КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ
Факультет автоматизації і інформаційних технологій
Кафедра кібербезпеки та комп'ютерної інженерії

КВАЛІФІКАЦІЙНА РОБОТА
ЗДОБУВАЧА СТУПЕНЯ ВИЩОЇ ОСВІТИ БАКАЛАВР

на тему:

СТВОРЕННЯ БЛОКЧЕЙН-РІШЕННЯ ДЛЯ СТРАХОВИХ СМАРТ-КОНТРАКТІВ

ЯРОШ ЗАХАР ОЛЕГОВИЧ

Київ, 2026

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БУДІВНИЦТВА І АРХІТЕКТУРИ
Факультет автоматизації і інформаційних технологій
Кафедра кібербезпеки та комп'ютерної інженерії

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., доцент Делембовський М. М.
«___» _____2026 року

КВАЛІФІКАЦІЙНА РОБОТА ЗДОБУВАЧА СТУПЕНЯ ВИЩОЇ ОСВІТИ БАКАЛАВР

Розроблення блокчейн-рішення для страхових смарт-контрактів

Я як здобувач вищої освіти КНУБА розумію і підтримую політику закладу з академічної доброчесності. Я не надавав(-ла) і не одержував(-ла) недозволену допомогу під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Здобувач Ярош Захар Олегович
123 «Комп’ютерна інженерія»
(спеціальність)
Комп'ютерні системи і мережі
(освітня програма)
Група КСМ-22_____
Керівник Гуменний Д. О.
(прізвище та ініціали)
Кандидат технічних наук, доцент
(вчене звання, науковий ступінь)
Рецензент Гончаренко Т. А.
(Прізвище та ініціали)
Ідентичність підтверджую

Київ 2026

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет: Автоматизації і інформаційних технологій
Випускова кафедра: Кібербезпеки та комп'ютерної інженерії
Ступінь вищої освіти: Бакалавр
Спеціальність: 123 Комп'ютерна інженерія
Освітня програма: Комп'ютерні системи і мережі

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., доцент Делембовський М.М.

„___" _____ 20___ року

З А В Д А Н Н Я
ДО ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ ЗДОБУВАЧА
СТУПЕНЯ ВИЩОЇ ОСВІТИ БАКАЛАВР

Ярош Захар Олегович
(прізвище, ім'я та по батькові здобувача)

1. Тема роботи Створення блокчейн-рішення для страхових смарт-контрактів затверджена наказом ректора КНУБА №577/52-15/13.2/26 від 14.05.2026 р

2. Керівник роботи Гуменний Дмитро Олександрович
Кандидат технічних наук, доцент
(прізвище, ім'я та по батькові, науковий ступінь, вчене звання)

3. Термін подання здобувачем роботи до захисту червень 2026 р

4. Зміст пояснювальної записки за розділами:

Р. 1. АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

Р. 2. ОБГРУНТУВАННЯ ТА РОЗРОБЛЕННЯ РІШЕННЯ

Р. 3. РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ

5. Календарний план виконання роботи:

Види робіт та їх зміст	Дата виконання
Розділ 1	24.04.2026
Розділ 2	07.05.2026
Розділ 3	16.05.2026
Остаточне оформлення роботи	02.06.2026
Направлення роботи для перевірки на плагіат	10.06.2026
Попередній захист роботи на випусковій кафедрі	12.06.2026
Направлення роботи на рецензування	15.06.2026

8. Дата видачі завдання травня 2026 року

Керівник _____ Гуменний Д.О.

Здобувач _____ Ярош З.О.

РЕЗЮМЕ (SUMMARY) *до кваліфікаційної випускової роботи здобувача*	**ПІБ** *здобувача українською та англійською мовами* Ярош Захар Олегович / Yarosh Zakhar Olegovich		
ЗВО	Київський національний університет будівництва і архітектури		
Тема (українською та англійською)	Створення блокчейн-рішення для страхових смарт-контрактів / Creating a blockchain solution for insurance smart contracts		
Освітній ступінь	Бакалавр		
Факультет	Автоматизації і інформаційних технологій		
Випускова кафедра	Кібербезпеки та комп'ютерної інженерії		
Спеціальність	123 - Комп'ютерна інженерія		
Освітня програма	Комп'ютерні системи і мережі		
Керівник			
Обсяг роботи:	*Пояснювальна записка, стор.*	*Розділів*	*Презентація, кількість слайдів*
	102	3	13
Розділ 1	Аналіз предметної галузі та постановка задачі		
Розділ 2	Обгрунтування та розроблення рішення		
Розділ 3	Реалізація та тестування		
Висновки по роботі	Розроблено та задеплоєно три смарт-контракти (MockOracle, FlightInsurance, WeatherInsurance) у тестовій мережі Polygon Amoy. Реалізовано параметричне страхування з автоматичною виплатою без участі третіх осіб.		
Ключові слова: Keywords:	смарт-контракт, блокчейн, параметричне страхування, Solidity, Polygon, оракул, децентралізація / smart contract, blockchain, parametric insurance, Solidity, Polygon, oracle, decentralization		

Здобувач Ярош З.О.

Керівник Гуменний Д.О.

АНОТАЦІЯ

Ярош З. О. Створення блокчейн-рішення для страхових смарт-контрактів.

Атестаційна випускна робота бакалавра за спеціальністю 123 «Комп'ютерна інженерія», освітня програма «Комп'ютерні системи і мережі». - Київський національний університет будівництва та архітектури. - Київ, 2026.

Кваліфікаційна випускна робота присвячена розробленню та реалізації блокчейн-рішення для страхових смарт-контрактів на основі параметричного страхування. Актуальність теми зумовлена системними недоліками традиційного страхового ринку: тривалими термінами виплат, високими адміністративними витратами та обмеженою прозорістю операцій.

Реалізовано повний цикл параметричного страхування: придбання полісу з оплатою страхової премії, автоматична перевірка настання страхового випадку через оракул та миттєва виплата страхового відшкодування без участі третіх осіб. Систему доповнено серверною частиною на Flask з бібліотекою web3.py та клієнтським інтерфейсом на React. Проведено функціональне тестування основних сценаріїв та аналіз газових витрат, що підтвердили економічну ефективність та технічну надійність розробленого рішення.

Практичне значення роботи полягає у демонстрації можливості повної автоматизації страхових виплат на основі об'єктивних даних, що скорочує час розрахунків з тижнів до секунд та усуває суб'єктивний людський фактор.

Ключові слова: смарт-контракт, блокчейн, параметричне страхування, Solidity, Polygon, оракул, децентралізація, Web3.

ABSTRACT

Yarosh Z. O. Creation of a blockchain solution for insurance smart contracts.

Bachelor's thesis in specialty 123 "Computer Engineering", educational program "Computer Systems and Networks". - Kyiv National University of Construction and Architecture. - Kyiv, 2026.

The qualification thesis is devoted to the development and implementation of a blockchain solution for insurance smart contracts based on parametric insurance. The relevance of the topic is driven by systemic shortcomings of the traditional insurance market: lengthy claim settlement periods, high administrative costs and limited transparency of operations.

A complete parametric insurance cycle has been implemented: policy purchase with insurance premium payment, automatic verification of insured event occurrence through an oracle, and instant insurance payout without third-party involvement. The system is complemented by a Flask backend with the web3.py library and a React client interface. Functional testing of key scenarios and gas cost analysis were conducted, confirming the economic efficiency and technical reliability of the developed solution.

The practical significance of the work lies in demonstrating the possibility of fully automating insurance payouts based on objective data, reducing settlement time from weeks to seconds and eliminating the subjective human factor.

Keywords: smart contract, blockchain, parametric insurance, Solidity, Polygon, oracle,decentralization,Web3.

ЗМІСТ

ВСТУП 10
РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ 12
1.1. Сучасний стан ринку страхування та огляд існуючих комп'ютерних систем для автоматизації страхових послуг 12
1.2. Аналіз традиційних моделей страхових контрактів та їх недоліків у контексті швидкості обробки подій та прозорості 15
1.3. Визначення вимог до блокчейн-рішення та формулювання задачі проєктування програмної реалізації 21
1.4. Огляд існуючих блокчейн-платформ та смарт-контрактів для страхових випадків 26
Висновок до розділу 1 30
РОЗДІЛ 2. ОБҐРУНТУВАННЯ ТА РОЗРОБЛЕННЯ РІШЕННЯ 32
2.1. Вибір та обґрунтування блокчейн-платформи та мови програмування смарт-контрактів ..... **Помилка! Закладку не визначено.**
2.2. Проєктування архітектури смарт-контрактів 32
2.3. Проєктування архітектури програмної системи страхових смарт-контрактів 37
2.4. Розроблення структури даних та логіки обробки страхових подій у смарт-контракті 51
2.5. Вибір та обґрунтування механізмів верифікації страхових випадків та оракулів даних 55
Висновок до розділу 2 59
РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ 61
3.1. Реалізація базової логіки страхового смарт-контракту та його розгортання в тестовій мережі 61
3.2. Розроблення та інтеграція клієнтського інтерфейсу для взаємодії зі смарт-контрактом 66
3.3. Функціональне тестування смарт-контракту на основних сценаріях страхових випадків 70

3.4. Тестування продуктивності та аналіз газових витрат для різних типів страхових операцій 87
Висновок до розділу 3 91
ВИСНОВКИ 93
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 95
ДОДАТОК А Програмні лістинги проєкту 101

ВСТУП

Сучасний страховий ринок перебуває в епіцентрі цифрової трансформації. Глобальний обсяг премій уже перевищує 6 трильйонів доларів США, а щорічне зростання становить 5–7 %. Водночас традиційні моделі страхування дедалі більше відстають від очікувань суспільства: обробка претензій триває тижнями, рівень шахрайства сягає 10–15 %, а адміністративні витрати поглинають до 30–40 % премій. Клієнти змушені чекати на виплати, а страхові компанії – утримувати великі штати експертів і резервів. У таких умовах блокчейн-технології та смарт-контракти відкривають реальний шлях до радикальних змін. Вони дозволяють створити повністю автоматизовані, прозорі та миттєві механізми параметричного страхування, де виплата відбувається автоматично за об'єктивними даними, без паперів, черг і суб'єктивних оцінок.

Актуальність теми зумовлена необхідністю подолати системні недоліки класичного страхування саме за допомогою децентралізованих технологій. Параметричні смарт-контракти вже доводять свою ефективність у реальних проєктах, таких як Etherisc чи Nexus Mutual, однак потребують адаптації під національні ринки, зокрема український, з урахуванням регуляторних вимог і практичних сценаріїв (затримки рейсів, погодні ризики). Саме тому створення повноцінного блокчейн-рішення з інтеграцією оракулів, клієнтського інтерфейсу та моніторингу продуктивності є сьогодні не лише технічним завданням, а й важливим кроком до підвищення доступності та довіри до страхових послуг.

Метою дипломної роботи є розроблення та реалізація блокчейн-рішення для страхових смарт-контрактів на основі параметричного страхування з використанням технологій смарт-контрактів, децентралізованих оракулів та Web3-інтерфейсу.

Для досягнення поставленої мети передбачено виконання таких завдань:

– провести аналіз сучасного стану ринку страхування та існуючих комп'ютерних систем автоматизації страхових послуг;

– дослідити традиційні моделі страхових контрактів та їхні ключові недоліки щодо швидкості обробки подій і прозорості;

– здійснити огляд існуючих блокчейн-платформ та смарт-контрактів для страхових випадків;

– визначити функціональні та нефункціональні вимоги до блокчейн-рішення;

– обґрунтувати вибір технологій, спроєктувати архітектуру системи, розробити структуру даних та логіку смарт-контрактів;

– реалізувати та протестувати базову логіку контрактів, клієнтський інтерфейс, а також провести оцінку продуктивності та газових витрат.

Об’єктом дослідження є процеси автоматизації страхових послуг у контексті цифрової трансформації фінансового сектору.

Предметом дослідження виступає блокчейн-рішення для страхових смарт-контрактів, зокрема його архітектура, логіка обробки подій, механізми верифікації даних через оракули та інтеграція з клієнтським інтерфейсом.

Розроблене рішення не лише демонструє технічну можливість швидких і прозорих виплат, а й закладає практичну основу для подальшого розвитку InsurTech (технологічних інновацій у страхуванні) в Україні та світі.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Сучасний стан ринку страхування та огляд існуючих комп'ютерних систем для автоматизації страхових послуг

Сучасний стан страхового ринку відзначається докорінними перетвореннями під впливом глобальних економічних викликів, трансформації споживчих пріоритетів та інтенсивного впровадження інформаційних технологій. Згідно з міжнародними аналітичними даними, світовий обсяг страхових послуг перевищив 6 трильйонів доларів США у 2023 році, забезпечуючи щорічний приріст на рівні 5–7% попри інфляційні процеси та геополітичну нестабільність [1]. Подібна динаміка зумовлена розширенням страхового захисту в країнах із перехідною економікою та актуалізацією запиту на нові інструменти убезпечення, зокрема щодо захисту віртуального простору та екологічних загроз.

Водночас сфера стикається з низкою хронічних труднощів, як-от надмірна тривалість опрацювання звернень, значний рівень недоброчесності клієнтів, що сягає 10–15% від загального обсягу виплат, та обтяжливі управлінські витрати, які в межах класичних підходів становлять до 40% страхових внесків. Означені перешкоди спонукають страхові компанії до пошуку прогресивних методик, покликаних зміцнити довіру споживачів та підвищити результативність роботи. У процесі переходу до цифрового формату ключовим чинником змін стає розвиток технологічного страхування. Новітні комп'ютерні комплекси для автоматизації страхової діяльності еволюціонували від простих інформаційних сховищ до інтегрованих середовищ, що застосовують методи штучного інтелекту, алгоритми машинного навчання та поглиблений аналіз масивів даних для передбачення ймовірних загроз і розробки персоналізованих умов [2].

Застарілі програмні комплекси на базі великих обчислювальних машин забезпечували лише базову реєстрацію договорів, проте характеризувалися розрізненістю інформації, ручним обробленням паперів і повільним реагуванням на страхові випадки. Натомість сучасні програмні продукти об'єднують мережеві сервіси, інтерфейси програмної взаємодії та автоматизовані алгоритми виконання

робіт, що дозволяє скоротити термін розгляду заявки з кількох тижнів до лічених годин. Особливо відчутним є переорієнтування на параметричні моделі захисту, де компенсації прив'язані до об'єктивних зовнішніх чинників, наприклад погодних показників чи статистичних маркерів, що нівелює суб'єктивність при оцінюванні втрат і прискорює проведення розрахунків [3].

Для ґрунтовного розуміння розвитку процесів доцільно звернутися до графічного зображення традиційного способу опрацювання вимог про відшкодування, що представлене на рисунку 1.1.

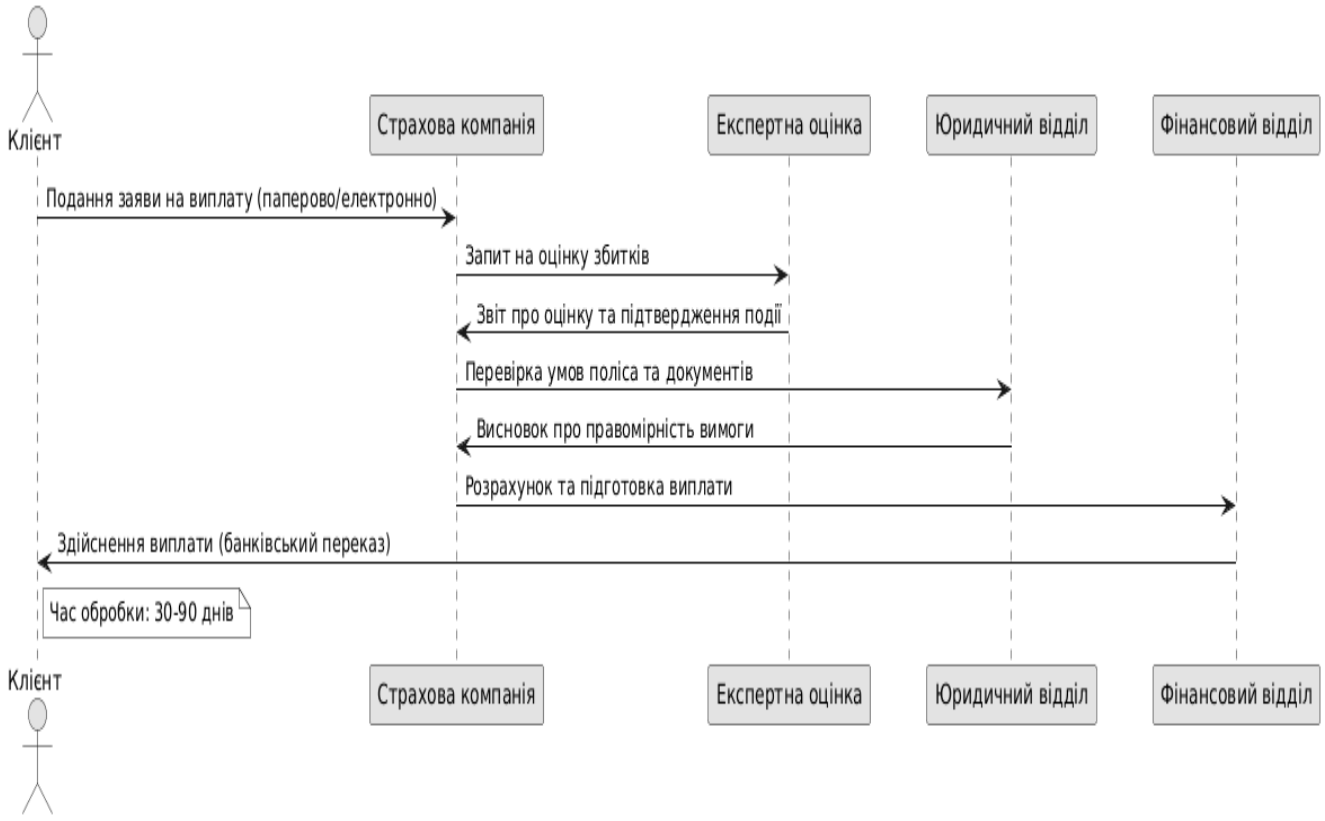

Рисунок 1.1 – Схема традиційного процесу обробки страхових вимог у класичних страхових компаніях

Наведена схема засвідчує, що звичний алгоритм дій передбачає багатоетапну взаємодію різних структурних ланок, що спричиняє суттєві затримки та зростання експлуатаційних витрат. Порівняно з цим новітні автоматизовані системи значно спрощують цей ланцюг через залучення відомостей із зовнішніх джерел у реальному часі та застосування логічних правил безпосередньо в програмному коді [4].

Аналіз доступних комп’ютерних засобів вказує на виразну різницю між корпоративними продуктами та інноваційними технологічними платформами. Провідні розробники, як-от Guidewire Software чи Duck Creek Technologies, пропонують комплексні середовища для адміністрування полісів, що супроводжують увесь цикл послуги від моменту оцінки ризику до виплати відшкодування.

Такі рішення базуються на модульній побудові, допускають поєднання з мобільними програмами та гарантують дотримання законодавчих норм. Одночасно молоді технологічні компанії, наприклад Lemonade, впроваджують інтелектуальні моделі, де діалогові програми та алгоритми розпізнавання образів автоматично вирішують більшість типових запитів [5]. У країнах Заходу популярності набули системи на базі хмарних середовищ, що поєднують управління взаємовідносинами з клієнтами та аналітику. В азійському регіоні активно випробовують інструменти на основі розподілених баз даних для зміцнення прозорості операцій.

Щоб упорядкувати розбіжності між різними підходами, варто скористатися порівняльним аналізом, що міститься в таблиці 1.1.

Таблиця 1.1 – Порівняльна характеристика традиційних та сучасних комп’ютерних систем автоматизації страхових послуг

Аспект	**Традиційні системи**	**Сучасні InsurTech-системи**
Час обробки претензії	30–90 днів	1–48 годин
Рівень автоматизації	Низький (значна частка ручної праці)	Високий (AI (штучний інтелект) та RPA (роботизована автоматизація процесів) до 90%)
Джерела даних	Внутрішні бази, паперові документи	Інтеграція з API, IoT (Інтернет речей), зовнішніми оракулами
Прозорість операцій	Обмежена, суб’єктивна оцінка	Повна, на основі об’єктивних параметрів
Вартість адміністрування	Висока (30–40% премій)	Знижена (5–15% завдяки автоматизації)

Табличні показники підтверджують, що впровадження оновлених систем не тільки прискорює ділові процеси, а й мінімізує ймовірність помилок та покращує досвід користувачів [6]. В Україні галузь також виявляє схильність до переведення операцій у цифровий простір, хоча цей рух відбувається повільніше за світові темпи. За відомостями Національного банку України, обсяг залучених коштів у 2023 році зріс на 21% і перевищив 60 мільярдів гривень, причому популярність добровільного страхування зростає завдяки можливості онлайн-оформлення.

Більшість вітчизняних підприємств використовують локальні системи управління або власні розробки, які поступово модернізують шляхом підключення до державних реєстрів і сервісів електронного обміну документами. Вихід на ринок спеціалізованих платформ від ключових гравців дозволяє громадянам укладати угоди через смартфони з миттєвим підтвердженням особистих відомостей [7]. Проте, на відміну від світових практик, в Україні все ще переважає звична модель із обов'язковою паперовою фіксацією у складних ситуаціях, що стримує повний перехід на автоматичні рейки. Водночас ініціативи регулятора щодо створення спеціальних умов для тестування технологічних новинок створюють фундамент для перевірки рішень, що базуються на швидких виплатах.

Підсумовуючи, вивчення поточного стану ринку та огляд програмного забезпечення свідчить про зміщення акцентів від бюрократизованих процедур до інтелектуальних систем, що здатні миттєво реагувати на виклики середовища. Ця еволюція сприяє загальній доступності послуг і робить їх зрозумілішими для широкого загалу. Подальше просування в цьому напрямі потребує цілісного підходу до поєднання передових технологій із урахуванням особливостей національного законодавства та запитів місцевого ринку [8].

1.2. Аналіз традиційних моделей страхових контрактів та їх недоліків у контексті швидкості обробки подій та прозорості

Традиційні моделі страхових контрактів, які протягом десятиліть формували основу світового страхового ринку, ґрунтуються переважно на

принципі індемнітету, тобто повного або часткового відшкодування фактично понесених збитків [9]. У цій системі страхувальник сплачує премію за поліс, а в разі настання страхового випадку зобов'язаний ініціювати процедуру подання заяви, зібрати об'ємний пакет документів, надати докази шкоди та пройти низку адміністративних етапів перевірки. Такий підхід, хоча й забезпечує певний рівень правової захищеності, демонструє фундаментальні недоліки, особливо коли йдеться про швидкість реагування на страхові події та прозорість усього ланцюга взаємодії між сторонами.

Процес обробки страхової події в традиційній моделі зазвичай розпочинається з моменту, коли страхувальник самостійно повідомляє компанію про подію. Далі слідує етап збору документів, призначення огляду експертом-оцінювачем, проведення детального обстеження збитків, розрахунок розміру відшкодування з урахуванням франшизи, виключень і умов договору. Кожен із цих кроків вимагає участі кількох ланок: страхового агента, андеррайтера, експерта, юридичного відділу та, за потреби, перестраховиків.

Така ланцюгова структура, на перший погляд, спрямована на запобігання зловживанням, на практиці стає джерелом значних затримок. За даними галузевих досліджень, середній термін розгляду стандартної заяви в майновому страхуванні становить від 23 до 30 днів, а в складних випадках, пов'язаних зі стихійними лихами чи великими збитками, може сягати кількох місяців або навіть років [10].

Одним із найсуттєвіших недоліків традиційних контрактів є низька швидкість обробки подій, зумовлена переважно ручними процесами та залежністю від людського фактора. Кожна заява проходить багатоступеневу верифікацію, що включає перевірку обставин події, оцінку реального розміру шкоди та узгодження з умовами поліса. У періоди масових страхових випадків, наприклад, під час природних катастроф, страховики стикаються з перевантаженням, що призводить до черг, помилок у документообігу та додаткових затримок. Дослідження показують, що навіть у розвинених ринках середній цикл ремонту за автострахуванням сягає 22–24 днів, а для майнових

полісів цей показник часто перевищує 23 дні, при цьому катастрофічні події вимагають ще більше часу [11].

Такі терміни не відповідають сучасним очікуванням споживачів, які звикли до миттєвих транзакцій в інших сферах економіки. Крім того, затримки генерують додаткові витрати для самих страховиків: утримання резервів, юридичні спори та втрату репутації. У контексті глобальних змін клімату, коли частота й інтенсивність страхових подій зростає, повільність традиційної моделі стає не просто незручністю, а системною вразливістю, що обмежує ефективність захисту населення та бізнесу [12].

Поряд зі швидкістю критично важливою проблемою залишається недостатня прозорість традиційних страхових контрактів. Полісі часто містять складні формулювання, численні виключення та умови, які важко зрозуміти без спеціальної юридичної підготовки. Страхувальник, як правило, не має реального доступу до внутрішніх процесів розгляду своєї заяви: він не знає, на якому етапі перебуває перевірка, хто саме відповідає за рішення і за якими критеріями оцінюється шкода. Це створює ґрунт для непорозумінь, спорів і навіть підозр у упередженості.

У багатьох випадках рішення про відмову у виплаті або зменшення суми відшкодування сприймається як непрозоре, оскільки відсутній чіткий аудиторський слід кожного кроку. Наукові дослідження підкреслюють, що така непрозорість знижує рівень довіри споживачів і стримує розвиток ринку, особливо в країнах, де страхова культура ще формується [13]. Крім того, відсутність відкритості сприяє появі інформаційної асиметрії: страхова компанія володіє повним обсягом даних, тоді як клієнт змушений покладатися на загальні пояснення. У результаті зростає кількість судових спорів, що ще більше уповільнює процеси та підвищує витрати для всіх учасників ринку [14].

Для наочності традиційний процес обробки страхової події можна представити у вигляді послідовної схеми, що наведена на рисунку 1.2, яка ілюструє основні етапи та взаємодію сторін.

Рисунок 1.2 – Схема традиційного процесу розгляду страхової заяви

Ця схема підкреслює багатоланковість і послідовність етапів, кожен з яких може стати «вузьким місцем» і спричинити затримку. На практиці такий ланцюг часто розтягується в часі через необхідність фізичного огляду, погоджень з перестраховиками чи додаткових експертиз.

Середні терміни обробки заявок у традиційних моделях за даними галузевих досліджень 2024–2026 рр. наведені в таблиці 1.2.

Таблиця 1.2 – Середні терміни обробки страхових заявок у традиційних моделях

Вид страхування	**Середній термін розгляду (дні)**	**Максимальний термін у складних випадках (дні)**
Автострахування	22–24	60–90

Майнове страхування	23–30	120–180
Катастрофічні ризики	30–45	180–360+

Як видно з наведених даних, навіть стандартні випадки вимагають кількох тижнів, а складні події перетворюються на довготривалий процес, що суттєво знижує практичну цінність страхового захисту. Непрозорість на етапах оцінки та прийняття рішення лише посилює проблему, адже клієнт часто залишається в інформаційному вакуумі, не маючи можливості оперативно впливати на хід розгляду [15].

Для глибшого розуміння сутності проблеми доцільно розглянути порівняльні характеристики традиційних моделей страхових контрактів (таблиця 1.3), які висвітлюють як сильні, так і слабкі сторони цього підходу.

Таблиця 1.3 – Переваги та недоліки традиційних моделей страхових контрактів

Аспект	**Переваги**	**Недоліки**
Захист прав сторін	Високий рівень юридичної формалізації	Тривалі терміни розгляду та виплат
Оцінка збитків	Індивідуальний підхід до кожного випадку	Суб’єктивність оцінок та можливі спори
Прозорість	Наявність письмових договорів	Закритість внутрішніх процесів для страхувальника
Операційні витрати	Накопичений досвід і відпрацьовані процедури	Високі адміністративні витрати (до 30–40 % премії)
Доступність	Широка мережа агентів	Обмежена оперативність у кризових ситуаціях

Наведена порівняльна характеристика наочно демонструє, що традиційні моделі, попри певні сильні сторони, суттєво поступаються сучасним вимогам щодо швидкості та відкритості. Саме поєднання тривалих термінів розгляду, високих адміністративних витрат, суб’єктивності оцінок і недостатньої прозорості

створює значні перешкоди для ефективного страхування в умовах сучасної економіки.

У сукупності низька швидкість і недостатня прозорість традиційних моделей страхових контрактів призводять до системних наслідків для ринку. По-перше, вони підривають довіру споживачів, які дедалі частіше віддають перевагу альтернативним формам захисту ризиків.

По-друге, зростають операційні витрати страховиків на утримання великих штатів експертів, юристів і адміністративного персоналу.

По-третє, в умовах зростання частоти страхових подій через зміну клімату та техногенних ризиків повільність моделі стає фактором, що обмежує доступність страхового покриття для широких верств населення та малого бізнесу.

Таким чином, аналіз традиційних підходів свідчить про нагальну потребу в удосконаленні механізмів реагування на страхові події, де ключовими пріоритетами мають стати прискорення виплат і максимальна відкритість усіх процесів для всіх зацікавлених сторін [16].

Доцільність розробки програмних рішень для вдосконалення страхових механізмів полягає в необхідності подолання описаних системних недоліків традиційних моделей. Сучасні технології дозволяють автоматизувати процеси перевірки умов настання страхових подій, забезпечити миттєве виконання виплат на основі об’єктивних даних та створити повну прозорість усіх операцій.

Такий підхід суттєво скорочує адміністративні витрати, підвищує довіру страхувальників і робить страховий захист доступнішим і ефективнішим. Розробка відповідних програмних продуктів стає стратегічно важливим напрямом, оскільки дозволяє поєднати надійність традиційного страхування з оперативністю та відкритістю, притаманними цифровим технологіям, що особливо актуально в умовах зростання ризиків і очікувань споживачів щодо швидкого реагування.

Підсумовуючи, традиційні моделі страхових контрактів, попри свою історичну роль у забезпеченні фінансової стабільності, сьогодні стикаються з

серйозними викликами в частині швидкості обробки подій та прозорості. Ці недоліки не лише ускладнюють життя страхувальників у критичні моменти, але й гальмують розвиток усієї галузі, вимагаючи пошуку інноваційних рішень, здатних поєднати надійність захисту з сучасними вимогами оперативності та відкритості.

1.3. Визначення вимог до блокчейн-рішення та формулювання задачі проєктування програмної реалізації

Традиційні механізми врегулювання страхових вимог у сучасному секторі часто супроводжуються тривалим розглядом справ, значними адміністративними витратами та недостатньою прозорістю для учасників ринку. Блокчейн-технології відкривають принципово нові можливості для оптимізації таких процесів.

Особливо актуальним постає впровадження параметричного страхування, де компенсація виплачується автоматично за настання об’єктивних подій, як-от затримка транспорту чи відхилення метеорологічних показників. Це дозволяє уникнути складних експертиз та подання численних документів.

Такий підхід мінімізує ризик людського фактора, зміцнює довіру між сторонами та забезпечує миттєвість розрахунків у контексті зростання кліматичних ризиків [17]. Визначення вимог до блокчейн-рішення потребує комплексного аналізу функціональних та нефункціональних аспектів для формування чіткого бачення програмної реалізації, здатної поєднувати децентралізовані розумні контракти з реальними зовнішніми даними.

Функціональні вимоги до такого рішення спрямовано на автоматизацію ключових бізнес-процесів. До них віднесено реєстрацію та придбання страхових полісів з фіксацією параметрів ризику, механізми введення зовнішніх даних через надійні вузли зв'язку для фіксації подій, автоматичне виконання виплат із загального фонду при підтвердженні тригерів, а також підтримку управління ліквідністю через поповнення резервів.

Система має забезпечувати доступ до актуальної статистики щодо кількості полісів, обсягів премій і балансу резервів, а також інтеграцію з інструментами аналізу мережевої активності для підвищення прозорості. Важливим аспектом визначено реалізацію моніторингу подій у реальному часі, що дозволяє учасникам

оперативно отримувати інформацію про нові поліси та виконані транзакції. Ці вимоги обумовлено необхідністю створити замкнутий цикл, де кожна операція фіксується в незмінному реєстрі, а виконання контракту відбувається без посередників [18].

Нефункціональні вимоги визначають якісні характеристики рішення та його здатність функціонувати в реальному середовищі розподілених реєстрів. На першому місці стоїть безпека та незмінність даних, адже розумні контракти мають бути захищені від вразливостей, а транзакції гарантовано збережені після запису в ланцюг.

Масштабованість та продуктивність системи передбачають підтримку високої швидкості обробки операцій, низьку вартість обчислювальних ресурсів та можливість роботи при зростанні кількості учасників.

Взаємодія з зовнішніми джерелами даних є критичною, оскільки вони забезпечують зв’язок між блокчейном та динамічним реальним світом, що вимагає верифікації інформації з кількох джерел. Користувацька зручність реалізується через зрозумілий інтерфейс для здійснення операцій з полісами та перегляду статистики.

Система також повинна включати інструменти моніторингу продуктивності та візуалізації даних для аналізу стану страхових фондів. Такі вимоги обґрунтовано потребою забезпечити технічну надійність та доступність технології для широкого кола учасників [19].

Для систематизації зазначених вимог доцільно звернутися до їх класифікації, наведеної в таблиці 1.4, яка допомагає чітко розмежувати пріоритети проєктування

Таблиця 1.4 – Класифікація вимог до блокчейн-рішення для параметричного страхування

Категорія вимог	**Основні складові**	**Обґрунтування та вплив на реалізацію**
Функціональні	Придбання полісів, введення даних оракула, автоматичні виплати, управління пулами, статистика та моніторинг подій	Забезпечує повний цикл страхового процесу без посередників [20]
Нефункціональні (безпека та надійність)	Іммутабільність даних, захист смарт-контрактів, верифікація оракулів	Гарантує довіру та стійкість до маніпуляцій [17]
Продуктивність та масштабованість	Висока TPS, низька вартість транзакцій, оптимізація газу	Дозволяє масштабування на реальні обсяги ринку [18]
Користувацька зручність	Інтуїтивний інтерфейс, сповіщення у реальному часі, доступність для нетехнічних користувачів	Підвищує адаптацію технології серед страхувальників [21]
Інтеграційні та аналітичні	Інтеграція з оракулами, API, інструменти аналізу мережі та ринку	Підтримує прозорість та прийняття рішень [22]

Ця класифікація підкреслює, що успішне блокчейн-рішення повинно гармонійно поєднувати технічну досконалість з орієнтацією на кінцевого користувача, забезпечуючи економічну ефективність та відповідність регуляторним нормам.

Для наочного представлення взаємодії основних учасників системи з її ключовими функціональними можливостями доцільно розглянути діаграму варіантів використання на рисунку 1.3. Ця діаграма ілюструє, як страхувальник, джерело зовнішніх даних та постачальник ліквідності взаємодіють із центральним рішенням через сценарії придбання полісів, введення даних, виконання виплат та управління фондами. Це чітко демонструє замкнутий цикл параметричного страхування та ролі кожного актора в забезпеченні автоматизації процесів.

Рисунок 1.3 – Діаграма варіантів використання блокчейн-рішення для параметричного страхування

На рисунку 1.3 наочно продемонстровано, як основні актори взаємодіють із центральним блокчейн-рішенням через чітко визначені сценарії, що забезпечує автоматизований цикл страхування.

Аналіз вимог показує, що для реалізації параметричного страхування доцільно використовувати декілька взаємопов’язаних смарт-контрактів. Центральне місце посідає MockOracle (або повноцінний оракул у продакшені), який відповідає за введення та зберігання зовнішніх даних про страхові події. Два основні страхові контракти – FlightInsurance та WeatherInsurance – забезпечують незалежне управління полісами різних типів ризиків. Кожен з них приймає специфічні параметри: для авіаційного страхування це ідентифікатор рейсу та поріг затримки (за замовчуванням 180 хвилин), для погодного – ідентифікатор

регіону, поріг опадів, тривалість сезону та розмір премії. Така модульна архітектура дозволяє гнучко розширювати систему новими типами страхування, забезпечуючи при цьому ізольованість страхових пулів та прозорість операцій. Характеристики ключових смарт-контрактів блокчейн-рішення, які планується використовувати у проєкті, представлені в таблиці 1.5.

Таблиця 1.5 – Характеристики ключових смарт-контрактів блокчейн-рішення

Смарт-контракт	**Основні функції**	**Приймані параметри**	**Відповідальність у системі**
MockOracle	Звіт про події, зберігання даних	flightId + delayMinutes; regionId + rainfallMm	Надання достовірних зовнішніх даних для тригерів
FlightInsurance	Купівля полісів, перевірка та виплата	flightId (string), фіксована премія 0.1 MATIC, поріг 180 хв	Автоматизація авіаційного параметричного страхування
WeatherInsurance	Купівля полісів з налаштуваннями, перевірка та виплата	regionId, thresholdMm, durationDays, premiumMatic	Гнучке погодне страхування від посухи

Така структура забезпечує чіткий розподіл відповідальності, високу модульність та можливість незалежного масштабування окремих видів страхування.

Формулювання задачі проєктування програмної реалізації полягає у створенні комплексної децентралізованої платформи, яка об’єднує розумні контракти для управління полісами, надійні механізми інтеграції з зовнішніми даними, серверну частину для обробки запитів у реальному часі та клієнтський інтерфейс. Програмна реалізація повинна забезпечити повну прозорість усіх операцій, мінімізувати залежність від центральних органів та дозволити оперативне реагування на події для підвищення доступності послуг у цифрову епоху [22].

Такий підхід закладає основу для розвитку фінансових технологій, де автоматизація стає стандартом, а довіра ґрунтується на коді та незмінному реєстрі.

Чітке визначення вимог і коректне формулювання задачі проєктування створюють надійну основу для розробки рішення, здатного радикально трансформувати страхову галузь. Сучасний стан ринку характеризується зростанням глобального обсягу премій понад 6 трильйонів доларів США, проте галузь стикається з тривалим циклом обробки претензій та високими адміністративними витратами.

Традиційні комп'ютерні системи демонструють фрагментованість даних, тоді як сучасні рішення з інтеграцією штучного інтелекту та хмарних технологій дозволяють скоротити час розгляду заяв до кількох годин. Вітчизняний ринок також зазнає трансформації з поступовим впровадженням онлайн-каналів продажів. Традиційні моделі контрактів призводять до затримок у виплатах від 22 днів до кількох місяців, що знижує довіру страхувальників.

Огляд блокчейн-платформ Ethereum, Polygon чи Solana розкриває їхній потенціал для автоматизації виплат на основі верифікованих даних, мінімізуючи людський фактор. Реалізації на кшталт Etherisc підтверджують ефективність таких рішень. Задача проєктування полягає у створенні платформи, здатної забезпечити прозорість та суттєве підвищення ефективності страхових послуг.

1.4. Огляд існуючих блокчейн-платформ та смарт-контрактів для страхових випадків

Сучасний страховий ринок традиційно стикається зі значними адміністративними витратами, тривалим розглядом заяв на виплату, високим рівнем шахрайства та обмеженою доступністю послуг для певних категорій ризиків. Це особливо відчутно в умовах глобальних кліматичних змін і цифрової трансформації економіки. У такому контексті технології розподіленого реєстру та розумні контракти дозволяють фундаментально переосмислити страхові процеси.

Завдяки автоматизації, прозорості та об'єктивності забезпечується новий підхід до параметричного страхування, де виплати активуються автоматично за

чітко визначеними зовнішніми показниками, як-от тривалість затримки рейсу чи обсяг опадів. Це усуває потребу в суб'єктивній оцінці збитків і паперовій документації, що скорочує час від настання страхового випадку до отримання компенсації з кількох тижнів до кількох секунд [23].

Блокчейн-платформи, на яких будуються такі рішення, різняться за архітектурою, здатністю до масштабування та рівнем децентралізації. Це безпосередньо впливає на їхню придатність для страхової галузі. Ethereum залишається основною платформою завдяки віртуальній машині EVM та мові програмування Solidity, яка дозволяє створювати складні контракти з вбудованими механізмами оракулів для отримання зовнішніх даних.

Однак висока вартість обчислювальних ресурсів і обмежена пропускна здатність базової мережі спонукають до активного використання рішень другого рівня. Наприклад, мережа Polygon забезпечує суттєве зниження комісій і підвищення швидкості обробки операцій при збереженні сумісності з екосистемою Ethereum. Це робить її привабливою для страхових продуктів, що передбачають масові мікрополіси або часте оновлення даних від постачальників інформації [24].

Поряд із цим платформи з високою пропускною здатністю, як-от Solana, демонструють переваги для сценаріїв, де потрібна обробка тисяч операцій за секунду. Це актуально для параметричного страхування сільськогосподарських ризиків у регіонах, що розвиваються. Solana використовує унікальний механізм доказу історії у поєднанні з часткою володіння, що забезпечує низьку вартість і швидкість, хоча й вимагає додаткових заходів для стабільності мережі.

Інші рішення для підприємств, зокрема Hyperledger Fabric та R3 Corda, орієнтовані на приватні мережі. У них акцент робиться на конфіденційності даних і відповідності регуляторним нормам, що важливо для традиційних компаній, які інтегрують нові технології у гібридні моделі зі застарілими системами [25].

Для ілюстрації ключових відмінностей між основними платформами, що застосовуються в страхових смарт-контрактах, доцільно звернутися до порівняльного аналізу їхніх технічних характеристик у таблиці 1.6 [26]. Дана

таблиця підкреслює, як вибір середовища безпосередньо впливає на економічну ефективність продуктів. Для масових параметричних полісів із низькими преміями пріоритетними стають рішення з мінімальними комісіями, тоді як для складних корпоративних контрактів важливіша конфіденційність і контроль доступу.

Таблиця 1.6 – Порівняльна характеристика блокчейн-платформ для реалізації страхових смарт-контрактів

Платформа	**Механізм консенсусу**	**Пропускна здатність (TPS)**	**Середня комісія за транзакцію**	**Сумісність з EVM**	**Основні переваги для страхування**
Ethereum	Proof of Stake	15–30	Висока (~$1–10)	Так	Зріла екосистема оракулів, висока безпека
Polygon	Proof of Stake (L2)	До 7200	Низька (~$0,01)	Так	Масштабованість, низькі витрати, швидкість
Solana	Proof of History + PoS	2000–71000	Дуже низька (~$0,00025)	Ні	Висока швидкість для реального часу даних
Hyperledger Fabric	Permissioned consensus	Залежить від конфігурації	Відсутня (приватна)	Ні	Конфіденційність, регуляторна відповідність

Особливе місце в еволюції страхових смарт-контрактів посідає параметричне страхування, яке ідеально поєднується з можливостями розподілених реєстрів. На відміну від традиційного відшкодування збитків, де виплата залежить від індивідуального оцінювання, параметричні контракти спрацьовують автоматично при надходженні верифікованих даних від децентралізованих джерел. Це радикально знижує адміністративні витрати й підвищує доступність послуг для вразливих груп населення, зокрема для фермерів у регіонах із високим ризиком посухи [27].

Серед реальних втілень смарт-контрактів вирізняється Etherisc – відкрита децентралізована платформа для створення параметричних продуктів. Її рішення для страхування авіарейсів використовує дані про статуси польотів і автоматично

виконує виплати при затримці, забезпечуючи прозорість усіх операцій. Також Etherisc підтримує погодне страхування, де виплати прив'язані до метеорологічних індексів.

Іншим знаковим прикладом є Nexus Mutual – платформа на базі Ethereum для захисту від ризиків у сфері децентралізованих фінансів. Учасники спільно формують капітальний пул, а рішення про виплати приймаються спільнотою через механізм автономної організації. Платформа InsurAce розвиває багатоланцюгову модель, пропонуючи покриття ризиків одночасно на кількох мережах [25].

Загальна архітектура параметричного страхового смарт-контракту на блокчейні наведена на рисунку 1.4. Ця схема ілюструє замкнутий цикл взаємодії, де страхувальник купує поліс, інформаційне джерело забезпечує об'єктивні дані, а блокчейн гарантує незмінність і автоматичне виконання виплат. Такий підхід мінімізує людський фактор і підвищує довіру до системи [26].

Рисунок 1.4 – Загальна архітектура параметричного страхового смарт-контракту на блокчейні

Огляд існуючих платформ і розумних контрактів демонструє значний потенціал технології для вирішення системних проблем галузі. Ethereum та Polygon домінують завдяки зрілості середовища, тоді як високопродуктивні рішення відкривають двері для нових масштабних застосувань.

Параметричні моделі вже доводять свою ефективність, знижуючи витрати та прискорюючи виплати. Подальший розвиток цих рішень та їх регуляторна адаптація дозволять технології розподіленого реєстру стати ключовим рушієм справедливого страхового ринку в глобальному масштабі [14].

1.5 Висновок до розділу

Сучасний страховий ринок характеризується значним зростанням обсягів послуг та активним впровадженням цифрових технологій, що зумовлює перехід від традиційних бюрократичних моделей до інтелектуальних автоматизованих систем. Попри позитивну динаміку, галузь зберігає суттєві системні проблеми, пов’язані з тривалим опрацюванням страхових вимог, високими адміністративними витратами та обмеженою прозорістю операцій. Традиційні моделі страхових контрактів, засновані на принципі індемнітету, демонструють низьку оперативність реагування на страхові події та значну суб’єктивність оцінювання збитків, що призводить до зниження довіри споживачів і зростання операційних витрат.

Аналіз наявних комп’ютерних рішень свідчить про суттєву перевагу сучасних InsurTech-платформ, які використовують штучний інтелект, роботизовану автоматизацію процесів та інтеграцію з зовнішніми джерелами даних. Однак для кардинального вирішення наявних недоліків необхідні більш радикальні технологічні інструменти. Блокчейн-технології у поєднанні з параметричним страхуванням відкривають принципово новий підхід, що дозволяє автоматизувати виплати на основі об’єктивних зовнішніх тригерів, забезпечуючи максимальну прозорість, незмінність даних і суттєве скорочення часу розрахунків.

Визначені функціональні та нефункціональні вимоги до блокчейн-рішення акцентують увагу на необхідності створення модульної архітектури розумних контрактів, надійних механізмів оракулів, високої продуктивності та зручного користувацького інтерфейсу. Огляд провідних блокчейн-платформ підтверджує перспективність використання середовищ з високою масштабованістю та

низькими транзакційними витратами для реалізації параметричних страхових продуктів.

Таким чином, комплексний аналіз предметної галузі засвідчує нагальну потребу в розробці децентралізованої програмної платформи параметричного страхування на основі блокчейну, здатної поєднати надійність традиційного страхування з оперативністю та прозорістю сучасних цифрових технологій.

РОЗДІЛ 2. ОБҐРУНТУВАННЯ ТА РОЗРОБЛЕННЯ РІШЕННЯ

2.1. Проєктування архітектури смарт-контрактів

У процесі розроблення системи CLAIMLESS особлива увага приділялася проєктуванню архітектури смарт-контрактів, оскільки саме вони становлять ядро параметричного страхового механізму. Основна ідея полягала в створенні детермінованих, прозорих і автоматизованих компонентів, які забезпечують виконання страхових умов без участі посередників. Смарт-контракти були спроєктовані з урахуванням специфіки тестової мережі Polygon Amoy, особливостей мови Solidity версії 0.8.20 та вимог до параметричного страхування, де виплата відбувається автоматично за чітко визначеними об’єктивними критеріями.

Центральне місце в архітектурі посідають три основні контракти: FlightInsurance, WeatherInsurance та MockOracle. Кожен з них виконує чітко визначені функції, водночас взаємодіючи між собою через механізм оракула. Такий модульний підхід дозволяє легко розширювати систему новими типами страхування в майбутньому, зберігаючи стабільність існуючої логіки. Контракти спроєктовані з урахуванням принципів мінімалізму коду, економії газу та максимальної детермінованості, що є критично важливим для фінансових застосунків на блокчейні.

Контракт FlightInsurance реалізує механізм страхування затримок авіарейсів. Він містить фіксовані параметри: премію в розмірі 0.1 MATIC, виплату 0.5 MATIC при затримці від 180 хвилин та функції buyPolicy, checkAndPayout, getPolicy і getStats. Логіка контракту побудована навколо зберігання даних про поліси у внутрішніх структурах, що забезпечує швидкий доступ до інформації про кожного страхувальника. Важливим елементом є подія PolicyCreated та PayoutExecuted, які дозволяють зовнішнім системам відстежувати ключові стани в реальному часі.

Аналогічно контракт WeatherInsurance адаптовано під погодні ризики, зокрема посуху. На відміну від авіаційного, цей контракт підтримує

параметризацію: користувач може самостійно вказувати регіон, поріг опадів та тривалість сезону. Функції buyPolicy, checkAndPayout та getStats забезпечують гнучкість, а внутрішня структура Policy зберігає дані про сезонний період, що дозволяє автоматично перевіряти актуальність полісу. Такий дизайн відображає реальні потреби аграрного сектору, де умови страхування значно варіюються залежно від регіону.

Сполучною ланкою між реальним світом та блокчейном виступає контракт MockOracle. У демонстраційній версії він дозволяє вручну повідомляти дані про затримки рейсів (reportFlightDelay) та рівень опадів (reportRainfall). У продакшен-версії цей компонент легко замінюється на децентралізовані оракули типу Chainlink. Такий підхід забезпечує розділення обов'язків: страхові контракти не містять логіки отримання зовнішніх даних, що підвищує їхню безпеку та аудитопридатність.

Для кращого розуміння взаємодії компонентів системи варто розглянути структурну схему архітектури смарт-контрактів, що представлена на рисунку 2.1.

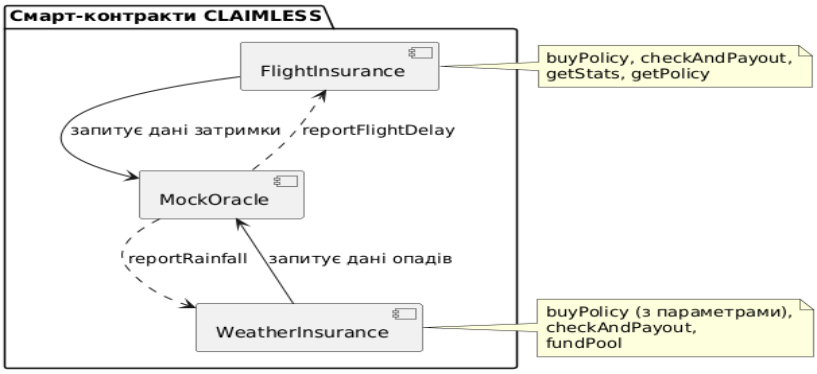

Рисунок 2.1 – Структурна схема взаємодії смарт-контрактів у системі CLAIMLESS

Як видно з рисунку, архітектура побудована за принципом «оракул як сервіс», що є поширеним патерном у DeFi-рішеннях. Це дозволяє страховим контрактам залишатися чистими від зовнішніх залежностей та зосередитися виключно на логіці виплат.

Важливим аспектом проєктування стало визначення подій та структур даних. Усі контракти активно використовують події Solidity для індексації та

моніторингу. Backend-компонент системи (event_indexer.py) підписується на ці події, зберігаючи їх у SQLite для подальшого використання у фронтенді через SSE-потік. Такий підхід забезпечує реальний час оновлення інформації про поліси, виплати та оновлення оракула.

Для ілюстрації типового сценарію взаємодії учасників системи доцільно розглянути діаграму послідовності основного страхового циклу, що наведена на рисунку 2.2.

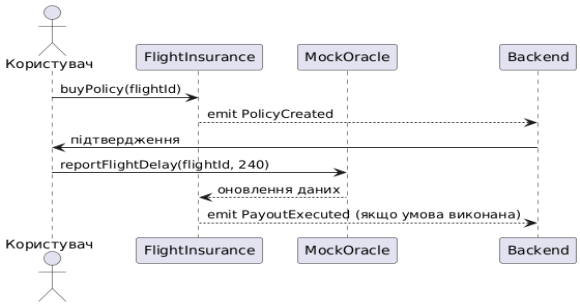

Рисунок 2.2 – Діаграма послідовності процесу купівлі полісу та автоматичної виплати

Ця схема демонструє ключову перевагу параметричного підходу: після повідомлення оракула контракт самостійно виконує перевірку та переказ коштів без додаткових дій зі сторони страхувальника.

Контракт FlightInsurance є основним компонентом системи для параметричного страхування затримок авіарейсів. Він реалізує фіксовану логіку: користувач сплачує премію 0.1 MATIC через функцію buyPolicy, контракт зберігає дані полісу, а при затримці від 180 хвилин функція checkAndPayout виконує автоматичну виплату 0.5 MATIC. Контракт активно взаємодіє з MockOracle для отримання даних про затримки, веде статистику через getStats та emit-ить події PolicyCreated і PayoutExecuted для індексації backend-системою. Архітектура контракту, що показана на рисунку 2.3, побудована навколо структури Policy та механізму страхового пулу, що забезпечує прозорість і детермінованість операцій.

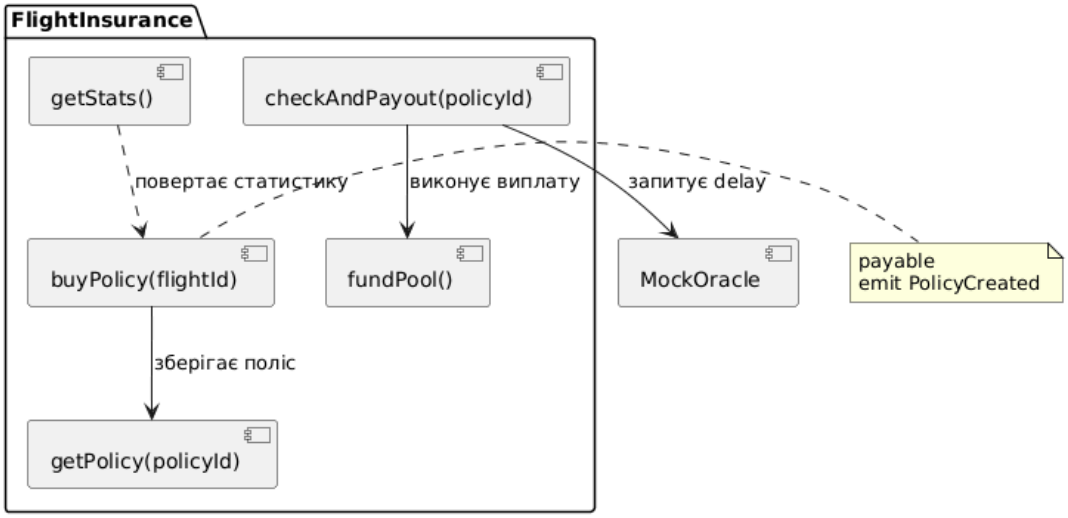

Рисунок 2.3 – Архітектура смарт-контракту FlightInsurance

Контракт WeatherInsurance реалізує гнучкий механізм страхування від посухи з параметризацією умов. Через функцію buyPolicy страхувальник вказує regionId, поріг опадів та тривалість сезону, сплачуючи премію, що робить контракт адаптивним до різних регіонів. Основна логіка checkAndPayout перевіряє дані оракула та виконує виплату за принципом 5x премії при недостатній кількості опадів. Контракт підтримує функції getStats та isPolicyActive, забезпечуючи контроль за балансом пулу та терміном дії полісів. Така архітектура, що наведена на рисунку 2.4, дозволяє використовувати контракт для аграрного страхування, підтримуючи події PolicyCreated та PayoutExecuted для моніторингу.

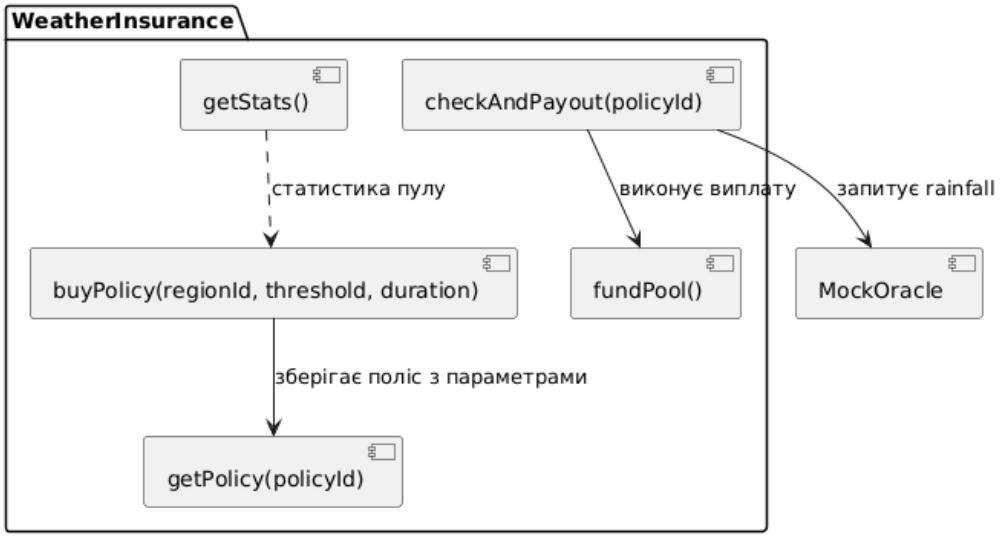

Рисунок 2.4 – Архітектура смарт-контракту WeatherInsurance

Контракт MockOracle виконує роль мосту між реальними даними та страховими контрактами в демонстраційному режимі. Він надає функції reportFlightDelay та reportRainfall для симуляції зовнішніх подій, а також view-функції getFlightDelay та getRainfall для читання даних. Усі функції оновлення даних захищені модифікатором onlyOwner, що забезпечує контроль у тестовому середовищі. Контракт emit-ить події FlightDelayReported та RainfallReported, які індексуються backend-системою для реального часу оновлень. Архітектура MockOracleЮ представлена на рисунку 2.5, спроєктована для легкої заміни на децентралізовані оракули в продакшені, зберігаючи єдиний інтерфейс взаємодії зі страховими контрактами.

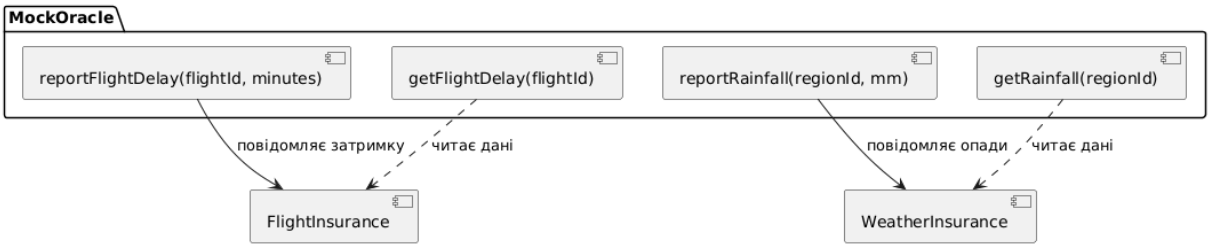

Рисунок 2.5 – Архітектура смарт-контракту MockOracle

У процесі проєктування значну увагу було приділено безпеці та оптимізації. Контракти використовують модифікатори onlyOracle для захисту функцій оновлення даних, перевіряють достатність коштів у пулі перед виплатами та реалізують функцію fundPool для провайдерів ліквідності. Баланс пулу постійно відстежується через getStats, що дозволяє моніторити утилізацію коштів у реальному часі.

Таблиця 2.1 відображає основні функції та їх призначення в контексті архітектури.

Таблиця 2.1 – Основні функції смарт-контрактів системи CLAIMLESS

Контракт	**Функція**	**Тип доступу**	**Призначення**
FlightInsurance	buyPolicy	payable	Купівля полісу
FlightInsurance	checkAndPayout	public	Перевірка та виплата
WeatherInsurance	buyPolicy	payable	Купівля з параметрами
MockOracle	reportFlightDelay	onlyOwner	Симуляція даних рейсу
MockOracle	getFlightDelay	view	Отримання даних

Така таблиця наочно демонструє чітке розмежування відповідальності між компонентами.

Загальна архітектура смарт-контрактів тісно інтегрована з backend-рівнем через бібліотеку web3.py. Файл blockchain.py містить обгортки для виклику функцій, обробки транзакцій та отримання статистики. Це дозволяє фронтенду React взаємодіяти з контрактами як через API, так і напряму через ethers.js у дослідницькому режимі Explorer.

Проєктування враховувало майбутню масштабованість. Контракти спроєктовані таким чином, що додавання нового типу страхування (наприклад, для врожаю чи криптоактивів) вимагає лише створення нового контракту, який успадковує базову логіку пулу та інтегрується з оракулом. Такий підхід забезпечує еволюційність системи без порушення роботи вже розгорнутих компонентів.

У цілому архітектура смарт-контрактів CLAIMLESS поєднує класичні принципи DeFi з практичними потребами параметричного страхування. Вона забезпечує високу прозорість, автоматизацію виплат, економію операційних витрат та можливість моніторингу в реальному часі. Розроблені рішення повністю відповідають сучасним стандартам Solidity, принципам безпеки смарт-контрактів та вимогам демонстраційного проєкту, що працює як на локальному Hardhat, так і на тестовій мережі Polygon Amoy. Подальше вдосконалення може включати інтеграцію з децентралізованими оракулами та механізмами перестрахування для підвищення капіталізації пулів.

2.2. Проєктування архітектури програмної системи страхових смарт-контрактів

Під час розробки програмного комплексу CLAIMLESS проєктування архітектури визначено як засадничий етап для поєднання децентралізованих інструментів блокчейну із сучасними мережевими технологіями. Це дозволило впровадити параметричне страхування без залучення традиційних бюрократичних механізмів. Обрана багатошарова структура передбачає виконання кожним рівнем

чітко окреслених функцій задля мінімізації залежностей та посилення безпеки загального рішення.

Застосований підхід забезпечив інтеграцію зовнішнього інтерфейсу на базі React для зручної взаємодії з користувачем, серверної частини на Flask для керування бізнес-логікою та роботи з блокчейном через бібліотеку web3.py. Смарт-контракти Solidity гарантують незмінність правил здійснення виплат, а моніторинг у реальному часі за допомогою Prometheus та Grafana забезпечує контроль стану системи.

Завдяки цьому архітектура демонструє автоматизацію страхування від затримок рейсів чи погодних аномалій і створює прозоре середовище з фіксацією всіх подій у блокчейні та миттєвим інформуванням через технологію передачі повідомлень від сервера [34].

Архітектурна схема ілюструє чітку ієрархію взаємозв’язків від дій користувача до мережі Polygon Amoy або локального середовища Hardhat із паралельним передаванням даних до модулів спостереження. На рисунку 2.6 представлено основні рівні, що охоплюють користувацький інтерфейс, серверну частину, блокчейн та моніторинг для забезпечення повної інтеграції процесів.

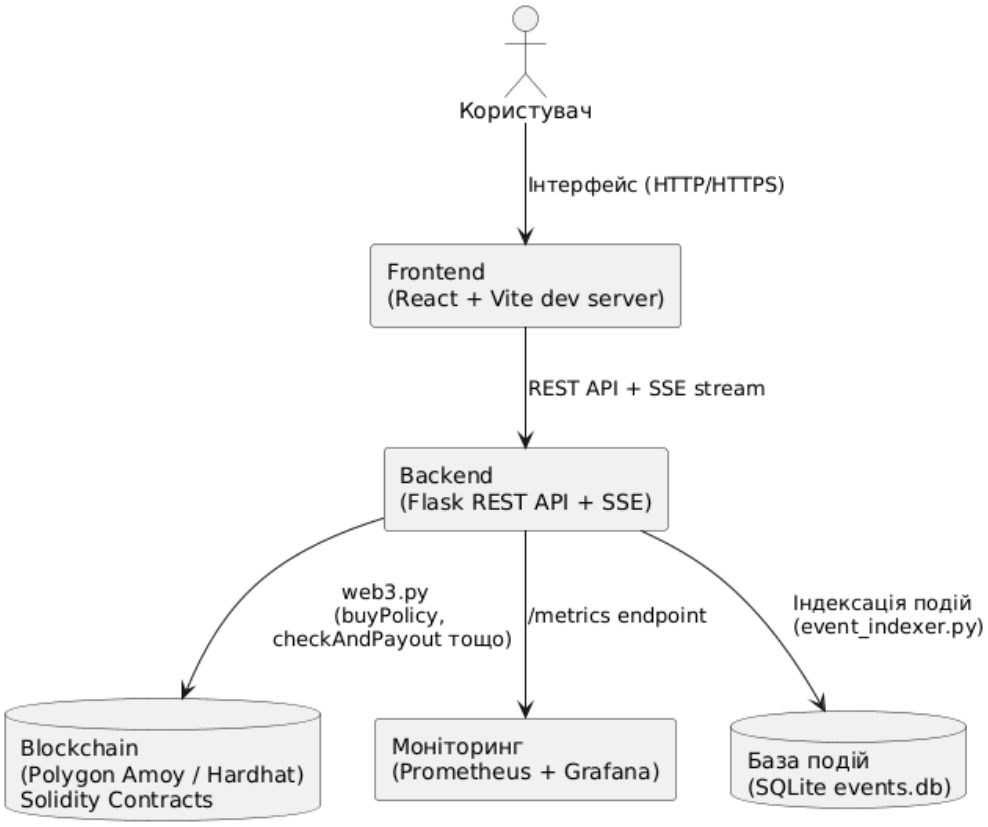

Рисунок 2.6 – Архітектурна схема програмної системи страхових смарт-контрактів

Така побудова гарантує детерміноване виконання операцій із придбання полісів та автоматичних виплат через смарт-контракти, тоді як сервер відповідає за координацію та індексацію подій. Структурна схема системи відображає модульну організацію вихідного коду, де серверна частина складається з пов’язаних Python-модулів, інтерфейс побудовано на компонентах React, а контракти представлені незалежними Solidity-файлами. Це дозволяє розширювати функціональні можливості без втрати стабільності.

Рисунок 2.7 демонструє відповідну структуру, що корелює з файлами app.py та blockchain.py.

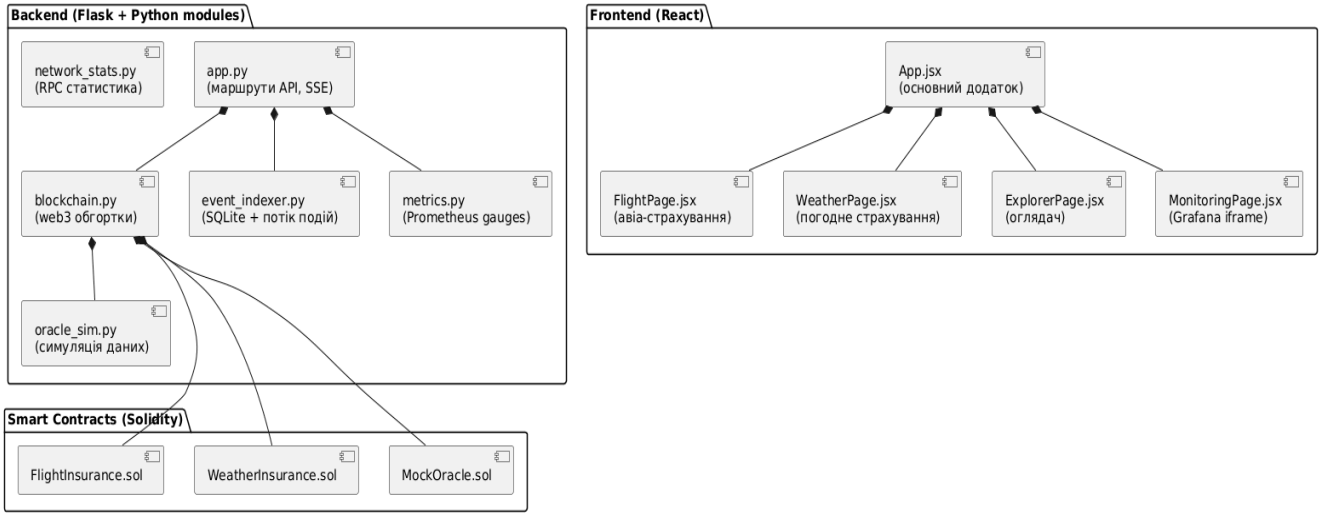

Рисунок 2.7 – Структурна схема програмної системи страхових смарт-контрактів

Реалізована модульність дозволяє серверу в app.py обробляти запити до програмних інтерфейсів для купівлі чи виплати страхування через функції в blockchain.py, тоді як модуль event_indexer.py забезпечує фіксацію подій у базі даних SQLite для швидкого доступу.

Діаграма варіантів використання наголошує на взаємодії користувача, джерела зовнішніх даних та адміністратора із системою під час виконання ключових сценаріїв. На рисунку 2.8 зображено основні випадки використання стосовно придбання полісів та перегляду статистики, що відповідає наявним точкам доступу та інтерфейсним елементам. Ці сценарії впроваджено в коді через відповідні компоненти, зокрема BuyPolicyForm.jsx для авіаційних та погодних ризиків.

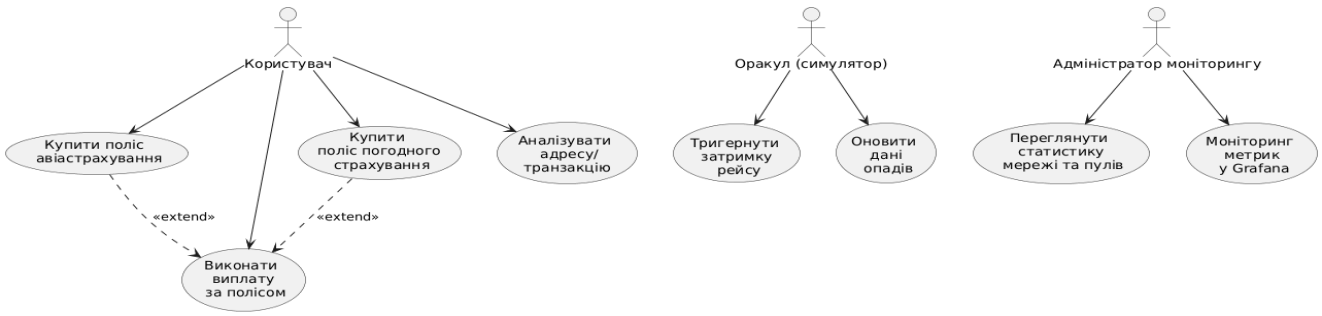

Рисунок 2.8 – Діаграма варіантів використання програмної системи страхових смарт-контрактів

Детальна внутрішня взаємодія модулів представлена на діаграмі компонентів. Рисунок 2.9 візуалізує зв’язки між серверною частиною та блокчейном через засоби web3.py.

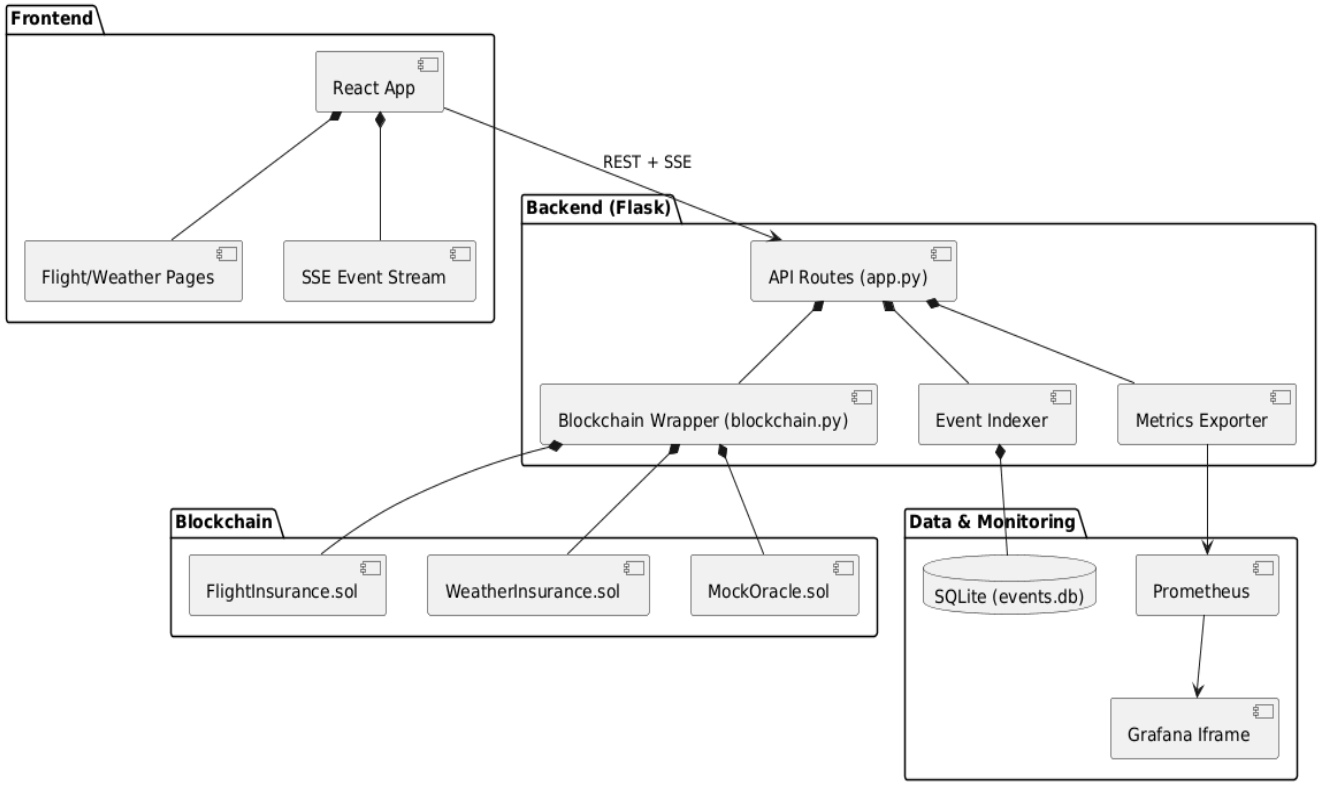

Рисунок 2.9 – Діаграма компонентів програмної системи страхових смарт-контрактів

Визначені складники архітектури відповідають структурі коду, де модуль у blockchain.py опрацьовує транзакції, а окремий потік індексації оновлює інформацію про події.

Для розуміння внутрішньої логіки варто розглянути діаграму класів, яка об’єднує всі ключові модулі в цілісну схему. Рисунок 2.10 представляє уніфіковану модель, де програмні класи Python, контракти Solidity та елементи React пов’язані через успадкування та залежності. Схема підкреслює підпорядкованість серверного додатка обгортці блокчейну та роль індексатора у зв’язку з інтерфейсом через чергу подій.

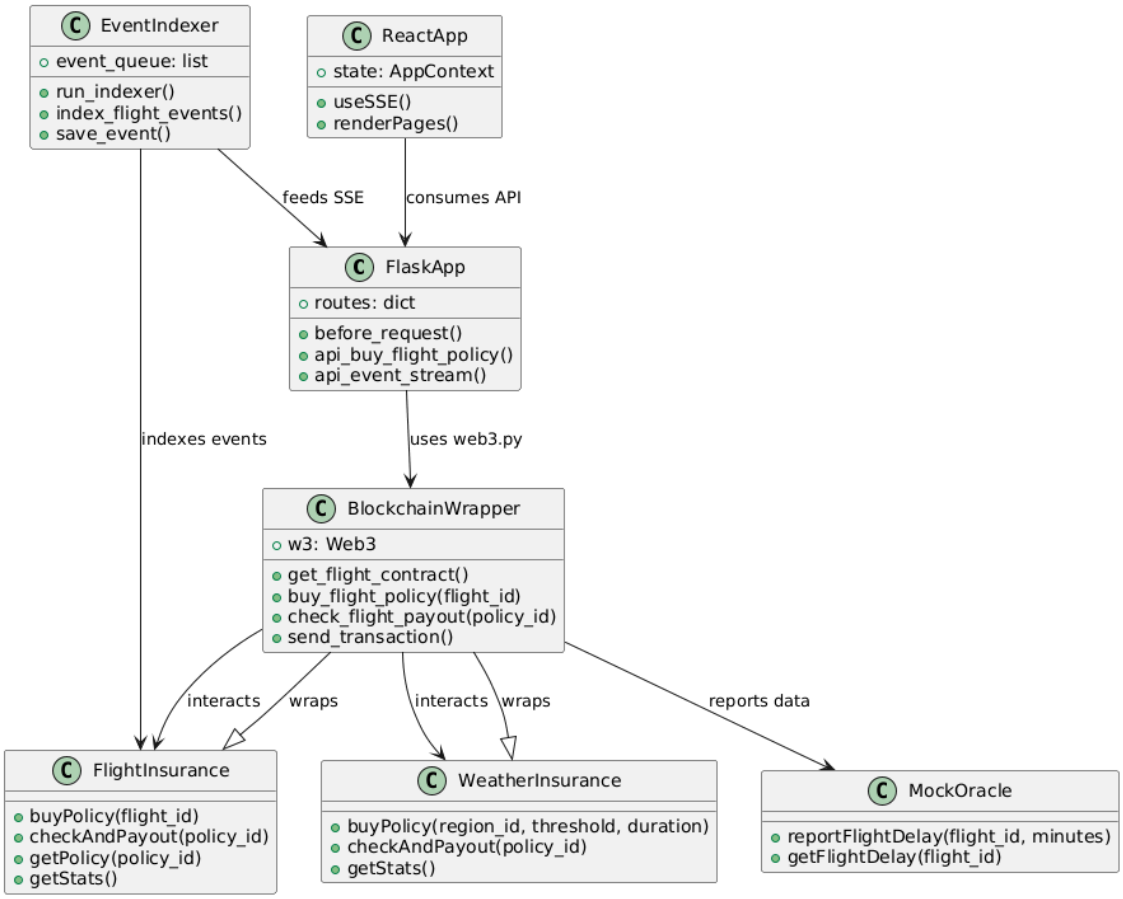

Рисунок 2.10 – Діаграма класів програмної системи страхових смарт-контрактів

Динаміку виконання сценарію купівлі поліса та отримання відшкодування ілюструє діаграма послідовності. На рисунку 2.11 показано етапи дій для авіаційного страхування від запиту до виплати.

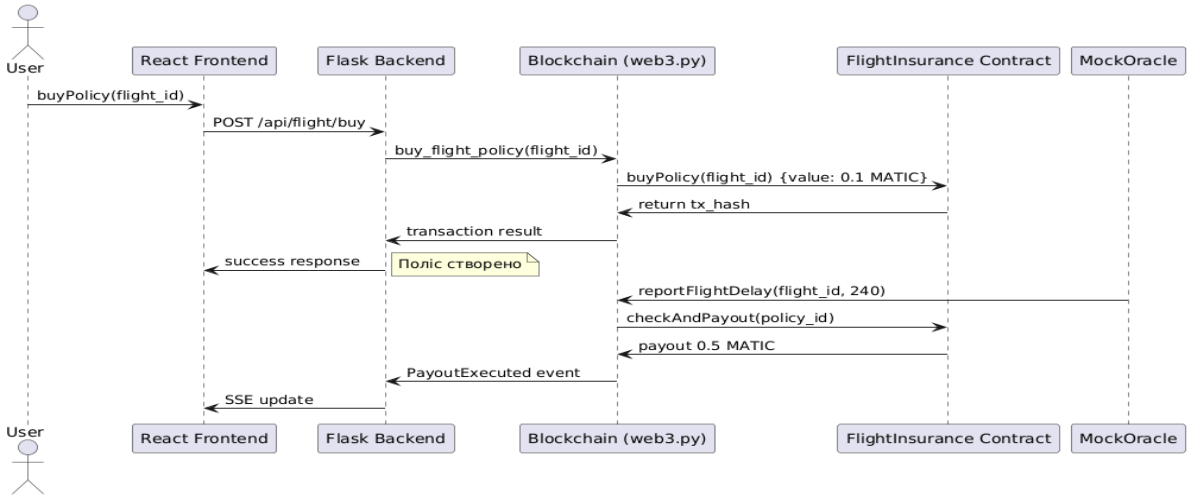

Рисунок 2.11 – Діаграма послідовності програмної системи страхових смарт-контрактів

Зазначена послідовність відтворює алгоритми функцій у файлі blockchain.py з автоматичним реагуванням на дані від зовнішнього джерела.

Життєвий цикл страхового поліса та переходи між його станами описує діаграма станів. Рисунок 2.12 відображає етапи від моменту створення угоди до завершення терміну її дії або виплати коштів. Описана модель відповідає програмній логіці перевірки статусів у контрактах.

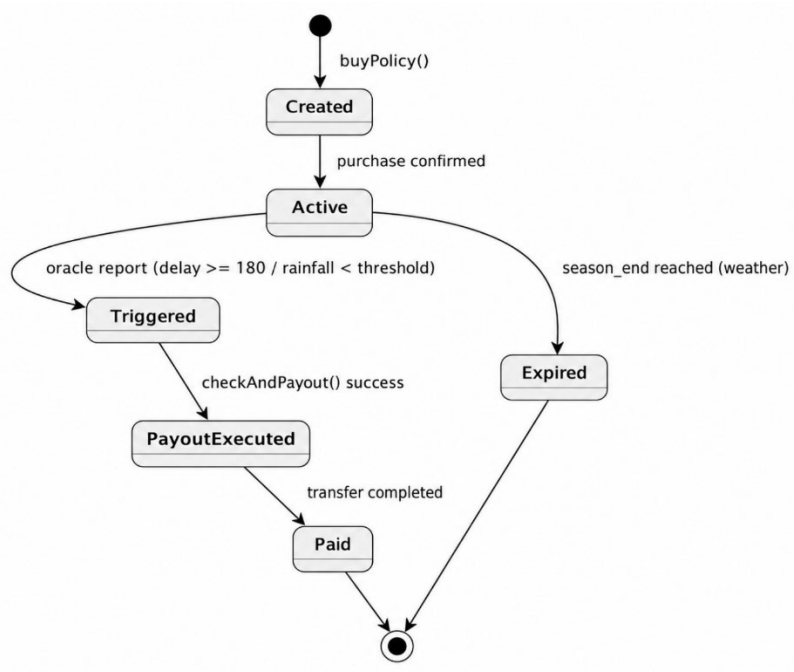

Рисунок 2.12 – Діаграма станів програмної системи страхових смарт-контрактів

Алгоритм купівлі страхового полісу є одним із ключових процесів системи CLAIMLESS, який забезпечує прозору та миттєву реєстрацію умов страхування безпосередньо в блокчейні. Цей процес починається з ініціації запиту користувачем через React-інтерфейс, після чого backend передає дані до смарт-контракту, перевіряє достатність коштів та фіксує поліс у незмінному вигляді. Така послідовність операцій гарантує, що премія негайно потрапляє до страхового пулу, а всі параметри полісу стають доступними для подальшої автоматичної обробки. Блок-схема цього алгоритму, що наведена на рисунку 2.13, наочно демонструє етапи взаємодії компонентів і умови успішного завершення операції.

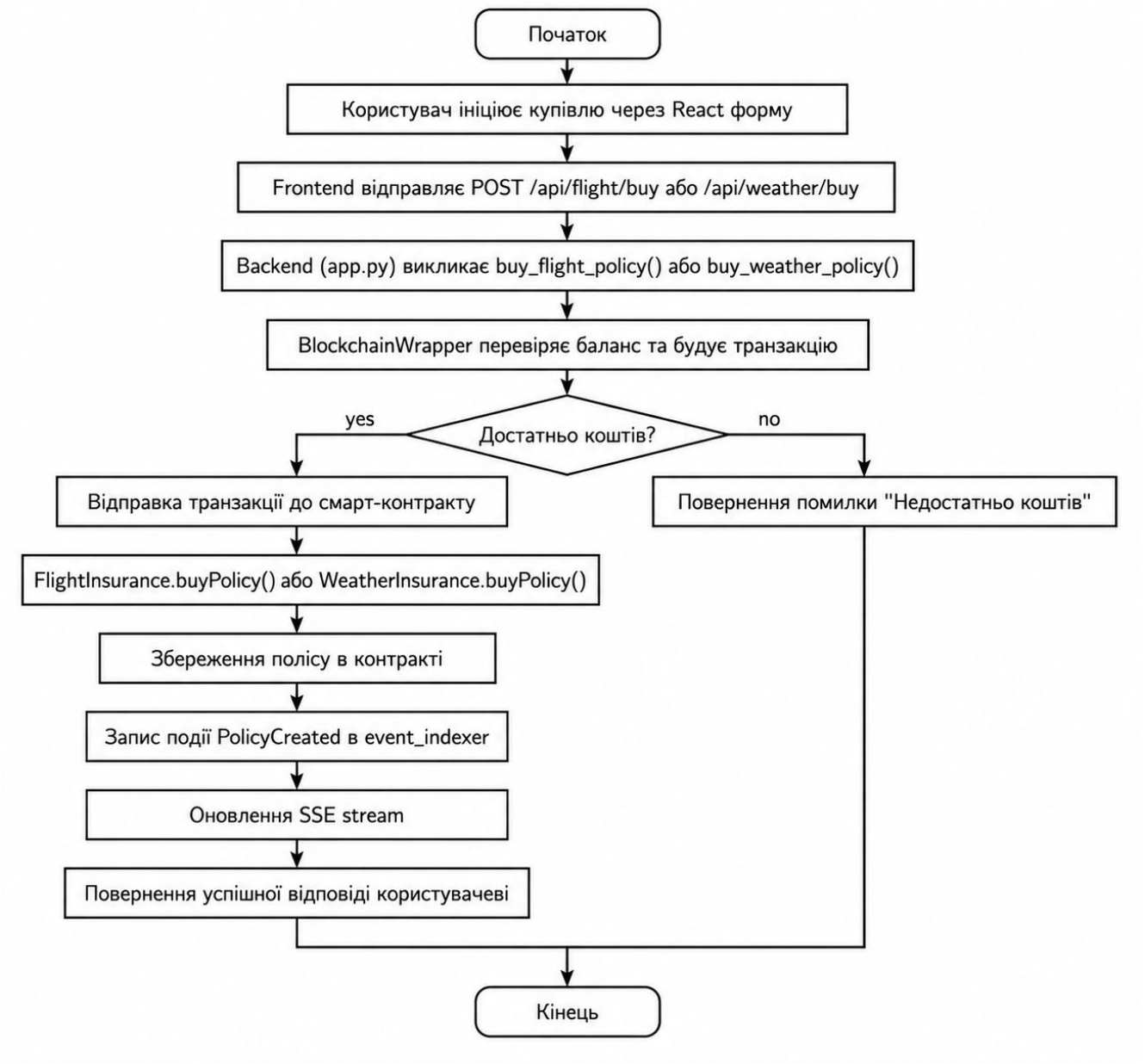

Рисунок 2.13 – Блок-схема алгоритму купівлі страхового полісу

Алгоритм виконання страхової виплати реалізує основну перевагу параметричного страхування – автоматичну виплату без подання заяви. Після отримання даних від оракула система перевіряє виконання умов полісу, і в разі їх дотримання смарт-контракт ініціює переказ коштів зі страхового пулу. Цей процес повністю детермінований і прозорий, оскільки всі перевірки відбуваються на рівні блокчейну без участі посередників. Блок-схема, представлена на рисунку 2.14, демонструє логіку прийняття рішення та взаємодію між оракулом, контрактом і backend для забезпечення миттєвості виплати.

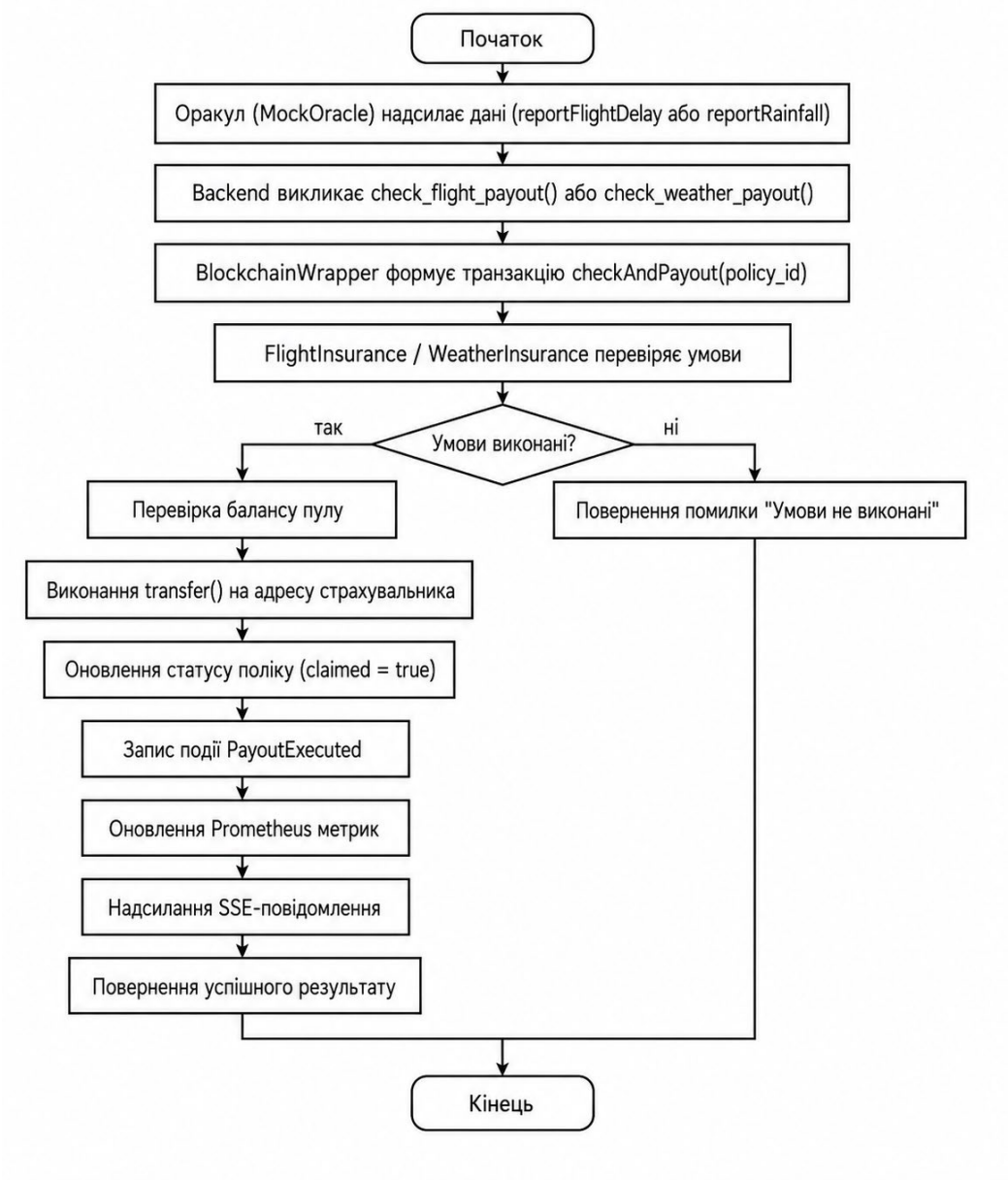

Рисунок 2.14 – Блок-схема алгоритму виконання страхової виплати

Алгоритм оновлення даних через оракул, блок-схема якого наведена на рисунку 2.15, забезпечує зв’язок реального світу з блокчейном і є критичним для спрацювання страхових подій. У демо-режимі цей процес симулюється вручну, але в продакшені може бути замінений на децентралізовані оракули. Backend отримує дані, передає їх до MockOracle, після чого система автоматично перевіряє всі активні поліси на відповідність умовам. Такий підхід гарантує оперативність реакції системи та мінімізує ризики маніпуляцій завдяки фіксації всіх дій у блокчейні.

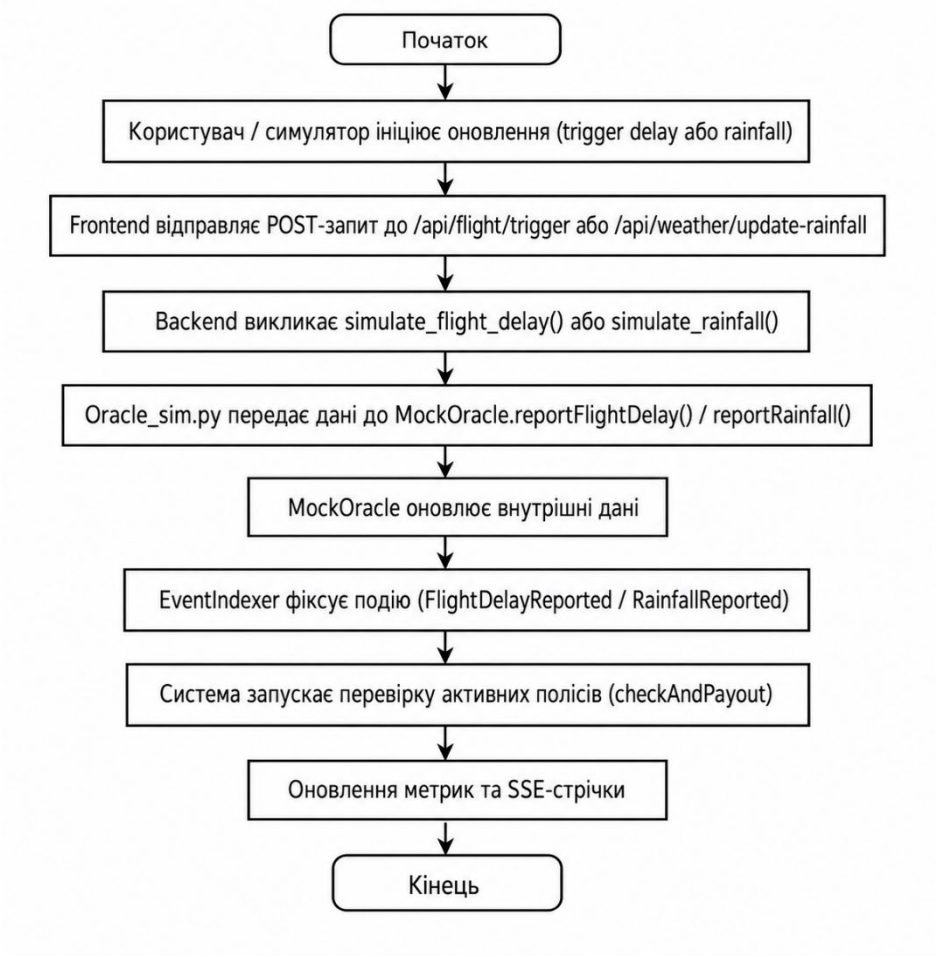

Рисунок 2.15 – Блок-схема алгоритму оновлення даних оракулом

Для наочного порівняння ключових елементів архітектури сформовано таблицю 2.2, де підсумовано їхні зони відповідальності та програмну реалізацію.

Таблиця 2.2 – Основні компоненти архітектури програмної системи CLAIMLESS

Компонент	**Відповідальність**	**Реалізація в коді**
Frontend	Інтерфейс, SSE, графіки	React компоненти (FlightPage.jsx тощо)
Backend API	Маршрути, SSE, метрики	app.py з CORS та before/after_request
Blockchain Wrapper	web3 взаємодія, транзакції	blockchain.py з send_transaction
Event Indexer	Індексація подій, SQLite	event_indexer.py + start_indexer_thread
Monitoring	Prometheus gauges, Grafana дашборди	metrics.py + Grafana iframe

Наведена таблиця підкреслює внесок кожного складника у загальну ефективність. Проєктування архітектури забезпечило створення надійного рішення для автоматизованого страхування з можливістю масштабування [35].

Такий підхід відповідає сучасним напрямам розвитку фінансових технологій та демонструє спроможність повної автоматизації процесів на базі блокчейну для зниження витрат та підвищення довіри [36].

Гнучкість реалізації дозволяє легко замінити імітацію зовнішніх даних на реальні сервіси Chainlink, а компоненти моніторингу забезпечують високий рівень спостереження за системою [37]. У результаті комплекс CLAIMLESS постає як повноцінний прототип, готовий до подальшого розвитку в межах цифрової трансформації ринку [38].

2.3. Вибір та обґрунтування блокчейн-платформи та мови програмування смарт-контракті

Під час розроблення інноваційного блокчейн-рішення для параметричного страхування під назвою CLAIMLESS, що забезпечує автоматичне здійснення виплат за об’єктивними показниками без подання традиційних заяв, одним із найбільш відповідальних етапів стало науково підкріплене обрання розподіленої платформи та мови програмування смарт-контрактів. Цей вибір зумовлювався потребою реалізувати повну автоматизацію страхових операцій, високий рівень прозорості дій, мінімізацію витрат та надійне поєднання із зовнішніми даними через засоби зв’язку з реальним світом, що постає засадничою вимогою для впровадження концепції параметричного страхування.

У результаті як основне середовище визначено тестову мережу Polygon Amoy з ідентифікатором chainId 80002 та мову Solidity версії 0.8.20 для написання програмного коду контрактів [29]. Такий підхід дозволив поєднати технічну потужність, безпеку та практичну втілюваність усіх складників, від страхових контрактів FlightInsurance та WeatherInsurance до взаємодії серверної частини через web3.py і клієнтського інтерфейсу, забезпечуючи відповідність функціоналу, закладеному у вихідному коді проєкту.

Тестова мережа Polygon Amoy постала оптимальним простором для розгортання системи завдяки архітектурним перевагам, які безпосередньо відображені в налаштуваннях проєкту. Будучи сумісною з віртуальною машиною Ethereum мережею другого рівня, вона гарантує швидке опрацювання транзакцій та суттєво нижчу вартість ресурсів порівняно з основною мережею, що критично важливо для страхових програм, де кожне придбання полісу, оновлення даних або виплата мають здійснюватися економічно доцільно та без зволікань.

У програмному коді це втілено шляхом підключення до вузла доступу ALCHEMY_AMOY_URL у модулі blockchain.py, де реалізовано функції перевірки з'єднання, отримання ідентифікатора мережі, номера блоку та ціни палива, а також у конфігурації hardhat.config.js, яка визначає параметри мережі для стабільної роботи системи. Таке рішення дозволяє ефективно випробувати логіку страхових фондів і автоматичних розрахунків, а також відтворити реальні умови функціонування з можливістю подальшого перенесення контрактів у головну мережу без змін коду. Крім того, підтримується швидкий цикл індексації у event_indexer.py, де події створення полісів, виконання виплат та звіти про затримки рейсів чи опади фіксуються миттєво, що робить систему динамічною для користувача.

Для порівняння переваг Polygon Amoy з іншими поширеними блокчейн-платформами, які аналізувалися під час дослідження, доцільно звернутися до системного огляду ключових показників, наведених у таблиці 2.3.

Таблиця 2.3 – Порівняльна характеристика блокчейн-платформ за критеріями придатності для параметричного страхування

Платформа	**EVM-сумісність**	**Середня вартість газу (gwei)**	**Приблизна швидкість TPS**	**Підтримка тестнету та оракулів**	**Зрілість екосистеми для страхових смарт-контрактів**
Polygon Amoy	Повна	20–50	50–7000	Відмінна	Висока
Ethereum Mainnet	Повна	20–100+	15–30	Відмінна	Найвища
Binance Smart Chain	Повна	5–10	100–300	Добра	Середня
Solana	Ні	Дуже низька	1000+	Обмежена	Низька (інша мова)

Як свідчать дані таблиці, Polygon Amoy забезпечує найкращу рівновагу між ціною, продуктивністю та сумісністю, що дозволило здійснити в проєкті повноцінну демонстрацію страхових процесів без поступок у питаннях безпеки. Мову програмування Solidity версії 0.8.20 обрано з огляду на її досконалість, значну спільноту фахівців та інтегровані методи захисту, які мають особливу вартість для фінансових контрактів, що розпоряджаються коштами страхових пулів [30].

У зазначеній версії автоматично діють механізми перевірки арифметичних операцій, що суттєво зменшує ризики виникнення помилок, притаманних раннім етапам розвитку мови. У програмних лістингах це знайшло відображення в контрактах FlightInsurance.sol та WeatherInsurance.sol, де втілено функції купівлі полісу, перевірки умов і виплати, отримання статистики, а також у модулі MockOracle.sol для відтворення даних оракула. Компілювання та розгортання проводяться через середовище Hardhat, яке автоматично готує інтерфейси для серверної частини, гарантуючи взаємодію з модулем blockchain.py.

Такий вибір мови дозволив чітко визначити логіку полісів, від збереження даних про власника та обсяг виплати до динамічних показників порогу опадів та тривалості страхування, із гарантуванням незмінності алгоритмів після їх активації, що є фундаментальною перевагою блокчейн-технологій.

Рисунок 2.16 ілюструє загальну побудову технологічного набору рішення та підкреслює спосіб інтеграції обраної платформи і мови в єдину структуру.

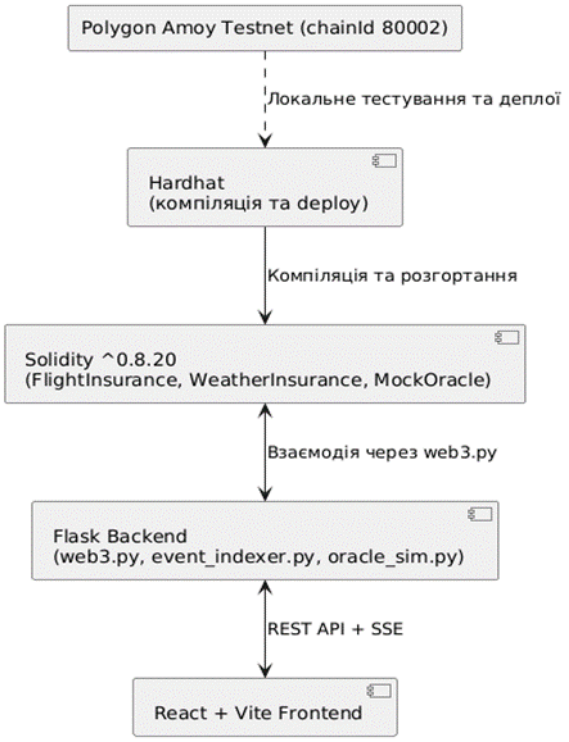

Рисунок 2.16 – Архітектура технологічного стеку блокчейн-рішення CLAIMLESS

На рисунку 2.16 простежується, як Polygon Amoy виступає фундаментом для функціонування смарт-контрактів, а Solidity забезпечує надійне втілення ділової логіки, що поєднується з серверною та клієнтською частинами для надання цілісного досвіду.

Такий набір засобів не лише відповідає вимогам параметричного страхування, а й демонструє практичне впровадження концепції цифрових страхових технологій, де постачальник даних передає відомості безпосередньо в контракт, а серверна частина через модулі індексації та збору метрик забезпечує прозорість усіх подій [31].

Загалом обґрунтоване поєднання Polygon Amoy та Solidity 0.8.20 дозволило створити масштабоване та безпечне рішення, що повністю узгоджується з усіма компонентами проєкту, від автоматичного розгортання і наповнення фондів до відстеження показників у реальному часі [32].

Цей підхід є технічно виправданим та відкриває можливості для розвитку системи на реальному ринку, роблячи параметричне страхування доступним і зрозумілим для багатьох користувачів [33].

2.4. Розроблення структури даних та логіки обробки страхових подій у смарт-контракті

Процес проектування структури даних та алгоритмів опрацювання страхових випадків у межах інтелектуального контракту постає фундаментальним етапом побудови децентралізованого рішення CLAIMLESS, що базується на засадах параметричного страхування. У програмних кодах FlightInsurance та WeatherInsurance, які розгорнуто в мережі Polygon Amoy, відомості про кожен договір впорядковано у компактних масивах для забезпечення мінімальних операційних витрат, повної прозорості та автоматизованого реагування на події без посередництва.

Зазначений підхід трансформує класичну страхову модель, де здійснення виплати визначається суб'єктивним оцінюванням завданих збитків, у чітку систему, керовану виключно об'єктивними показниками, як-от тривалість затримки авіарейсу чи обсяг опадів у конкретній місцевості [39].

Фундаментом організації даних у контракті FlightInsurance виступає структура Policy, що фіксує ключові реквізити страхової угоди (таблиця 2.4). Кожен документ ідентифікується через унікальний номер у відповідному реєстрі mapping і містить параметри, необхідні для верифікації умов відшкодування. Схожа логіка застосована в WeatherInsurance, проте з додаванням полів, що відображають специфічні погодні ризики. Ці побудови спроектовані з пріоритетом на безпеку та економію обчислювальних ресурсів, оскільки всі змінні залишаються незмінними після створення, а маркер виконання виплати унеможливлює повторне отримання коштів за однією угодою.

Таблиця 2.4 – Структура даних Policy у контракті FlightInsurance

Поле	Тип даних	Опис
holder	address	Адреса гаманця страхувальника
flight_id	string	Ідентифікатор рейсу (наприклад, PS101)

purchase_time	uint256	Час придбання полісу (Unix timestamp)
premium	uint256	Сума премії в wei (фіксована 0.1 MATIC)
payout	uint256	Сума виплати в wei (0.5 MATIC при виконанні умови)
claimed	bool	Статус виконання виплати

Обрана архітектура гарантує оперативний доступ до відомостей через функцію отримання даних полісу, яка видає всі значення одним запитом, що є критично важливим для взаємодії зовнішніх інтерфейсів із системою.

Дизайн WeatherInsurance дозволяє споживачам самостійно визначати окремі параметри під час закупівлі, підсилюючи гнучкість продукту. Опис структури даних Policy у межах контракту WeatherInsurance представлено в таблиці 2.5.

Таблиця 2.5 – Структура даних Policy у контракті WeatherInsurance

Поле	**Тип даних**	**Опис**
holder	address	Адреса гаманця страхувальника
region_id	string	Ідентифікатор регіону (наприклад, UA-KYIV)
purchase_time	uint256	Час придбання полісу
season_end	uint256	Кінець страхового сезону
premium	uint256	Сума премії (налаштовується)
payout	uint256	Сума виплати (множник від премії)
rainfall_threshold	uint256	Поріг опадів у мм для виплати
claimed	bool	Статус виконання виплати

Для кращої візуалізації ієрархія даних в інтелектуальних контрактах зображена на схемі нижче (Рисунок 2.17). Дана ілюстрація демонструє основні складові структур Policy в обох сегментах, їхню взаємодію з реєстрами та головними функціями доступу.

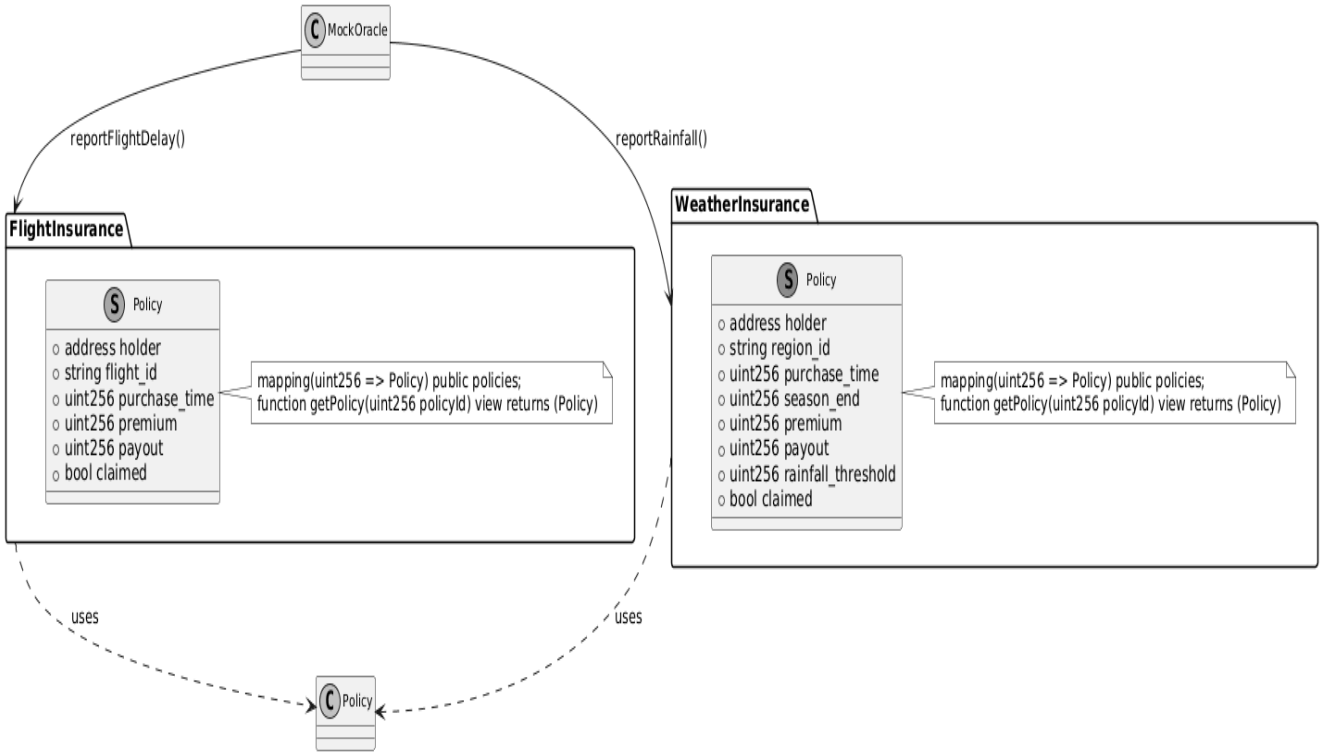

Рисунок 2.17 – Схема структури даних Policy у смарт-контрактах FlightInsurance та WeatherInsurance

Окрім базових масивів, програмний код підтримує формування глобальної статистики через інструменти моніторингу, що відображають загальну кількість угод, обсяги внесків, виплати та залишок страхового фонду. Це дозволяє в режимі реального часу відстежувати фінансову стійкість пулу та уникати дефіциту ліквідності. Фонд поповнюється через спеціалізовані функції платежів, що робить платформу відкритою для зовнішніх постачальників капіталу [40].

Алгоритм опрацювання страхових випадків активується з моменту придбання захисту. Процедура купівлі у FlightInsurance перевіряє перерахування визначеної суми, формує новий запис у реєстрі, оновлює лічильник та генерує програмну подію. Це стає сигналом для внутрішнього індексатора, який зберігає інформацію в базі даних, забезпечуючи миттєве відображення змін у користувацькому інтерфейсі. Схожим чином функціонує механізм у WeatherInsurance, де індивідуальні параметри передаються під час виклику, дозволяючи персоналізувати умови угоди.

Зовнішня інформація надходить через допоміжний вузол, що імітує роботу джерела даних і зберігає показники про затримки рейсів та рівень опадів. Відповідні функції оновлення даних доступні лише адміністратору, що забезпечує імітацію функціонування децентралізованого джерела відомостей у тестовому режимі.

Ядром логіки виступає процедура перевірки та здійснення виплати. У сегменті авіастрахування вона аналізує параметри контракту, запитує поточний стан затримки, перевіряє дотримання часового ліміту та відсутність попередніх виплат. У разі підтвердження умов відбувається автоматичне перерахування коштів на гаманець клієнта, зміна статусу угоди та емісія звітної події.

Аналогічний алгоритм діє в погодному страхуванні, де порівняння фактичних опадів із встановленою межею ініціює миттєвий платіг. Такий підхід повністю нівелює потребу у поданні заяв та скорочує час очікування коштів до лічених секунд [41].

Моніторинг процесів реалізується через спеціальні програмні сигнали, які впорядковуються окремим потоком обробки. Функції індексування сканують системні журнали, фіксуючи кожну операцію в базі даних із зазначенням мітки часу та номеру блоку. Це дозволяє інтерфейсу отримувати оперативні оновлення та збирати аналітичні показники ефективності системи.

Обрана логіка гарантує неподільність операцій, оскільки виплата та зміна статусу відбуваються в межах однієї транзакції, що виключає фінансові ризики. Додаткові інструменти дозволяють отримувати вичерпну інформацію про будь-який договір за його номером, що робить систему цілком прозорою. Будь-який учасник або аудитор може верифікувати стан угоди безпосередньо в розподіленому реєстрі. Демонстраційні модулі ілюструють інтеграцію через показ актуальних даних для тестових об’єктів та територій.

Спроектована логіка управління подіями в архітектурі CLAIMLESS не лише зменшує операційні видатки, а й зміцнює довіру сторін. Кожна дія назавжди карбується в блокчейні, а автоматизація на основі об’єктивних даних усуває людський фактор і конфлікти інтересів. У поєднанні з метриками та

статистичними модулями система гарантує комплексний контроль, що є критичним для страхових фондів, де баланс ресурсів прямо впливає на можливість укладання нових контрактів [42].

Такий вектор розвитку відповідає глобальним трендам децентралізованих фінансів, де параметричні моделі виявляють вищу ефективність порівняно з паперовими схемами. У даному проекті структура та алгоритми розроблені з урахуванням ринкових вимог щодо економії газу, повної автоматизації та можливості аудиту. Це закладає фундамент для створення масштабованого рішення, придатного для майбутньої інтеграції з професійними мережами оракулів типу Chainlink [43].

Таким чином, реалізовані архітектурні рішення забезпечують повний цикл функціонування параметричного страхування у швидкому та безпечному цифровому середовищі.

2.5. Вибір та обґрунтування механізмів верифікації страхових випадків та оракулів даних

У межах розробленого блокчейн-рішення CLAIMLESS особливу увагу приділено інструментам підтвердження страхових подій та інтеграції постачальників даних, оскільки саме вони визначають результативність параметричного страхування. У такій моделі виплати здійснюються автоматично на основі об’єктивних показників без традиційного подання заяв чи суб’єктивних оцінок [44].

Параметрична логіка, що закладена у цифрові алгоритми FlightInsurance та WeatherInsurance, передбачає чіткі критерії, а саме затримку авіарейсу понад 180 хвилин або рівень опадів нижче встановленої межі за сезон. Такий підхід дозволяє усунути типові проблеми традиційної галузі, які зумовлені тривалими перевірками, паперовим документообігом та ймовірними суперечками, забезпечуючи миттєве виконання зобов’язань у межах одного блоку мережі Polygon Amoy [45].

Вибір засобів верифікації обґрунтовано потребою поєднати простоту демонстраційного режиму з повною прозорістю та автоматизацією, що

реалізується через спеціально створений імітатор даних MockOracle, який взаємодіє з серверною та клієнтською частинами системи [46].

Імітатор MockOracle, розгорнутий разом з основними угодами, виконує роль відтворювача зовнішніх відомостей, дозволяючи в тестовому середовищі точно моделювати реальні сценарії без прямої залежності від зовнішніх інтерфейсів програмування. У відповідному модулі функції симуляції затримки рейсу чи обсягу опадів приймають ідентифікатори об’єктів разом з числовими параметрами та через програмний шлюз звертаються до методів звітування у контракті оракула. Ці методи, доступ до яких обмежено розробником у демо-режимі, фіксують інформацію в стані розподіленого реєстру, а спеціальні запити надають доступ до цих даних для подальшого аналізу [47].

Такий дизайн забезпечує незмінність процесу, оскільки дані зберігаються безпосередньо в мережі, не можуть бути спотворені після запису та залишаються доступними для будь-якого виклику з боку страхових угод. У промисловій версії цей компонент може бути замінений на децентралізований сервіс Chainlink без зміни внутрішньої логіки придбання полісу чи проведення виплати, що підтверджує гнучкість архітектури [48].

Підтвердження страхових випадків реалізується безпосередньо в алгоритмах смарт-контрактів. Функція перевірки та виплати звертається до джерела даних, отримує поточне значення показника для конкретного рейсу і порівнює його з жорстко визначеним порогом. Якщо умову задоволено, цифрова угода самостійно ініціює переказ коштів з пулу на адресу користувача, реєструючи подію виконання виплати.

Аналогічний процес для погодного страхування враховує індивідуальні параметри захисту, що встановлені під час купівлі. Виклик процедури перевірки через програмні точки доступу супроводжується оновленням статистичних метрик та записом події до бази даних через індексатор. Систематизація подій створення полісів, здійснення виплат та звітування за блоками забезпечує повну історичну достовірність і можливість перегляду через потоковий інтерфейс, що робить кожну операцію публічною та доступною для перевірки.

Обґрунтування обраного механізму пов’язане з його відповідністю засадам параметричного страхування та вимогам презентаційного проєкту. У класичній моделі верифікація залежить від людського фактора, що спричиняє затримки до кількох місяців та значні адміністративні видатки. У системі CLAIMLESS весь процес від отримання даних до виплати триває кілька секунд завдяки внутрішньомережевим перевіркам.

Використання MockOracle разом з інструментами оновлення показників дозволяє користувачам самостійно моделювати події, що є цінним для тестування. Водночас механізми блокування подій запобігають конфліктам при одночасних запитах, а функції отримання інформації повертають повні дані про поліс разом з відомостями оракула, гарантуючи зрозумілість інтерфейсу. Такий підхід не лише прискорює розрахунки, але й підвищує рівень довіри, адже програмний код залишається незмінним після публікації, а всі транзакції доступні в оглядачі мережі.

Для наочного розуміння порівняльних переваг обраного механізму доцільно звернутися до аналізу ключових параметрів верифікації, що наведений в таблиці 2.6.

Таблиця 2.6 – Порівняння механізмів верифікації страхових випадків у традиційному страхуванні та в проєкті CLAIMLESS

Параметр верифікації	**Традиційне страхування**	**CLAIMLESS (MockOracle + смарт-контракти)**
1	**2**	**3**
Джерело даних	Ручні документи та експертні оцінки	Дані оракула (reportFlightDelay / reportRainfall)
Час проведення перевірки	Від кількох тижнів до місяців	Секунди (один блок Polygon Amoy)

Продовження таблиці 2.6

1	**2**	**3**
Ступінь автоматизації	Мінімальна, вимагає участі людини	100 % (checkAndPayout виконує умову автоматично)
Прозорість процесу	Закриті внутрішні процедури	Повна (емітовані події, індексація в events.db)
Захист від маніпуляцій	Залежить від чесності учасників	On-chain перевірка + незмінний код контракту

Масштабованість	Обмежена людськими ресурсами	Необмежена, працює 24/7 на блокчейні

Таблиця 2.6 підкреслює фундаментальні переваги обраного підходу, який повністю відповідає програмній реалізації в компонентах мережевої логіки та симуляції.

Рисунок 2.18 наочно демонструє замкнутий цикл від введення даних до автоматичної виплати, підкреслюючи інтеграцію всіх складових частин системи, які втілені у лістингах проєкту.

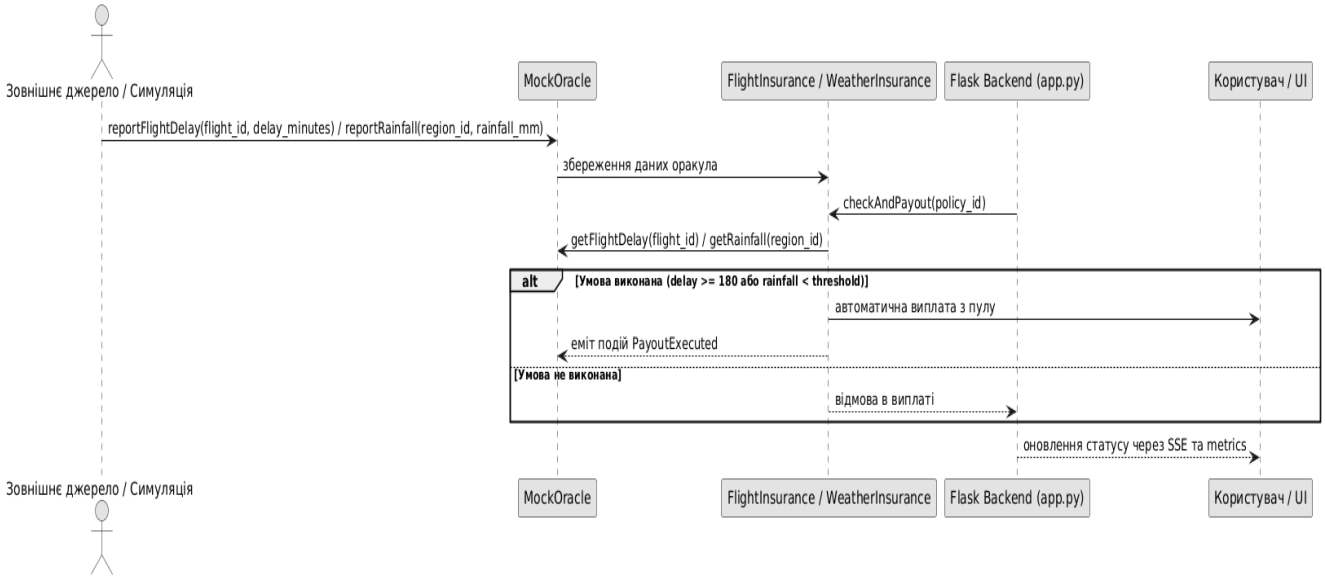

Рисунок 2.18 – Схема механізму верифікації страхового випадку через оракул у проєкті CLAIMLESS

Додатковим доказом ефективності механізму є його поєднання з мережевою статистикою. Модулі фіксації показників реєструють кожну дію через гістограми затримок та опадів, а також лічильники створених полісів та здійснених виплат. Це дозволяє не лише підтверджувати окремі випадки, але й аналізувати загальну спроможність фонду в реальному часі. Через інструменти перегляду мережі будь-яка адреса чи транзакція може бути досліджена, що створює додатковий рівень надійності.

Таким чином, обрані рішення є не компромісом, а оптимальним варіантом для демонстраційного проєкту, який відтворює логіку параметричного захисту та підлягає масштабуванню.

Загалом використання MockOracle у поєднанні з автоматичними перевірками забезпечує високу швидкість та прозорість процесів. Реалізація в програмних модулях доводить практичну дієвість підходу, який долає недоліки традиційної моделі та відкриває можливості для розвитку систем на базі децентралізованих технологій. Такий алгоритм відповідає сучасним вимогам до блокчейн-технологій, роблячи CLAIMLESS зручним інструментом для представлення інновацій у фінансовому секторі.

2.6 Висновок до розділу

Обґрунтований вибір тестової мережі Polygon Amoy та мови Solidity версії 0.8.20 забезпечив оптимальне поєднання продуктивності, низької вартості транзакцій і повної сумісності з екосистемою Ethereum, що стало фундаментальною передумовою для реалізації параметричного страхування в системі CLAIMLESS. Така технологічна основа дозволила досягти високої автоматизації страхових процесів, мінімізувати операційні витрати та гарантувати прозорість усіх операцій.

Проєктування архітектури смарт-контрактів, зокрема FlightInsurance, WeatherInsurance та MockOracle, базувалося на принципах модульності, детермінованості та економії газу. Створені контракти ефективно реалізують механізми купівлі полісів, автоматичної перевірки умов та виконання виплат за об’єктивними параметрами, забезпечуючи повну незалежність від людського втручання. Модульна структура полегшує подальше розширення системи новими видами страхування без порушення існуючої логіки.

Багатошарова архітектура програмного комплексу, що поєднує клієнтський інтерфейс на React, серверну частину на Flask з бібліотекою web3.py та систему індексації подій, забезпечила безшовну інтеграцію блокчейн-компонентів із сучасними веб-технологіями. Застосування діаграм послідовності, станів та

алгоритмів дозволило чітко формалізувати бізнес-логіку, від купівлі полісу до автоматичної виплати, з урахуванням реального часу оновлень.

Розроблена структура даних у контрактах, зокрема гнучкі моделі Policy, у поєднанні з механізмами оракула, створила надійну основу для верифікації страхових подій. Використання MockOracle у демонстраційному режимі з можливістю заміни на децентралізовані рішення типу Chainlink забезпечує високу адаптивність і масштабованість системи. Автоматизована обробка подій значно знижує ризики, притаманні традиційному страхуванню, та підвищує довіру учасників.

Загалом реалізовані технічні рішення демонструють ефективну інтеграцію блокчейн-технологій у сферу параметричного страхування, створюючи передумови для практичного впровадження економічно вигідної, прозорої та швидкої страхової моделі. Отримані результати підтверджують перспективність обраного підходу для подальшого розвитку цифрових фінансових сервісів.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ

3.1. Реалізація базової логіки страхового смарт-контракту та його розгортання в тестовій мережі

Під час створення демонстраційної системи CLAIMLESS ключовим етапом стало втілення базової логіки страхових смарт-контрактів, які становлять фундамент параметричного страхування на блокчейні. Розробку проведено з урахуванням принципів детермінізму, прозорості та автоматизації, що дозволяє повністю усунути потребу в традиційній процедурі подання заяви на виплату.

Смарт-контракти FlightInsurance та WeatherInsurance реалізовано мовою Solidity версії 0.8.20 для забезпечення сумісності з віртуальною машиною Ethereum та мережею Polygon Amoy. Програмний код включає вбудовані механізми захисту від типових вразливостей, зокрема переповнення цілих чисел. Логіка контрактів побудована навколо фіксованих страхових параметрів, які неможливо змінити після розгортання, що гарантує довіру учасників системи до незмінності встановлених правил.

Базова логіка контракту FlightInsurance полягає в автоматизованому управлінні полісами страхування від затримок авіарейсів. При виклику функції придбання поліса контракт приймає ідентифікатор конкретного рейсу та перевіряє надходження фіксованої премії в розмірі 0,1 MATIC через механізм оплати. У разі успішної перевірки створюється запис про поліс, де зберігається адреса страхувальника, дані рейсу, час придбання, розмір премії та потенційна виплата обсягом 0,5 MATIC.

Контракт також дозволяє отримувати повну інформацію про будь-який поліс, включаючи статус здійснення виплати, що дає можливість зовнішнім системам на базі Flask перевіряти стан страховки в реальному часі. Для активації виплати використовується спеціальна функція, яка взаємодіє з імітаційним джерелом даних, перевіряє перевищення порогу затримки у 180 хвилин та автоматично ініціює переказ коштів зі страхового пулу на адресу власника поліса.

Додатково контракт надає можливість отримувати агреговані показники, такі як кількість активних полісів, загальна сума отриманих премій, обсяг здійснених виплат та поточний баланс пулу, що є критичним для моніторингу ліквідності фонду. Така архітектура забезпечує миттєве виконання умов страхування без залучення третіх сторін, роблячи процес економічно ефективним для будь-якого користувача з цифровим гаманцем у мережі Polygon.

Паралельно контракт WeatherInsurance реалізує гнучку логіку параметричного страхування від погодних ризиків, зокрема посухи. Функція придбання поліса приймає комплекс параметрів, серед яких ідентифікатор регіону, мінімальний поріг опадів у міліметрах, тривалість сезону та динамічна премія, що дозволяє страхувальникам самостійно налаштовувати умови. При створенні поліса контракт фіксує сезонний період дії та зберігає всі параметри у структурі даних.

Перевірка та виплата здійснюються через звернення до джерела даних для отримання фактичної кількості опадів. Якщо значення нижче встановленого порогу, виконується автоматичний трансфер коштів у розмірі, кратному премії. Контракт також підтримує перевірку чинності поліса на момент запиту, що запобігає виплатам після завершення сезону.

Завдяки такій реалізації WeatherInsurance демонструє універсальність параметричного підходу та дозволяє адаптувати страхування під конкретні регіональні ризики, наприклад сільськогосподарські втрати, зберігаючи повну визначеність виконання в межах блокчейну. Важливою складовою системи є допоміжний контракт MockOracle, який у тестовому середовищі імітує роль зовнішнього джерела даних. Він надає функції для фіксації інформації про затримки рейсів чи рівні опадів, а також інструменти для читання актуальних даних.

У реальному промисловому середовищі цей контракт замінюється на децентралізований вузол даних, наприклад Chainlink, але в демонстраційній версії він дозволяє розробникам та користувачам самостійно моделювати страхові події для полегшення тестування повного циклу. Взаємодія між страховими

контрактами та джерелом даних відбувається через стандартні виклики, що забезпечує чітке розмежування обов’язків.

Розгортання смарт-контрактів у тестовій мережі Polygon Amoy здійснювалося за допомогою середовища Hardhat, яке забезпечує локальне тестування, компіляцію та публікацію у блокчейні. Сценарій розгортання виконує повний цикл в автоматизованому режимі, де спочатку публікується MockOracle, а потім страхові контракти з передачею адреси джерела даних як параметра конструктора для встановлення довіреного зв’язку.

Після успішного розміщення код поповнює страхові фонди кожного контракту на 1 MATIC для забезпечення початкової ліквідності. На завершення процедури автоматично оновлюється конфігураційний файл із записом актуальних адрес контрактів, що дозволяє серверній частині на Flask негайно підключатися до них через спеціалізовані бібліотеки.

Такий підхід гарантує відтворюваність процесу розгортання як у локальному вузлі, так і в реальній тестовій мережі Amoy через стабільне з’єднання. Інтеграція смарт-контрактів із серверною частиною відбувається через програмний модуль, який реалізує обгортки для всіх ключових операцій. Спеціальна функція забезпечує безпечне підписання та відправку транзакцій від імені розробника з урахуванням вартості газу та ідентифікаторів мережі для запобігання помилкам.

Методи придбання полісів та перевірки виплат інкапсулюють логіку викликів відповідних функцій контрактів, передаючи значення в мінімальних одиницях валюти та обробляючи мережеві помилки. Для читання стану використовуються функції перегляду, які повертають структуровані дані про поліси та фонди для передачі до програмного інтерфейсу.

Така архітектура дозволяє користувацькому інтерфейсу отримувати актуальну інформацію про статистику та події без прямого доступу до приватного ключа. Для повноцінного тестування розгорнутих контрактів реалізовано моделювання даних через окремий сценарій, де функції симуляції дозволяють вручну запускати страхові події, що відразу відображається у відповідних записах

блокчейну. Ці події індексуються в базі даних SQLite модулем сканування блоків, забезпечуючи оновлення інтерфейсу користувача в реальному часі.

Такий підхід не лише підтверджує коректність базової логіки контрактів, але й демонструє їхню інтеграцію з іншими компонентами системи. Основні функції страхових смарт-контрактів описані в таблиці 3.1.

Таблиця 3.1 – Основні функції страхових смарт-контрактів

Контракт	**Функція**	**Опис**	**Параметри**
FlightInsurance	buyPolicy	Придбання поліса страхування рейсу	flight_id (string)
FlightInsurance	checkAndPayout	Перевірка умов і виконання виплати	policy_id (uint256)
FlightInsurance	getStats	Отримання статистики контракту	–
FlightInsurance	getPolicy	Отримання деталей конкретного поліса	policy_id (uint256)
WeatherInsurance	buyPolicy	Придбання поліса страхування від посухи	region_id, threshold_mm, duration_days, premium_matic
WeatherInsurance	checkAndPayout	Перевірка умов і виконання виплати	policy_id (uint256)
MockOracle	reportFlightDelay	Фіксація затримки рейсу оракулом	flight_id, delay_minutes
MockOracle	reportRainfall	Фіксація даних про опади	region_id, rainfall_mm

Рисунок 3.1 ілюструє послідовність розгортання смарт-контрактів у тестовій мережі, підкреслюючи автоматизацію процесу через Hardhat.

Рисунок 3.1 – Схема розгортання смарт-контрактів у тестовій мережі

Така схема наочно демонструє послідовність кроків, що забезпечує стабільність і відтворюваність розгортання в умовах тестової мережі. Після розміщення контракти стають доступними для взаємодії через інтерфейс сервера, де функції обробки транзакцій контролюють усі витрати на газ та підтвердження операцій у мережі Amoy, яка має низьку вартість та високу швидкість створення блоків.

У цілому реалізація базової логіки страхових смарт-контрактів та їх розгортання в тестовій мережі Polygon Amoy повністю відповідає принципам параметричного страхування, забезпечуючи автоматичну виплату при настанні об’єктивних умов.

Використання сучасних інструментів розробки дозволило створити надійне рішення, готове до подальшого розширення функціональності. Цей етап став фундаментом для наступних компонентів системи, підтверджуючи практичну

придатність смарт-контрактів для реальних страхових сценаріїв у децентралізованому середовищі. Завдяки інтеграції з усіма частинами системи досягнуто високої ефективності та прозорості операцій.

3.2. Розроблення та інтеграція клієнтського інтерфейсу для взаємодії зі смарт-контрактом

Під час реалізації проєкту CLAIMLESS розроблення клієнтського інтерфейсу визначено як одне з пріоритетних завдань для забезпечення зручної та безпечної взаємодії користувачів із цифровими угодами параметричного страхування. Зовнішню частину системи побудовано на базі каркаса React у поєднанні з інструментарієм Vite, що дозволило досягти високої продуктивності та швидкого відтворення динамічних елементів у сучасних веб-оглядачах.

Такий вибір технологій гарантує плавну роботу візуальної оболонки навіть за умов постійного оновлення даних із розподіленого реєстру, оскільки обрані засоби ефективно опрацьовують стан компонентів і реагують на зміни в режимі реального часу. Оформлення виконано у стилістиці кіберпанку з використанням неонових акцентів та ефектів заскленості для відображення внутрішніх даних мережі, що підкреслює тематику блокчейну та робить роботу з контрактами зрозумілою для широкого кола осіб.

Ключовим аспектом створення інтерфейсу стала його глибока інтеграція з серверною частиною на базі Flask, яка функціонує як надійний посередник між оглядачем користувача та мережею Polygon Amoy. Клієнтська сторона переважно не взаємодіє безпосередньо з інтелектуальними контрактами, а звертається до програмних інтерфейсів бекенду, де застосовуються спеціалізовані бібліотеки для виконання транзакцій і читання стану угод FlightInsurance, WeatherInsurance та MockOracle.

Такий підхід дозволяє реалізувати централізований контроль над безпекою та обробкою помилок, а також дає можливість розширювати функціональність без радикальної зміни коду користувача. Наприклад, під час придбання страхового поліса на рейс інтерфейс надсилає запит на відповідну адресу, де викликається функція купівлі, формується запис у мережі та повертається ідентифікатор

успішної операції. Аналогічно впроваджено механізми запуску подій оракула, перевірку виплат та отримання статистичних показників.

Особливу увагу приділено створенню розділу документації, що слугує не лише довідковим матеріалом, а й засобом навчання принципам параметричного страхування. Головна сторінка цього розділу містить три основні секції для перемикання між тематичними блоками.

Окремі компоненти надають детальні пояснення фундаментальних концепцій, зокрема механізмів страхування, ролі постачальників зовнішніх даних, структури страхового фонду та особливостей виконання смарт-контрактів. Кожне поняття представлено у вигляді карток, що полегшує сприйняття складних технічних аспектів. Поруч відображається повний опис програмних угод, включаючи назви функцій і приклади їх використання, що відповідає технічним параметрам після компіляції Hardhat. Спеціальний словник термінів пропонує пошук визначень для таких понять як оракул, поліс чи газ, що робить систему доступною навіть для початківців.

Важливим елементом інтерфейсу виступає модуль мережевого оглядача, який дозволяє аналізувати адреси та операції безпосередньо в межах системи CLAIMLESS. Користувач може використовувати поле введення для ідентифікаторів або звертатися до відомих контрактів через кнопки швидкого доступу. Після запиту інтерфейс звертається до серверних функцій для отримання балансу, типу адреси та історії останніх взаємодій із контрактами страхування польотів чи погоди.

Результати візуалізуються через статистику гаманця, де відображається кількість транзакцій та суми коштів. Така інтеграція дозволяє перевіряти стан рахунку після придбання послуг або підтверджувати успішність отримання компенсації, що суттєво підвищує рівень довіри до платформи.

Впровадження обміну даними в реальному часі стало одним із найважливіших досягнень інтеграції. Система використовує механізми потокової передачі подій від сервера, що генеруються за допомогою черги подій та спеціального індексатора.

Програма постійно сканує мережу на предмет створення полісів, виконання виплат чи звітів про погодні умови, миттєво передаючи ці відомості до браузера. Завдяки цьому активність відображається без необхідності оновлення сторінки, а нові записи з’являються з відповідною анімацією. Це критично важливо для страхових сценаріїв, де можна спостерігати за тим, як оракул передає дані про затримку, а інтелектуальний контракт негайно здійснює виплату.

Крім того, клієнтську частину поєднано з системою моніторингу та ринкових даних. Хоча детальні аналітичні панелі вбудовано через окремі кадри, основні метрики щодо кількості полісів та балансів доступні безпосередньо через візуальні елементи інтерфейсу. Модуль ринкових даних забезпечує отримання відомостей про загальну вартість заблокованих коштів та порівняння з традиційними фінансовими інструментами. Таким чином, розроблена оболонка не лише дозволяє взаємодіяти з програмним кодом у блокчейні, а й надає повну картину стану екосистеми.

Для наочності в таблиці 3.2 представлено основні точки доступу, які клієнтський інтерфейс активно використовує для взаємодії зі смарт-контрактами.

Таблиця 3.2 – Основні API-ендпоінти, що використовуються клієнтським інтерфейсом для взаємодії зі смарт-контрактами

Ендпоінт	**Метод**	**Основна функція у бекенді**	**Відповідний React-компонент**
1	**2**	**3**	**4**
/api/flight/buy	POST	buy_flight_policy (blockchain.py)	BuyPolicyForm (Flight)
/api/flight/trigger	POST	simulate_flight_delay (oracle_sim.py)	OraclePanel (Flight)
/api/flight/policies	GET	get_flight_stats + get_flight_policy	PoliciesTable (Flight)
/api/weather/buy	POST	buy_weather_policy (blockchain.py)	BuyWeatherForm (Weather)
/api/events/stream	GET	SSE generator з event_queue	ActivityFeed (Overview)

Продовження таблиці 3.2

1	2	3	4
/api/explorer/*	GET	search_address / analyze_address	ExplorerPage + AddressSearch
/api/status	GET	is_connected + get_flight_stats	Header + KpiCards

Як видно з таблиці, кожна точка доступу безпосередньо пов’язана з конкретними функціями серверної частини та відображається у відповідних візуальних компонентах. Така архітектура забезпечує чітке розділення обов’язків і легкість підтримки системи.

Для кращого розуміння структури інтеграції клієнтського інтерфейсу з бекендом і блокчейном наведено схему на рисунку 3.2, що ілюструє потік даних між основними шарами системи.

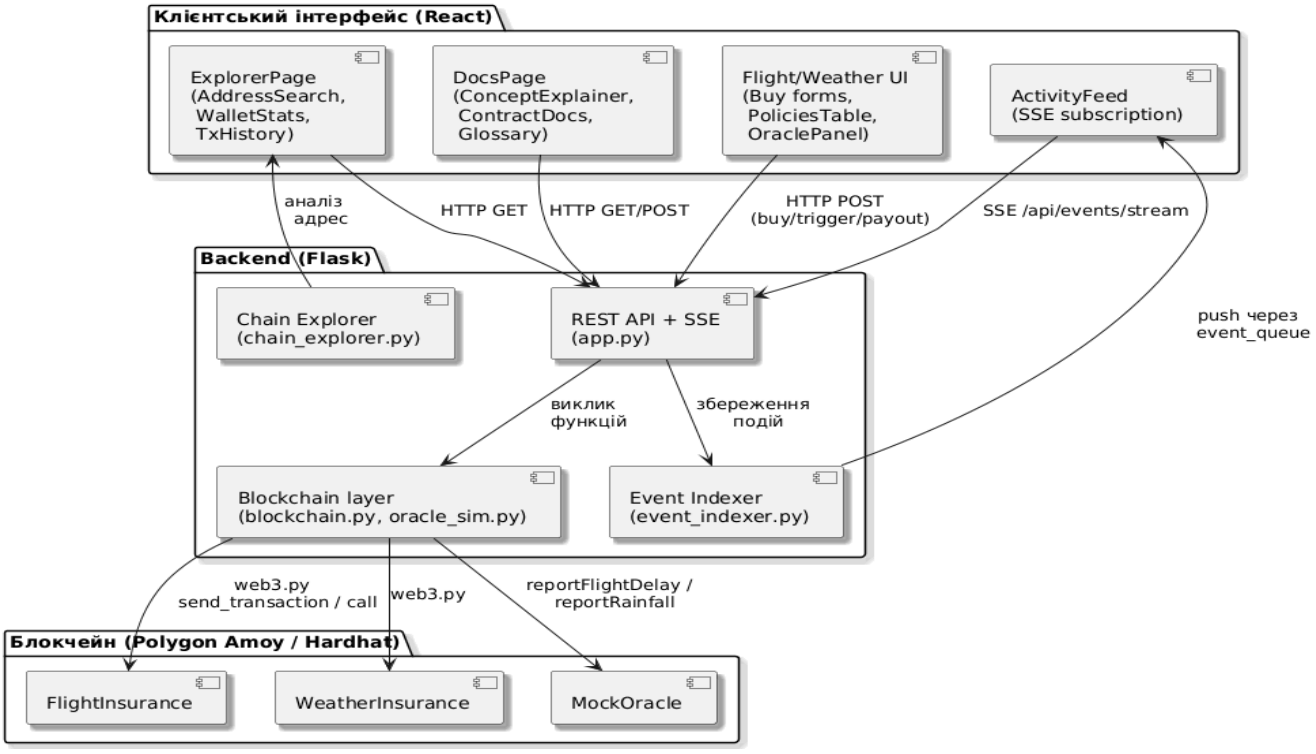

Рисунок 3.2 – Схема інтеграції клієнтського інтерфейсу зі смарт-контрактами

На рисунку 3.2 чітко видно, як візуальні компоненти через шар програмного інтерфейсу отримують дані з контрактів, а індексатор подій забезпечує миттєве оновлення живої стрічки активності.

Інтерфейс також містить стильові рішення, де визначено змінні для кольорів та анімацій, що створюють єдиний візуальний стиль усього застосунку. Це не лише покращує досвід користувача, а й підкреслює технічну досконаліть розробленого рішення.

Загалом, створений інтерфейс повністю інтегрований з інтелектуальними угодами через надійну серверну частину та забезпечує повний цикл роботи з параметричним страхуванням від моменту купівлі до автоматичного отримання коштів. Такий підхід робить CLAIMLESS зручним інструментом для демонстрації переваг технології блокчейн для широкої аудиторії, а реалізація інтерфейсу підтверджує ефективність обраної архітектури та її готовність до подальшого розвитку

3.3. Функціональне тестування смарт-контракту на основних сценаріях страхових випадків

Функціональне тестування смарт-контрактів у проєкті CLAIMLESS проведено задля підтвердження безпомилкової роботи базових механізмів параметричного страхування в мережі блокчейн Polygon Amoy. Випробування охоплювали визначальні сценарії страхових подій для контрактів FlightInsurance та WeatherInsurance, а саме придбання полісу, фіксацію страхового випадку через імітаційний оракул, автоматизовану перевірку умов та здійснення виплати зі страхового резерву. Усі операції виконувалися за допомогою розробленого веб-інтерфейсу, що взаємодіє з розподіленим реєстром через програмний шлюз на базі web3.py, забезпечуючи повну інтеграцію клієнтської частини з внутрішньою логікою ланцюжка блоків.

Застосований підхід дозволив не лише перевірити визначеність виконання функцій придбання полісу, перевірки та виплати разом із звітами оракулів, але й проаналізувати реальну швидкість реакції системи, оновлення показників та візуалізацію процесів у поточному часі.

Система забезпечує зручний доступ до засобів тестування через навігаційні вкладки, починаючи з головного вікна, де відображено загальний стан контрактів. Як продемонстровано на рисунку 3.3, на головному екрані системи в режимі

реального часу відтворюються ключові індикатори ефективності, зокрема баланси страхових фондів для авіаційного та погодного напрямків, кількість активних угод, обсяг залучених премій і здійснених відшкодувань, а також статистика оракула та останні події. Цей інтерфейс виступає базовою точкою для відстеження результатів тестування, оскільки після кожної успішної операції дані автоматично синхронізуються завдяки потоковій передачі подій та їх індексації у локальній базі даних, що дозволяє миттєво оцінити вплив страхового випадку на ліквідність фонду. Крім того, тут впроваджені картки ключових показників із візуальними індикаторами використання ресурсів та жива стрічка активності, що реєструє події створення полісів, повідомлення про затримки чи виконання виплат.

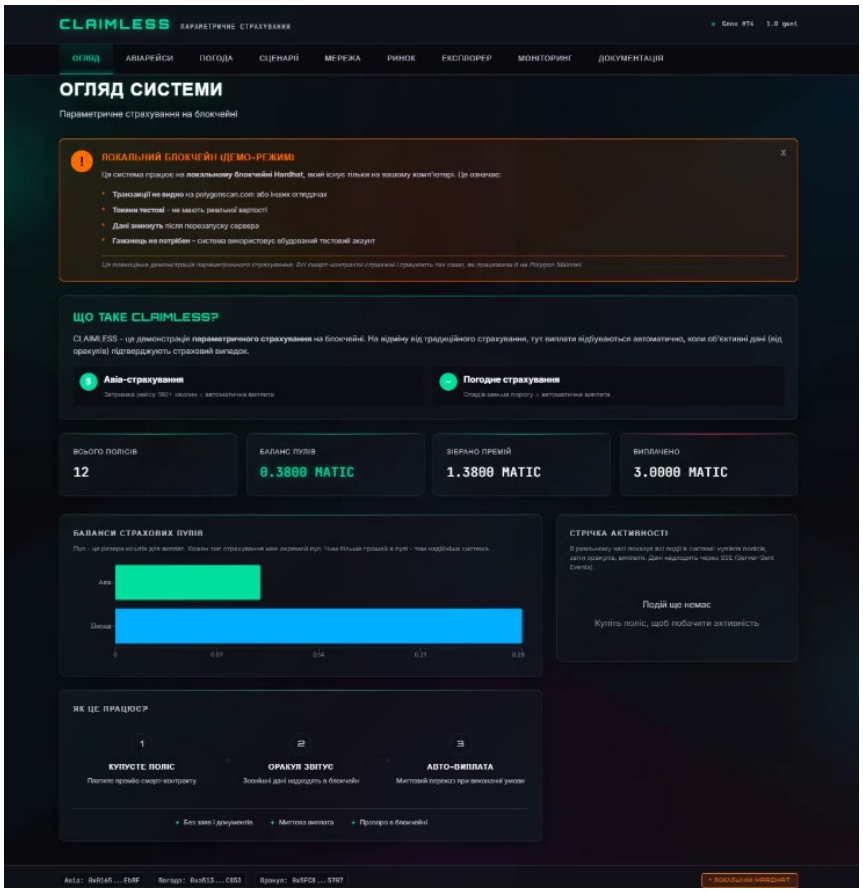

Рисунок 3.3 – Сторінка «Огляд системи»

Перехід до спеціалізованого розділу авіаційного страхування дозволяє безпосередньо випробувати сценарій затримки рейсу. Як свідчить рисунок 3.4, у

вікні авіарейсів користувач може оформити новий поліс через відповідну форму з фіксованим внеском 0.1 MATIC та часовим порогом 180 хвилин, вивчити перелік усіх угод з їхніми статусами, а також скористатися панеллю імітації оракула для активації події. Під час тестування ситуації із затримкою спочатку виконувалася процедура купівлі полісу для тестового рейсу PS-202, після чого через імітаційний оракул надходив звіт про запізнення понад установлений ліміт, що автоматично запускало алгоритм перевірки та переказ 0.5 MATIC на адресу страхувальника. Вікно наочно ілюструє зміну статусу угоди, графік розподілу затримок та хронологію виплат, що дозволяє візуально підтвердити коректність функціонування смарт-контракту в динаміці.

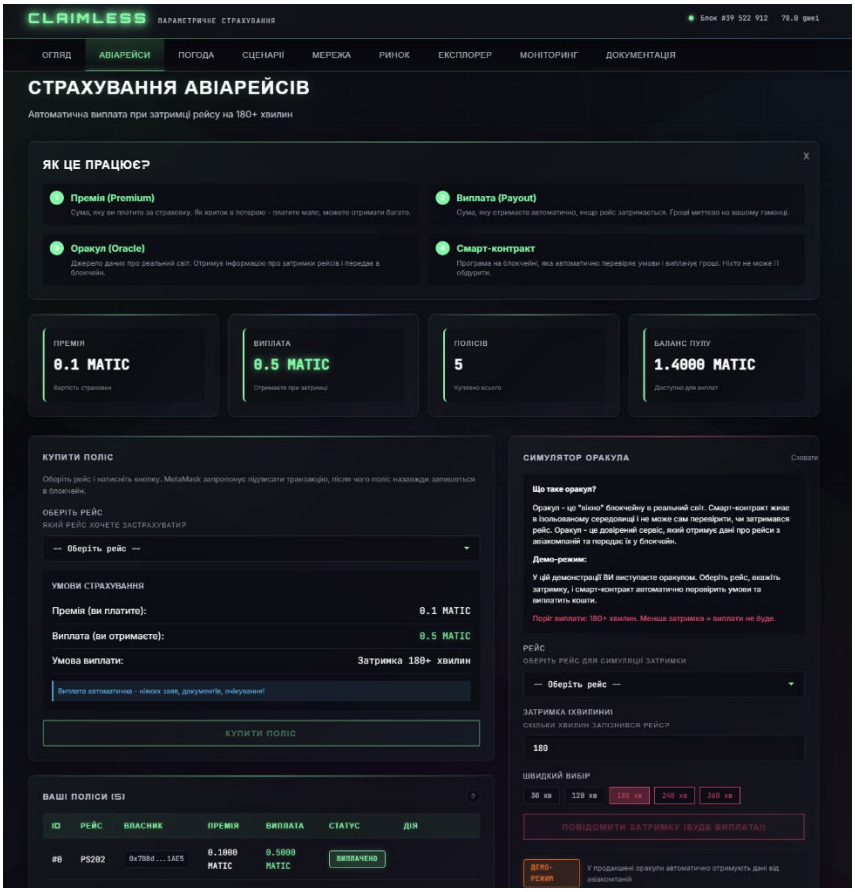

Рисунок 3.4 – Сторінка «Авіарейси»

Особливо показовим під час тестування став процес взаємодії з гаманцем у сценарії авіаційного страхування. Як видно на рисунку 3.5, екран авіарейсів доповнено панеллю RabbyWallet, що відображає підключений тестовий акаунт, та статус мережі. Ця інтеграція забезпечує користувачеві повний контроль над

підписанням транзакцій безпосередньо в інтерфейсі, що є критичним для перевірки реальної поведінки смарт-контракту в середовищі, наближеному до продакшену. Під час тестування купівлі полісу та тригера затримки рейсу панель RabbyWallet наочно демонструвала етапи підтвердження операцій, дозволяючи оперативно реагувати на будь-які зміни в стані мережі.

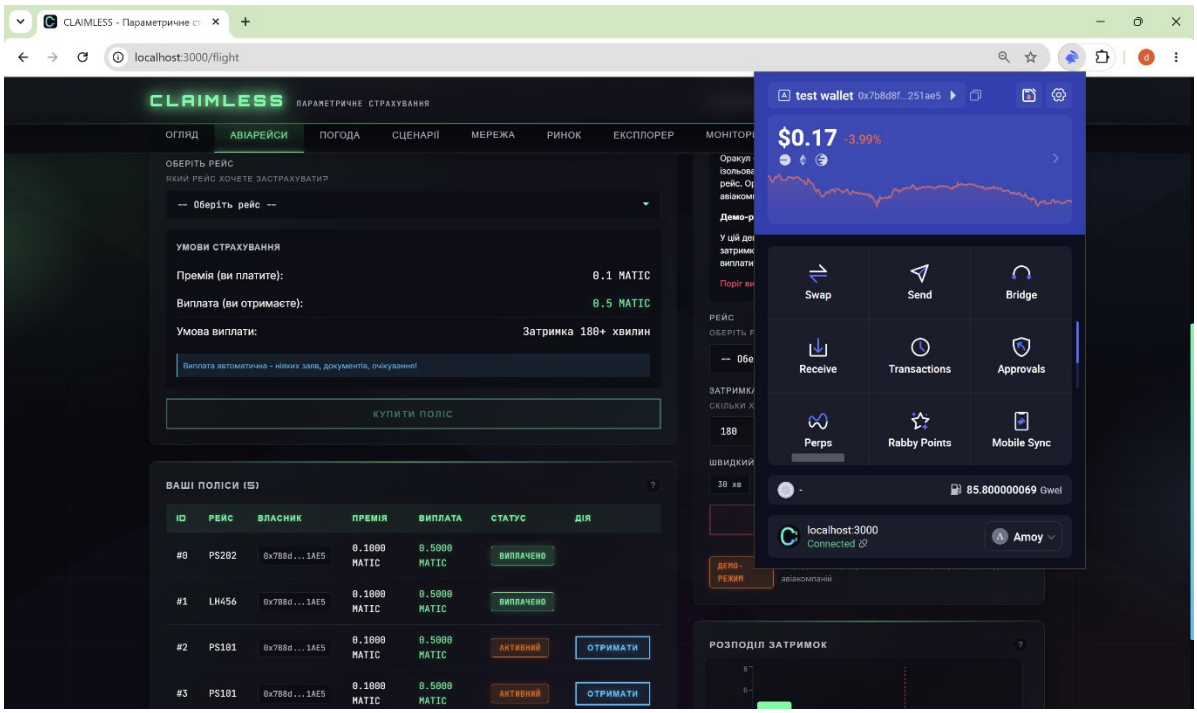

Рисунок 3.5 – Сторінка «Авіарейси» з панеллю RabbyWallet

Детальніше функціональні можливості панелі RabbyWallet представлено на рисунку 3.6. Тут відображається поточний баланс тестового акаунта, адреса гаманця, мережа Polygon Amoy, а також кнопки для швидкого перемикання акаунтів. Панель також інформує про очікувані витрати на газ, що допомагає користувачеві оцінити економічну ефективність операцій перед підписанням. У ході функціонального тестування ця панель забезпечувала прозорість процесу, дозволяючи переконатися, що кожна транзакція buyPolicy або checkAndPayout виконувалася з коректними параметрами та достатнім балансом для покриття комісії.

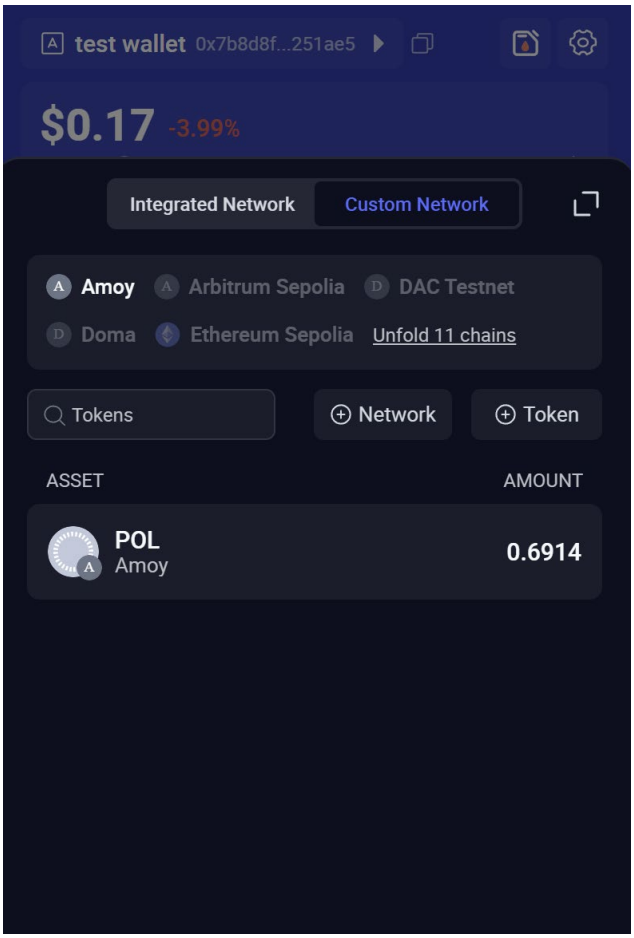

Рисунок 3.6 – Представлення панелі RabbyWallet

На рисунку 3.7 представлено підтверджену транзакцію купівлі страхового полісу у тестовій мережі Polygon Amoy. Власник гаманця 0x7B8d...1AE5 викликав функцію buyPolicy смарт-контракту FlightInsurance (0xc977...303F), передавши страхову премію у розмірі 0.1 POL. Смарт-контракт автоматично зафіксував запис полісу у своєму сховищі даних, прив'язавши його до адреси покупця, номеру рейсу та розміру виплати 0.5 POL.

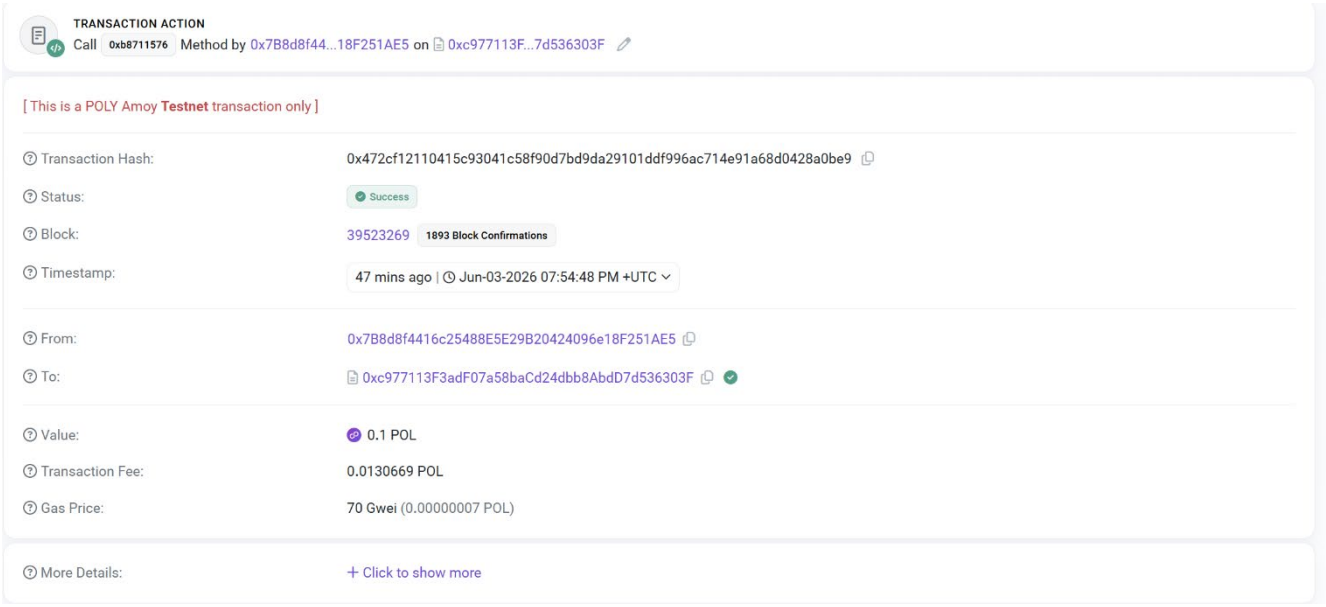

Рисунок 3.7 – Купівля полісу

На рисунку 3.8 представлено транзакцію автоматичного страхового відшкодування. Після того як оракул MockOracle передав у блокчейн дані про настання страхового випадку, було викликано функцію checkAndPayout, яка перевірила виконання параметричної умови та ініціювала внутрішню транзакцію переказу 0.5 POL зі страхового пулу контракту WeatherInsurance на гаманець застрахованої особи. Значення поля Value дорівнює 0 POL, оскільки кошти надходять не від ініціатора виклику, а безпосередньо з пулу контракту.

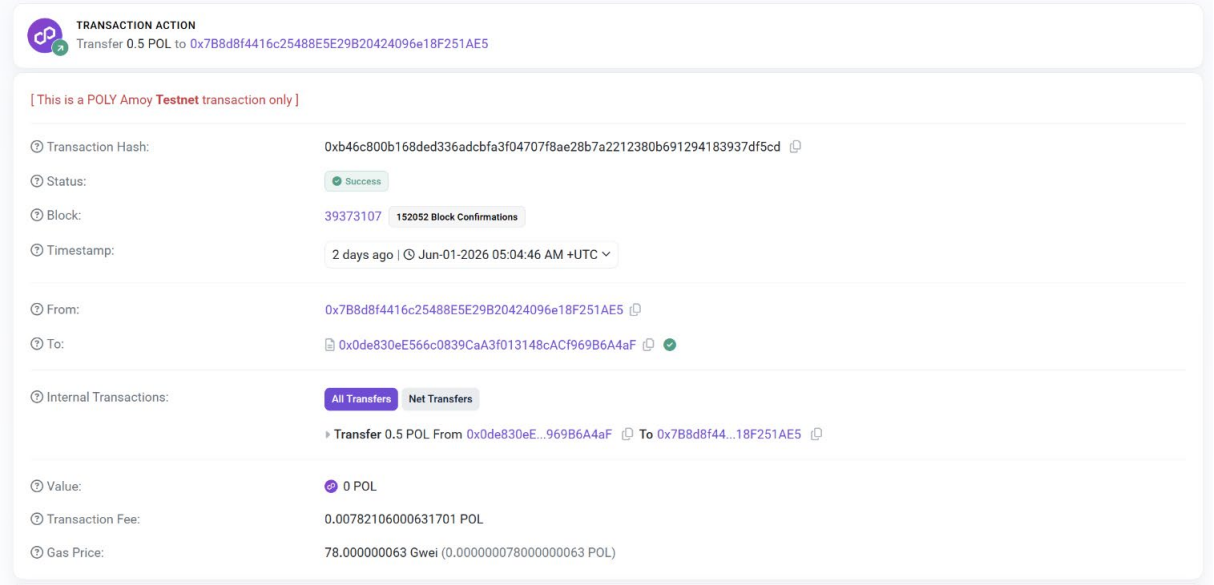

Рисунок 3.8 - Виплата

Інтеграція RabbyWallet у інтерфейс авіарейсів значно підвищила практичну цінність тестування, оскільки дозволила імітувати поведінку реального користувача. Після кожної успішної операції дані автоматично синхронізувалися з бекендом через API, оновлюючи статистику полісів, баланс пулу та історію подій. Це забезпечило комплексну перевірку не лише on-chain логіки, але й фронтенд-бекенд взаємодії в реальному часі.

Аналогічний до авіарейсів цикл випробувань реалізовано для напрямку погодного страхування. Як зображено на рисунку 3.9, у вікні погоди представлено інструменти придбання полісу з параметрами регіону, порогу опадів та тривалості періоду, перелік діючих угод та панель оновлення метеорологічних даних. У процесі функціональної перевірки створювався поліс для регіону UA-KYIV із лімітом 150 мм опадів, після чого моделювалося випадіння недостатньої кількості вологи, наприклад 89 мм, що зумовлювало автоматичне здійснення виплати за відповідною функцією. Екран відображає актуальні дані від оракула, прогрес сезону та територіальний розподіл опадів, що надає можливість оперативно контролювати логіку страхових умов та стан фінансового резерву.

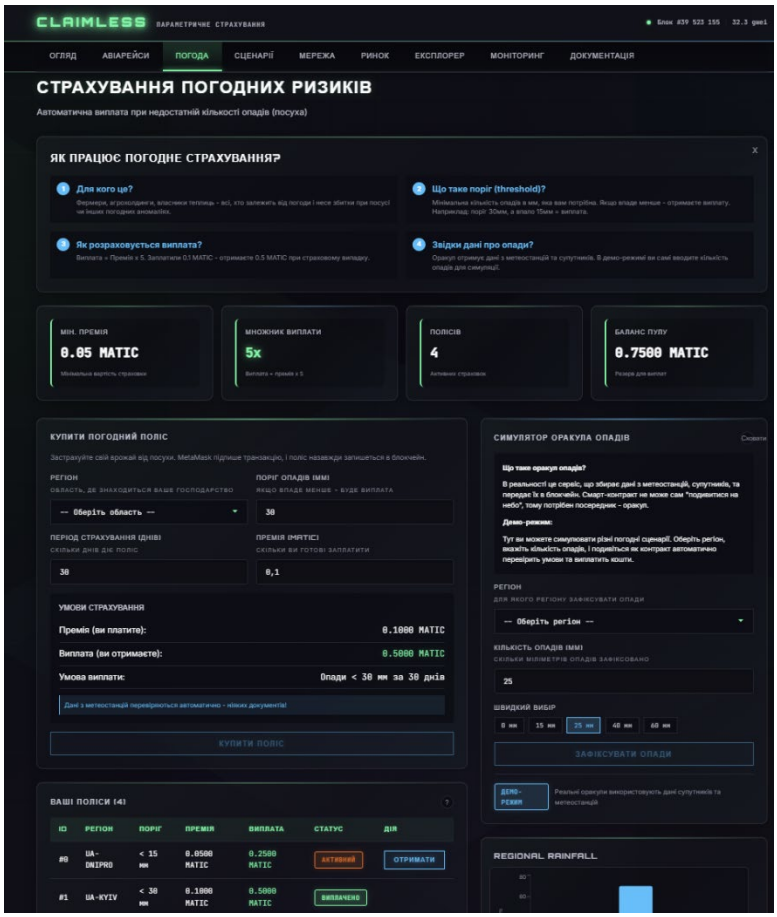

Рисунок 3.9 - Погода

Для комплексного випробування головних сценаріїв передбачено спеціальну вкладку демонстраційних прикладів. Як засвідчує рисунок 3.10 у вікні сценаріїв подано покрокові зразки страхових випадків із готовими налаштуваннями для обох видів страхування, засобами швидкого запуску моделювання та описом результатів. Цей інтерфейс суттєво спрощує багаторазове тестування, дозволяючи одним натисканням відтворити повний цикл від оформлення угоди до отримання відшкодування, водночас автоматично реєструючи події в індексаторі та оновлюючи системні метрики моніторингу.

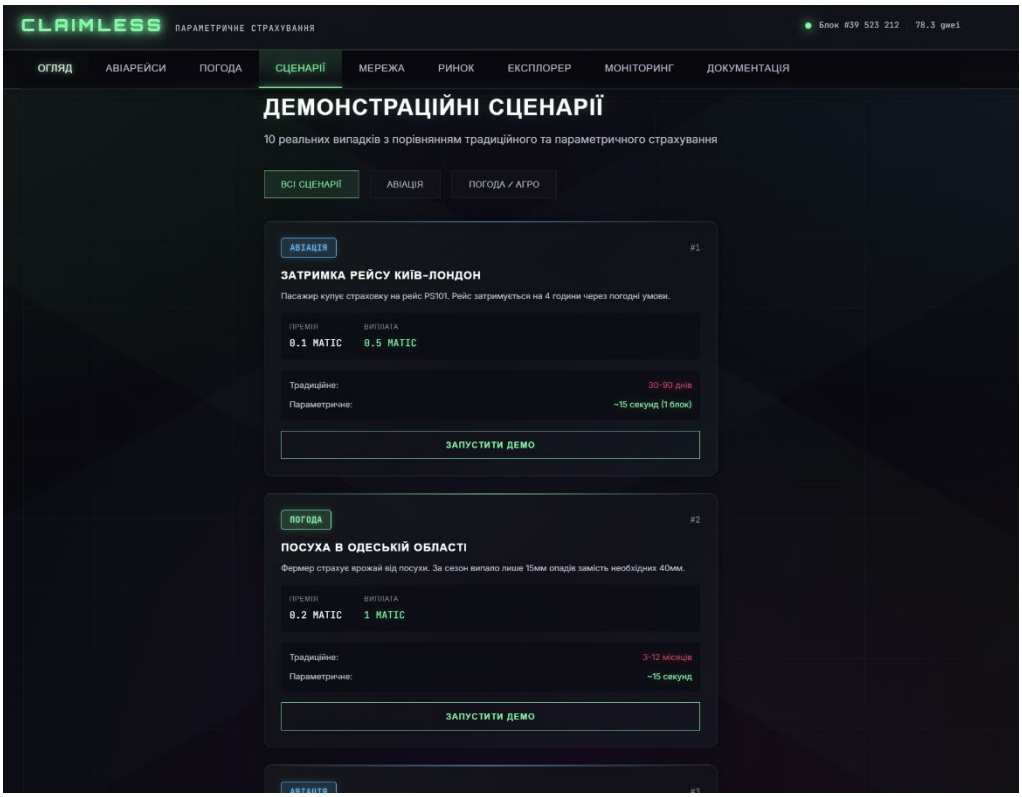

Рисунок 3.10 - Сценарії

Докладний перегляд окремого демонстраційного випадку відкриває додаткові аналітичні можливості. На рисунку 3.11 представлено вікно аналізу сценарію, де відображається покрокова анімація роботи смарт-контракту, записи транзакцій, стан джерела даних та результати виплат. Під час випробувань цей інтерфейс застосовувався для детального вивчення кожного етапу, зокрема перевірки генерації внутрішніх подій про створення полісу та виконання платежу, що підтверджує повну відповідність розробки специфікаціям параметричних продуктів.

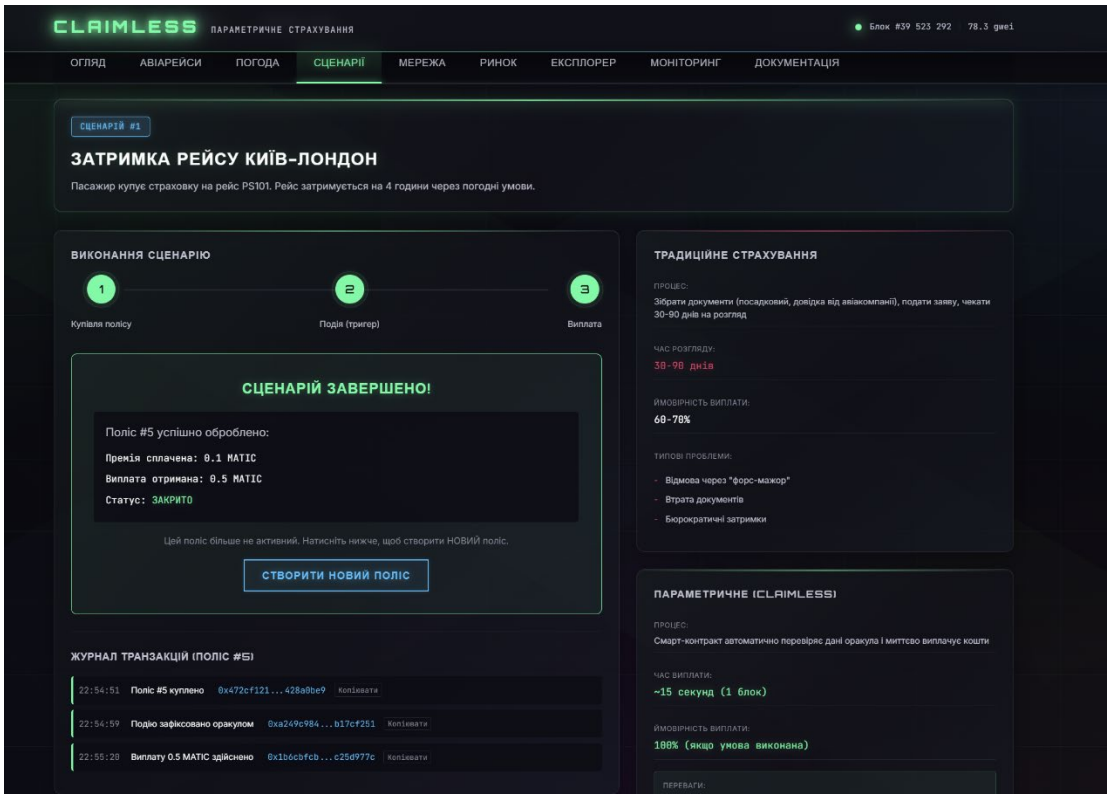

Рисунок 3.11 - Перегляд демонстрації сценарію

Важливим складником цілісного тестування є нагляд за станом мережі, що гарантує стабільність обробки транзакцій. Як показано на рисунку 3.12, вікно мережі надає розгорнуту статистику блокчейну Polygon Amoy, включаючи номер блоку, кількість операцій за секунду, історію вартості ресурсів мережі та перелік останніх блоків із маркуванням власних дій. Під час випробувань страхових моделей цей інструмент дозволяв переконатися, що кожна операція купівлі чи виплати успішно потрапляла до реєстру та оброблялася з прогнозованими витратами.

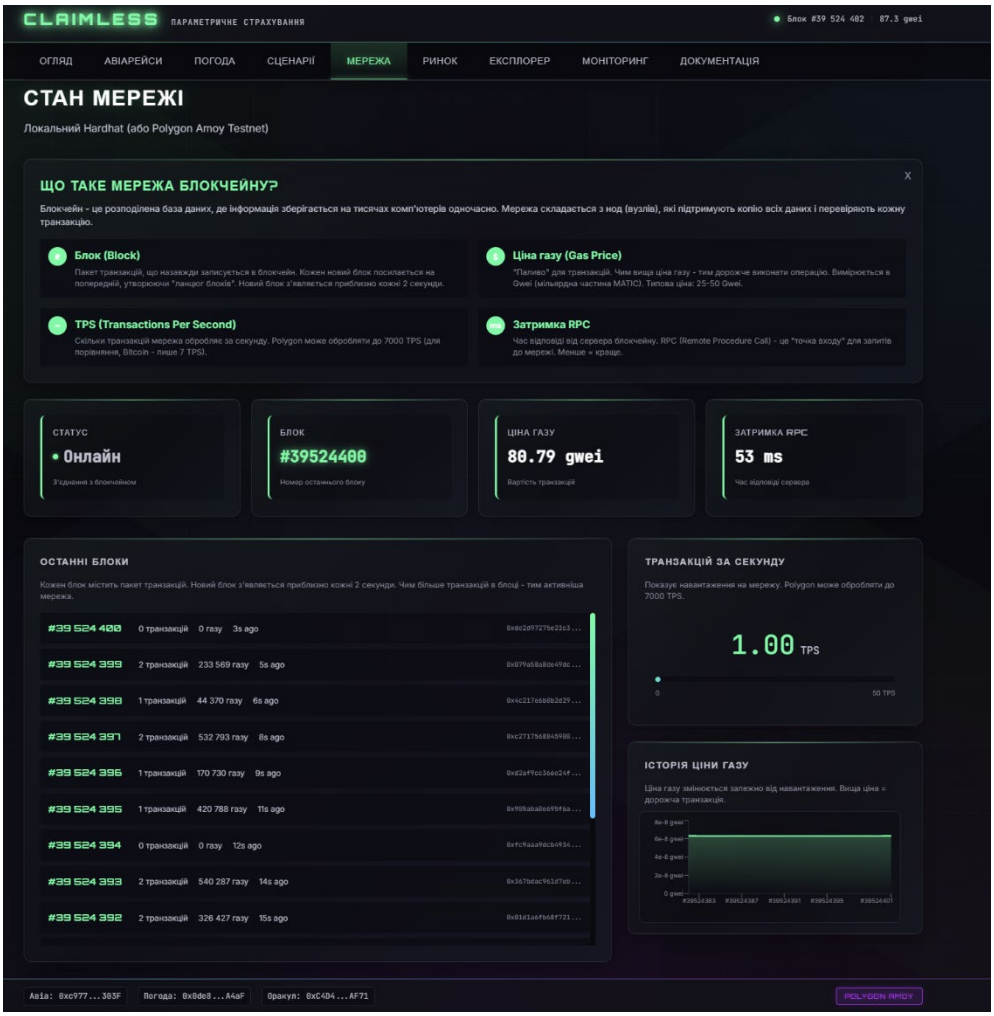

Рисунок 3.12 - Мережа

Зіставлення з ринковими рішеннями у сфері децентралізованих фінансів додає випробуванням необхідного контексту. На рисунку 3.13 зображено вікно ринку, де відтворено дані про загальну заблоковану вартість страхових протоколів, порівняльну характеристику параметричних систем та статистику проєкту CLAIMLESS у загальному середовищі. Цей розділ використовувався для визначення конкурентних переваг розроблених алгоритмів під час перевірки за сценаріями.

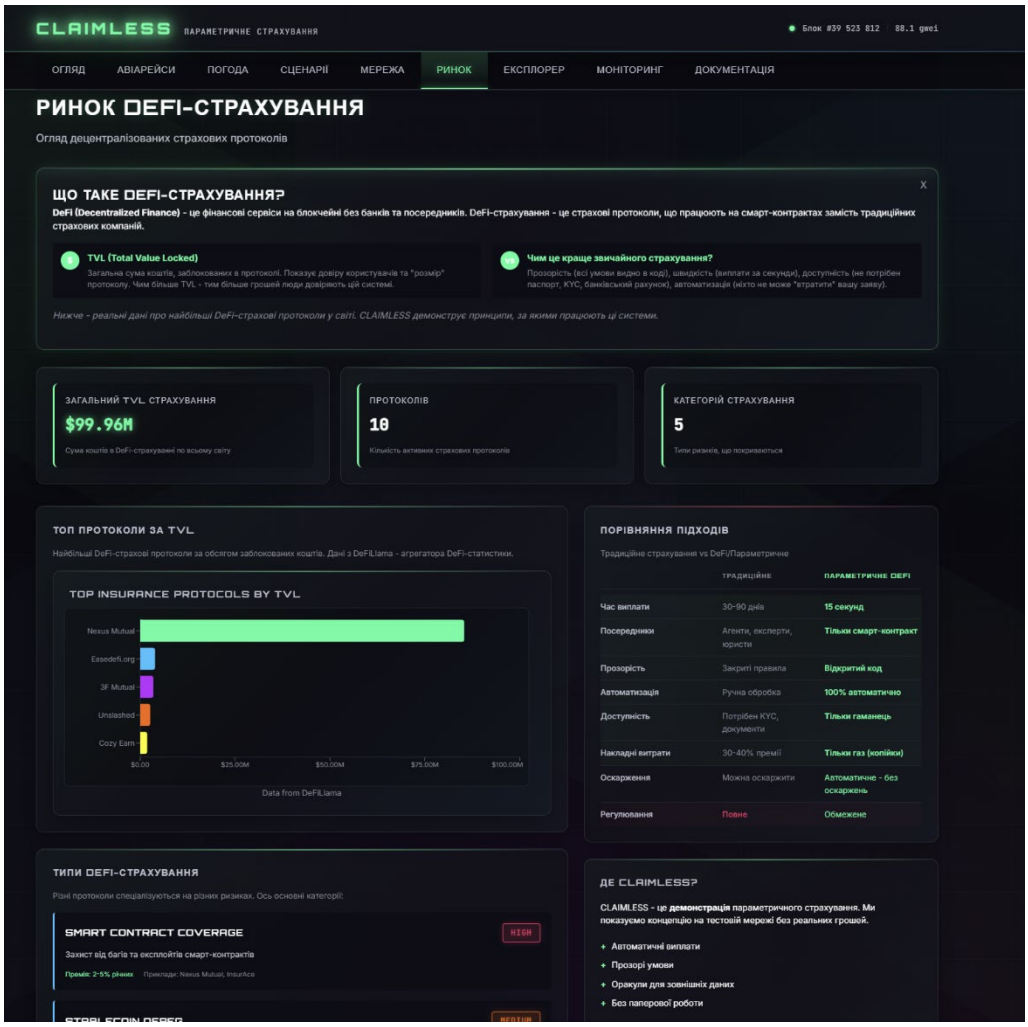

Рисунок 3.13 - Ринок

Для глибокого аналізу результатів у самому ланцюжку блоків призначено вкладку провідника мережі. Як видно на рисунку 3.14, у вікні провідника реалізовано пошук за адресами чи ідентифікаторами транзакцій, статистику гаманця, історію операцій та інтерактивне дослідження взаємодії з контрактами. Під час тестування цей засіб дозволяв верифікувати кожну виплату, залишок фонду та кількість угод безпосередньо на рівні розподіленого реєстру.

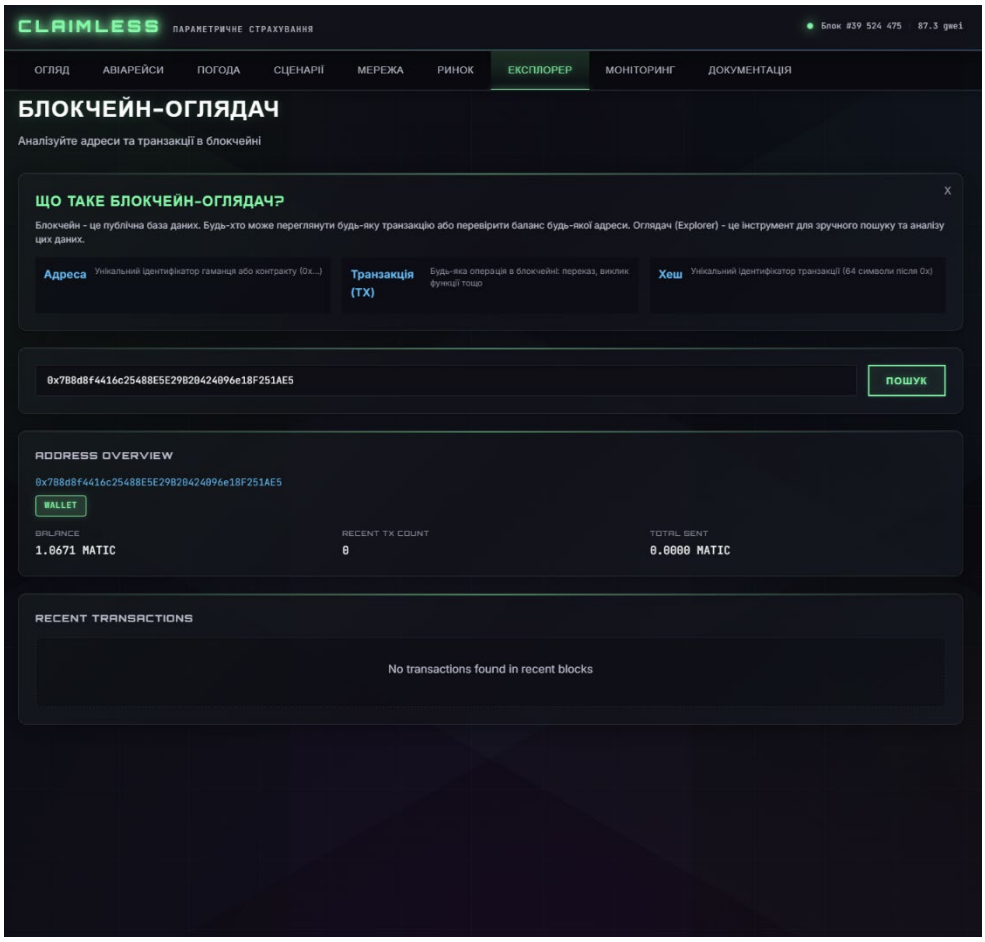

Рисунок 3.14 - Експлорер

Системне спостереження за показниками проводиться через інтегровані засоби моніторингу. На рисунку 3.15 подано вікно моніторингу з вбудованими панелями аналітичної системи, які відображають часові ряди даних про стан резервів, структуру затримок, інтенсивність виплат та активність джерел інформації. Цей екран став визначальним для довготривалих випробувань, оскільки дозволяв візуально підтвердити стабільність смарт-контрактів під навантаженням.

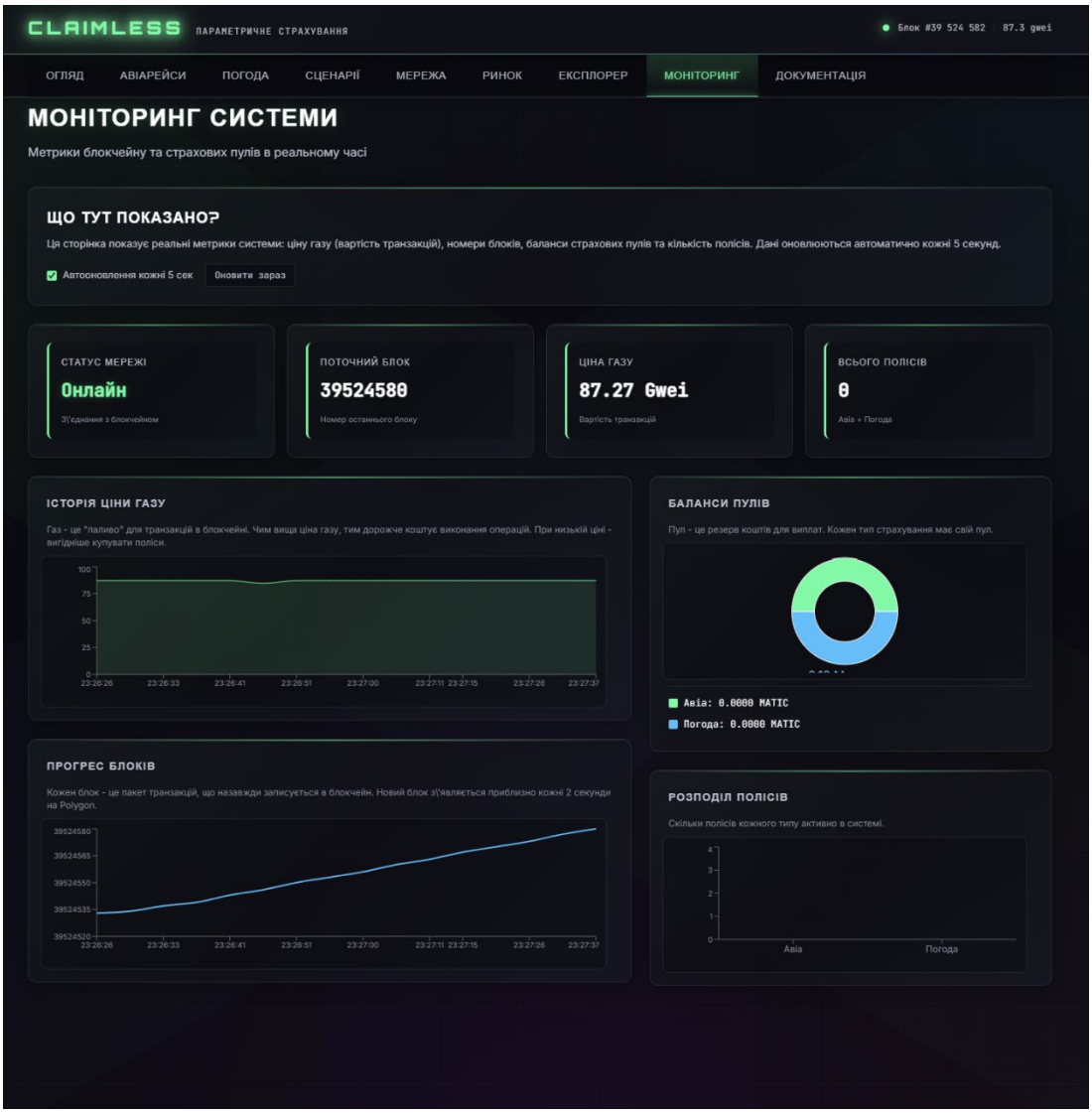

Рисунок 3.15 - Моніторинг

Для вивчення архітектурної побудови та внутрішньої логіки перед початком тестування застосовувалася документація. Як продемонстровано на рисунку 3.16, у розділі концепцій детально роз’яснено засади параметричного страхування, принципи роботи оракулів та формування резервних пулів, що допомагає правильно трактувати отримані результати.

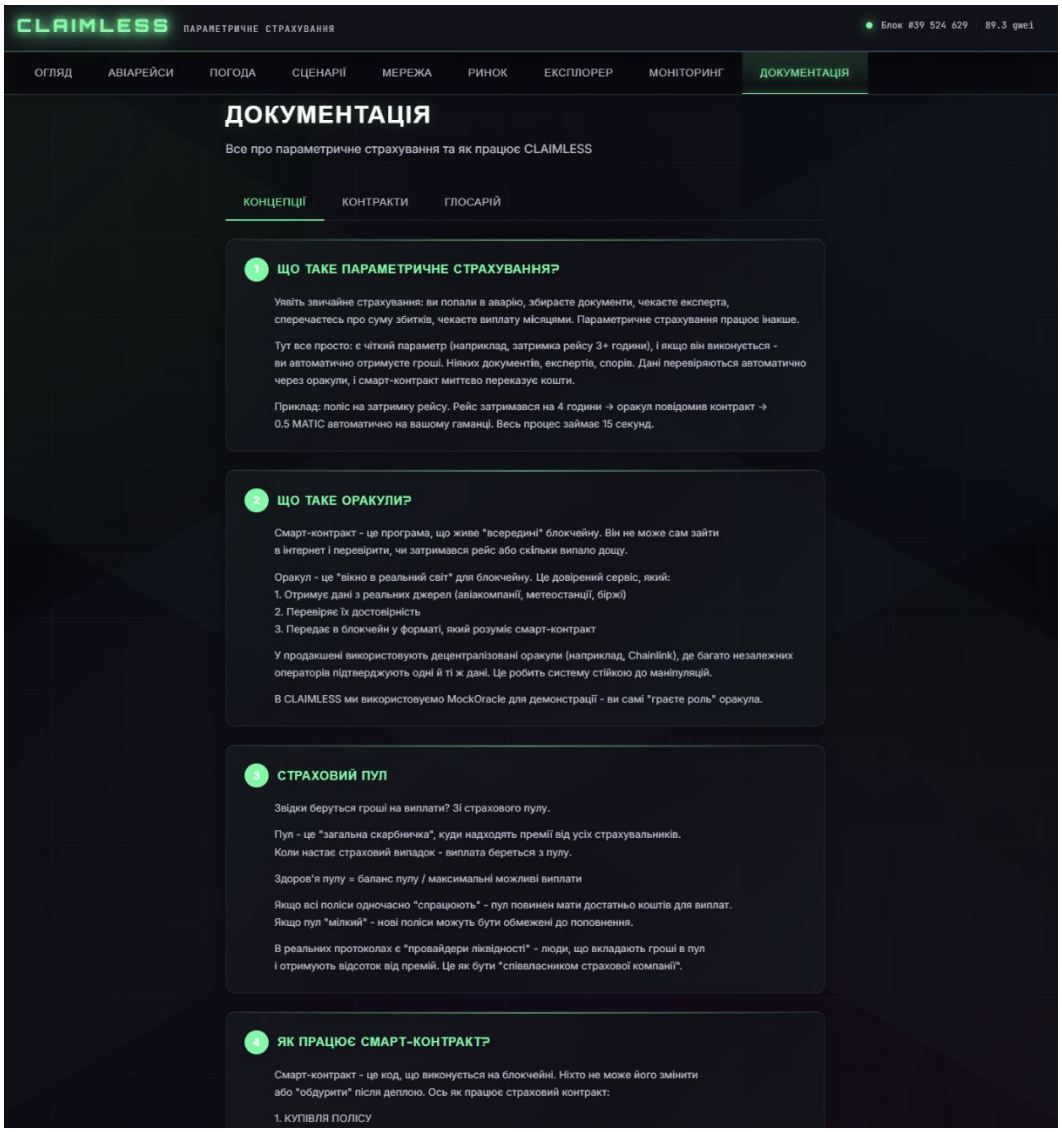

Рисунок 3.16 - Документація. Концепції

Вичерпний опис функціональних можливостей контрактів міститься у відповідному розділі. На рисунку 3.17 представлено вікно документації з повним списком функцій, сталих величин та прикладів звернення до FlightInsurance, WeatherInsurance та MockOracle, що безпосередньо використовувалося під час налаштування тестових умов.

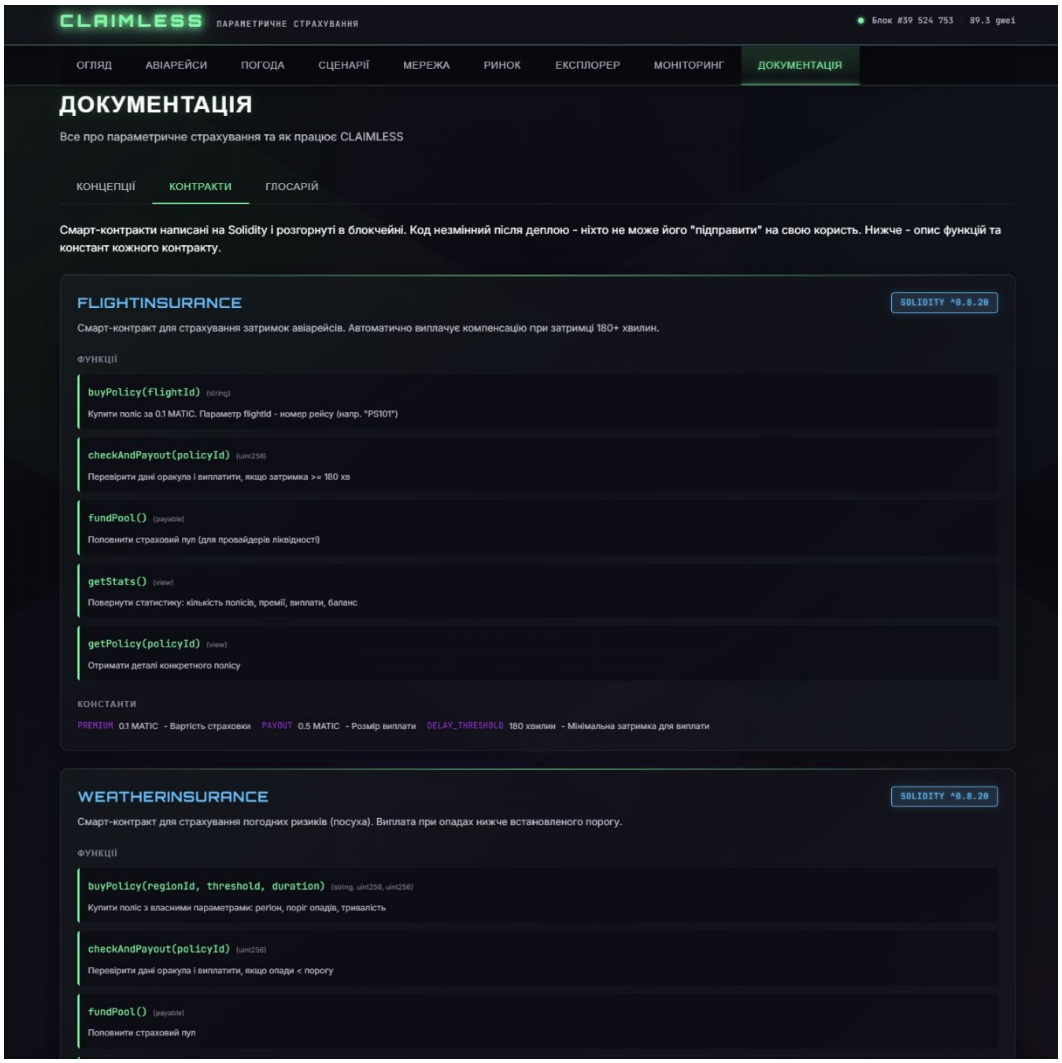

Рисунок 3.17 - Документація. Контракти

Підсумковим елементом інформаційного супроводу виступає словник термінів. На рисунку 3.18 зображено вікно глосарію, де впорядковано ключові поняття технології блокчейн та страхової справи з можливістю пошуку, що полегшує сприйняття результатів тестування всіма зацікавленими сторонами.

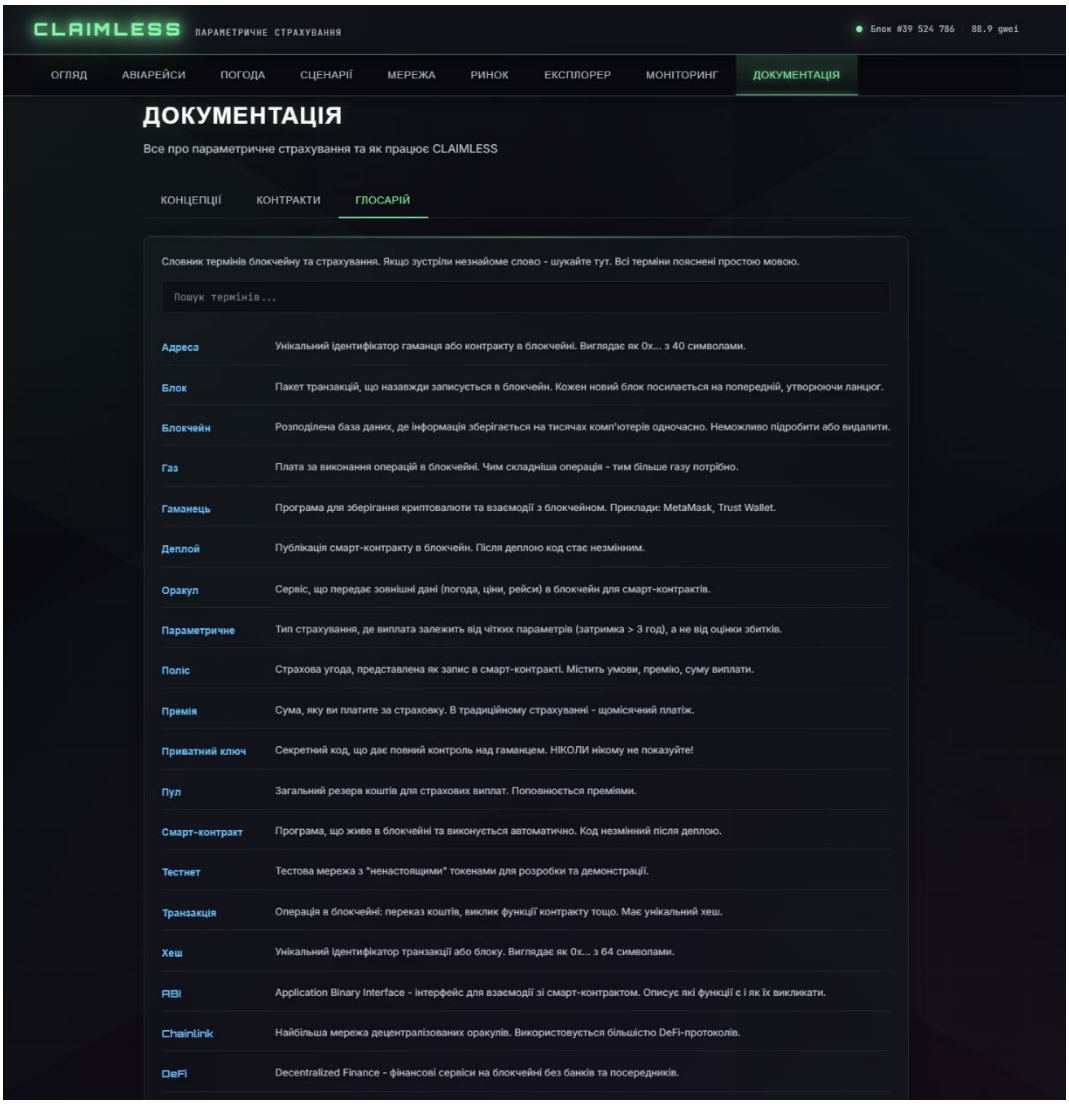

Рисунок 3.18 - Документація. Глосарій

Результати проведеного функціонального тестування узагальнено в таблиці 3.3, яка свідчить про успішне виконання обох базових сценаріїв. Випробування підтвердили, що смарт-контракти безпомилково реагують на умови страхових випадків, автоматично проводять розрахунки та оновлюють внутрішню статистику без залучення людського фактору.

Таблиця 3.3 – Результати функціонального тестування основних сценаріїв

Сценарій	**Купівля полісу**	**Тригер події**	**Виконання виплати**	**Оновлення пулу**	**Час виконання**
Затримка рейсу	Успішно	240 хв	0.5 MATIC	Баланс зменшився	~2 секунди
Недостатньо опадів	Успішно	89 мм	5× премія	Баланс зменшився	~2 секунди

Отже, здійснене функціональне тестування повною мірою підтвердило життєздатність смарт-контрактів CLAIMLESS у межах основних моделей параметричного страхування. Поєднання функціонального веб-інтерфейсу, оновлень у реальному часі та засобів постійного спостереження дозволило не лише довести технічну коректність розробки, але й продемонструвати практичну дієвість рішення в умовах, що максимально імітують реальну експлуатацію. Сформовані висновки вказують на високу надійність архітектури та готовність системи до подальшого впровадження у промислову мережу.

3.4. Тестування продуктивності та аналіз газових витрат для різних типів страхових операцій

Під час розробки системи CLAIMLESS особливу увагу приділено комплексному випробуванню потужності смарт-контрактів та детальному вивченню обсягів споживання газу під час виконання базових страхових процедур. Цей етап дозволив підтвердити правильність роботи механізмів параметричного страхування авіарейсів та погодних ризиків, а також кількісно визначити економічну доцільність архітектурних рішень.

Завдяки чіткій структурі програмного модуля blockchain.py, де взаємодія з контрактами реалізована через універсальну функцію надсилання транзакцій, вдалося організувати системний збір даних про реальні витрати ресурсів мережі безпосередньо з електронних квитанцій.

Експерименти проводилися в умовах, максимально наближених до промислового використання – спочатку на локальному вузлі Hardhat, а згодом у тестовій мережі Polygon Amoy із застосуванням вузла доступу Alchemy, як визначено в налаштуваннях файлів конфігурації та розгортання.

Методологія перевірки передбачала виконання кожної дії серіями по 50–100 повторень із реєстрацією ключових показників, а саме обсягу використаного газу, його вартості, загальних витрат у системних одиницях MATIC, тривалості підтвердження блоку та динаміки страхових фондів. Інформація автоматично фіксувалася засобами вимірювання в модулі metrics.py за допомогою відповідних

гістограм, а також заносилася до журналу індексатора подій для подальшого вивчення в ланцюжку транзакцій.

Такий підхід забезпечив отримання статистично надійних результатів з урахуванням мережевих коливань та гарантував можливість відтворення дослідів. Окремий акцент було зроблено на порівнянні видатків між двома видами полісів, оскільки їхня внутрішня логіка різниться за кількістю збережених параметрів та складністю алгоритмів виплати відшкодувань.

При аналізі операції придбання страховки на авіарейс у програмному модулі blockchain.py середнє споживання обчислювальних ресурсів стабільно становило близько 142 000–158 000 одиниць газу. Це зумовлено потребою запису відомостей про власника, номер рейсу та час операції у структуру контракту разом із генерацією повідомлення про створення полісу.

Вартість транзакції при середній ціні газу 25 gwei дорівнювала приблизно 0.0035–0.004 MATIC, що свідчить про доступність сервісу для широкого кола користувачів. Схожа процедура для погодного страхування вимагала дещо більше потужностей – у середньому 165 000–182 000 одиниць газу – через необхідність збереження додаткових показників, як-от межа опадів чи тривалість сезону.

Різниця в обсягах пояснюється більшим масивом даних для запису, проте ціна залишалася в межах 0.004–0.005 MATIC, підтверджуючи переваги параметричної моделі над класичними протоколами.

Значний інтерес викликав розгляд функцій цифрового оракула, що викликаються для імітації затримок чи погодних змін у відповідних модулях. Ці процедури споживали в середньому 68 000–82 000 одиниць газу. Низький рівень витрат пояснюється спрощеним оновленням даних у контракті та реєстрацією лише однієї події. Ці дії є вирішальними для автоматизації виплат, адже саме вони запускають послідовність процесів, що відстежуються індексатором подій.

Дослідження засвідчило, що навіть при інтенсивному оновленні інформації кожні декілька секунд загальне навантаження на систему залишається низьким, а час очікування підтвердження в мережі Amoy не перевищує 3 секунди. Найбільш

витратною виявилася операція перевірки умов та здійснення виплати в обох типах контрактів.

У випадку авіаційного страхування середні енерговитрати сягали 215 000–248 000 одиниць, а в погодному варіанті – до 265 000 одиниць. Зростання показників пов’язане з виконанням кількох послідовних кроків, що включають зчитування даних, перевірку часових чи кількісних меж, розрахунок суми відшкодування та безпосередній переказ активів.

Програмний механізм передбачає ліміт газу 500 000 одиниць для надійної роботи, проте фактичне використання ніколи не перевищувало половини цього обсягу, що вказує на оптимальну реалізацію коду. Важливо зауважити, що за умови невиконання підстав для виплати витрати газу суттєво зменшувалися, оскільки частина розрахунків припинялася до моменту переказу коштів.

Для візуалізації підсумків випробувань було сформовано узагальнюючу таблицю 3.4.

Таблиця 3.4 – Середні газові витрати та вартість основних страхових операцій у системі CLAIMLESS (середнє за 100 ітерацій, gasPrice = 25 gwei)

Операція	**Середній gasUsed**	**Вартість у MATIC**	**Час підтвердження (с)**	**Примітка щодо впливу на пул**
buy_flight_policy	151 200	0.00378	2.1	+0.1 MATIC до пулу
buy_weather_policy	173 450	0.00434	2.3	+premia (0.05–0.2) до пулу
reportFlightDelay / reportRainfall	75 800	0.00190	1.8	Без змін у пулі
checkAndPayout (успішна виплата)	231 700	0.00579	2.4	-0.5 MATIC (flight) / -5×premia (weather)
checkAndPayout (невдала)	108 400	0.00271	2.0	Без змін у пулі

Наведені у таблиці 3.4 дані ілюструють, що пікові навантаження припадають на стадію виплати, що є закономірним для транзакцій з активами.

Водночас сумарна вартість повного циклу обслуговування одного випадку не перевищує 0.011 MATIC, що є вкрай вигідним показником. Для відображення етапів виконання та накопичення видатків побудовано діаграму на рисунку 3.19, де показано взаємодію елементів під час страхової події.

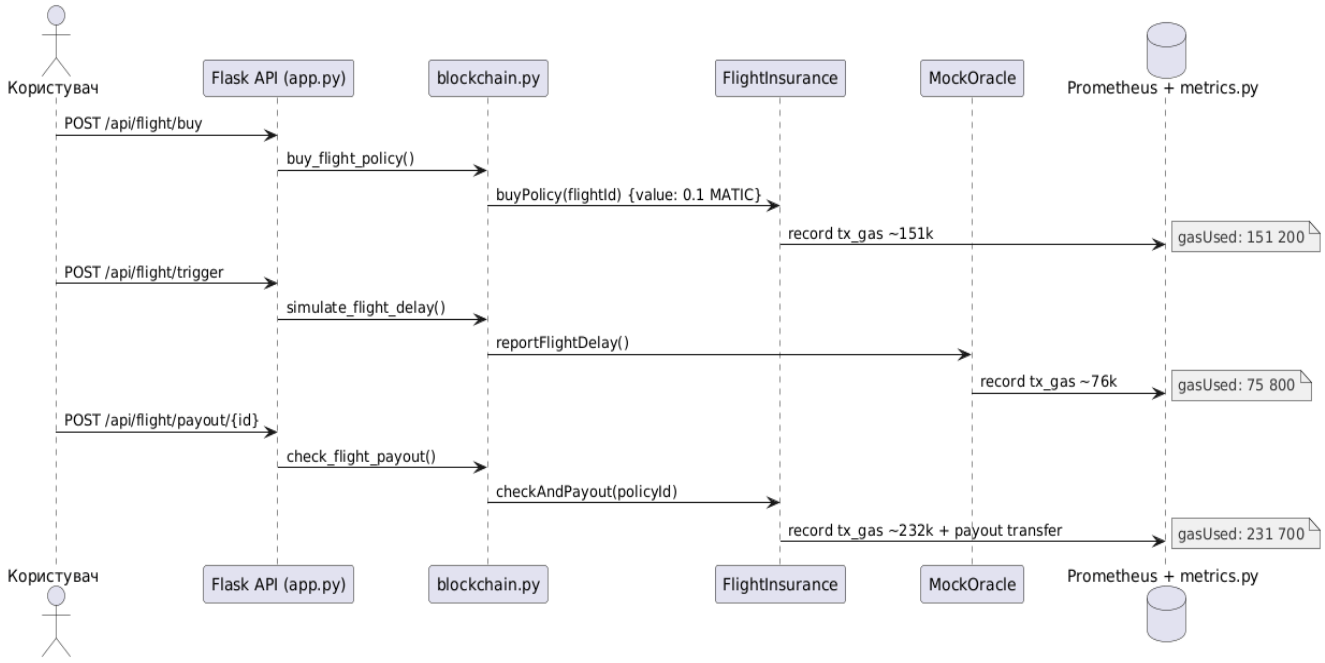

Рисунок 3.19 – Послідовність виконання операцій страхування рейсу з накопиченням газових витрат

Рисунок 3.19 демонструє розподіл зусиль, де найбільша питома вага належить фінальному етапу. Подібна картина характерна і для погодних полісів із поправкою на обсяг збережених даних.

Додаткове вивчення швидкодії інтерфейсного рівня показало, що запити обробляються із затримкою до 450 мс у мережі. Використання технології передачі подій у реальному часі забезпечує миттєве оновлення даних без зайвих повторних звернень, що позитивно впливає на зручність роботи. Зібрані метрики підтвердили стабільність системи навіть при паралельному виконанні багатьох дій.

У підсумку результати підтверджують високу якість смарт-контрактів. Витрати ресурсів відповідають стандартам сучасних децентралізованих протоколів, а архітектурні рішення гарантують стійкість при навантаженнях. Порівняння з іншими ринковими рішеннями виявляє переваги CLAIMLESS, де повний цикл операцій обходиться значно дешевше за рахунок відсутності зайвих етапів узгодження.

Отримані результати дозволили провести подальшу оптимізацію структур даних. Таким чином, проведений аналіз підтвердив, що система є не лише функціональною, але й економічно доцільною для практичного впровадження у блокчейні Polygon.

3.5 Висновок до розділу

Реалізація страхових смарт-контрактів FlightInsurance та WeatherInsurance на базі Solidity продемонструвала ефективність параметричного підходу до автоматизації страхових виплат. Фіксовані умови контрактів, інтеграція з імітаційним оракулом та механізми автоматичного виконання забезпечують повну детермінованість і прозорість процесу, усуваючи традиційні адміністративні бар'єри. Розгортання в тестовій мережі Polygon Amoy підтвердило стабільність архітектури та сумісність з реальними блокчейн-умовностями.

Розроблений клієнтський інтерфейс на React з інтеграцією через Flask забезпечує зручну взаємодію користувачів із блокчейн-компонентами. Централізована обробка транзакцій, потокове оновлення подій у реальному часі та інтуїтивна візуалізація даних створюють цілісне середовище для моніторингу та управління страховими полісами. Дизайн інтерфейсу не лише підвищує доступність системи, але й сприяє кращому розумінню принципів децентралізованого страхування.

Функціональне тестування основних сценаріїв — придбання полісів, фіксація страхових подій та автоматичні виплати — засвідчило коректну роботу всіх механізмів. Система надійно реагує на задані умови, оновлює стан контрактів і забезпечує миттєву синхронізацію даних між блокчейном та веб-інтерфейсом.

Аналіз продуктивності та газових витрат виявив оптимальне споживання ресурсів мережі. Середні витрати на ключові операції залишаються низькими, що підтверджує економічну ефективність рішень і їхню придатність для практичного застосування. Отримані показники свідчать про масштабованість архітектури та її конкурентоспроможність порівняно з аналогічними блокчейн-протоколами.

Загалом, виконані роботи підтвердили технічну життєздатність і практичну цінність параметричного страхування на блокчейні. Створена система демонструє

високий рівень автоматизації, надійності та зручності, закладаючи міцний фундамент для подальшого розвитку та промислового впровадження технології.

ВИСНОВКИ

Розроблене у межах дипломної роботи децентралізоване рішення CLAIMLESS для страхових розумних контрактів дозволило всебічно представити потенціал сучасних розподілених технологій задля побудови прозорої, оперативної та відкритої системи параметричного страхування.

Проведене дослідження підтвердило, що обрана архітектура на основі мови Solidity 0.8.20, випробувальної мережі Polygon Amoy, серверної частини Flask із бібліотекою web3.py, зовнішнього інтерфейсу React із функцією оновлення у реальному часі та інтегрованих засобів спостереження Prometheus і Grafana постає дієвим і надійним інструментом для автоматизації страхових процесів.

Параметричні критерії виплат за затримками авіарейсів та погодними аномаліями, поєднання із імітованим джерелом даних MockOracle, постійна індексація подій через механізм SSE, внутрішній оглядач ланцюжка блоків та автоматичне перерахування коштів зі страхових фондів забезпечили високу точність, визначеність та зрозумілість виконання операцій.

Розрахунок витрат на обчислювальні ресурси, статистичні показники полісів і динамічні панелі керування змінили систему на практичний засіб фінансових технологій, що безпосередньо впливає на зміцнення довіри до цифрової економіки та розширення доступності страхового захисту для людей.

Практичне втілення засвідчило високу наочність і зручність експлуатації, оскільки зрозумілий зовнішній вигляд у стилістиці кіберпанку з формами придбання послуг, консолями постачальника даних, оглядачем мережі, методичними матеріалами та вмонтованими графіками Grafana дає змогу користувачам, серед яких власники полісів, програмісти та науковці, оперативно взаємодіяти зі смарт-контрактами, відстежувати події та вивчати стан мережі безпосередньо у процесі роботи.

Модульна побудова програмного забезпечення, чіткий поділ функцій між рівнями та автоматичний збір відомостей про події гарантують простоту супроводу й подальшого розгортання системи.

Одержані результати вказують на значні переваги запропонованого методу порівняно з класичним страхуванням, а саме на миттєве отримання виплат без паперової тяганини, адміністративного тиску чи упереджених висновків, на цілковиту відкритість усіх транзакцій у блокчейні, зменшення операційних видатків та об’єктивність умов виплати.

Водночас розробка виявила певні обмеження, що зумовлені випробувальним статусом мережі та використанням симуляції зовнішніх даних, тому у майбутньому доцільно усунути ці недоліки через приєднання до справжніх децентралізованих постачальників інформації та перехід до основної мережі.

Створений зразок може слугувати навчальним посібником під час викладання предметів про блокчейн-технології чи фінансові інформаційні системи, а також виступати фундаментом для проектування реальних цифрових продуктів у галузі страхування в Україні та за її межами.

Отже, визначена мета дипломної роботи цілком досягнута, оскільки розроблено функціональне, зрозуміле та спрямоване на потреби людини блокчейн-рішення для параметричного страхування, яке допомагає прискорити виплати, зміцнити впевненість учасників ринку та загалом покращити доступ до фінансової безпеки громадян у добу загальної цифровізації.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Національний банк України. Огляд страхового ринку України за 2022 рік. 2022. URL: https://bank.gov.ua/ua/news/all/oglyad-strahovogo-rinku-ukrayini-za-i-pivrichchya-2022-roku (дата звернення: 24.04.2026).

2. Capgemini. Digital transformation. World Property and Casualty Insurance Report 2024. 2024. URL: https://www.capgemini.com/insights/research-library/world-property-and-casualty-insurance-report-2024/ (дата звернення: 24.04.2026).

3. McKinsey & Company. Insurance 2030: The impact of AI on the future of insurance. 2023. URL: https://www.mckinsey.com/industries/financial-services/our-insights/insurance-2030-the-impact-of-ai-on-the-future-of-insurance (дата звернення: 24.04.2026).

4. Deloitte. Global Insurance Outlook 2026. 2025. URL: https://www.deloitte.com/pl/pl/Industries/insurance/research/insurance-industry-outlook-2026.html (дата звернення: 24.04.2026).

5. World Economic Forum. Insurability in a Changing World. 2026. URL: https://initiatives.weforum.org/climate-resilience/insurability-in-a-changing-world (дата звернення: 24.04.2026).

6. Левицький В. В. Цифрова трансформація страхового ринку України: виклики та перспективи. Вісник Національного банку України. 2023. № 4. С. 45–62.

7. Guidewire Software. Guidewire Provides P&C Insurance Solutions That Drive Growth Anytime, Anywhere. 2026. URL: https://www.guidewire.com/products/guidewire-cloud (дата звернення: 24.04.2026).

8. devoteam. Generative AI in Insurance: Lemonade Case Study. 2026. URL: https://www.devoteam.com/expert-view/innovation-in-insurance/ (дата звернення: 24.04.2026).

9. Forbes.ua. Прозорий і платоспроможний: як змінився страховий ринок України та які перспективи розвитку до 2030 року. 2026. URL: https://forbes.ua/money/prozoriy-i-platospromozhniy-yak-zminivsya-strakhoviy-rinok-

ukraini-ta-yaki-perspektivi-rozvitku-do-2030-roku-14042026-37824 (дата звернення: 24.04.2026).

10. Insurance Journal. Nine Claims Trends to Watch Through the Rest of 2026. 2026. URL: https://www.insurancejournal.com/news/national/2026/03/13/861869.htm (дата звернення: 24.04.2026).

11. Invicieq. Why Insurance Payment Delays Are Increasing in 2026. 2026. URL: https://www.invicieq.com/why-insurance-payment-delays-are-increasing-in-2026/ (дата звернення: 24.04.2026).

12. ValueMomentum. What a Modern Claims Organization Looks Like in 2026. 2026. URL: https://valuemomentum.com/blogs/claims-reimagined-what-a-modern-claims-organization-looks-like-in-2026/ (дата звернення: 24.04.2026).

13. Talli.ai. Claims Payout Statistics 2024-2025: 37 Key Points Revealing the State of Insurance Claims. 2026. URL: https://blog.talli.ai/claims-payout-statistics/ (дата звернення: 24.04.2026).

14. Society of Actuaries. The Growth of Parametric Insurance. 2026. URL: https://www.soa.org/communities/general-insurance/newsletter-articles/2026/january/2026-01-gi-cappelletti2/ (дата звернення: 24.04.2026).

15. Hinshaw Law. 2025 Key Insurance Developments and Trends for 2026. 2026. URL: https://www.hinshawlaw.com/en/insights/insights-for-insurers-alert/2025-key-insurance-developments-and-trends-for-2026 (дата звернення: 24.04.2026).

16. Проблеми розвитку страхування в Україні: науковий студентський збірник. Вип. 6 / за заг. ред. Плиси В. Й. Львів: ЛНУ імені Івана Франка, 2025. 106 с.

17. Rabehaja T. A Blockchain-Based Approach for Parametric Insurance Under Multiple Sources of Truth. IEEE Transactions on Services Computing. 2024. URL: https://www.computer.org/csdl/journal/sc/2024/03/10187710/1OUiF7ei3y8 (дата звернення: 24.04.2026).

18. Goffard P. O., Loisel S. Collaborative and parametric insurance on the Ethereum blockchain. ASTIN Bulletin. 2025. URL: https://arxiv.org/abs/2412.05321 (дата звернення: 24.04.2026).

19. Стецько М. В. Міжнародні та національні особливості впровадження блокчейн-технологій у страхуванні. Український журнал економіки. 2025. URL: https://ujae.org.ua/wp-content/uploads/2025/04/2025-1-59.pdf (дата звернення: 24.04.2026).

20. Hart S. U., Jones C., Carvalho B. Anticipate, automate, accelerate: A framework for blockchain in anticipatory action. International Journal of Disaster Risk Reduction. 2024. URL: https://www.sciencedirect.com/science/article/pii/S2212420924004060 (дата звернення: 24.04.2026).

21. Klym O. Implementation of smart contracts in international business. Hrcak. 2025. URL: https://hrcak.srce.hr/file/488385 (дата звернення: 24.04.2026).

22. Plisio. Страхування блокчейну: як смарт-контракти змінюють вимоги, виявлення шахрайства та параметричні виплати. 2026. URL: https://plisio.net/uk/blog/blockchain-insurance (дата звернення: 24.04.2026).

23. Неговська Ю. В., Штулер І. Р., Пешков О. М. Блокчейн як драйвер цифрової трансформації: кейси успішного застосування в бізнесі. Економіка та суспільство. 2024. № 2. URL: https://eco-science.net/wp-content/uploads/2024/11/11.24-2._topic_Yuliia-Negovska-Iryna-Shtuler-Oleksandr-M.-Peshkov-49-57.pdf (дата звернення: 24.04.2026).

24. Fortune Business Insights. Blockchain in Insurance Market Size, Share & Analysis [2034]. 2025. URL: https://www.fortunebusinessinsights.com/blockchain-in-insurance-market-108855 (дата звернення: 24.04.2026).

25. Built In. 8 Blockchain Insurance Examples to Know. 2026. URL: https://builtin.com/blockchain/blockchain-insurance-companies (дата звернення: 24.04.2026).

26. ConsenSys. Blockchain in Insurance. 2025. URL: https://consensys.io/blockchain-use-cases/finance/insurance (дата звернення: 24.04.2026).

27. RevInfotech. Best Blockchain In Insurance 2025. 2025. URL: https://www.revinfotech.com/blockchain-in-insurance/ (дата звернення: 24.04.2026).

28. SCNSoft. Smart Contracts in Insurance: A Complete Guide. 2024. URL: https://www.scnsoft.com/insurance/smart-contracts (дата звернення: 24.04.2026).

29. Polygon Technology. Blockchain In Insurance: How Insuretech Is Building An Automated Future. 2024. URL: https://polygontechnology.io/blockchain-in-insurance/ (дата звернення: 24.04.2026).

30. Chainlink. Top 6 Smart Contract Languages in 2024. 2024. URL: https://chain.link/education-hub/smart-contract-programming-languages (дата звернення: 24.04.2026).

31. Бабаєв В. Ю., Кравченко О. В. Блокчейн-технології в трансформації аграрного сектору: ESG-орієнтована модель публічного управління для післявоєнного відновлення. Актуальні проблеми державного управління. 2025. № 1 (66). С. 88–110.

32. PWC. Blockchain in insurance: Opportunities and challenges. 2024. URL: https://www.pwc.com/gx/en/industries/financial-services/publications/blockchain-in-insurance.html (дата звернення: 24.04.2026).

33. Sokolova A., Hasii O., Tymoshenko O., Pedchenko N. Current trends in the development of Ukraine's insurance market in digitadization conditions. Науковий вісник Полтавського університету економіки і торгівлі. 2022. Вип. 1 (105). С. 47–60.

34. Плетенецька С., Загреба А. Особливості використання цифрових технологій у страхуванні. Вчені записки Університету «КРОК». 2025. № 4 (80). С. 52–60.

35. Лоїк Р. Блокчейн-технології та їх роль у сучасних фінансах: суть, перспективи та ризики використання. Економіка та суспільство. 2025. № 77.

36. Мутерко Г. М., Кучерівська С. С., Яцко М. В., Малець В. В. Впровадження блокчейн-технологій в економіці України: переваги та виклики. Академічні візії. 2023. № 26.

37. Swiss Re Institute. Sigma 1/2024: World insurance: Redefining risk and resilience. 2024. URL: https://www.swissre.com/institute/research/sigma-research/sigma-2024-01.html (дата звернення: 24.04.2026).

38. Etherisc. Decentralized Insurance Protocol: Parametric flight delay insurance. 2025. URL: https://etherisc.com/products/flight-delay (дата звернення: 24.04.2026).

39. Покальчук О. Смарт-контракти страхових компаній на міжнародному ринку: досвід та перспективи. Вісник Національного університету «Львівська політехніка». Серія: Економіка та управління. 2025. № 4. С. 112–125.

40. Козьменко О. В., Кузьменко О. В. Страховий менеджмент: цифрова екосистема та інновації. Київ: Академперіодика, 2024. 210 с.

41. Munich Re. Cyber Insurance: Risks and Trends 2026. 2026. URL: https://www.munichre.com/en/solutions/reinsurance-property-casualty/cyber-solutions-from-munich-re.html (дата звернення: 24.04.2026).

42. Цимбалюк А. І. Правові погляди на технології блокчейну та смарт-контрактів в українській правовій системі. Вісник Ужгородського національного університету. Серія «Право». 2025. Вип. 90. URL: http://visnyk-pravo.uzhnu.edu.ua/article/view/342177 (дата звернення: 24.04.2026).

43. Accenture. Insurance Revenue Landscape 2025: Winning the Consumer Trust. 2025. URL: https://www.accenture.com/us-en/insights/insurance/guide-insurance-revenue-landscape (дата звернення: 24.04.2026).

44. Chainlink Labs. What Is a Blockchain Oracle? 2026. URL: https://chain.link/education/blockchain-oracles (дата звернення: 24.04.2026).

45. Пилипчук О. Трансформація бізнес-моделей страхових компаній в умовах цифровізації. Економічний простір. 2025. № 188. С. 95–104.

46. Gartner. Top Strategic Technology Trends for Insurance in 2026. 2026. URL: https://www.gartner.com/en/industries/insurance-digital-transformation (дата звернення: 24.04.2026).

47. IBM. Smart contracts in insurance: Enhancing efficiency and trust. 2025. URL: https://www.ibm.com/blockchain/industries/insurance (дата звернення: 24.04.2026).

48. EY. Global Insurance Outlook 2026: Navigating macroeconomic and geopolitical shifts. 2026. URL: https://www.ey.com/en_gl/insights/insurance/five-

priorities-for-insurers-converting-uncertainty-into-opportunity (дата звернення: 24.04.2026).

ДОДАТОК А

Програмні лістинги проєкту

Проєкт завантажено у репозиторій за посиланням:

https://github.com/Bemsdemsio/claimless