Київський столичний університет імені Бориса Грінченка
Факультет інформаційних технологій та математики
Кафедра комп'ютерних наук

Допущено до захисту

Завідувач кафедри комп'ютерних наук,
Доктор техн. наук, професор
_____ Андрій БОНДАРЧУК

« ___ » _____ 202__ р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня «Магістр»
Спеціальність 122 Комп'ютерні науки
Освітня програма 122.00.02 Інформаційно-аналітичні системи

Тема: Смарт-контракти для автоматизації бізнес-процесів в сфері оренди житла

Виконав
студент групи ІАСм-1-24-1.4д
Радука Олександр Олегович
_____

Науковий керівник:
канд. технічних наук,
доцент Машкіна І.В,
_____

Київ – 2025

Київський столичний університет імені Бориса Грінченка
Факультет інформаційних технологій та математики
Кафедра комп'ютерних наук

«Затверджую»

Завідувач
кафедри комп'ютерних наук,
канд. техн. наук, доцент

_____ Ірина МАШКІНА
(підпис)

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ МАГІСТРА

«Смарт-контракти для автоматизації бізнес-процесів в сфері оренди житла»

Виконавець – студент групи
спеціальності 122 Комп'ютерні наук,
освітньої програми 122.00.02 Інформаційно-аналітичні системи
Радука Олександр Олегович

1. Вихідні дані: *Публікації за напрямом дослідження, нормативно-правова база України щодо цифрових активів, технічна документація стандартів EIP та ERC.*
2. Основні завдання: *Здійснити огляд наукових публікацій, існуючих аналогів та законодавства у сфері смарт-контрактів. Обрати та обґрунтувати методи і засоби дослідження. Підготувати матеріали до розділів роботи відповідно до індивідуального завдання. Провести декомпозицію та аналіз бізнес-процесів оренди житла. Обґрунтувати вибір гібридної архітектури та технологічного стеку. Розробити смарт-контракти для автоматизації ескроу та механізм конфіденційної верифікації користувачів. Виконати програмну реалізацію серверної та клієнтської частин платформи. Провести розгортання та тестування системи у тестовій мережі Sepolia. Оцінити ефективність запропонованого рішення.*
3. Пояснювальна записка: *Обсяг – 68 стор. формату А4 комп'ютерного набору з дотриманням вимог стандарту і методичних рекомендацій кафедри.*
4. Графічні матеріали: *Презентація.*
5. Додатки: Додаток А. Код смарт-контракту, Додаток Б. Конфігурація середовища, Додаток В. Скрипт розгортання, Додаток Г. Серверна частина, Додаток І. Гаманець, Додаток Д. Клієнтська частина.
6. Строк подання роботи на кафедру: *«___» _____2025 р.*

Науковий керівник
канд. техн. наук, доцент
_____ Машкіна І.В.
дата

Виконавець:
_____ Радука О.О.
дата

Календарний план

№	Назва етапу дипломної роботи	Строк виконання етапу роботи	Примітка
1	Формування теми дипломної роботи магістра	03.11.2025 - 05.11.2025	виконано
2	Ознайомлення з методичними рекомендаціями до дипломної роботи	06.11.2025 - 07.11.2025	виконано
3	Складання змісту та календарного плану роботи	08.11.2025 - 10.11.2025	виконано
4	Підбір літератури, складання завдання	11.11.2025 - 14.11.2025	виконано
5	Написання реферату та вступу	15.11.2025 - 18.11.2025	виконано
6	Написання першого розділу	19.11.2025 - 24.11.2025	виконано
7	Написання другого розділу	25.11.2025 - 30.11.2025	виконано
8	Написання третього розділу	01.12.2025 - 06.12.2025	виконано
9	Написання висновків	07.12.2025 - 08.12.2025	виконано
10	Виправлення зауважень	09.12.2025 - 10.12.2025	виконано
11	Створення презентації	11.12.2025	виконано
12	Захист матеріалів дипломної роботи на засіданні кафедри	12.12.2025	виконано
13	Офіційний захист матеріалів дипломної роботи на засіданні екзаменаційної комісії	19.12.2025	виконано

Здобувач Радука О.О._____
Керівник роботи Машкіна І.В _____

РЕФЕРАТ магістерської роботи

Дипломна робота : 68 с., 8 рис., 2 табл., 19 літературних джерел, 6 додатків.

Магістерську роботу присвячено розробці та впровадженню гібридної платформи для короткострокової оренди житла. Актуальність теми зумовлена тим, що у 2025 році ринок оренди в Україні переживає значні трансформації через нові законодавчі вимоги та цифровізацію. Традиційні договори часто супроводжуються високими комісіями посередників, затримками платежів та кризою довіри, тоді як блокчейн-технології дозволяють вирішити ці проблеми шляхом автоматизації. Впровадження смарт-контрактів створює передумови для мінімізації людського фактору, прискорення розрахунків та забезпечення безпеки угод, що відповідає потребам сучасної економіки.

Об’єктом дослідження є бізнес-процеси у сфері оренди житла, а предметом — методи, моделі та технології розробки і впровадження смарт-контрактів для автоматизації укладання та контролю орендних угод.

Мета роботи-обґрунтування та розробка смарт-контрактів для автоматизації ключових бізнес-процесів оренди задля підвищення прозорості, безпеки та ефективності взаємодії між орендодавцем і орендарем. Для досягнення поставленої мети було вирішено низку завдань: проаналізовано існуючі бізнес-процеси та їхні проблемні точки, досліджено сучасні технології блокчейну, розроблено модель процесів оренди з використанням смарт-контрактів, створено прототип для автоматизації бронювання й оплати, а також надано рекомендації щодо впровадження технології.

У результаті виконання роботи було реалізовано технологічне рішення, що базується на використанні смарт-контрактів на EVM-сумісній мережі.

Смарт-контракт, написаний мовою Solidity, виступає в ролі автоматизованого ескроу-агента, який надійно утримує орендну плату та заставу у стейблкоїнах до завершення угоди. Цей Web3-компонент поєднано з традиційним бекендом на Node.js та фронтендом на React, що забезпечують швидкий інтерфейс користувача. Такий гібридний підхід дозволив поєднати надійність та прозорість блокчейн-платежів зі швидкістю та зручністю звичних веб-додатків.

Ключові слова: Смарт-контракти (Smart contracts),автоматизація бізнес-процесів (Business process automation), технологія Blockchain, Solidity, Web3

ВСТУП

У сучасному цифровому світі блокчейн-технології та смарт-контракти революціонізують бізнес-процеси, забезпечуючи автоматизацію, прозорість і надійність угод без посередників. Сфера оренди житла, де традиційні договори часто супроводжуються затримками платежів, спорами щодо умов та людським фактором, є ідеальним полем для впровадження смарт-контрактів. Ці самовиконувані програми, записані в блокчейні, автоматично ініціюють платежі орендної плати, контроль доступу до помешкання чи розрахунок комунальних послуг за об'єктивними даними лічильників.

У 2025 році правова основа застосування смарт-контрактів в Україні перебуває в активній стадії становлення, що створює передумови для глибокої цифровізації ринку нерухомості. Експерти прогнозують, що після завершення воєнних дій ринок короткострокової оренди зіткнеться з піковим попитом, який потребуватиме миттєвої обробки угод. В умовах очікуваного зростання попиту блокчейн-технології виступають ефективним інструментом автоматизації: вони здатні нівелювати ризики, пов'язані з затримками платежів та вирішенням спорів. Розробка прототипів децентралізованих реєстрів та інтеграція смарт-контрактів дозволить вирішити проблеми довіри, забезпечити динамічне керування орендою в умовах нестабільної кон'юнктури ринку.

Використання смарт-контрактів у оренді житла дозволяє усунути посередників, зменшити адміністративні витрати та мінімізувати ризики невиконання зобов'язань, наприклад, через автоматичне списання коштів у встановлені терміни. Інтеграція з системами "розумного дому" та Інтернетом речей може блокувати доступ до приміщення у разі прострочки, забезпечуючи безпеку для орендодавців. Такий підхід підвищує довіру сторін і прогнозованість процесів, перетворюючи орендні відносини на ефективну цифрову екосистему. Ця робота присвячена розробці та впровадженню гібридної платформи для короткострокової оренди житла.

Впровадження смарт-контрактів у сферу оренди житла створює передумови для мінімізації людського фактору. Зростання популярності цифрових платформ для оренди та розвиток екосистем Web3 підсилюють потребу в дослідженні інноваційних підходів до управління орендними відносинами. Тому аналіз можливостей, викликів та практичних аспектів застосування смарт-контрактів у цій сфері є актуальним і відповідає потребам сучасної економіки та ринку нерухомості.

Обєкт дослідження: Бізнес-процеси у сфері оренди житла.

Предмет дослідження: Методи, моделі та технології розробки і впровадження смарт-контрактів для автоматизації процесів укладання, виконання та контролю орендних угод.

Мета роботи: Обґрунтувати та розробити смарт-контракти для автоматизації ключових бізнес-процесів у сфері оренди житла з метою підвищення прозорості, безпеки, ефективності взаємодії та мінімізації ризиків між орендодавцем і орендарем. Це допоможе вирішенню ключових проблем ринку, зокрема високих комісій централізованих посередників та відсутності довіри при фінансових розрахунках.

Для досягнення мети робити необхідно вирішити наступні завадання:

- Проаналізувати існуючі бізнес-процеси у сфері оренди житла та визначити їхні проблемні точки.
- Дослідити сучасні технології блокчейну та принципи роботи смарт-контрактів.
- Розробити модель бізнес-процесів оренди житла з використанням смарт-контрактів.
- Створити прототип смарт-контрактів для автоматизації процесів бронювання, оплати, перевірки виконання умов та завершення оренди.
- Провести тестування та оцінити ефективність розроблених смарт-контрактів.

- Надати рекомендації щодо впровадження технології у реальні бізнес-процеси.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ТЕОРЕТИЧНІ ОСНОВИ ДОСЛІДЖЕННЯ.

Технологія блокчейн останнім часом спричинила значний сплеск інтересу як серед академічних кіл, так і в індустріальному секторі, оскільки вона являє собою розподілену програмну систему, що дозволяє учасникам здійснювати транзакції та обмінюватися цінностями без необхідності залучення довіреної централізованої третьої сторони. Завдяки цій децентралізованій структурі, бізнес-операції можуть бути завершені з меншими витратами та значно швидше. Додатковою перевагою, що забезпечує розподілену довіру, є незмінність блокчейнів: внесені в них дані майже неможливо підробити чи змінити, а всі історичні транзакції залишаються повністю прозорими, такими, що можна перевірити та відстежити. Сьогодні технологія блокчейн стала широко обговорюваним поняттям, головним чином завдяки унікальній технологічній основі, на якій вона функціонує. По суті, блокчейн – це безперервно зростаючий ланцюг записів транзакцій, які називаються блоками, і всі ці записи криптографічно пов'язані між собою, формуючи ланцюг, захищений за допомогою складних криптографічних методик. Кожен блок може містити один або кілька записів транзакцій, при цьому обов'язково включаючи хеш-функцію попереднього блоку для забезпечення цілісності, власні дані транзакції та позначку часу. Як тільки блок успішно валідовано, він додається до блокчейну в хронологічному порядку і після цього не може бути змінений. Ці ключові характеристики - безпека, надійність, прозорість та гарантована можливість моніторингу цифрових транзакцій - роблять технологію блокчейн надзвичайно цінною та корисною. Найактивніше блокчейн використовується в контексті криптовалют, зокрема Біткоїна та Ефіріума. Криптовалюта розглядається як віртуальна або цифрова валюта, яка використовує криптографію для максимального підвищення безпеки фінансових операцій. Таким чином, вона слугує безпечним засобом обміну, оскільки застосовує високорівневу криптографію

для забезпечення перевірності передачі активів, контролює процес створення одиниць валюти та дозволяє обходити регулятивні механізми, які можуть бути нав'язані традиційними установами, як-от урядові інститути.

Блокчейн — це розподілена програмна система (ланцюг записів транзакцій, або блоків), яка:

- Автономно зростає.
- Безпечно захищена криптографічними техніками.
- Містить блоки, які пов'язані між собою (кожен блок містить хеш-функцію попереднього блоку, дані транзакції та часову мітку).
- Після завершення та коміту блок хронологічно додається і не може бути модифікований (незмінність).

Блокчейн забезпечує надійний, прозорий і незворотний розподілений реєстр, що дозволяє здійснювати транзакції без потреби в довіреній третій стороні. Це забезпечує розподілену довіру.

1.1 Смарт-контракти та технологія Блокчейн

Виникнення передових технологій призвело до посилення конкуренції між бізнесами, оскільки кожен з них намагається використовувати останні для підвищення продуктивності працівників і загальної ефективності діяльності компанії [1, 6]. Як наслідок, технологія стала критично важливою основою організаційних операцій і основною рушійною силою інновацій та конкурентоспроможності організацій [4]. Підприємства перейшли від традиційних способів ведення бізнесу та, як наслідок, прийняли сучасніші, надійніші та більш економічно ефективні механізми; смарт контракти є одним із таких механізмів.

Смарт-контракти — це самовиконувані програми, в яких контрактні умови закодовані у вигляді комп'ютерного коду, що дозволяє автоматично виконувати угоди після виконання певних умов. Останніми роками термін «смарт» став дуже популярним: ми живемо в «розумних містах», користуємося «розумними холодильниками» чи «розумними духовками», і, звісно, не можемо обійтися без наших «смартфонів». Прикметник «смарт» означає, що функціональність об'єкта була значно покращена за допомогою програмних застосунків, тобто частину функціональності було автоматизовано. «Розумна духовка» готує їжу так само, як її застарілий аналог, але може бути увімкнена і вимкнена віддалено та контролюватися на відстані. Смартфон — це телефон, який дозволяє нам телефонувати людям, не торкаючись клавіатури, і він виконує інші цінні (хоча й повторювані) завдання за нас. У контексті смарт контракту, це означає, що положення, записані в комп’ютерних програмах, будуть автоматично виконуватись, коли будуть виконані заздалегідь визначені умови.

Концепцію смарт-контракту вперше представив американський вчений у галузі інформатики Нік Сабо у 1994 році. Його бачення полягало в автоматизації юридичних контрактів за допомогою комп'ютерного коду та криптографії, щоб:

- Виконати умови контракту (наприклад, оплату, конфіденційність).
- Мінімізувати зловмисні чи випадкові винятки.
- Зменшити потребу в довірених посередниках.
- Знизити втрати від шахрайства, витрати на арбітраж та інші транзакційні витрати.

Сабо наголошував, що для того, щоб смарт-контракти стали ціннішими для суспільства, вони повинні бути: верифікованими, спостережуваними та примусовими до виконання. Таким чином, смарт-контракти будуть частиною суспільства, що зменшить юридичні бар'єри, знизить витрати на транзакції, скоротить час на виконання контракту і створить можливості для нових типів бізнесу.

1.2 Застосування смарт контрактів, переваги і виклики

Будь-яку фінансову транзакцію, здійснену організацією з третіми сторонами, можна розглядати як форму контракту, якою б простою або складною не була транзакція. По суті, фінансова відкритість і прозорість є одними з ключових аспектів успішного управління організацією, оскільки вони створюють сприятливе середовище не лише для інвестицій, але й для встановлення довіри з різними зацікавленими сторонами організації.

Смарт-контракти сприяють розвитку широкого спектру застосувань:

- Децентралізовані Фінанси (DeFi): Кредитування, страхування, токенізація активів без посередників.
- Управління Ланцюгами Поставок: Автоматизація замовлень, доставок, умов оплати та відстеження походження товарів.
- Децентралізовані Автономні Організації (DAO): Керування організаціями через кодовані правила та права голосу.
- Інтернет Речей (IoT): Безпечне керування даними, спільне використання ресурсів та автоматизовані сервісні угоди між пристроями.

– Торгівля енергією: Одноранговi транзакції та розрахунки в розумних мережах.

Рис. 1 Розвиток застосування смарт- контрактів.

Сьогодні багато користувачів використовують смарт-контракти, не усвідомлюючи цього: безконтактні платежі банківськими картами за проїзд в

транспорті або прокат велосипедів є прикладами розгортання смарт-контрактів. Традиційно оренда велосипедів передбачала підписання фізичного документа, що роз’яснює ціну й умови оренди; заставу могли брати для покриття потенційних збитків. Документ є фізичним доказом контракту, і платіж здійснювався, коли велосипед повертався. Для порівняння, коли ми використовуємо банківські картки для безпосередньої оренди "розумних велосипедів" на станціях докування у "розумних містах" на зразок Лондона, Парижа чи Берліна, велосипед випускається, а правильна сума грошей списується з банківського рахунку, коли велосипед повертається. Умови договору автоматично управляються без потреби людського втручання. Для цього типу транзакцій існують вагомі аргументи, не в останню чергу через те, що виключення людського втручання прискорює процес і знижує витрати. Інтернет прискорив впровадження "розумних угод": стаття 9 Директиви Європейського Союзу з електронної комерції (Директива про електронну комерцію, 2000) вимагає від усіх держав-членів забезпечити, щоб їхні правові системи сприяли впровадженню електронних контрактів.Директива ЄС з комерції (Директива про електронну комерцію, 2000) використовує термін "електронний контракт", що в основному охоплює визначення смарт-контрактів, що розгорнуті у цій статті.

Розробка смарт-контрактів розпочинається з критично важливого етапу - проектування та узгодження, де всі учасники майбутнього контракту детально обговорюють та фіксують бізнес-логіку та правила, які будуть автоматично виконуватися. Після узгодження, контракт програмується, переважно із застосуванням спеціалізованих мов, таких як Solidity, Yul або Vyper, які спеціально розроблені для платформ, сумісних із віртуальною машиною Ethereum (EVM). Застосування шаблонів проектування та загальноприйнятих патернів значно спрощує процеси кодування, тестування та налагодження, що є життєво необхідним для мінімізації потенційних загроз безпеки та вразливостей у фінальному контракті. Проте, широкий спектр

різних блокчейн-платформ ускладнює створення та стандартизацію єдиних, універсальних шаблонів. Solidity є найбільш поширеною мовою; вона є високорівневою, статично типізованою та об'єктно-орієнтованою, беручи своє натхнення від таких мов, як C++, Python та JavaScript. Її архітектура спеціально оптимізована для ефективної роботи у середовищі віртуальної машини Ethereum (EVM). Типова структура смарт-контракту в Solidity включає обов'язкові специфікатори ліцензії, директиви pragma для визначення сумісної версії компілятора, змінні стану (наприклад, беззнакові цілі числа uint), які зберігають дані на блокчейні, а також функції для читання або зміни цього стану. Наступний етап — компіляція смарт-контрактів, під час якої високорівневий код (наприклад, на Solidity) трансформується у байт-код EVM за допомогою компілятора Solidity (SOLC), який, у свою чергу, інтерпретується EVM під час виконання. Сучасні середовища розробки, такі як Remix Integrated Development Environment (IDE), надають розробникам потужний інструментарій, що охоплює написання коду, його компіляцію, налагодження та безпосереднє розгортання смарт-контрактів. Інтеграція з інструментами, як-от MetaMask, забезпечує можливість прямої взаємодії з тестовими чи основними мережами Ethereum. Розгортання є фінальним кроком, який здійснюється виключно після ретельного та вичерпного тестування, оскільки ключовою особливістю смарт-контракту є його незмінність (*immutability*) після розміщення на блокчейні: код контракту стає постійним і не може бути змінений. Змінюватися можуть лише змінні стану та локальні змінні, тоді як основна логіка та структура залишаються фіксованими. Це означає, що виправлення помилок чи усунення критичних багів, виявлених вже після розгортання, вимагає створення та перерозгортання абсолютно нової версії контракту, що завжди тягне за собою додаткові транзакційні витрати (газ). Виконання смарт-контрактів активується або надходженням зовнішніх транзакцій, або при досягненні заздалегідь визначених умов, і завжди відбувається автоматично та детерміновано. EVM

гарантує, що результат виконання контракту завжди буде однаковим для всіх валідаторів мережі, а механізм газу використовується для регулювання та пріоритезації цього виконання. Газ функціонує як міра обчислювального ресурсу, необхідного для виконання операцій, і користувачі зобов'язані сплачувати комісію за газ пропорційно до складності та обчислень, потрібних для виконання коду контракту. Наприклад, проста транзакція переказу Ethereum фіксовано коштує 21,000 одиниць газу, тоді як мінімальна вартість розгортання абсолютно нового контракту становить щонайменше 32,000 газу, при цьому загальні витрати на розгортання складних контрактів можуть легко перевищувати 200,000 одиниць газу. Це наочно демонструє, що складність і обсяг логіки контракту безпосередньо корелюють зі споживанням газу та фінальною вартістю операції.

Смарт-контракти мають наступні переваги в порівнянні з традиційними контрактами :

Безпека. Завдяки незмінності блокчейну та криптографічній валідації смарт-контракти пропонують надійну безпеку.

Смарт-контракт, який був впроваджений, неможливо змінити, що гарантує виконання умов без можливості маніпуляцій. Це особливо корисно в таких галузях, як фінанси, де безпека є надзвичайно важливою. Смарт-контракти, наприклад, можуть автоматизувати видачу кредитів і погашення на платформах однорангового кредитування, гарантуючи належне дотримання угод без небезпеки людської помилки чи втручання (Swan, 2015).

Смарт-контракти надзвичайно стійкі до шахрайства та злому, оскільки вони захищені за допомогою криптографічних методів. Розподіляючи свої дані по великій мережі вузлів, децентралізована структура контракту додає ще більше безпеки.

Прозорість. Усі сторони можуть незалежно перевірити умови та

виконання смарт-контракту без допомоги третьої сторони, переглядаючи код і те, як контракт виконується в блокчейні.

Традиційні контракти часто вимагають певного рівня довіри між сторонами та незалежними третіми сторонами, як-от нотаріуси чи юристи. Наприклад, щоб переконатися, що умови працевлаштування є справедливими та відповідають законодавству, як роботодавець, так і працівник можуть покладатися на юрисконсульта в трудовому договорі. Формулювання цих контрактів може бути неоднозначним або відкритим для різних тлумачень, що може призвести до розбіжностей.

З іншого боку, смарт-контракти забезпечують прозорість, дозволяючи всім сторонам перевіряти умови через публічний реєстр блокчейну. Смарт-контракти, наприклад, прозоро контролюють кредитну та позикову діяльність на платформах децентралізованих фінансів (DeFi), де всі умови доступні в блокчейні. Завдяки своїм незмінним записам і криптографічній безпеці, вони пропонують високий ступінь довіри, одночасно знижуючи можливість втручання.

Можливість моніторингу. Гарантується можливість моніторингу цифрових транзакцій. Самовиконувані угоди: Смарт-контракти створені для автоматичного початку роботи, щойно будуть виконані певні вимоги. Умови контракту жорстко закодовані в блокчейні, що усуває потребу в посередниках і знижує ймовірність людської помилки.

Децентралізація. Оскільки смарт-контракти працюють у децентралізованій мережі блокчейн, вони не керуються однією стороною. Цілісність контракту зберігається на численних вузлах, що підвищує безпеку та надійність завдяки цій децентралізації.

Ефективність. Смарт-контракти кардинально скорочують час і витрати, пов'язані з традиційним управлінням контрактами, завдяки автоматизації процесу виконання. Ця ефективність може зменшити адміністративний тягар та оптимізувати процеси.

Через необхідність обговорень, юридичної експертизи та примусового виконання, традиційні контракти можуть бути дорогими та займати багато часу. Наприклад, переговори та оплата юридичних послуг можуть тривати місяцями при підготовці угоди про корпоративне злиття (Scott & Kraus, 2013). Вони тягнуть за собою можливість затримок у виконанні та вирішенні спорів, а також адміністративні витрати.

Завдяки автоматизації виконання контрактів і скороченню часу та витрат, пов'язаних з операціями, що виконуються вручну, смарт-контракти підвищують ефективність. Значна економія коштів є результатом того, що ця автоматизація зменшує адміністративний тягар і прискорює транзакції (Werbach, 2018). Наприклад, після підтвердження інформації про рейс через API, смарт-контракт може негайно ініціювати врегулювання претензії щодо затримки рейсу в страховій галузі, усуваючи необхідність ручної обробки претензій.

Економія коштів. Смарт-контракти можуть призвести до значної економії коштів завдяки автоматизації виконання контрактів та усуненню потреби в посередниках.

Звичайні контракти іноді включають тривалі процеси переговорів, значні юридичні витрати та адміністративні накладні витрати. Ці процедури оптимізуються смарт-контрактами, що знижує час і витрати на адміністрування контрактів [11, 12]. Наприклад, місяці переговорів і дорогі юридичні витрати зазвичай пов'язані з переговорами щодо угоди про корпоративне злиття. Багато з цих процесів може бути автоматизовано смарт-контрактом, що робить транзакції швидшими та доступнішими (Robert, Cooter Jr., & Ulen, 2011).

Ці можливості демонструють, як смарт-контракти здатні революціонізувати низку секторів. Очікується, що використання смарт-контрактів зростатиме в міру зміни законодавства та прогресу технологій, стимулюючи ефективність та інновації в комерційних угодах.

Відкриті виклики. Існує багато перешкод для інтеграції смарт-контрактів, які впливають як на суспільство, так і на користувачів. Оскільки багато юрисдикцій ще не визнають смарт-контракти юридично обов'язковими, користувачі ризикують залишитися без засобів правового захисту. Правова та регуляторна невизначеність залишається основною проблемою (Allen & Hunn, 2022). Технічні недоліки, такі як проблеми масштабованості та помилки коду, можуть мати несподівані наслідки або призвести до неефективності.

Ще однією складністю при створенні та читанні смарт-контрактів є їхня комплексність; перетворення складних юридичних угод на точний код вимагає високого рівня кваліфікації та особливої уваги до деталей. Через негнучку природу блокчейну та вразливість смарт-контрактів до хакерів, міркування безпеки мають найбільше значення (Atzei, Bartoletti, & Cimoli, 2017).

Нарешті, фінансові фактори є вирішальними. Створення, впровадження та управління смарт-контрактами може бути дорогим, особливо для малих і середніх підприємств, а деякі мережі блокчейн можуть стягувати надмірні комісії за транзакції. Щоб повною мірою використати переваги смарт-контрактів, мінімізуючи їхні ризики, гравці галузі, законодавці, юристи та інженери повинні співпрацювати для вирішення цих складних питань.

Технологічні виклики. Незважаючи на всі свої фундаментальні переваги у сфері автоматизації та децентралізації, смарт-контракти все ще стикаються з численними технологічними викликами, які необхідно ефективно вирішити, перш ніж вони зможуть досягти масштабного та широкого впровадження у ключових галузях промисловості. Однією зі значних труднощів є жорсткість і негнучкість коду (*code inflexibility*). Смарт-контракти, за своєю природою, можуть виконувати тільки ті умови та інструкції, які були абсолютно точно прописані в їхній логіці. Це може призвести до серйозних проблем у непередбачуваних ситуаціях або при виникненні юридичних чи бізнес-колізій, які розробники не змогли передбачити на етапі проєктування.

Класичним і найбільш відомим прикладом цієї вразливості стала подія, відома як «злам DAO» на блокчейні Ethereum у 2016 році. У цьому випадку була експлуатована логічна вразливість у коді смарт-контракту децентралізованої автономної організації (DAO), що призвело до значної втрати активів. Цей інцидент наголосив на критичній небезпеці помилок кодування та ілюструє, як навіть невеликі баги можуть мати катастрофічні та незворотні наслідки у незмінному середовищі блокчейну. Крім того, точність і надійність вхідних даних є абсолютно вирішальною для коректного виконання смарт-контрактів. Виконання контракту може бути скомпрометовано або призвести до неправильних результатів, якщо дані, які він споживає, є помилковими, неточними або підробленими. Наприклад, у високоавтоматизованому сценарії ланцюга поставок, смарт-контракт може помилково переказати великий платіж завчасно або не тій стороні, якщо в нього було введено неправильні або сфальсифіковані дані про статус доставки товару. Ця необхідність використання зовнішніх джерел даних, відомих як «оракули» (*oracles*), додає додатковий рівень складності, ризику та довіри до загальної архітектури, оскільки потрібно довіряти не лише контракту, але й джерелу, яке надає йому інформацію. Необхідність ретельного тестування та незалежного аудиту безпеки є життєво важливою для гарантування безпечності та надійності смарт-контрактів перед їх розгортанням.

Незважаючи на потенційну здатність смарт-контрактів заощаджувати адміністративні витрати та підвищувати продуктивність у довгостроковій перспективі, ресурсомісткий характер створення, верифікації та управління абсолютно безпечними смарт-контрактами залишається значною початковою перешкодою. Для повного розкриття трансформаційного потенціалу цієї технології необхідно сфокусувати зусилля на усуненні цих перешкод. Окрім проблем з безпекою та точністю даних, існують також технічні обмеження самої базової інфраструктури. Масштабованість залишається гострим питанням, оскільки публічні блокчейни, такі як Ethereum, часто мають

відносно низьку пропускну здатність за кількістю транзакцій на секунду, оскільки кожна транзакція має бути оброблена та підтверджена кожним вузлом у мережі, що створює вузьке місце, відоме як проблема «трилеми блокчейну». Це активно вирішується за допомогою таких рішень, як шардинг (поділ блокчейну на менші частини) та протоколи другого рівня (*Layer 2 solutions*), такі як Rollups (Optimistic та ZK), які обробляють транзакції поза основним ланцюгом, значно знижуючи навантаження на головний блокчейн. Іншим важливим обмеженням є міжопераційність (*interoperability*), оскільки наразі існує обмежена або складна взаємодія між смарт-контрактами, розгорнутими на різних, несумісних між собою платформах блокчейну. Міжопераційність між гетерогенними платформами блокчейн обмежується відмінностями у фундаментальних елементах, таких як механізми консенсусу (наприклад, Proof-of-Work проти Proof-of-Stake), мовами смарт-контрактів та структурами даних, створюючи так звані «ізольовані ділянки» (*silos*) та ускладнюючи операції між ланцюгами (*cross-chain operations*). Активні дослідження зосереджені на розробці моделей виконання через ланцюги (*cross-chain execution*) та мостових протоколів (*bridge protocols*) для полегшення безперешкодної взаємодії та співпраці між цими різними мережами блокчейну. Для вирішення проблем із прозорістю, властивою публічним блокчейнам, техніки збереження конфіденційності активно інтегруються у смарт-контракти. Зокрема, використання нульових доказів (*Zero-Knowledge Proofs*, наприклад, zk-SNARKs) та захищених багатосторонніх обчислень (*Secure Multi-Party Computation*, MPC) дозволяють проводити верифікацію транзакцій та обчислень без необхідності розкривати чутливі дані самого контракту або учасників. Моделі виконання поза ланцюгом (*Off-chain execution models*), такі як Secure Remote Procedure (SRP) і Incentive-Compatible Off-chain Execution (ICOE), додатково зменшують споживання дорогоцінних ресурсів у ланцюзі, підвищуючи як масштабованість, так і конфіденційність. Проте, варто зазначити, що складність цих криптографічних протоколів та процесів захисту

конфіденційності може, у свою чергу, обмежити пропускну здатність та ефективність транзакцій. Вразливості безпеки та обчислювальні витрати залишаються постійними проблемами, оскільки незмінність смарт-контрактів після розгортання робить будь-які помилки постійними та потенційно призводить до значних фінансових втрат. Для протидії цьому розробляються формальні фреймворки перевірки, такі як ZEUS та Securify, які використовують математичні моделі та автоматизований статичний аналіз для суворої перевірки правильності логіки та виявлення вразливостей у коді смарт-контрактів ще до розгортання. Автоматизовані інструменти аудиту та семантичні фреймворки аналізу додатково підтримують ідентифікацію та зменшення ризиків безпеки. Окрім технічних аспектів, виникають правові, етичні та управлінські розгляди, пов'язані з децентралізованою та автоматизованою природою смарт-контрактів. Ці фактори кидають виклик традиційним принципам контрактного права, піднімаючи складні питання щодо їхньої юридичної можливості виконання та вирішення спорів за повної відсутності центральних регуляторних органів. Майбутні напрямки досліджень включають оптимізацію механізмів консенсусу, покращення ефективності використання даних, зниження затримок у мережі та розробку надійних фреймворків управління (*governance frameworks*) для забезпечення автоматизованого та відповідального дотримання контрактів.

1.3 Нормативно-правове регулювання. Аналіз чинного законодавства України щодо цифрових активів та смарт-контрактів.

Коли сторони укладають транзакцію, смарт-контракти слугують інноваційним та автоматизованим механізмом для створення правових відносин між ними, оскільки вони цифровим чином кодують і фіксують всі права, обов'язки та відповідальність учасників. У цій децентралізованій екосистемі кожен залучений суб'єкт — розробники (які несуть відповідальність за безпеку коду), кінцеві користувачі, контрагенти та сторонні постачальники послуг (наприклад, оракули, що надають зовнішні дані) — несе

певні юридичні обов'язки та зобов'язання. Питання юридичної відповідальності стає критично важливим у разі виникнення розбіжностей, порушень, помилок коду або непередбачуваних зовнішніх подій; тому ретельне врахування правових аспектів є обов'язковою передумовою при проєктуванні та впровадженні будь-якого смарт-контракту. З юридичної точки зору, вкрай важливо розуміти, що смарт-контракти є насамперед технологічною інновацією — це інструмент, який використовує блокчейн для автоматизації та оптимізації виконання вже існуючих договірних відносин, а не являють собою фундаментальний відрив від традиційного договірного права. Це розуміння передбачає, що правовий аналіз смарт-контрактів полягає у вивченні їхньої керівної логіки та їхніх наслідків у контексті усталеної системи юриспруденції. Більшість юрисдикцій схиляються до того, щоб розглядати смарт-контракти як нову форму вираження юридично обов'язкової угоди, якщо в коді зафіксовані ключові елементи контракту: пропозиція (оферта), прийняття (акцепт) та зустрічне задоволення (compensation). На відміну від цього, традиційні контракти — це, як правило, усні або письмові угоди, які ґрунтуються на текстових документах і можуть багаторазово обговорюватися та змінюватися в процесі переговорів. Наприклад, класичний контракт на купівлю-продаж нерухомості передбачає ретельні обговорення умов, часто за посередництва агентів з нерухомості та юристів, і завершується фізичними або кваліфікованими електронними підписами покупця та продавця. Ці контракти покладаються на ручні процедури примусового виконання, що часто вимагає залучення судових позовів у разі виникнення суперечок, і вимагають фізичних підписів або інших форм підтвердження, щоб стати юридично обов'язковими. Наприклад, у разі порушення будівельної угоди, постраждала сторона змушена подати судовий позов для витребування засобів правового захисту, що є процесом повільним, дорогим та залежним від зовнішнього арбітражу (суду). Смарт-контракти, навпаки, є цифровими контрактами у формі коду, який зберігається та виконується безпосередньо в

блокчейні. Їхня ключова відмінність полягає в тому, що вони автоматично виконують заздалегідь визначені дії щойно будуть виконані зазначені вимоги (умови), забезпечуючи детермінізм (однозначність результату). У системі управління ланцюгом поставок, наприклад, смарт-контракт може бути закодований таким чином, щоб автоматично перерахувати платіж постачальнику, як тільки відстеження в блокчейні підтвердить доставку товарів до місця призначення. Смарт-контракти виключають потребу у фізичних підписах, покладаючись виключно на криптографічну валідацію (підтвердження за допомогою приватних ключів) для ідентифікації та намірів сторін. Самовиконувана та самозабезпечувана природа смарт-контрактів — їхня здатність автоматично здійснювати примусове виконання завдяки коду — усуває потребу в посередниках (для виконання) та ручному примусовому виконанні. Це робить виконання угод негайним, ефективним та знижує ризик затримки чи недобросовісності, проте створює юридичну проблему щодо того, як змінити або скасувати угоду в разі помилки, оскільки код є незмінним.

Станом на листопад 2025 року правовий ландшафт для смарт-контрактів в Україні перебуває у перехідному стані та на активній стадії формування. Хоча очевидно, що технологія блокчейн та смарт-контракти мають величезний потенціал для автоматизації економічних та правових відносин, комплексного та спеціалізованого законодавчого акту, який би вичерпно визначав та регулював саме поняття смарт-контракту як унікальної форми юридичного інструменту, наразі не прийнято [13].

Правовою основою для смарт-контракту виступає правочин — дія особи, спрямована на набуття, зміну або припинення цивільних прав і обов'язків, який вчиняється через мережу «Інтернет» і може бути кваліфікований відповідно до частини 1 статті 202 Цивільного кодексу України (ЦК України). Для того, щоб смарт-контракт вважався дійсним правочином і укладеним договором, він має відповідати всім вимогам закону, зокрема загальним вимогам, необхідним для чинності правочину згідно зі статтею 203

ЦК України (що включає вільне волевиявлення, відповідність закону, здатність сторін тощо). Основні законодавчі засади, на які спирається ця сфера, були закладені у 2022 році із прийняттям Закону України «Про віртуальні активи». Цей Закон був сфокусований насамперед на визначенні статусу, обігу та правових аспектів віртуальних активів (як-от криптовалюти та токени) та їх оподаткуванні. Проте, він не надав чіткого та прямого правового визначення смарт-контракту, залишивши його регулювання у площині аналогії з чинним цивільним законодавством.

Ключовим моментом для набуття юридичної сили є те, що договір вважається укладеним згідно з частиною 1 статті 638 ЦК України, якщо сторони в належній формі досягли згоди з усіх істотних умов. Отже, визнання смарт-контракту юридично зобов'язувальним договором є найважливішим для забезпечення того, щоб результати самостійного виконання «розумного контракту» підлягали примусовому виконанню та правовому захисту. Якщо, наприклад, «розумний контракт» не може законно передавати право власності на криптоактив або якщо в ньому відсутні всі істотні умови договору купівлі-продажу чи страхування (наприклад, предмет, ціна), смарт-контракт може бути визнаний недійсним або неукладеним.

Правова природа смарт-контракту в Україні інтерпретується як автоматизована, програмна форма цивільно-правового договору — сукупність програмного коду, що виконується детерміновано в розподіленій мережі. Законодавство України, зокрема Закон «Про електронну комерцію» та стаття 626 Цивільного кодексу України, визнає юридичну силу електронного правочину за умови, що сторони належно ідентифіковані, а їхня воля та згода на умови угоди зафіксовані (наприклад, шляхом використання криптографічних електронних підписів або підтвердження транзакції приватним ключем). Це дозволяє прирівнювати смарт-контракти, які відповідають цим вимогам, до традиційних письмових договорів.

Незважаючи на цю можливість застосування за аналогією, існують суттєві обмеження: хоча смарт-контракти можуть використовуватися для внутрішнього обліку, взаєморозрахунків та автоматизації процесів між сторонами угоди, державні реєстри та офіційні інституції (наприклад, реєстрація прав власності) не визнають напряму результати виконання цих контрактів. Це означає, що смарт-контракти та токени, які вони генерують, не породжують прямих юридичних наслідків для третіх осіб чи держави у правовідносинах, що потребують державної реєстрації.

Прихильники визнання смарт-контракту різновидом договору, укладеного в електронній формі, аргументують це тим, що електронною формою є електронний документ, у якому сторони закріплюють істотні умови договору і який може бути візуально сприйнятий людиною. Оскільки сторони не завжди мають можливість перевірити коректність трансформації текстових умов договору у програмний код, вони вважають необхідним збереження електронної форми договору як основи для формування коду. У разі виникнення спорів та виявлення технічних помилок чи спотворення інформації в коді, саме ця електронна форма може слугувати доказом у суді. При цьому не виключається можливість залучення судових експертів для аналізу інформації, що міститься в програмному коді.

Опоненти такого підходу зазначають, що однозначно не можна розглядати смарт-контракт як різновид договору, укладеного виключно в електронній формі, оскільки, відповідно до пункту 3 статті 3 Закону України «Про електронну комерцію», електронною формою подання інформації вважається документування інформації, що дає змогу її відтворювати у візуальній формі, придатній для сприйняття людиною. Вони стверджують, що програмний код, який є сутністю смарт-контракту, не завжди відповідає цій вимозі візуального сприйняття та розуміння людиною без спеціалізованих знань, що не узгоджується з традиційною правовою природою електронного документу.

Повноцінне впровадження смарт-контрактів стримується низкою невирішених проблем:

Необхідність законодавчого визначення. Відсутність чітко закріпленого поняття смарт-контракту створює правову невизначеність, ускладнюючи розробку стандартних процедур.

Доказовий статус блокчейну. Потребує унормування доказова сила інформації, зафіксованої у блокчейні, у судовому процесі. Необхідно розробити протоколи, які б дозволяли належним чином верифікувати та приймати як докази код контракту, дані транзакцій та мітки часу.

Відповідальність у разі збою. Виникають складні питання щодо відповідальності сторін у випадку помилок коду (багів), збоїв в оракулах або непередбачених технічних ситуаціях. Традиційні норми про порушення договору можуть бути недостатньо адаптовані до автоматизованого, самовиконуваного характеру смарт-контрактів, де людське втручання мінімізовано.

Судова практика. Судова практика щодо спорів, пов'язаних зі смарт-контрактами, в Україні залишається обмеженою та не має сталих підходів, що ускладнює прогнозування судових рішень та збільшує юридичні ризики для учасників.

Правова база криптовалют в Україні.

Правовий режим криптовалют в Україні на кінець 2025 року визначається насамперед Законом України «Про віртуальні активи», який створив початкову основу для легалізації цієї сфери. Водночас, ключові регуляторні зміни очікуються вже з 2026 року завдяки новій законодавчій реформі, над якою активно працює Верховна Рада. Ця реформа має остаточно унормувати статус, обіг та оподаткування цифрових активів, що є критично важливим для інтеграції в міжнародний фінансовий простір.

Основним законодавчим актом є Закон «Про віртуальні активи» (2021–2022), який легалізував обіг віртуальних активів, визнавши

криптовалюту майном або нематеріальним благом, яким можна володіти та розпоряджатися. Цей закон створив умови для легалізації роботи криптобірж, захисту прав власників та офіційної реєстрації компаній у криптосфері. Наразі у Верховній Раді на опрацюванні перебуває низка законопроєктів (зокрема, №10225 та №10225-1), які мають на меті повністю врегулювати обіг, оподаткування та діяльність криптовалютних компаній. Проєкт №10225-1, підготовлений Міністерством цифрової трансформації, також спрямований на адаптацію українського законодавства до європейського регламенту MiCA (Markets in Crypto-Assets), впровадження якого очікується з 1 січня 2026 року.

Згідно з позицією Національного банку України (НБУ), підтвердженою ще у 2014 році, та чинним законодавством, криптовалюта НЕ є законним платіжним засобом на території України, оскільки є грошовим сурогатом і не є офіційною грошовою одиницею (гривнею). Її офіційний статус визначений як цифровий актив — нематеріальне благо, що є об'єктом цивільних прав і має вартість. Цей вибір — визнання криптовалюти майном (об'єктом цивільних прав) — спрощує інтеграцію, оскільки дозволяє застосовувати вже врегульовані норми Цивільного кодексу України. Водночас, використання її як традиційного платіжного засобу вимагало б масштабної законодавчої роботи та перегляду законів про платіжні системи.

Криптовалюта, на відміну від фіатних грошей, характеризується як цифровий код, що генерується складними математичними алгоритмами; їй притаманна децентралізованість, анонімність учасників та ринкове формування курсу без державного контролю. Законодавство дозволить офіційне купівлю і продаж криптовалюти компаніями, декларування доходів та їх оподаткування, а також запровадить ліцензування та регуляторний нагляд для криптобірж і компаній, що надають послуги, пов'язані з обігом віртуальних активів.

Нові правила оподаткування, які набудуть чинності з 2026 року, мають замінити поточні загальні ставки (18% ПДФО та 1,5% військовий збір як

іноземний дохід) на пільгову ставку (наприклад, 5% для фізичних осіб). Це має на меті стимулювати легалізацію. Наразі існує проблема з декларуванням доходів, отриманих у криптовалюті без її конвертації у фіат, оскільки НБУ не публікує офіційних курсів до криптовалют, що унеможливлює перерахунок доходу у гривню для подання декларації за чинною формою. Нові законопроєкти покликані вирішити цю колізію та забезпечити прозорі податкові зобов'язання для бізнесу.

Правовий статус криптовалют у світі не є єдиним: США розглядають її як майно, Японія — як законний засіб платежу, а ЄС працює над уніфікованим регламентом MiCA [14, 15]. Адаптація українського законодавства до стандартів ЄС є стратегічним кроком. Повноцінна інтеграція у фінансову систему очікується після ухвалення необхідних законодавчих змін, що дозволить українському бізнесу повною мірою скористатися перевагами смарт-контрактів та цифрових активів **[16, 17]**.

1.4 Етапи та методи дослідження.

Послідовність дослідження для досягнення поставленої мети є логічним процесом, що складається з чотирьох основних етапів [18, 19].

1. Теоретико-аналітичний етап: глибокий аналіз поточного стану автоматизації бізнес-процесів та теоретичних основ технології блокчейн, смарт-контрактів і їхньої правової бази, зокрема в контексті України.
2. Проєктно-модельний етап: на основі аналізу визначаються ключові технічні та функціональні вимоги до системи, проводиться порівняльна оцінка та вибір оптимальної технологічної платформи (EVM-сумісна мережа), та розробляється концептуальна модель автоматизації бізнес-процесу оренди.
3. Практично-реалізаційний етап: розробка алгоритму впровадження та створення програмного прототипу. Він охоплює

написання смарт-контракту на Solidity для функції ескроу та розробку гібридної архітектури (Web2+Web3) з використанням Node.js та React, як визначено у завданні.

4. Оціночно-рекомендаційний етап: Завершальний етап присвячений всебічному тестуванню розробленого прототипу для оцінки його ефективності, оцінці економічного ефекту від впровадження та формулюванню фінальних рекомендацій щодо інтеграції, безпеки та подальшого розвитку системи.

Висновки до першого розділу.

У першому розділі досліджено теоретичні засади та правове поле використання технології блокчейн. Встановлено, що смарт-контракти трансформують економічні відносини, замінюючи інституційну довіру програмним кодом, що дозволяє автоматизувати виконання угод та мінімізувати участь посередників. Аналіз нормативної бази України станом на 2025 рік засвідчив активне формування законодавства про віртуальні активи, проте правовий статус смарт-контрактів як цивільно-правових договорів все ще потребує гармонізації з європейськими стандартами MiCA. Визначено, що для досягнення мети роботи найефективнішим є поєднання теоретичного аналізу з практичним моделюванням програмної системи.

РОЗДІЛ 2. АНАЛІЗ БІЗНЕС-ПРОЦЕСІВ ТА ВИБІР ТЕХНОЛОГІЇ РЕАЛІЗАЦІЇ

2.1 Визначення та декомпозиція актуальних бізнес-процесів оренди.

В рамках теми можна логічно розділити на три послідовні фази: верифікація учасників, укладення та фінансове забезпечення угоди, а також виконання та врегулювання.

Етап 1: Ініціація та Верифікація Учасників

Першочерговим процесом є ідентифікація орендаря. Для забезпечення приватності використано підхід **Off-chain Verification with On-chain Proof**. Серверна частина виконує перевірку документів користувача і, у разі успіху, генерує унікальний цифровий підпис (згідно зі стандартом EIP-712). Користувач передає цей підпис у смарт-контракт, який математично перевіряє його валідність. Цей метод емулює властивості Zero-Knowledge (доведення факту без розкриття даних), але є значно ефективнішим для реалізації у прототипі.

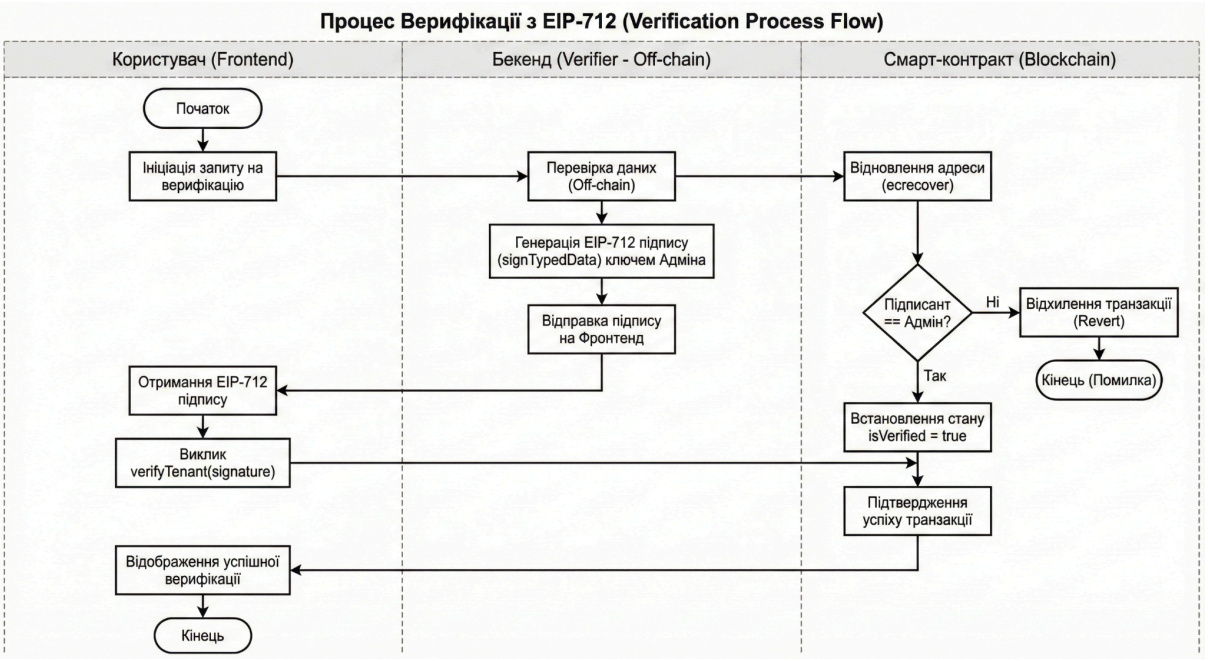


Рис 2.1 Процес верифікації з EIP-712**.**

Етап 2: Укладення угоди та фінансове забезпечення

Після успішної верифікації настає процес бронювання. Клієнт взаємодіє з децентралізованою системою календарів для вибору доступних дат. Смарт-контракт виступає як автоматизований ескроу-агент (депонування). При підписанні угоди обома сторонами, контракт автоматично блокує на гаманці орендаря необхідну суму, що включає орендну плату та заставу. Для уникнення волатильності криптовалют, розрахунки проводяться у стейблкоїнах. Кожна транзакція — депонування, оплата, повернення — незмінно фіксується в блокчейні, що забезпечує повну прозорість та значно спрощує облік.

Етап 3: Виконання, Розрахунок та Вирішення Спорів

Успішний сценарій («Happy Path») виконується автоматично: якщо час оренди вичерпано і претензій немає, смарт-контракт самостійно розподіляє кошти. У разі виникнення конфлікту («UnHappy Path») активується механізм спору. На поточному етапі реалізації функцію арбітра виконує **довірений оракул (адміністратор системи)**. Це дозволяє оперативно вирішувати суперечки без складних механізмів голосування, що є оптимальним для запуску платформи, з архітектурною можливістю заміни адреси адміністратора на адресу DAO-контракту в майбутньому.

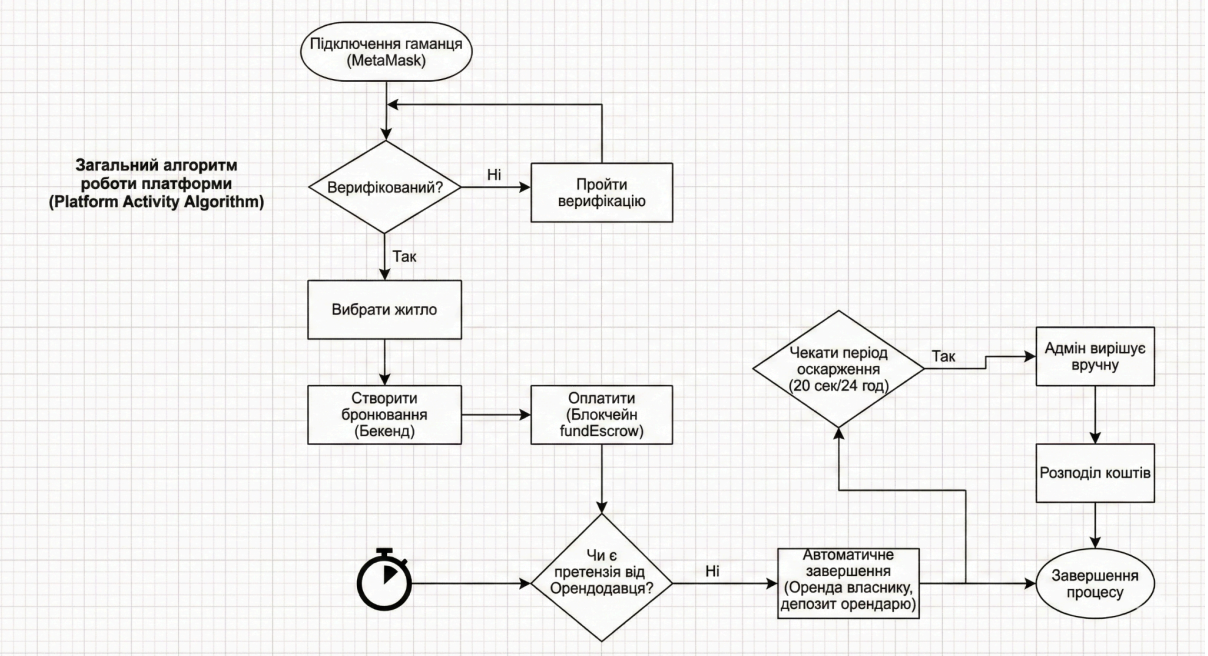


Рис 2.2 Загальний алгоритм роботи платформи.

На наступній діаграмі можна побачити перехід між трьома фазами та ключові точки прийняття рішень (успішна верифікація, наявність спору).

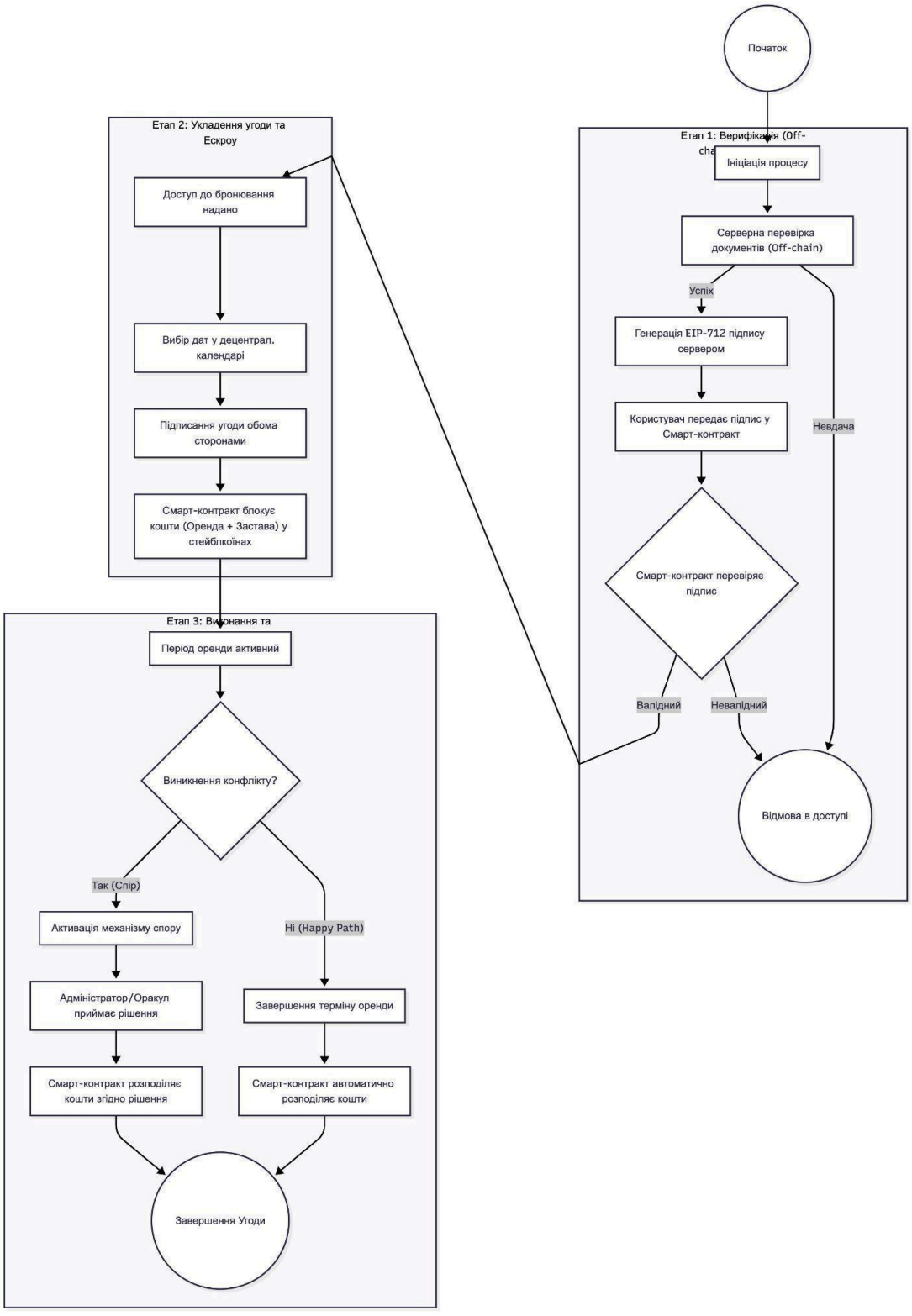


Рис 2.3 UML Діаграма діяльності (Бізнес-потік високого рівня)

Ця діаграма показує, які повідомлення передаються між Орендарем, Офчейн-сервером, Смарт-контрактом та Адміністратором (Оракулом) у хронологічному порядку.

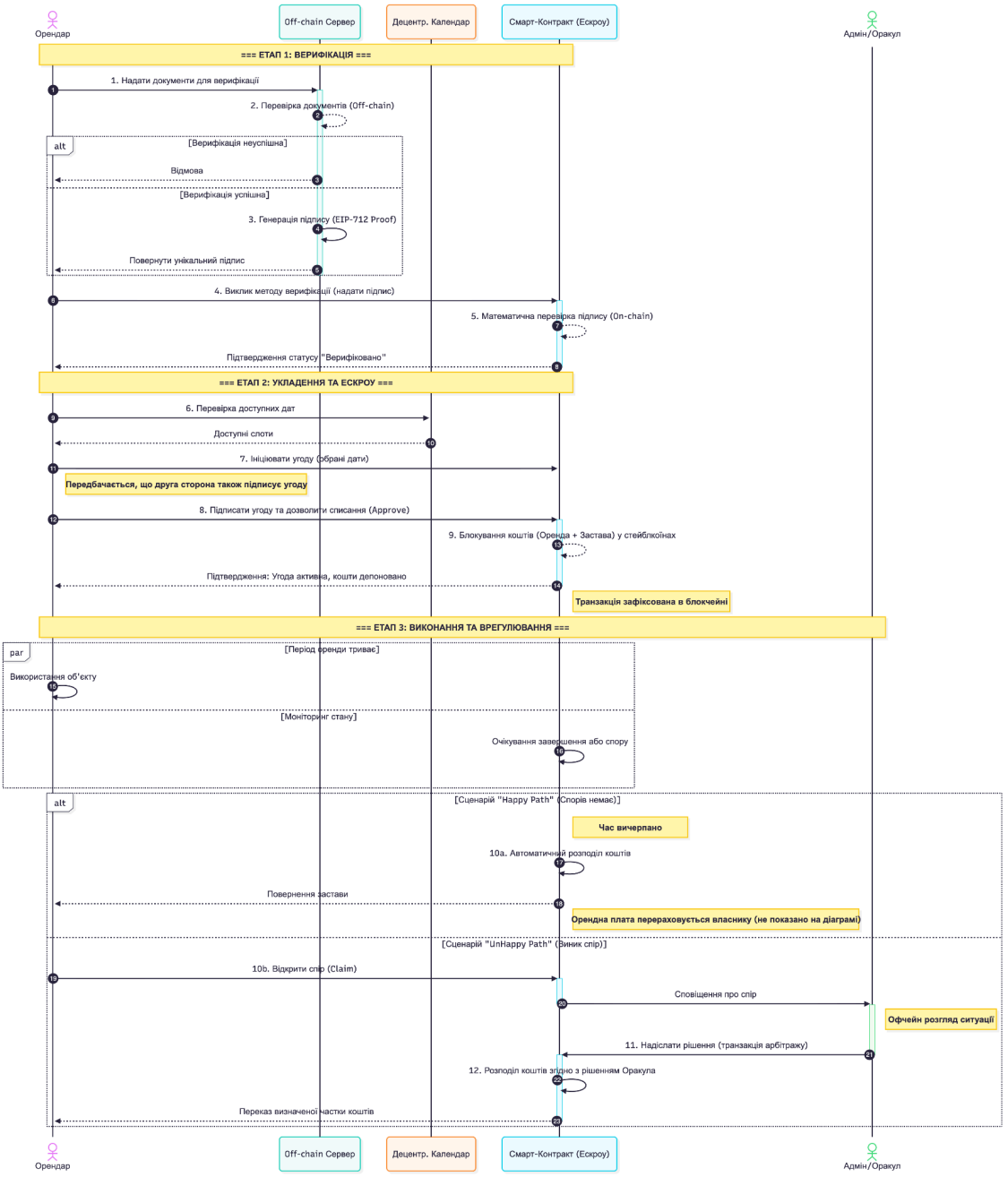


Рис. 2.3 UML Діаграма послідовності (Детальна технічна взаємодія)

2.2 Аналіз існуючих аналогів та їх функціоналу.

Платформа Airbnb є еталонним представником централізованих рішень (Web2) на ринку короткострокової оренди. Її функціонал базується на повній

централізації: всі дані користувачів, фінансові потоки та історія бронювань зберігаються на приватних серверах компанії. Основна орієнтація платформи — масовий ринок з використанням фіатних валют та банківських карток. Хоча Airbnb реалізує механізм утримання коштів (escrow), він є непрозорим для користувача: гроші знаходяться на рахунках компанії, а рішення про їх виплату або повернення приймається виключно менеджерами служби підтримки, що створює ризик людського фактору та суб'єктивності при вирішенні спорів.

Сервіс Travala займає нішу Web 2.5, виступаючи мостом між традиційними подорожами та криптовалютами. Функціонально це класичне онлайн-турагентство (OTA), яке інтегрувало можливість оплати у понад 90 видах криптовалют [3]. Однак, архітектурно Travala залишається централізованою системою: смарт-контракти не використовуються для управління процесом бронювання чи депонування коштів. Платформа орієнтована на власників криптовалют, які шукають способи витратити цифрові активи на реальні послуги (готелі, авіаквитки), але не пропонує децентралізації самого процесу угоди.

Платформа Dtravel є прямим конкурентом у сфері Web3, оскільки пропонує децентралізовану екосистему для оренди житла [9, 10]. Вона використовує смарт-контракти для проведення угод та управляється через DAO (децентралізовану автономну організацію), де власники токенів TRVL мають право голосу. Функціонал Dtravel включає значно нижчі комісії порівняно з Airbnb та прозорість транзакцій. Проте, платформа орієнтована на глибоко занурених у криптосферу користувачів, вимагаючи від них повного розуміння роботи гаманців та блокчейн-мереж, що часто стає бар'єром для масового впровадження.

Проект Rentberry фокусується переважно на ринку довгострокової оренди. Унікальною функцією цієї платформи є механізм аукціону орендної плати, де потенційні орендарі можуть пропонувати власну ціну [10]. Також реалізовано функцію токенізації страхового депозиту, що дозволяє

розморозити кошти орендарів. Rentberry використовує блокчейн для прозорості історії оренди та скорингу, але їхня модель більше орієнтована на вирішення проблем ринку нерухомості США та довгострокових контрактів, аніж на швидку автоматизацію короткострокових бронювань.

2.3 Обґрунтування гібридної архітектури та функціональних інновацій.

Опираючись на досвід конкурентів для реалізації проекту, а також з метою внесення новизни, було обрано реалізацію наступних речей:

Гібридна архітектура системи (Hybrid Web2+Web3) На відміну від повністю централізованих платформ (як Airbnb), де всі процеси контролюються сервером, або повністю децентралізованих (як Dtravel), які часто є складними для пересічного користувача, мною було розроблено компромісну модель. Вона поєднує звичний та швидкий веб-інтерфейс (Web2) для пошуку житла із безпекою блокчейну (Web3) для фінансових розрахунків. Це дозволило усунути потребу в банківських посередниках, перенісши зберігання коштів безпосередньо в код смарт-контракту, при цьому зберігаючи високу зручність для користувача.

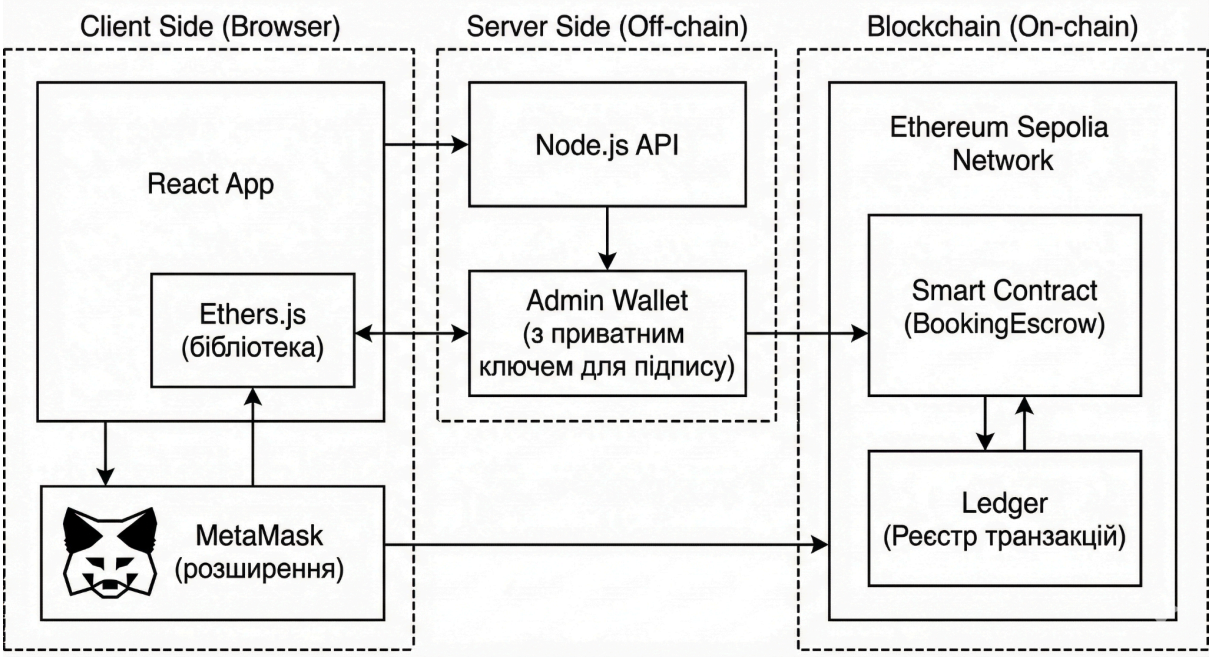


Рис. 2.4 Загальна взаємодія фронтенду, бекенду та взаємодії з блокчейном.

Механізм конфіденційної верифікації (Privacy-preserving Verification) Враховуючи ризики витоку персональних даних у конкурентів, було імплементовано протокол верифікації на базі стандарту EIP-712. Замість збереження чутливих даних (паспортів, імен) у публічному реєстрі блокчейну, система проводить перевірку off-chain (на сервері) та генерує унікальний криптографічний підпис. Смарт-контракт верифікує лише цей математичний доказ. Такий підхід забезпечує надійну ідентифікацію учасників без порушення їхньої приватності.

Автоматизоване депонування коштів (Time-locked Escrow) Для вирішення проблеми непрозорості руху коштів, притаманної класичним сервісам, розроблено смарт-контракт із детермінованою логікою. Функція ескроу блокує кошти орендаря (плату та заставу) до моменту завершення угоди. Новизною є використання автоматичних часових тригерів: якщо орендодавець не відкриває спір протягом встановленого «вікна оскарження», контракт автоматично розподіляє кошти. Це гарантує виплати програмним кодом і усуває людський фактор затримок.

Модель «оптимістичного виконання» (Optimistic Execution) З метою зниження транзакційних витрат та спрощення бізнес-процесу, систему побудовано за принципом «презумпції успішної угоды». На відміну від громіздких арбітражних систем, механізм вирішення спорів активується лише у виняткових випадках (за ініціативою орендодавця). Це дозволяє проводити переважну більшість угод в автоматичному режимі, залучаючи адміністративного арбітра лише при реальному конфлікті, що значно підвищує ефективність платформи.

2.4 Аналіз блокчейн-платформ та вибір технологічного стеку

Вибір технологічної основи для побудови децентралізованої системи оренди нерухомості є стратегічним рішенням, що визначає архітектурну стійкість, економічну доцільність та користувацький досвід майбутнього продукту. Цей процес вимагає вирішення класичної «трилеми блокчейну», яка постулює складність одночасного досягнення децентралізації, безпеки та масштабованості. Для забезпечення відповідності бізнес-вимогам, які передбачають миттєве бронювання та захист страхових депозитів, відбір платформи здійснювався на основі комплексу критичних показників: масштабованості (TPS), вартості транзакцій (Gas Fee), стійкості алгоритму консенсусу та зрілості екосистеми розробки.

У рамках дослідження було проведено порівняльний аналіз чотирьох репрезентативних архітектур: класичного Ethereum (Layer 1), високопродуктивного монолітного блокчейну Solana, приватного рішення для корпоративного сектору Hyperledger Fabric та мереж, сумісних з віртуальною машиною Ethereum (EVM), на прикладі Polygon. Результати порівняння наведено у таблиці 2.1.

Таблиця 2.1. Порівняння блокчейн платформ

Критерій порівняння	**Ethereum (Layer 1)**	**Solana**	**Hyperledger Fabric**	**Polygon (EVM Sidechain/ L2)**
Архітектура	Монолітний блокчейн	Монолітний (Proof of History)	Приватний (Permissioned)	Модульний / Сайдчейн
Алгоритм консенсусу	Proof-of-Stake (PoS)	PoH + PoS	Pluggable (Raft/Kafka)	Proof-of-Stake (Heimdall/ Bor)

Пропускна здатність (TPS)	~15–30	~65 000 (теор.)	~3 000+	~65 000 (теор.)
Час фіналізації блоку	~12–15 сек	~400 мс	Миттєво	~2 сек
Середня вартість транзакції	Висока ($2–$50+)	Наднизька (<$0.01)	Відсутня (корп. інфраструктура)	Низька ($0.01–$0.1)
Мова смарт-контрактів	Solidity, Vyper	Rust, C, C++	Go, Java, Node.js	Solidity, Vyper
Рівень децентралізації	Дуже високий	Середній	Низький (Централізований)	Середній/ Високий
Придатність для B2C	Низька (через вартість)	Висока	Низька (закритий доступ)	**Висока**

На основі проведеного аналізу було зроблено висновок про недоцільність використання Ethereum Mainnet для основних операцій через високу волатильність транзакційних комісій, що створює значний економічний бар'єр для користувачів при сплаті оренди або внесенні депозитів. Водночас Hyperledger Fabric було відхилено через його закриту архітектуру (permissioned), що суперечить концепції прозорої публічної системи, доступної для будь-якого орендодавця. Хоча мережа Solana демонструє високі показники швидкодії, її використання пов'язане з необхідністю розробки на мові Rust, що підвищує поріг входження та обмежує доступ до стандартизованих бібліотек безпеки, властивих EVM-екосистемі.

Оптимальним рішенням для реалізації проєкту обрано EVM-сумісну архітектуру (з орієнтацією на мережі типу Polygon або L2-рішення). Такий

підхід забезпечує необхідний баланс між економічною ефективністю та сумісністю. Використання технології сайдчейнів або Rollups дозволяє знизити вартість газу у 50–100 разів порівняно з базовим рівнем Ethereum, залишаючи транзакції рентабельними навіть для мікроплатежів. Крім того, повна підтримка Ethereum Virtual Machine (EVM) гарантує можливість використання стандартних користувацьких гаманців (наприклад, MetaMask) та перевірених інструментів аудиту безпеки.

2.5 Обґрунтування вибору інструментарію та архітектурних рішень

Проектування та програмна реалізація системи децентралізованої оренди вимагали ретельного підбору технологічного стеку, здатного забезпечити баланс між безпекою зберігання активів, вартістю транзакцій (Gas fees) та зручністю користувацького досвіду. Оскільки сервіс оперує фінансовими коштами користувачів у форматі ескроу, головним критерієм вибору інструментів стала їх надійність, перевіреність часом та відповідність сучасним стандартам безпеки Web3-індустрії.

Базовий рівень: Блокчейн та смарт-контракти Фундаментом системи обрано мову програмування Solidity (версії 0.8.20 і вище). Вибір цієї версії є критичним, оскільки вона містить вбудований захист від переповнення числових типів (overflow/underflow), що раніше було джерелом критичних вразливостей. Solidity дозволяє створювати складні структури даних (у проекті це `struct Escrow`) та ефективно керувати станами життєвого циклу угоди.

В якості середовища розробки (Development Environment) використано фреймворк Hardhat. На відміну від застарілого Truffle, Hardhat надає розробнику гнучкішу архітектуру на основі плагінів та вбудовану локальну мережу *Hardhat Network*. Ця мережа дозволяє не лише миттєво виконувати транзакції, але й, що найважливіше для даного проекту, маніпулювати часом блокчейну (EVM time manipulation). Це дало змогу протестувати логіку

часових вікон для відкриття спорів (змінна `DISPUTE_PERIOD_SECONDS`) без необхідності реального очікування. Крім того, можливість використання `console.log` безпосередньо у Solidity-коді значно пришвидшила процес відлагодження складної бізнес-логіки на етапі прототипування.

Криптографія та безпека (On-chain Layer) Для забезпечення цілісності даних та авторизації дій користувачів проект відмовляється від написання власної криптографічної логіки ("Don't roll your own crypto") на користь аудитрованих бібліотек OpenZeppelin Contracts. Ключовим архітектурним рішенням стало впровадження стандарту EIP-712 ("Typed Structured Data Hashing and Signing"). На відміну від звичайного підпису повідомлень, EIP-712 дозволяє користувачеві бачити у своєму гаманці чітку структуру даних, яку він підписує (наприклад, "VerifyTenant"), замість незрозумілого шістнадцяткового рядка. Використання бібліотеки `ECDSA` від OpenZeppelin у поєднанні з EIP-712 забезпечило надійний механізм офчейн-верифікації: бекенд перевіряє документи користувача і видає криптографічний доказ (підпис), який смарт-контракт може верифікувати дешево та безпечно, захищаючись від атак повторного відтворення (replay attacks).

Для розмежування прав доступу використано модуль Ownable, який надає адміністратору платформи (або децентралізованому арбітру) виключні права на вирішення спорів (`resolveDispute`), гарантуючи, що кошти не будуть заблоковані назавжди у випадку конфлікту сторін.

Серверна частина та взаємодія (Hybrid Web2/Web3 Layer) Для реалізації гібридної моделі верифікації обрано середовище Node.js із фреймворком Express. Це рішення дозволило винести ресурсоємні операції (перевірку реальних документів орендаря) за межі блокчейну, суттєво зекономивши кошти користувачів на комісіях. Сервер виконує роль довіреного оракула, який генерує підписи для смарт-контракту. Взаємодія між сервером, клієнтом та блокчейном реалізована через бібліотеку Ethers.js. Її перевагою над Web3.js є модульність та повна підтримка статичної типізації. Це дозволило чітко

розділити логіку читання даних (через `JsonRpcProvider`) та виконання транзакцій (через `Wallet` або `Signer`), а також безпечно працювати з великими числами (BigInt), які використовуються в Ethereum для позначення балансів.

Клієнтський рівень (Frontend) Інтерфейс користувача побудовано на бібліотеці React із використанням збирача Vite. Специфіка децентралізованих додатків (dApps) полягає у необхідності постійної синхронізації стану інтерфейсу зі станом блокчейну (очікування підтвердження блоку, зміна акаунту в MetaMask, оновлення балансу). Реактивна модель React ідеально підходить для цього завдання, дозволяючи створити плавний UX, де користувач отримує миттєвий зворотний зв'язок на кожному етапі — від підключення гаманця до фінального розрахунку за оренду. Vite, у свою чергу, забезпечує високу швидкість розробки завдяки нативній підтримці ES-модулів.

Узагальнений перелік використаних інструментів та їх призначення наведено у таблиці 2.2.

Таблиця 2.2 Стек технологій програмної реалізації.

Категорія	**Інструмент / Бібліотека**	**Призначення в системі**
Smart Contracts	Solidity (v0.8.20+)	Написання бізнес-логіки смарт-контрактів.
Framework	Hardhat	Розробка, тестування, розгортання та відлагодження коду.
Security	OpenZeppelin (ECDSA, EIP712, Ownable)	Криптографічна верифікація підписів, контроль доступу, безпека стандартів.
Network	Sepolia Testnet	Публічне тестове середовище для емуляції реальної роботи.

Backend	Node.js + Express	API для генерації криптографічних доказів (підписів) для верифікації.
Frontend	React + Vite	Створення реактивного інтерфейсу користувача.
Web3 Integration	Ethers.js	Взаємодія фронтенду з блокчейном (RPC-виклики, управління гаманцем).

2.6 Обмеження реалізації та припущення.

Обмеження поточної реалізації та архітектурні компроміси В рамках кваліфікаційної роботи розроблено прототип (MVP), який має певні свідомі спрощення порівняно з промисловою версією:

Централізований арбітраж: Для спрощення логіки вирішення спорів використано модель OnlyOwner (адміністративний арбітраж). Це дозволяє уникнути вразливостей складних систем голосування на етапі запуску.

Довірений Верифікатор: Система покладається на чесність серверної частини (Off-chain Verifier) для генерації підписів EIP-712. Ці компроміси дозволили зосередитися на головній меті роботи — створенні надійного та прозорого механізму фінансового ескроу (Escrow), який гарантує збереження коштів програмним кодом, що є ключовою перевагою над традиційними системами.

Висновки до другого розділу.

У другому розділі здійснено декомпозицію бізнес-процесів оренди та обґрунтовано архітектурні рішення проєкту. На основі аналізу існуючих аналогів запропоновано гібридну модель системи, яка поєднує off-chain верифікацію для захисту приватності користувачів та on-chain розрахунки для

гарантії безпеки коштів. Для програмної реалізації обрано EVM-сумісний стек технологій, що забезпечує необхідний баланс між швидкістю, вартістю транзакцій та надійністю. Ключовою інновацією стало впровадження механізму «оптимістичного виконання» та стандарту EIP-712, що дозволяє автоматизувати успішні угоди без втручання арбітрів та оптимізувати витрати на газ.

РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ СИСТЕМИ.

Враховуючи, що проєкт **"Crypto Booking Demo"** моделює публічну, фінансово-орієнтовану операцію (ескроу) з токенами ETH, критичним є вибір платформи, яка забезпечує найвищу ліквідність, безпеку та зрілу екосистему інструментів. Саме тому була обрана архітектура, сумісна з Ethereum Virtual Machine (EVM), яка підтримує мову Solidity та бібліотеки OpenZeppelin. EVM-сумісні мережі домінують у сферах DeFi та NFT, мають найбільшу спільноту розробників та пропонують перевірені рішення другого рівня (Layer 2) для вирішення проблем масштабованості та вартості транзакцій (Gas Fees), які є основними викликами для базового рівня Ethereum. Таким чином, цей вибір забезпечує технологічну стійкість та найширшу сумісність з існуючими криптоактивами.

Для цілей цього пілотного проєкту та демонстрації архітектури було обрано Ethereum Sepolia (EVM-сумісна тестова мережа). Використання Sepolia та відповідного технологічного стеку (Solidity, Ethers.js, Hardhat) забезпечує кілька критичних переваг для тестування:

- Зрілість Інструментів: Доступ до найбільш зрілої та широкодокументованої екосистеми EVM прискорює розробку та налагодження.
- Безкоштовне Тестування: Тестова мережа дозволяє проводити необмежену кількість транзакцій без реальних витрат на газ, що є ідеальним для ітеративного тестування функціоналу смарт-контракту.
- Імітація Реальних Активів: Використання тестового токена ETH дозволяє повністю імітувати логіку роботи з реальними стейблкоїнами та механізмами approve/transferFrom без фінансових ризиків.

- Таким чином, вибір EVM-сумісних інструментів для тестового проєкту є прагматичним кроком, що забезпечує швидкість розробки, нульову вартість тестування та повну відповідність логіці майбутнього розгортання у робочій публічній мережі (наприклад, на Layer 2 рішенні).

3.1. Загальна архітектура програмного комплексу.

Програмний комплекс розроблено на основі гібридної архітектури, що поєднує в собі переваги традиційних веб-технологій (Web2) та децентралізованих рішень (Web3). Такий підхід дозволяє створити збалансовану систему, яка забезпечує високу швидкість обробки даних і зручність для користувача, водночас гарантуючи безпеку фінансових операцій та незмінність домовленостей. Архітектура системи розділена на три логічні рівні, кожен з яких виконує чітко визначені функції та взаємодіє з іншими компонентами для забезпечення цілісності бізнес-процесу оренди житла.

Фундаментом системи виступає On-chain рівень, реалізований у вигляді смарт-контракту, розгорнутого в тестовій мережі блокчейну Ethereum Sepolia. Цей рівень відповідає за критично важливі аспекти угоди, які вимагають максимальної довіри та прозорості, а саме за фінансові розрахунки та механізм депонування коштів (Escrow). Смарт-контракт діє як незалежний арбітр, що програмно блокує активи орендаря та автоматично розподіляє їх між учасниками угоди за виконання заданих часових умов. Крім того, на цьому рівні здійснюється криптографічна валідація підписів, що дозволяє перевіряти статус верифікації користувачів без необхідності зберігання їхніх персональних даних у відкритому реєстрі.

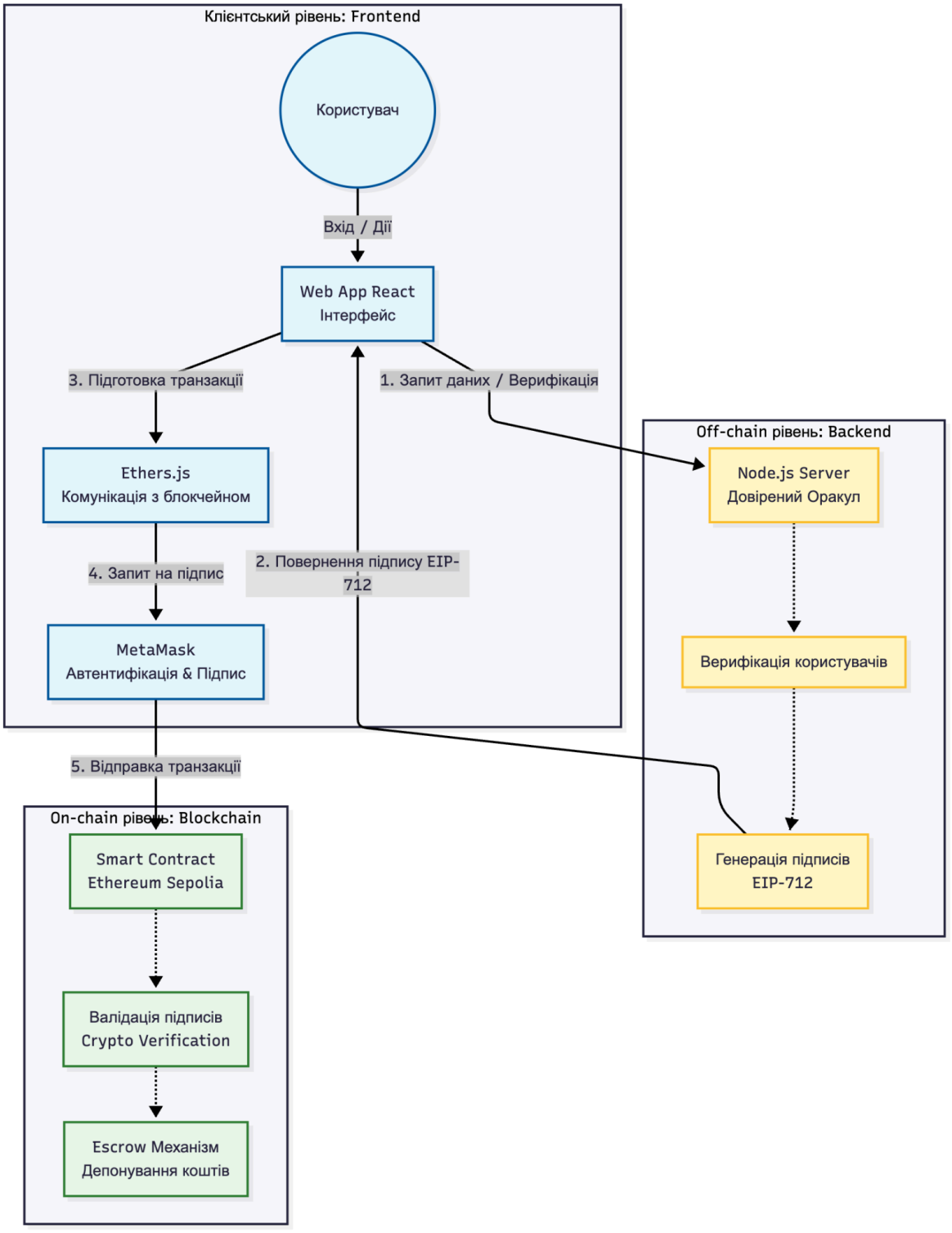


Рис. 3.1 Архітектура програмної частини

Сполучною ланкою комплексу є Off-chain рівень, представлений сервером на базі середовища виконання Node.js. Цей компонент виконує роль

довіреного оракула та адміністратора системи. Основною функцією серверної частини є проведення верифікації користувачів поза блокчейном та генерація відповідних криптографічних підписів згідно зі стандартом EIP-712, які згодом використовуються смарт-контрактом для підтвердження прав доступу. Також на цей рівень винесено бізнес-логіку, зберігання якої в блокчейні було б економічно недоцільним через вартість газу, зокрема ініціалізацію параметрів бронювання та підготовку даних для створення смарт-контрактних угод.

Взаємодію кінцевого користувача з системою забезпечує клієнтський рівень, реалізований у вигляді веб-додатку на бібліотеці React. Фронтенд слугує єдиною точкою доступу, що об’єднує функціонал обох попередніх рівнів у зрозумілий інтерфейс. Для комунікації з блокчейном додаток використовує бібліотеку ethers.js, яка дозволяє відправляти транзакції та зчитувати стан смарт-контракту. Автентифікація користувачів та підписання транзакцій здійснюються через інтеграцію з некастодіальним гаманцем MetaMask, що забезпечує безпечне управління цифровими активами безпосередньо у браузері.

3.2 Реалізація смарт-контракту (On-chain логіка)

Ключовим компонентом розробленої системи є смарт-контракт BookingEscrow.sol, який реалізує детерміновану логіку виконання угод та зберігання активів. Архітектура контракту побудована навколо модуля конфіденційної верифікації, що дозволяє перевіряти статус учасників без розкриття їхніх персональних даних у публічному реєстрі. Функція verifyTenant використовує криптографічний механізм відновлення адреси підписанта за допомогою бібліотеки ECDSA. Контракт отримує цифровий підпис, згенерований сервером, і відновлює з нього адресу. Якщо відновлена адреса збігається з адресою довіреного адміністратора, верифікація вважається успішною. Такий підхід гарантує, що смарт-контракт оперує лише математичним фактом перевірки особи, забезпечуючи високий рівень

приватності користувачів, оскільки жодні особисті дані не записуються в блокчейн.

Фундаментом фінансової взаємодії виступає модуль ескроу, який базується на структурі даних struct Escrow. Ця структура зберігає повний контекст угоди, включаючи адреси сторін, суми оренди та застави, а також поточний статус, який може набувати значень Pending (очікування), Funded (оплачено), Resolved (завершено) або Disputed (спір). Процес депонування коштів ініціюється функцією fundEscrow. Вона містить суворі перевірки безпеки: контракт порівнює суму транзакції (msg.value) із заявленою вартістю бронювання. Лише за умови повної відповідності сум кошти блокуються на балансі смарт-контракту, виключаючи можливість часткової оплати або шахрайства, а угода переходить у стан активного виконання.

Для забезпечення справедливості угоди імплементовано модуль вирішення спорів, що спирається на логіку часових блокувань (Time-locks). Система передбачає дві альтернативні гілки завершення угоди. Пріоритетним сценарієм є так званий «Happy Path», реалізований через функцію automaticResolve. Якщо встановлений часовий ліміт (у межах прототипу — 20 секунд) вичерпано, і статус спору не було активовано, смарт-контракт автоматично розблоковує кошти: орендна плата перераховується власнику, а страховий депозит повертається орендарю. Альтернативний сценарій («Dispute Path») активується функцією raiseDispute. Якщо орендодавець фіксує порушення умов до завершення таймера, ця функція переводить угоду в стан спору та заморожує автоматичний розподіл активів до моменту винесення рішення адміністратором, що захищає майнові інтереси власника житла.

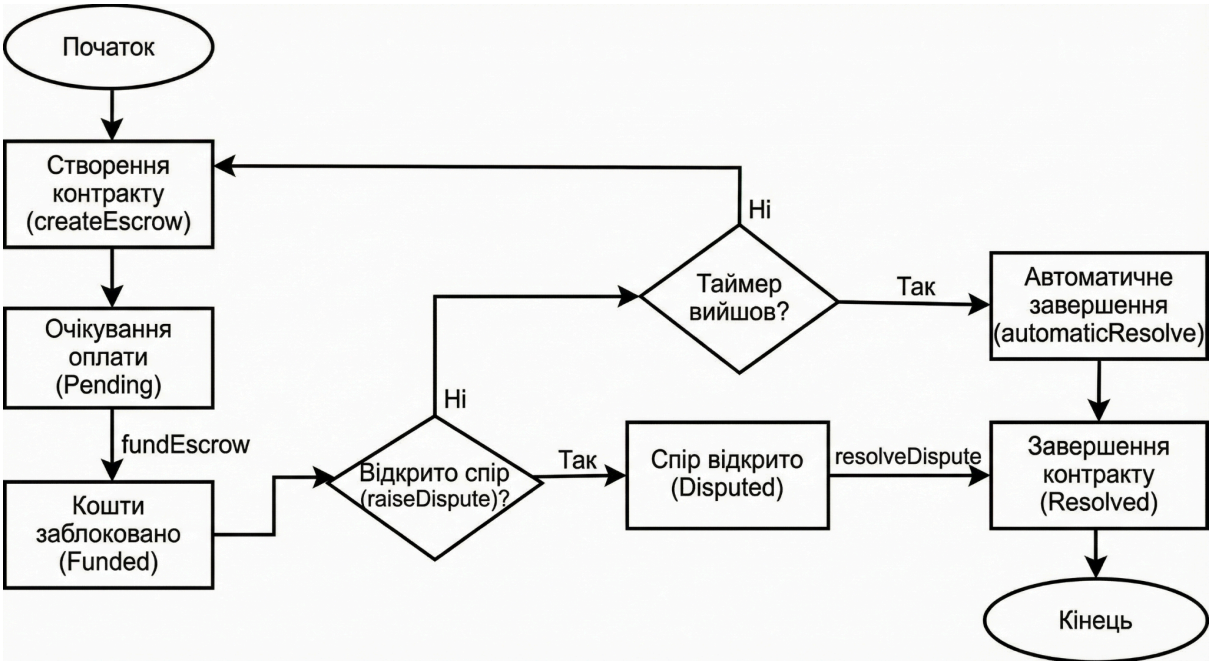


Рис. 3.2 Алгоритм роботи смарт контракту.

3.3 Реалізація серверної частини (Off-chain логіка)

Серверна складова програмного комплексу, реалізована у файлі index.js, виступає критично важливою сполучною ланкою між інтерфейсом користувача та блокчейн-мережею. Центральним елементом безпеки та функціонування цього рівня є управління приватним ключем адміністратора (ADMIN_PRIVATE_KEY або SEPOLIA_PRIVATE_KEY). Володіння цим ключем наділяє сервер повноваженнями довіреного Верифікатора, дозволяючи йому виконувати криптографічні операції від імені платформи. Це перетворює звичайний Web2-сервер на повноцінного учасника блокчейн-екосистеми, який має право авторизувати дії користувачів та керувати адміністративними функціями смарт-контракту без необхідності розкриття ключів доступу на клієнтській стороні.

Ключову роль у забезпеченні конфіденційності відіграє реалізація API-ендпоінту /api/get-verification-proof. Цей метод відповідає за імітацію механізму доказів з нульовим розголошенням (ZKP). При зверненні клієнта сервер формує структуровані дані згідно зі стандартом EIP-712, які містять параметри домену та адресу користувача, після чого підписує їх своїм

приватним ключем. Згенерований цифровий підпис повертається на фронтенд і слугує математичним доказом успішної перевірки. Такий підхід дозволяє перенести процедуру верифікації документів за межі блокчейну (off-chain), передаючи в смарт-контракт лише криптографічне підтвердження, що гарантує валідність даних без порушення приватності.

Для покращення користувацького досвіду (UX) та оптимізації витрат розроблено ендпоінт /api/create-escrow. Замість того, щоб змушувати користувача самостійно ініціювати складну транзакцію створення угоди, цю функцію бере на себе сервер. Отримавши деталі бронювання, сервер викликає метод createEscrow смарт-контракту, використовуючи власний гаманець адміністратора. Це означає, що витрати на газ (Gas Fees) за ініціалізацію запису в реєстрі покриваються платформою, а не користувачем. Таке архітектурне рішення значно знижує поріг входження для клієнтів, залишаючи за ними необхідність сплачувати комісію лише на етапі безпосереднього переказу коштів.

3.4. Реалізація клієнтського інтерфейсу (Web3 Integration)

Клієнтська частина програмного комплексу, зосереджена у файлі App.jsx, відповідає за безшовну інтеграцію користувача з децентралізованою мережею. Управління станом додатку реалізовано з урахуванням асинхронної природи блокчейну: інтерфейс повинен миттєво реагувати на зміни, що відбуваються як у глобальному реєстрі, так і в локальному гаманці користувача. Завдяки використанню React-хуків useState та useEffect, додаток підтримує актуальність даних у реальному часі. Зокрема, реалізовано динамічне відстеження зміни активного акаунту в MetaMask через подію accountsChanged. Це гарантує, що при перемиканні користувачем гаманця інтерфейс автоматично скидає статус верифікації та бронювання, запобігаючи виконанню транзакцій від імені невірного користувача.

Взаємодія з системою розпочинається з процедури ініціалізації Web3-провайдера через функцію connectWallet. Додаток використовує об'єкт window.ethereum, що інжектується розширенням браузера, для створення екземпляра ethers.BrowserProvider. Це дозволяє безпечно запитувати доступ до рахунків користувача без передачі приватних ключів. Логіка інтерфейсу побудована за принципом послідовного доступу до функціоналу, візуалізуючи три ключові етапи життєвого циклу угоди. На першому етапі користувачу доступна лише функція верифікації; після успішного отримання криптографічного доказу відкривається інтерфейс бронювання та оплати; і лише після підтвердження транзакції депонування коштів відображаються елементи управління завершенням угоди або відкриттям спору.

Критично важливим аспектом є забезпечення реактивності інтерфейсу під час виконання блокчейн-операцій. Оскільки запис даних у реєстр потребує часу на підтвердження блоку, фронтенд реалізує механізм очікування завершення транзакцій за допомогою методу .wait(). Додаток "слухає" результати виконання ключових функцій смарт-контракту, таких як fundEscrow або raiseDispute. Одразу після отримання підтвердження від мережі, локальний стан інтерфейсу оновлюється автоматично, миттєво відображаючи перехід угоди у статус Funded або Disputed. Такий підхід дозволяє користувачеві бачити результат своїх дій у реальному часі без необхідності ручного перезавантаження сторінки, що наближає досвід використання децентралізованого додатку (dApp) до стандартів сучасних веб-сервісів.

3.5. Розгортання (Deployment) та тестування смарт-контрактів у мережі Sepolia.

Фінальним етапом реалізації програмного комплексу стало розгортання смарт-контракту в публічній тестовій мережі. Для цієї мети було обрано мережу Ethereum Sepolia, яка наразі є рекомендованим середовищем для тестування децентралізованих додатків. Sepolia використовує механізм консенсусу Proof-of-Stake, ідентичний до основної мережі Ethereum (Mainnet),

що дозволяє з високою точністю симулювати умови реальної експлуатації, включаючи час підтвердження блоків та вартість газу, без фінансових ризиків, пов'язаних з використанням реальної криптовалюти. Вибір саме цієї мережі забезпечує стабільність тестування та сумісність з більшістю сучасних інструментів розробки та гаманців.

Процес перенесення коду в блокчейн було автоматизовано за допомогою спеціалізованого скрипта deploy.ts. Ключовою особливістю цього сценарію є динамічне налаштування параметрів безпеки безпосередньо під час ініціалізації. Скрипт автоматично зчитує приватний ключ адміністратора з файлу конфігурації середовища, вираховує відповідну публічну адресу та передає її як аргумент у конструктор смарт-контракту VerifiedBookingEscrow. Це критично важливий крок, оскільки саме ця адреса жорстко фіксується в незмінному коді контракту як єдиний довірений Верифікатор, підписи якого будуть прийматися системою для авторизації користувачів.

Результатом виконання скрипта розгортання стало успішне створення екземпляра контракту в мережі Sepolia, що було підтверджено відповідними записами в консолі розробника. Система повідомила про присвоєння контракту унікальної адреси (формату 0x...) та підтвердила встановлення коректної адреси верифікатора. Отримані адреси були інтегровані в конфігураційні файли серверної та клієнтської частин додатку, що дозволило об'єднати всі розрізнені модулі в єдину працездатну екосистему. Проведені після цього інтеграційні тести, що включали повний цикл від верифікації користувача до успішного завершення ескроу-угоди, підтвердили коректність взаємодії компонентів та працездатність закладеної бізнес-логіки в умовах реальної блокчейн-мережі.

Висновки до третього розділу.

Третій розділ присвячено програмній реалізації та тестуванню розробленого комплексу в мережі Ethereum Sepolia. Створено працездатний прототип, що складається зі смарт-контракту для ескроу-операцій, сервера-оракула для криптографічної перевірки даних та клієнтського веб-інтерфейсу. Практичні випробування підтвердили коректність роботи алгоритмів бронювання, надійність механізму блокування коштів та ефективність автоматичного розподілу активів. Реалізація підтвердила технічну життєздатність запропонованої архітектури, забезпечивши прозорість фінансових потоків та високий рівень безпеки без збереження чутливих даних у публічному реєстрі.

Висновки.

У результаті виконання роботи було досягнуто основної мети дослідження, що полягала у створенні, теоретичному обґрунтуванні та практичній валідації комплексної моделі децентралізованої платформи оренди. Робота розпочалася з комплексного аналізу предметної області, в ході якого було ідентифіковано критичні недоліки існуючих централізованих платформ, такі як надмірно високі комісії, непрозорість механізмів арбітражу та суттєві ризики збереження персональних даних користувачів. На основі отриманих даних було обґрунтовано доцільність застосування технології блокчейн та спроектовано гібридну архітектуру програмного комплексу, що гармонійно поєднує швидкість та зручність традиційних веб-інтерфейсів (Web2) із надійністю та безпекою децентралізованих реєстрів (Web3). Розроблена модель взаємодії компонентів включає клієнтський додаток на базі React, інтегрований з гаманцем MetaMask, серверну частину на Node.js, що виконує роль оракула, та систему смарт-контрактів.

Практична складова роботи представлена у реалізації програмного прототипу (MVP) та його розгортанні в тестовій мережі Ethereum (Sepolia). Ключовим елементом системи став смарт-контракт на мові Solidity, який виконує функції автоматизованого ескроу-агента. В ході тестування було підтверджено коректність виконання усіх етапів бізнес-процесу: транзакцій бронювання, депонування коштів, автоматичного розподілу фінансів та механізму вирішення спорів. Для забезпечення фінансової стабільності та нівелювання волатильності криптовалют бізнес-логіка смарт-контракту базується на використанні стейблкоїнів, що робить систему життєздатною для реального бізнесу.

Наукова новизна одержаних результатів полягає у формуванні цілісної системної моделі, яка інтегрує раніше розрізнені елементи в єдиний процес та вирішує трилему прозорості, приватності та ефективності. Вперше для сфери

орендних угод запропоновано комбіновану модель конфіденційної верифікації, що базується на емуляції підходу Zero-Knowledge через використання криптографічних підписів стандарту EIP-712. Це дозволяє здійснювати перевірку особистості учасників off-chain (поза блокчейном) та передавати у смарт-контракт лише математичний доказ валідності, що забезпечує відповідність сучасним вимогам захисту даних (GDPR) та суттєво знижує транзакційні витрати (Gas fees) порівняно з повним зберіганням даних on-chain.

Значним внесок у розвиток технології децентралізованого ескроу стало впровадження моделі «оптимістичного виконання» (Optimistic Execution) з використанням часових блокувань (Time-locks). На відміну від існуючих аналогів, дана модель автоматизує успішний сценарій завершення угоди без необхідності втручання арбітрів, що мінімізує транзакційні витрати та скорочує час фіналізації розрахунків. Роль арбітра виконує адміністративний оракул лише у виняткових випадках виникнення спору, що є ефективним рішенням для етапу MVP. Крім того, набуло подальшого розвитку застосування гібридних архітектур в електронній комерції: розроблено алгоритм, де сервер діє як довірений оракул лише для ініціалізації параметрів, тоді як фінансові потоки залишаються повністю децентралізованими, вирішуючи проблему масштабованості блокчейн-систем.

Практичне значення роботи полягає у створенні дієвого інструменту для нішевого ринку короткострокової оренди (наприклад, погодинна оренда студій або просторів). Розроблений прототип реалізує принцип дисінтермедіації, знижуючи комісійні витрати за рахунок усунення фінансових посередників. Впровадження смарт-контрактів гарантує орендодавцю отримання коштів за надані послуги, а орендарю — повернення страхового депозиту, оскільки логіка виплат зафіксована у незмінному коді. Важливою відмінністю роботи від суміжних досліджень є адаптація розробленої моделі

до правового поля України, зокрема аналіз сумісності з Цивільним правом та надання рекомендацій щодо інтеграції з державними реєстрами нерухомості. Запропонована архітектура є універсальною та може бути адаптована для інших сфер (фріланс, e-commerce) або перенесена на L2-мережі (Polygon, Optimism) для масштабування.

Достовірність отриманих результатів забезпечена використанням комплексу наукових методів. На теоретичному етапі застосовувалися методи аналізу та синтезу для опрацювання літератури, а також порівняльний аналіз технологічних платформ. Метод системного аналізу дозволив декомпозувати бізнес-процеси оренди. На практичному етапі ключову роль відіграли методи моделювання та прототипування через проведення програмного експерименту, що надав емпіричні дані для підтвердження ефективності автоматизації, яких раніше бракувало в академічних дослідженнях цієї сфери.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Wallis K., Stodt J., Jastremskoj E., Reich C. Agreements between Enterprises digitized by Smart Contracts in the Domain of Industry 4.0 [Електронний ресурс] // arXiv. 2020. Режим доступу: https://arxiv.org/abs/2007.14181

2. Горбачук В. М. та ін. Смарт-контракти в енергетиці. Використання блокчейн технологій в енергетиці [Електронний ресурс] // ResearchGate. 2024. Режим доступу: https://www.researchgate.net/publication/382183240

3. Трішин Ф. А., Трач О. Р. Процесний підхід у формуванні системи управління якістю бізнес-процесів в операційній діяльності туристичних підприємств України // Food Industry Economics. 2022. Т. 14, No 4. Режим доступу: https://www.researchgate.net/publication/369601389

4. Таранюк Л. М. Теоретико-методологічні засади управління вибором напрямів реінжинірингу бізнес-процесів промислових підприємств // Вісник СумДУ. Серія: Економіка. 2015. No 1. С. 13. Режим доступу: http://www.irbis-nbuv.gov.ua/cgi-bin/irbis_nbuv/cgiirbis_64.exe?I21DBN=LINK&P21DBN=UJRN&Z21ID=&S21REF=10&S21CNR=20&S21STN=1&S21FMT=ASP_meta&C21COM=S&2_S21P03=FILA=&2_S21STR=VSU_ekon_2015_1_13

5. Шевчун М. Б. Моделювання результативного управління логістичними процесами підприємства торгівлі // Бізнес Інформ. 2021. С. 234–241. Режим доступу: http://jnas.nbuv.gov.ua/uk/article/UJRN-0001269641

6. Бавико О. Є. Цифровізація бізнес-процесів як елемент стратегії сталого смарт-розвитку підприємницьких структур // Економічний журнал Одеського політехнічного університету. 2023. No 2(24). С. 15–23. Режим доступу: https://economics.net.ua/ejopu/2023/No2/15.pdf

7. Машкіна, І., Носенко, Т., Глушак, О., Співак, С., & Білоус, В. (2025). ОПТИМІЗАЦІЯ CUSTOMER SUPPORT ЗА ДОПОМОГОЮ AI ЧАТ-БОТІВ: ПРАКТИЧНИЙ КЕЙС. *Електронне фахове наукове видання «Кібербезпека:*

освіта, наука, техніка», *4*(28), 727-739.https://www.csecurity.kubg.edu.ua/index.php/journal/article/view/838

8. Chou Y. et al. An Artwork Rental System Based on Blockchain Technology [Електронний ресурс] // Symmetry. 2023. Vol. 15(2). Режим доступу: https://www.mdpi.com/2073-8994/15/2/341

9. Yuan S. Towards a Trustworthy Rental Market: A Blockchain-Based Housing System Architecture [Електронний ресурс] // Electronics. 2025. Vol. 14(15). Режим доступу: https://www.mdpi.com/2079-9292/14/15/3121

10. Li R., Shen W., Yang Y., Zhou L. Housing rental system based on blockchain Technology [Електронний ресурс] // Journal of Physics: Conference Series. 2021. Vol. 1948. Режим доступу: https://iopscience.iop.org/article/10.1088/1742-6596/1948/1/012058/meta

11. Chen Qi-Long1 , Ye Rong-Hua1, Lin Fei-Long. [Insert Title from PDF] [Електронний ресурс] // IEEE Xplore. 2019. Режим доступу: https://ieeexplore.ieee.org/abstract/document/8844943

12. Vikas Hassija, Mohd Zaid, Gurjot Singh, Amit Srivastava, Vikas Saxena [Електронний ресурс] // ScitePress Digital Library. 2019. Режим доступу: https://www.scitepress.org/Papers/2019/80972/80972.pdf

13. Козенкова В. Д. Розвиток ринку криптовалют: глобальні тренди та виклики [Електронний ресурс] // Економічний простір. 2025. № 198. С. 180–187. Режим доступу: https://economic-prostir.com.ua/wp-content/uploads/2025/03/198-180-187-kozenkova.pdf

14. Global Legal Group. Blockchain & Cryptocurrency Laws and Regulations 2025 [Електронний ресурс] // Global Legal Insights. 2025. Режим доступу: https://www.globallegalinsights.com/practice-areas/blockchain-cryptocurrency-laws-and-regulations/

15. PwC. Global Crypto Regulation Report 2025: Navigating the Global Crypto Landscape [Електронний ресурс] // Crystal Intelligence / PwC. 2025. Режим

доступу: https://crystalintelligence.com/crypto-regulations/pwc-global-crypto-regulation-trends-for-2025/

16. OSL Academy. The growing need for clear crypto regulations in 2025 [Електронний ресурс] // OSL. 2025. Режим доступу: https://osl.com/academy/article/the-growing-need-for-clear-crypto-regulations-in-2025/

17. Гашко Андрій Олександрович, et al. Автоматизований метод перевірки правильності виконання смарт-контрактів в блокчейн мережі. Телекомунікаційні та інформаційні технології, 2025, 1: 13-20. DOI: 10.31673/2412-4338.2025.014506

18. Sumsub. Crypto regulations in the US: A complete guide [Електронний ресурс] // Sumsub Blog. 2025. Режим доступу: https://sumsub.com/blog/crypto-regulations-in-the-us-a-complete-guide/

19. Abramov, V., Astafieva, M., Boiko, M., Bodnenko, D., Bushma, A., Vember, V., Hlushak, O., Zhyltsov, O., Ilich, L., Kobets, N., Kovaliuk, T., Kuchakovska, H., Lytvyn, O., Lytvyn, P., Mashkina, I., Morze, N., Nosenko, T., Proshkin, V., Radchenko, S., … Yaskevych, V. (2021). *Theoretical and practical aspects of the use of mathematical methods and information technology in education and science*. https://doi.org/10.28925/9720213284km.

20. Бушма, Олександр Володимирович та Машкіна, Ірина Вікторівна та Носенко, Тетяна Іванівна та Яскевич, Владислав Олександрович (2024) *Кваліфікаційна робота магістра: Навчально-методичний посібник для спеціальності «Комп'ютерні науки»* Київський столичний університет імені Бориса Грінченка, Україна.https://elibrary.kubg.edu.ua/id/eprint/50205/

Додаток А. (BookingEscrow.sol)

Це найголовніший файл. Він демонструє основну бізнес-логіку (створення ескроу, поповнення, виплату), використання Solidity та стандартів OpenZeppelin (Ownable).

BookingEscrow.sol M

blockchain-escrow > contracts > BookingEscrow.sol

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "@openzeppelin/contracts/access/Ownable.sol";
import "@openzeppelin/contracts/utils/cryptography/EIP712.sol";
import "@openzeppelin/contracts/utils/cryptography/ECDSA.sol";

contract VerifiedBookingEscrow is Ownable, EIP712 {

    // ... (всі змінні: verifierAddress, isVerified, DISPUTE_PERIOD_SECONDS, і т.д.) ...
    address public verifierAddress;
    mapping(address => bool) public isVerified;
    uint256 public constant DISPUTE_PERIOD_SECONDS = 20;
    mapping(uint256 => uint256) public fundedTimestamp;
    mapping(uint256 => bool) public isDisputed;

    enum EscrowState { Pending, Funded, Resolved, Cancelled }

    struct Escrow {
        address tenant;
        address landlord;
        uint256 rentAmount;
        uint256 depositAmount;
        EscrowState state;
    }

    mapping(uint256 => Escrow) public escrows;

    // ... (всі події) ...
    event TenantVerified(address indexed tenant);
    event EscrowCreated(uint256 indexed bookingId, address indexed tenant, address indexed landlord, uint256 rentAmount, uin
    event EscrowFunded(uint256 indexed bookingId, uint256 amount, uint256 timestamp);
    event DisputeRaised(uint256 indexed bookingId, address indexed landlord);
    event EscrowResolved(uint256 indexed bookingId, uint256 toLandlord, uint256 toTenant);

    constructor(address _verifierAddress)
        Ownable(msg.sender)
        EIP712("VerifiedRentals", "1")
```

Додаток Б. (hardhat.config.ts)

Демонструє робоче середовище. Показує, як налаштування плагінів (hardhat-ethers), версію компілятора та підключення до тестової мережі (Sepolia), включаючи завантаження ключів з .env.

hardhat.config.cjs

blockchain-escrow > hardhat.config.cjs > ...

```
require("@nomicfoundation/hardhat-toolbox");
require("dotenv/config");

module.exports = {
  solidity: {
    version: "0.8.28",
    settings: {
      optimizer: {
        enabled: true,
        runs: 200,
      },
    },
  },
  networks: {
    sepolia: {
      url: process.env.SEPOLIA_RPC_URL || "",
      accounts: process.env.SEPOLIA_PRIVATE_KEY ? [process.env.SEPOLIA_PRIVATE_KEY] : [],
    },
  },
};
```

Додаток В. (Deploy.ts)

Скрипт для автоматичного розгортання (деплою) смарт-контракту VerifiedBookingEscrow у блокчейн-мережу (наприклад, Sepolia) за допомогою середовища Hardhat.

deploy.ts 1, M

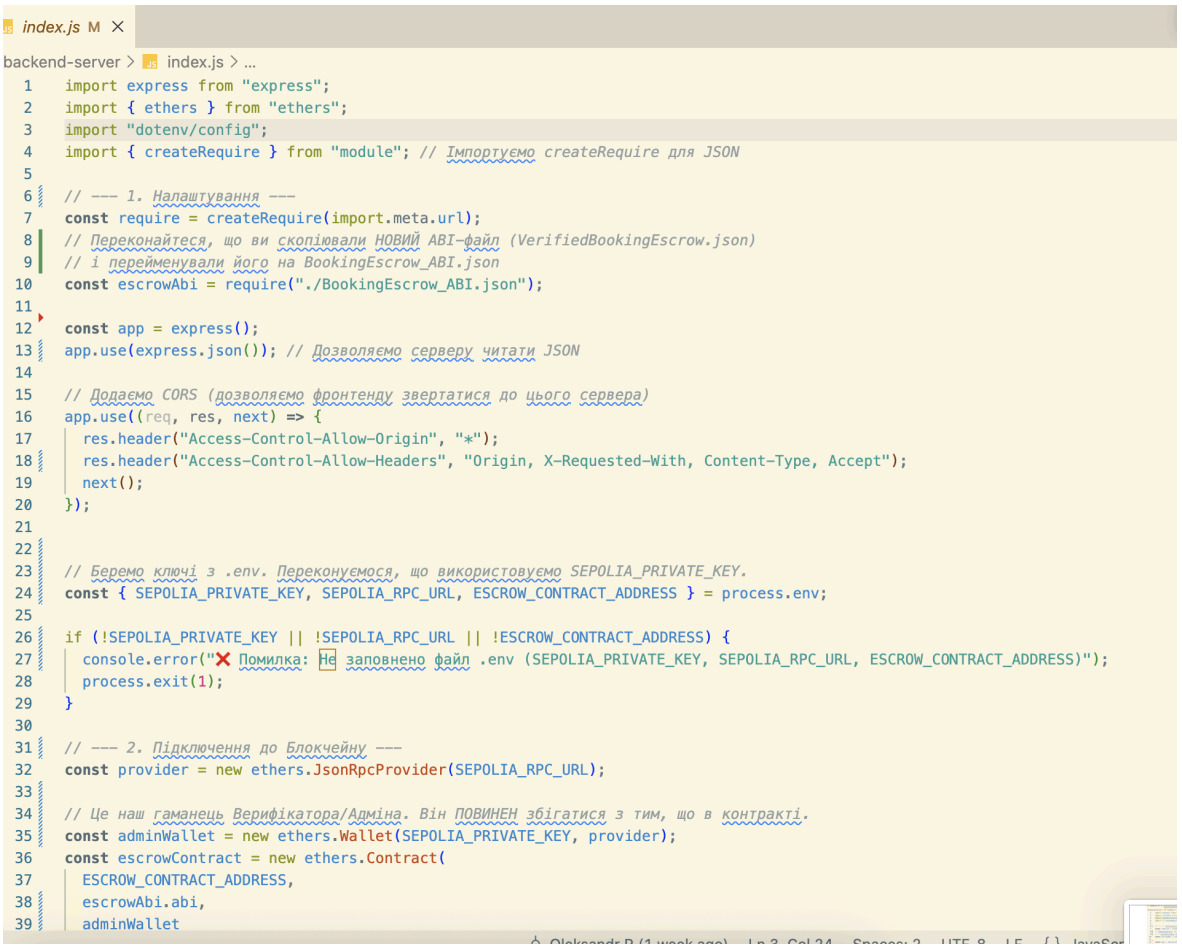
blockchain-escrow > scripts > deploy.ts > main

```
import pkg from "hardhat";
const { ethers } = pkg;

// 1. ІМПОРТУЄМО 'dotenv', щоб завантажити .env файл
// Це дозволить нам отримати доступ до process.env.SEPOLIA_PRIVATE_KEY
import "dotenv/config";

async function main() {
  console.log("Deploying Verifier-version of BookingEscrow contract...");

  // 2. ОТРИМУЄМО АДРЕСУ ВЕРИФІКАТОРА (АДМІНА)
  // Беремо приватний ключ з того ж .env файлу, який використовує hardhat
  const adminPrivateKey = process.env.SEPOLIA_PRIVATE_KEY;
  if (!adminPrivateKey) {
    throw new Error("SEPOLIA_PRIVATE_KEY не знайдено у .env файлі");
  }

  // Створюємо гаманець Адміна/Верифікатора
  const adminWallet = new ethers.Wallet(adminPrivateKey);
  const verifierAddress = adminWallet.address;

  console.log(`Гаманець Адміна (Верифікатора) визначено: ${verifierAddress}`);

  // Отримуємо "фабрику" контракту
  // ВАЖЛИВО: Переконайтеся, що назва тут "VerifiedBookingEscrow"
  const BookingEscrow = await ethers.getContractFactory("VerifiedBookingEscrow");

  // 3. РОЗГОРТАЄМО, ПЕРЕДАЮЧИ АДРЕСУ В КОНСТРУКТОР
  const bookingEscrow = await BookingEscrow.deploy(verifierAddress);

  // Чекаємо на завершення розгортання
  await bookingEscrow.waitForDeployment();
```

Додаток Г. (index.js)

Ключовий файл бекенду. Він показує:

- Підключення до розгорнутого контракту за допомогою ethers.js (адреса + ABI).
- Керування гаманцем "Адміністратора" (adminWallet).
- Реалізація API-ендпоінтів (/api/create-escrow) для зв'язку з фронтендом.
- Реалізація прослуховування подій блокчейну (escrowContract.on(...)).



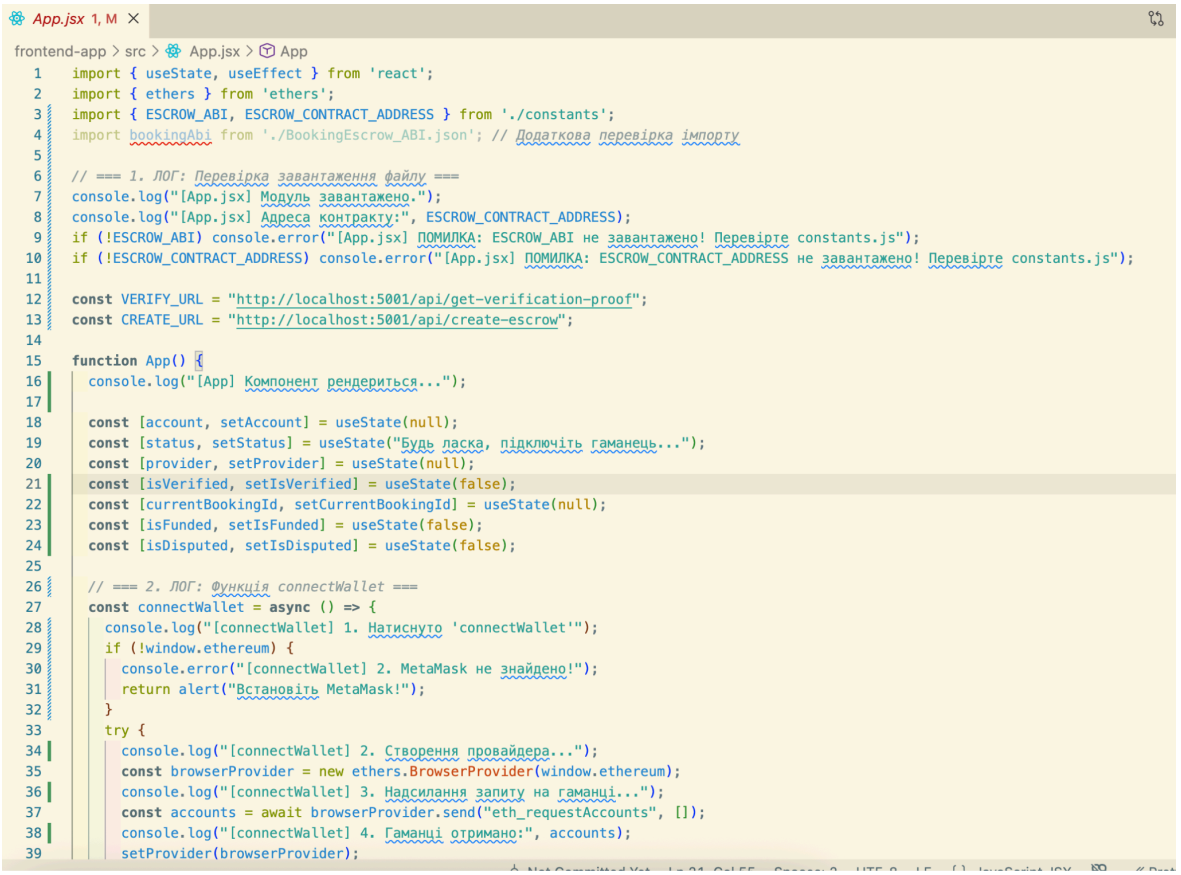
```
index.js M
backend-server > index.js > ...
1   import express from "express";
2   import { ethers } from "ethers";
3   import "dotenv/config";
4   import { createRequire } from "module"; // Імпортуємо createRequire для JSON
5
6   // --- 1. Налаштування ---
7   const require = createRequire(import.meta.url);
8   // Переконайтеся, що ви скопіювали НОВИЙ ABI-файл (VerifiedBookingEscrow.json)
9   // і перейменували його на BookingEscrow_ABI.json
10  const escrowAbi = require("./BookingEscrow_ABI.json");
11
12  const app = express();
13  app.use(express.json()); // Дозволяємо серверу читати JSON
14
15  // Додаємо CORS (дозволяємо фронтенду звертатися до цього сервера)
16  app.use((req, res, next) => {
17    res.header("Access-Control-Allow-Origin", "*");
18    res.header("Access-Control-Allow-Headers", "Origin, X-Requested-With, Content-Type, Accept");
19    next();
20  });
21
22
23  // Беремо ключі з .env. Переконуємося, що використовуємо SEPOLIA_PRIVATE_KEY.
24  const { SEPOLIA_PRIVATE_KEY, SEPOLIA_RPC_URL, ESCROW_CONTRACT_ADDRESS } = process.env;
25
26  if (!SEPOLIA_PRIVATE_KEY || !SEPOLIA_RPC_URL || !ESCROW_CONTRACT_ADDRESS) {
27    console.error("❌ Помилка: Не заповнено файл .env (SEPOLIA_PRIVATE_KEY, SEPOLIA_RPC_URL, ESCROW_CONTRACT_ADDRESS)");
28    process.exit(1);
29  }
30
31  // --- 2. Підключення до Блокчейну ---
32  const provider = new ethers.JsonRpcProvider(SEPOLIA_RPC_URL);
33
34  // Це наш гаманець Верифікатора/Адміна. Він ПОВИНЕН збігатися з тим, що в контракті.
35  const adminWallet = new ethers.Wallet(SEPOLIA_PRIVATE_KEY, provider);
36  const escrowContract = new ethers.Contract(
37    ESCROW_CONTRACT_ADDRESS,
38    escrowAbi.abi,
39    adminWallet
```

Додаток І. (App.jsx)

Демонструє повний користувацький сценарій (E2E). Цей файл показує:

- Підключення до MetaMask (connectWallet).

- Гібридну логіку: спочатку виклик вашого Web2-бекенду (fetch(BACKEND_URL)), а потім…
- ...прямий виклик Web3-контракту (escrowContract.fundEscrow(...)) для оплати.
- Обробку помилок та оновлення статусу для користувача.



```
App.jsx 1, M
frontend-app > src > App.jsx > App
1   import { useState, useEffect } from 'react';
2   import { ethers } from 'ethers';
3   import { ESCROW_ABI, ESCROW_CONTRACT_ADDRESS } from './constants';
4   import bookingAbi from './BookingEscrow_ABI.json'; // Додаткова перевірка імпорту
5
6   // === 1. ЛОГ: Перевірка завантаження файлу ===
7   console.log("[App.jsx] Модуль завантажено.");
8   console.log("[App.jsx] Адреса контракту:", ESCROW_CONTRACT_ADDRESS);
9   if (!ESCROW_ABI) console.error("[App.jsx] ПОМИЛКА: ESCROW_ABI не завантажено! Перевірте constants.js");
10  if (!ESCROW_CONTRACT_ADDRESS) console.error("[App.jsx] ПОМИЛКА: ESCROW_CONTRACT_ADDRESS не завантажено! Перевірте constants.js");
11
12  const VERIFY_URL = "http://localhost:5001/api/get-verification-proof";
13  const CREATE_URL = "http://localhost:5001/api/create-escrow";
14
15  function App() {
16    console.log("[App] Компонент рендериться...");
17
18    const [account, setAccount] = useState(null);
19    const [status, setStatus] = useState("Будь ласка, підключіть гаманець...");
20    const [provider, setProvider] = useState(null);
21    const [isVerified, setIsVerified] = useState(false);
22    const [currentBookingId, setCurrentBookingId] = useState(null);
23    const [isFunded, setIsFunded] = useState(false);
24    const [isDisputed, setIsDisputed] = useState(false);
25
26    // === 2. ЛОГ: Функція connectWallet ===
27    const connectWallet = async () => {
28      console.log("[connectWallet] 1. Натиснуто 'connectWallet'");
29      if (!window.ethereum) {
30        console.error("[connectWallet] 2. MetaMask не знайдено!");
31        return alert("Встановіть MetaMask!");
32      }
33      try {
34        console.log("[connectWallet] 2. Створення провайдера...");
35        const browserProvider = new ethers.BrowserProvider(window.ethereum);
36        console.log("[connectWallet] 3. Надсилання запиту на гаманці...");
37        const accounts = await browserProvider.send("eth_requestAccounts", []);
38        console.log("[connectWallet] 4. Гаманці отримано:", accounts);
39        setProvider(browserProvider);
```

Додаток Д. (Constants.js)

Це "міст", який з'єднує фронтенд з блокчейном. Показує, передачу адресу контракту та його ABI у клієнтський додаток.

constants.js M

frontend-app > src > constants.js > ...

```
1  // 1. ІМПОРТУЄМО ABI
2  // Цей рядок "зламається", якщо файл 'BookingEscrow_ABI.json'
3  // (новий) не лежить у папці 'frontend-app/src/'
4  import escrowAbi from './BookingEscrow_ABI.json';
5
6  // === ВАША ОСТАННЯ АДРЕСА КОНТРАКТУ ===
7  export const ESCROW_CONTRACT_ADDRESS = "0xFdb5a8397B033eB2aCA6b1C2e2bEDcE52aB63532";
8  // ABI контракту
9  export const ESCROW_ABI = escrowAbi.abi;
```