Національний лісотехнічний університет України
(повне найменування вищого навчального закладу)
Навчально-науковий інститут комп'ютерних наук та інформаційних технологій
(повне найменування інституту, назва факультету (відділення))
Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

Магістерська кваліфікаційна робота

другий (магістерський)
(рівень вищої освіти)

на тему: Бізнес-орієнтована платформа для транзакцій між крипто- та фіатними валютами

Виконав: студент 6 курсу групи КН-62м
спеціальності
122 "Комп'ютерні науки"
(шифр і назва напряму підготовки, спеціальності)

Лютий Назарій Ігорович
(прізвище та ініціали)

Керівник Думанський О.І.
(прізвище та ініціали)

Рецензент Карашевський В.Б.
(прізвище та ініціали)

Львів – 2025

Національний лісотехнічний університет України

ННІ комп'ютерних наук та інформаційних технологій

Кафедра _ комп'ютерних наук

Рівень вищої освіти _другий (магістерський)

Спеціальність _122 «Комп'ютерні науки»

(код і повна назва)

ЗАТВЕРДЖУЮ:
Завідувач кафедри комп'ютерних наук
_____ Борецька І.Б.
"10" _____ 2025 року

ЗАВДАННЯ

НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Лютий Назарій Ігорович

(прізвище, ім'я, по батькові)

1. Тема роботи Бізнес-орієнтована платформа для транзакцій між крипто- та фіатними валютами

Керівник роботи Думанський Остап Іванович канд. фіз-матем. наук, доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від "29" квітня 2025 року № С-288

2. Термін подання студентом роботи 10.12.2025

3. Вихідні дані до роботи Бізнес-орієнтована платформа для транзакцій між крипто- та фіатними валютами

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)

Розділ 1. Стан проблемної області

Розділ 2. Інформаційне забезпечення

Розділ 3. Математичне забезпечення

Розділ 4. Програмне забезпечення

Розділ 5. Розроблення стартап-проєкту

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Підготовка матеріалу до доповіді.

6. Дата видачі завдання 01.05.2025

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1.	Огляд літератури згідно досліджуваної теми. Збір необхідних матеріалів	02.05.2025-22.05.2025	Виконано
2.	Постановка задачі та її формалізація	23.05.2025-08.06.2025	Виконано
3.	Виконання вхідного етапу технології	09.06.2025-20.07.2025	Виконано
4.	Реалізація фундаментальних алгоритмів проєкту	21.07.2025-29.09.2025	Виконано
5.	Виконання етапу відлагодження проєкту	30.09.2025-13.11.2025	Виконано
6.	Оформлення записки до дипломного проєкту	14.11.2025-04.12.2025	Виконано

Студент _____ Лютий Н.І.
(підпис) (прізвище, ініціали)

Керівник роботи _____ Думанський О.І.
(підпис) (прізвище, ініціали)

АНОТАЦІЯ

Магістерська робота містить 91 сторінку пояснювальної записки, 9 рисунків, 1 таблицю, 1 додаток, 58 джерел.

У роботі розроблено програмне та алгоритмічне забезпечення бізнес-орієнтованої платформи для виконання транзакцій між криптовалютами та фіатними грошима. Платформа реалізована на основі мікросервісної архітектури та поєднує інструменти обміну, моніторингу та аналітики, що забезпечують прозорість і безпеку операцій. У процесі розроблення застосовано методи математичного моделювання, регресійного аналізу та технології обробки природної мови (NLP) для автоматизації аналізу транзакційних даних і перевірки користувачів.

Запропонована система може використовуватися фінансовими установами, біржами або бізнес-клієнтами для оптимізації крипто-фіатних операцій і підвищення ефективності процесів контролю комплаєнсу.

Ключові слова: криптовалюта, фіатні валюти, транзакції, DevOps, NLP, мікросервісна архітектура, безпека, фінансові технології.

ABSTRACT

The master's thesis consists of 91 pages of explanatory note, 9 figures, 1 table, 1 appendice and 58 literary sources.

The thesis presents the development of algorithmic and software solutions for a business-oriented platform designed to process transactions between cryptocurrencies and fiat currencies. The platform is based on a microservice architecture and integrates exchange, monitoring, and analytics tools to ensure transparency and data security. Mathematical modeling, regression analysis, and natural language processing (NLP) methods were applied to automate transaction data analysis and user verification.

The developed system can be used by financial institutions, exchanges, and business clients to optimize crypto-fiat operations and improve compliance control efficiency.

Keywords: cryptocurrency, fiat currencies, transactions, DevOps, NLP, microservice architecture, security, fintech.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

NLP – Обробка природньої мови (Natural language processing)

AI – Штучний інтелект (Artificial intelligence)

Micronaut – Фреймворк для веб-застосунку Java

Guice – Бібліотека для контролю та впровадження залежностей між класами

CPU – Центральний процесор (Central processing unit)

GPU – Графічний процесор (graphics processing unit)

Saas - Програмне забезпечення як сервіс (software as a service)

JPA – Бібліотека для керування і доступу до бази даних (Java Persistence Api)

REST – Принцип побудови зв’язку між клієнтом та сервером (Representational state transfer)

WS – Вебсокет, протокол для постійного з’єднання між клієнтом та сервером у режимі реального часу

ВСТУП

Сучасний етап розвитку глобальної фінансової системи характеризується стрімкою трансформацією, що супроводжується конвергенцією традиційних банківських інструментів і технологій розподіленого реєстру. Криптовалютний ринок, капіталізація якого зросла з 1,65 трильйона доларів на початку 2024 року до 3,76 трильйона доларів наприкінці періоду, перетворився з інструменту спекулятивних операцій на повноцінний клас активів з інституційною підтримкою. Впровадження спотових Bitcoin ETF у 2024 році засвідчило початок якісно нового етапу прийняття цифрових активів корпоративним сектором. Водночас інтеграція криптовалют у операційну діяльність підприємств стикається з суттєвими бар'єрами: регуляторною невизначеністю, високою волатильністю, технічними складнощами та відсутністю спеціалізованих інструментів для бізнес-застосувань.

Аналіз існуючих платформ для обміну криптовалютних і фіатних активів виявляє системні прогалини у функціональному забезпеченні потреб бізнес-сегменту. Провідні біржі Binance, Coinbase, Kraken орієнтовані переважно на роздрібних трейдерів або великих інституційних клієнтів, залишаючи незадоволеними потреби малого та середнього підприємництва. Відсутні інтегровані інструменти автоматизації бухгалтерського обліку, синхронізації з корпоративними ERP-системами та управління волатільністю у контексті комерційних транзакцій. Ця ситуація обґрунтовує необхідність створення спеціалізованої бізнес-орієнтованої платформи, що поєднає надійність регульованої фінансової інфраструктури з гнучкістю блокчейн-технологій.

Об'єктом дослідження є процеси обміну та конвертації між криптовалютними та фіатними валютами у контексті бізнес-транзакцій.

Предметом дослідження виступають математичні моделі, алгоритми та програмні рішення для реалізації бізнес-орієнтованої платформи транзакцій між криптовалютами та фіатними грошима.

Мета роботи полягає у розробленні архітектури, математичного та програмного забезпечення бізнес-орієнтованої платформи для транзакцій між

криптовалютними та фіатними валютами, що забезпечує інтеграцію цифрових активів у операційну діяльність підприємств.

Для досягнення поставленої мети визначено такі завдання:

- проаналізувати сучасний стан криптовалютного ринку та обґрунтувати необхідність створення спеціалізованої платформи;
- дослідити функціональні можливості існуючих сервісів та визначити прогалини у забезпеченні потреб корпоративних користувачів;
- розробити інформаційну модель платформи, що включає бізнес-процеси, структуру даних та архітектуру взаємодії модулів;
- формалізувати процеси обміну валют через математичні моделі визначення курсів, оцінювання ризиків і оптимізації транзакцій;
- реалізувати програмне забезпечення з використанням сучасних технологій та механізмів безпеки;
- розробити концепцію стартап-проєкту з аналізом ринкових можливостей та бізнес-моделі.

Наукова новизна роботи полягає у розробленні комплексної математичної моделі ціноутворення для крипто-фіатних транзакцій, що інтегрує агрегацію котирувань, адаптивний спред та багаторівневу структуру комісій. Удосконалено методи оцінювання ризиків через поєднання VaR-методології з EWMA-моделюванням волатильності. Запропоновано алгоритми оптимізації виконання транзакцій через динамічне розбиття обсягів та інтелектуальне балансування навантаження.

Практичне значення роботи визначається створенням функціональної архітектури та програмної реалізації бізнес-орієнтованої платформи, що забезпечує автоматизацію крипто-фіатних розрахунків для підприємств. Розроблені математичні моделі та алгоритми можуть використовуватися фінтех-компаніями для оптимізації курсів конвертації та управління ризиками. Результати дослідження формують технологічну основу для подальшого розвитку фінансових сервісів, що поєднують традиційні та цифрові активи у єдиній екосистемі.

РОЗДІЛ 1. СТАН ПРОБЛЕМНОЇ ОБЛАСТІ

1.1 Сучасний стан розвитку криптовалютного ринку

Криптовалютний ринок демонструє стрімку динаміку розвитку, що підтверджується зростанням загальної капіталізації цифрових активів. Протягом 2024 року ринок зріс більш ніж удвічі – з початкової капіталізації 1,65 трильйона доларів США до 3,76 трильйона доларів до кінця грудня. Це зростання відбувалося на тлі значних змін у регуляторному середовищі та інституційному сприйнятті цифрових активів. Станом на четвертий квартал 2024 року професійні інвестори з активами понад 100 мільйонів доларів володіли Bitcoin ETF на суму 27,4 мільярда доларів, що становить збільшення на 114% порівняно з попереднім. Трансформація ринку характеризується переходом від спекулятивного інструменту до інвестиційного активу з інституційною підтримкою та регуляторним визнанням [1].

Впровадження спотових Bitcoin ETF у 2024 році стало переломним моментом для інституційного прийняття криптовалют. Затвердження Комісією з цінних паперів та бірж США спотових Bitcoin ETF у 2024 році забезпечило необхідну регуляторну ясність. Приблизно 500 інституційних інвесторів розмістили кошти в спотові Bitcoin ETF протягом першого кварталу 2024 року. Цей механізм інвестування дозволив традиційним фінансовим установам отримати експозицію до криптовалют без необхідності прямого володіння цифровими активами та управління пов'язаними з цим технічними та регуляторними ризиками. Зростання інституційного інтересу створило передумови для подальшої інтеграції криптовалют у глобальну фінансову систему [2]. Географічне поширення використання криптовалют демонструє неоднорідність адаптації цифрових активів у різних регіонах світу. Понад 6,5 мільйона українців, що становить близько 15-17% дорослого населення, володіють або використовують криптовалюту. Україна посідає високі позиції в глобальних рейтингах прийняття криптовалют, що свідчить про зростаючу роль цифрових активів у національній економіці [3]. Водночас спостерігається диференціація між розвиненими та країнами, що розвиваються, у підходах до регулювання та

використання криптовалют. Ця географічна варіативність створює різні бізнес-моделі та вимоги до платформ для обміну цифровими активами.

Еволюція ринкової інфраструктури супроводжується розширенням сегменту стейблкоїнів, що виконують функцію моста між традиційними та криптовалютними фінансами. У грудні 2024 року капіталізація стейблкоїнів досягла понад 200 мільярдів доларів, а прогнози передбачають перевищення 400 мільярдів доларів у 2025 році. Стейблкоїни забезпечують стабільність вартості та ліквідність, необхідні для здійснення комерційних транзакцій та міжнародних переказів. Їх зростання відображає попит на інструменти, що поєднують переваги блокчейн-технологій з прогнозованістю традиційних валют. Розвиток цього сегменту стимулює появу нових бізнес-моделей, орієнтованих на забезпечення безперешкодного обміну між фіатними та цифровими валютами [3].

Технологічна інфраструктура криптовалютного ринку продовжує вдосконалюватися через впровадження рішень масштабування та покращення користувацького досвіду. Розвиток протоколів другого рівня, децентралізованих бірж та кастодіальних сервісів розширює функціональні можливості цифрових активів. Паралельно зростають виклики у сфері кібербезпеки та регуляторної відповідності, що потребує комплексних рішень для захисту користувачів та забезпечення прозорості операцій.

Сучасний стан ринку характеризується балансуванням між інноваційним потенціалом технологій розподіленого реєстру та необхідністю створення надійної, регульованої екосистеми для масового прийняття криптовалют у бізнес-середовищі.

1.2 Проблеми інтеграції криптовалют у бізнес-середовище

Регуляторна невизначеність залишається однією з найсуттєвіших перешкод для повноцінного впровадження криптовалют у корпоративну практику. Відсутність уніфікованих правових норм створює складнощі для бізнесу у питаннях бухгалтерського обліку, оподаткування та дотримання вимог фінансового моніторингу. В Україні ситуація характеризується неповною імплементацією законодавства про віртуальні активи, що призводить до правової невизначеності

щодо статусу криптовалют та обов'язків суб'єктів господарювання. Компанії стикаються з необхідністю самостійно інтерпретувати норми законодавства, що збільшує юридичні ризики та потенційну відповідальність перед регуляторними органами. Така ситуація уповільнює розвиток інфраструктури та стримує інвестиції у криптовалютний сектор. Волатильність цінових коливань криптовалют становить суттєву проблему для бізнес-операцій, що потребують прогнозованості фінансових потоків. Біткоїн демонструє волатильність у три-чотири рази вищу порівняно з різними індексами акцій США протягом періоду з 2020 по 2024 рік. Така нестабільність курсів ускладнює процес ціноутворення товарів та послуг, планування бюджетів та управління фінансовими ризиками [4].

Підприємства, що приймають криптовалютні платежі, змушені впроваджувати механізми миттєвої конвертації у фіатні гроші або використовувати складні стратегії хеджування для мінімізації збитків від несприятливих змін курсу. Необхідність постійного моніторингу ринкової ситуації та швидкого реагування на цінові коливання вимагає додаткових ресурсів та спеціалізованої експертизи [5].

Технічні бар'єри входу на криптовалютний ринок створюють додаткові виклики для традиційного бізнесу, що не має досвіду роботи з блокчейн-технологіями. Інтеграція криптовалютних платежів потребує розуміння принципів роботи розподілених систем, управління приватними ключами та забезпечення кібербезпеки цифрових активів. Ліцензований криптовалютний платіжний шлюз вирішує питання безпеки, відповідності нормативним вимогам та волатильності, дозволяючи бізнесу зосередитися на зростанні та спрощуючи процес платежів. Відсутність внутрішніх компетенцій змушує компанії звертатися до зовнішніх постачальників послуг, що збільшує операційні витрати та створює залежність від третіх сторін. Складність технологій також ускладнює процес навчання персоналу та подальшу підтримку криптовалютної інфраструктури.

Проблема відповідності вимогам боротьби з відмиванням коштів та фінансуванням тероризму набуває особливого значення у контексті криптовалютних операцій. Зростаюча легітимність криптовалют вплинула на необхідність відстеження клієнтів та транзакцій регуляторними органами, оскільки офіцери з

комплаєнсу часто є новачками у відстеженні потоків криптовалютних активів та потребують глибшого розуміння походження, маршрутів та призначення транзакцій [6]. Псевдонімний характер блокчейн-транзакцій ускладнює ідентифікацію бенефіціарних власників та перевірку джерел походження коштів. Бізнес змушений впроваджувати складні процедури верифікації клієнтів та моніторингу транзакцій, що вимагає інвестицій у спеціалізоване програмне забезпечення та навчання співробітників. Недотримання цих вимог може призвести до санкцій з боку регуляторів та репутаційних втрат.

Обмежена взаємосумісність між різними блокчейн-мережами та відсутність стандартизованих протоколів ускладнює створення ефективних бізнес-рішень. Фрагментація криптовалютної екосистеми примушує компанії підтримувати інтеграцію з численними блокчейнами, кожен з яких має власні технічні особливості та вимоги. Це збільшує складність розробки, тестування та підтримки програмного забезпечення, а також підвищує ймовірність виникнення помилок та вразливостей безпеки. Відсутність єдиних стандартів для обміну даними між платформами перешкоджає створенню комплексних рішень, що могли б забезпечити безперешкодну інтеграцію криптовалют у існуючі бізнес-процеси та інформаційні системи підприємств [7].

1.3 Аналіз існуючих платформ для обміну крипто та фіатних валют

Сучасний ринок криптовалютних бірж характеризується значною диференціацією платформ за функціоналом, цільовою аудиторією та бізнес-моделями. Станом на сьогодні відстежується 203 криптовалютні біржі із загальним 24-годинним обсягом торгів 246 мільярдів доларів, при цьому три найбільші біржі – Binance, MEXC та Gate – домінують на ринку. Централізовані платформи забезпечують основний обсяг ліквідності та пропонують широкий спектр торгових інструментів, від спотової торгівлі до деривативів [8]. Водночас спостерігається зростання сегменту децентралізованих бірж, що працюють на основі смарт-контрактів без посередників. Різноманітність архітектурних рішень відображає різні

підходи до балансування між зручністю використання, безпекою та рівнем децентралізації.

Платформа Binance представляє найбільшу за обсягами торгів біржу з екосистемою, що охоплює понад 500 криптовалют та підтримку 19 фіатних валют. Binance надає користувачам по всьому світу можливість торгувати понад 500 валютами та 19 фіатними валютами. Платформа орієнтована на масовий ринок та пропонує низькі торгові комісії, що починаються від 0,1%, що робить її привабливою для активних трейдерів. Функціонал включає спотову та маржинальну торгівлю, ф'ючерси, стейкінг та власний блокчейн для випуску токенів. Проте глобальна присутність Binance супроводжується регуляторними викликами у різних юрисдикціях, що впливає на доступність послуг для користувачів окремих країн та обмежує можливості для інституційних клієнтів у регульованих ринках [9]. Coinbase позиціонується як біржа для початківців з акцентом на простоту використання та регуляторну відповідність. Coinbase пропонує понад 120 валют як для роздрібних, так і для інституційних інвесторів. Платформа отримала ліцензію на роботу у Сполучених Штатах та є публічною компанією на NASDAQ, що забезпечує додатковий рівень довіри з боку традиційних інвесторів. Інтерфейс Coinbase спрощений для непрофесійних користувачів, проте це досягається за рахунок вищих комісій порівняно з конкурентами [10].

Для бізнес-клієнтів компанія розвиває окремі продукти, включаючи кастодіальні послуги та інфраструктуру для інституційної торгівлі. Обмеження платформи полягають у меншій кількості доступних криптовалют та географічній концентрації переважно на північноамериканському ринку.

Kraken виділяється фокусом на безпеці та розширеному функціоналі для досвідчених трейдерів. Функції Kraken включають маржинальну торгівлю, ф'ючерсну торгівлю, торгівлю NFT, винагороди за стейкінг та API-торгівлю. Платформа пропонує глибоку ліквідність та підтримує сім фіатних валют для прямого введення та виведення коштів. Kraken має репутацію однієї з найбезпечніших бірж завдяки багаторівневій системі захисту та відсутності значних інцидентів злому в історії. Для бізнес-користувачів доступні API з високою пропускною здатністю та можливості

автоматизованої торгівлі. Недоліком є менш інтуїтивний інтерфейс порівняно з Coinbase, що може ускладнювати адаптацію для нових користувачів без технічного досвіду.

Регіональні платформи демонструють спеціалізацію на локальних ринках з адаптованим функціоналом та підтримкою місцевих методів оплати. Такі біржі зазвичай мають глибше розуміння регуляторних вимог конкретних юрисдикцій та можуть швидше адаптуватися до змін законодавства. Перевагою регіональних платформ є інтеграція з національними банківськими системами, що забезпечує швидші та дешевші фіатні транзакції порівняно з міжнародними переказами. Водночас обмеженість ліквідності на окремих торгових парах та менший вибір криптовалют може створювати проблеми для користувачів, що потребують доступу до широкого спектру активів. Регіональні біржі часто виконують роль точок входу для локального бізнесу, що робить перші кроки у криптовалютній сфері.

Аналіз існуючих рішень виявляє відсутність платформ, повністю орієнтованих на специфічні потреби бізнес-клієнтів у сегменті малого та середнього підприємництва. Більшість бірж фокусуються або на роздрібних трейдерах, або на великих інституційних клієнтах, залишаючи прогалину у обслуговуванні компаній, що потребують інтеграції криптовалютних розрахунків у операційну діяльність. Існуючі рішення не завжди забезпечують необхідний рівень автоматизації, інтеграції з корпоративними системами обліку та звітності, а також не враховують специфіку комплаєнс-процедур для бізнесу [10]. Комісійні структури оптимізовані під активну торгівлю, але не під регулярні конвертаційні операції, що є типовими для підприємств. Це створює потребу у спеціалізованих платформах, що поєднують надійність та регуляторну відповідність із функціоналом, адаптованим під бізнес-процеси комерційних організацій.

1.4 Порівняльна характеристика функціональних можливостей існуючих сервісів

Комісійна політика платформ для обміну криптовалют демонструє широкий діапазон підходів до монетизації, що безпосередньо впливає на економічну доцільність використання сервісів для бізнесу. Kraken стягує комісію мейкера близько 0,25% та комісію тейкера приблизно 0,40%, проте ці комісії зменшуються для високооб'ємних трейдерів до 0,00% мейкер/0,10% тейкер. Binance пропонує базову ставку 0,10% для обох типів ордерів з можливістю зниження до 0,02% при використанні власного токена платформи. Gemini застосовує комісію за транзакції 1,49% та додаткову комісію за зручність 1,00%, що робить загальні витрати вищими порівняно з багатьма іншими біржами. Диференціація комісій залежить не лише від обсягів торгівлі, але й від методів поповнення рахунку та виведення коштів, що ускладнює порівняння загальних витрат для бізнес-користувачів з різними моделями операційної діяльності [11].

Технічна інтеграція через програмні інтерфейси визначає можливості автоматизації процесів для корпоративних клієнтів. Провідні платформи надають REST API та WebSocket з'єднання для отримання ринкових даних у реальному часі та виконання торгових операцій програмним шляхом. ChangeNOW підтримує понад 1100 криптовалют та більше 100 мереж, пропонуючи API для фіатних он-рампів та офф-рампів, що працює як криптовалютна біржа, гаманець та платіжний шлюз. Документація API, швидкість обробки запитів та ліміти на кількість викликів варіюються між платформами, що впливає на можливості масштабування рішень. Частина бірж обмежує доступ до розширеного функціоналу API лише для верифікованих корпоративних клієнтів, створюючи бар'єри для малого бізнесу. Процедури верифікації користувачів та комплаєнс-вимоги суттєво відрізняються між платформами, що впливає на швидкість онбордингу та рівень анонімності. Більшість централізованих бірж вимагають обов'язкової ідентифікації клієнтів відповідно до регуляторних норм юрисдикцій, де вони оперують. Процес KYC зазвичай включає верифікацію документів, що посвідчують особу, підтвердження адреси та біометричну ідентифікацію через селфі або відеодзвінок [9].

Для корпоративних рахунків додатково потрібні установчі документи, інформація про бенефіціарних власників та джерела походження коштів. Терміни проходження верифікації коливаються від кількох годин до тижнів залежно від завантаженості служб підтримки та складності корпоративної структури клієнта.

Підтримка фіатних валют та методів введення-виведення коштів визначає зручність використання платформ для локальних ринків. Таблиця 1.1 систематизує порівняння функціональних характеристик провідних криптовалютних бірж за параметрами, що мають значення для бізнес-застосування.

Таблиця 1.1 – Порівняльна характеристика криптовалютних платформ

Платформа	Кількість криптовалют	Фіатні валюти	Комісія спот-торгівлі	Підтримка API	KYC вимоги	Ліцензування
Binance	500+	19	0,10%	REST, WebSocket	Обов'язкове	Обмежене в окремих юрисдикціях
Coinbase	120+	8	0,40-0,60%	REST	Обов'язкове	Повне регулювання США
Kraken	200+	7	0,25-0,40%	REST, WebSocket	Обов'язкове	Регульоване в ЄС та США
ChangeNOW	1100+	12	Від 0,25%	REST	Опціонально	Non-custodial
Gemini	70+	USD, GBP, EUR, SGD	1,49% + 1,00%	REST, WebSocket	Обов'язкове	Повне регулювання США

Дані таблиці демонструють, що жодна з платформ не пропонує оптимального поєднання характеристик для бізнес-сегменту. Платформи з найширшим вибором активів часто мають регуляторні обмеження, тоді як найбільш регульовані біржі пропонують обмежений асортимент та вищі комісії.

Функціональні можливості щодо автоматизації бізнес-процесів залишаються недостатньо розвиненими на більшості платформ. Відсутні інтегровані інструменти для автоматичного виставлення рахунків у криптовалюті, синхронізації з бухгалтерськими системами та генерації звітності у форматах, прийнятних для податкових органів. Більшість бірж орієнтовані на трейдинг, а не на використання криптовалют як засобу розрахунків, що відображається у структурі комісій та доступному функціоналі. Платформи не забезпечують інструментів для управління волатильністю у контексті комерційних операцій, таких як автоматична конвертація отриманих коштів у стейблкоїни або фіат. Ці обмеження створюють потребу у розробці спеціалізованих рішень, що враховують специфіку використання криптовалют саме для бізнес-транзакцій, а не спекулятивної торгівлі.

Рівень безпеки та страхування коштів користувачів варіюється між платформами та залежить від їх архітектури. Централізовані біржі зберігають приватні ключі користувачів, що створює ризики масштабних злочинів у разі компрометації системи. Деякі платформи пропонують страхування коштів на холодних гаманцях, проте умови покриття часто обмежені та не поширюються на збитки від несанкціонованого доступу до облікового запису користувача. Децентралізовані біржі усувають ризик централізованого зберігання, але покладають повну відповідальність за безпеку коштів на користувача. Для бізнесу це означає необхідність впровадження власних процедур управління ключами та контролю доступу, що вимагає технічної експертизи та додаткових інвестицій у інфраструктуру безпеки.

1.5 Обґрунтування необхідності розроблення бізнес-орієнтованої платформи

Аналіз існуючих рішень виявляє системні прогалини у функціоналі, що перешкоджають ефективному використанню криптовалют у щоденній діяльності комерційних організацій. Більшість платформ проектувалися як інструменти для трейдингу та інвестування, де головною метою є максимізація прибутку від цінових коливань, а не забезпечення надійних розрахунків між контрагентами. Бізнес

потребує стабільності, прогнозованості та інтеграції з існуючими фінансовими процесами, тоді як трейдингові платформи оптимізовані під швидкість виконання угод та доступ до широкого спектру спекулятивних інструментів. Відсутність спеціалізованих рішень примушує підприємства адаптувати непридатні інструменти, що призводить до неефективності операцій, підвищених ризиків та додаткових витрат на підтримку технічної інфраструктури [12, с. 22-24].

Регуляторні вимоги та процедури комплаєнсу для бізнес-користувачів суттєво відрізняються від потреб роздрібних клієнтів, проте існуючі платформи не враховують цю специфіку. Компанії зобов'язані вести детальну документацію всіх фінансових операцій, підтверджувати економічну доцільність транзакцій та забезпечувати прозорість для податкових органів. Стандартні інструменти звітності криптовалютних бірж не відповідають форматам бухгалтерської та податкової звітності, що вимагає ручного перенесення даних та збільшує ймовірність помилок. Процеси верифікації юридичних осіб на універсальних платформах часто є надмірно складними або, навпаки, недостатньо ретельними для задоволення потреб регульованих галузей.

Спеціалізована платформа може інтегрувати комплаєнс-процедури безпосередньо у бізнес-процеси, автоматизуючи збір необхідної інформації та генерацію документації [13]. Економічна модель використання криптовалют у бізнесі передбачає інші пріоритети порівняно з активною торгівлею на біржах. Підприємства здійснюють регулярні конвертаційні операції відносно невеликих обсягів як частину розрахунків з постачальниками або клієнтами, де головним критерієм є надійність та передбачуваність витрат, а не мінімізація спреду на великих угодах. Існуючі комісійні структури, що заохочують високочастотну торгівлю великими обсягами, не є оптимальними для таких сценаріїв використання. Бізнес-орієнтована платформа може запропонувати модель тарифікації, адаптовану під паттерни корпоративних транзакцій, з фіксованими або передбачуваними комісіями за регулярні операції та інструментами для управління валютними ризиками через автоматичну конвертацію або лімітні ордери на певні періоди.

Технічна інтеграція криптовалютних розрахунків у корпоративні інформаційні системи вимагає спеціалізованих інструментів, яких бракує на універсальних біржових платформах. Підприємства використовують ERP-системи, бухгалтерське програмне забезпечення та інструменти управління грошовими потоками, що повинні безперешкодно взаємодіяти з криптовалютною інфраструктурою. Необхідна автоматизація процесів виставлення рахунків у цифрових активах, відстеження статусу платежів у блокчейні та автоматичне відображення операцій у бухгалтерському обліку з коректним розрахунком курсових різниць. Розроблення спеціалізованої платформи дозволить реалізувати ці можливості як інтегровану систему, а не набір розрізнених інструментів, що потребують ручної координації. Така платформа може стати мостом між традиційними фінансовими процесами та інноваційними можливостями блокчейн-технологій, знижуючи бар'єри входу для бізнесу у криптовалютну економіку.

Висновки до розділу 1

Проведений аналіз засвідчує трансформацію криптовалютного ринку у повноцінний елемент глобальної фінансової системи з капіталізацією 3,76 трильйона доларів, проте виявлено суттєві перешкоди для його інтеграції у бізнес-середовище: регуляторну невизначеність, високу волатильність, технічні бар'єри та складність дотримання вимог фінансового моніторингу. Порівняльний аналіз існуючих платформ продемонстрував їх орієнтацію на трейдингові операції та відсутність спеціалізованого функціоналу для бізнес-потреб, що підтверджується систематизацією характеристик провідних бірж у таблиці 1.1. Результати дослідження обґрунтовують необхідність розроблення спеціалізованої бізнес-орієнтованої платформи, що забезпечить інтеграцію криптовалютних транзакцій у операційну діяльність підприємств через адаптований функціонал, прозорі тарифи та інструменти управління ризиками.

РОЗДІЛ 2. ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Аналіз бізнес-процесів платформи та основних учасників транзакцій

Бізнес-орієнтована платформа для транзакцій між крипто- та фіатними валютами функціонує як багатостороння екосистема, де взаємодіють різні категорії учасників з відмінними цілями та моделями поведінки. Основними учасниками виступають корпоративні клієнти, що використовують платформу для операційних розрахунків, індивідуальні користувачі, які здійснюють епізодичні конвертаційні операції, постачальники ліквідності, що забезпечують можливість виконання угод за конкурентними цінами, та регуляторні органи, які контролюють дотримання вимог фінансового моніторингу. Кожна група учасників має специфічні вимоги до функціоналу, швидкості обробки транзакцій та рівня автоматизації процесів. Платформа виступає посередником, що координує взаємодію між учасниками та забезпечує виконання транзакцій з дотриманням встановлених правил та обмежень.

Центральний бізнес-процес платформи включає послідовність етапів від ініціювання транзакції до остаточного розрахунку та відображення операції в обліку користувача. Процес починається з автентифікації користувача та перевірки достатності коштів на рахунку для виконання операції з урахуванням комісій платформи. Далі система формує заявку на обмін, що містить параметри транзакції: вихідну валюту, цільову валюту, обсяг операції та тип ордеру [14]. Платформа здійснює матчинг заявки з доступною ліквідністю, визначає оптимальний курс конвертації та резервує кошти на час виконання транзакції. Після успішного виконання обміну відбувається блокчейн-транзакція для криптовалютної частини операції та банківський переказ для фіатної складової, з подальшим оновленням балансів рахунків учасників.

Корпоративні клієнти взаємодіють з платформою через розширений функціонал, що включає можливості автоматизації регулярних операцій та інтеграції з внутрішніми інформаційними системами. Ці користувачі реєструються як юридичні особи з проходженням посиленої процедури верифікації, що передбачає надання установчих документів, інформації про бенефіціарних власників та підтвердження

легітимності джерел походження коштів. Після успішної верифікації компанії отримують доступ до API платформи, що дозволяє програмно ініціювати транзакції, отримувати інформацію про курси валют у реальному часі та генерувати звітність для бухгалтерського обліку [15, с. 44]. Корпоративні користувачі можуть налаштовувати автоматичні правила конвертації, встановлювати ліміти на операції та делегувати повноваження різним співробітникам з гранульованим контролем доступу. Постачальники ліквідності виконують роль маркет-мейкерів, що постійно розміщують заявки на купівлю та продаж різних валютних пар, забезпечуючи можливість виконання транзакцій користувачів без значних затримок. Ці учасники підключаються до платформи через спеціалізовані торгові інтерфейси з низькою латентністю та можливістю швидкого оновлення котирувань залежно від ринкової ситуації. Платформа агрегує ліквідність від множини постачальників, формуючи консолідований стакан заявок та визначаючи найкращі ціни виконання для кінцевих користувачів. Взаємодія з постачальниками ліквідності включає процеси моніторингу їх фінансової стійкості, встановлення лімітів експозиції та розрахунків за надані послуги через систему ребейтів або преференційних комісій.

Комплаєнс-процеси пронизують усі етапи взаємодії учасників з платформою та включають процедури протидії відмиванню коштів, верифікації транзакцій на відповідність санкційним спискам та моніторингу підозрілої активності. Система автоматично аналізує паттерни поведінки користувачів, виявляючи аномальні операції, що потребують додаткової перевірки службою безпеки. Для кожної транзакції зберігається повний аудиторський слід, що містить інформацію про учасників, параметри операції, джерела та призначення коштів. Регуляторна звітність генерується автоматично на основі накопичених даних та передається до відповідних органів у встановленому форматі. Платформа забезпечує балансування між вимогами приватності користувачів та необхідністю прозорості для регуляторів через диференційований доступ до інформації залежно від ролі запитувача.

2.2 Визначення інформаційних потоків і структури даних

Інформаційні потоки платформи для обміну криптовалют організовані у вигляді багаторівневої архітектури, де дані циркулюють між зовнішніми джерелами, внутрішніми компонентами системи та кінцевими користувачами. Вхідні потоки формуються з ринкових даних про котирування криптовалют від агрегаторів цін, інформації про стан блокчейн-мереж через взаємодію з нодами, повідомлень про банківські транзакції від платіжних провайдерів та запитів користувачів через веб-інтерфейс або API. Вихідні потоки включають підтвердження виконаних транзакцій, оновлення балансів рахунків, повідомлення про зміну статусу операцій та звітні дані для регуляторних органів. Внутрішні потоки забезпечують синхронізацію стану між різними модулями системи, передачу команд між сервісами та реплікацію даних для забезпечення відмовостійкості [8]. Процеси обробки даних включають етапи валідації запитів, екстракції даних та індексування, що потребують ретельної координації для мінімізації затримок та уникнення дублювання інформації.

Структура даних користувачів організована навколо профілю, що містить ідентифікаційну інформацію, верифікаційні документи, налаштування безпеки та історію взаємодії з платформою. Для фізичних осіб зберігаються персональні дані відповідно до вимог KYC: прізвище, ім'я, дата народження, громадянство, номер документа та адреса проживання. Корпоративні профілі додатково включають реквізити юридичної особи, інформацію про структуру власності, дані контактних осіб та документи, що підтверджують повноваження представників.

Транзакційні дані формують ядро інформаційної системи платформи та включають детальну специфікацію кожної операції обміну. Базова структура транзакції містить унікальний ідентифікатор, тип операції (купівля/продаж, конвертація, поповнення, виведення), вихідну та цільову валюти, обсяги операції, застосований курс конвертації та комісійні платформи [16]. Для криптовалютних операцій зберігаються адреси гаманців відправника та одержувача, хеш транзакції в блокчейні, кількість підтверджень мережею та статус виконання. Фіатні транзакції включають банківські реквізити, номер платіжного доручення, дату валютування та інформацію про проміжних учасників переказу. Кожна транзакція супроводжується

метаданими про ініціатора операції, IP-адресу запиту, геолокацію та результати автоматичної перевірки на відповідність правилам безпеки.

Ринкові дані про котирування та ліквідність зберігаються у високопродуктивних структурах, оптимізованих для швидкого читання та оновлення. Стакан заявок для кожної валютної пари представлено у вигляді впорядкованих списків ордерів на купівлю та продаж з зазначенням ціни, обсягу та часу розміщення. Історичні дані про виконані угоди зберігаються з прив'язкою до часових міток для побудови графіків цін та обчислення технічних індикаторів. Платформа агрегує інформацію від зовнішніх джерел про котирування на інших біржах для визначення справедливої ринкової ціни та виявлення арбітражних можливостей [17].

Комплаєнс-дані організовані для забезпечення можливості швидкого аудиту та формування регуляторної звітності. Система зберігає результати перевірок транзакцій на відповідність санкційним спискам, скоринг ризиків для кожної операції на основі поведінкових паттернів та історію змін статусів верифікації користувачів. Для підозрілих операцій формуються окремі записи з детальним описом підстав для додаткової перевірки, результатами розслідування службою безпеки та прийнятими рішеннями. Структура даних включає зв'язки між транзакціями для відстеження ланцюжків переказів та виявлення схем відмивання коштів. Зберігаються документи, отримані під час верифікації користувачів, з контрольованим доступом відповідно до вимог захисту персональних даних.

Технічні логи та метрики системи формують допоміжний рівень даних, що забезпечує моніторинг працездатності платформи та виявлення аномалій. Реєструються всі звернення до API з параметрами запитів, часом обробки та кодами відповідей для аналізу навантаження та оптимізації продуктивності.

2.3 Опис інформаційної моделі взаємодії між модулями системи

Архітектура платформи побудована за принципом мікросервісної організації, де кожен модуль виконує специфічну функцію та взаємодіє з іншими компонентами через чітко визначені інтерфейси. Модуль автентифікації та управління сесіями забезпечує перевірку ідентичності користувачів, генерацію токенів доступу та

контроль за правами на виконання операцій. Цей компонент взаємодіє з модулем управління користувачами для отримання профільних даних та з модулем безпеки для перевірки підозрілих спроб доступу. Модуль балансів та рахунків відповідає за відстеження коштів користувачів у різних валютах, резервування сум під активні ордери та атомарне оновлення залишків після виконання транзакцій. Взаємодія між модулями відбувається як синхронно через REST API для операцій, що потребують миттєвої відповіді, так і асинхронно через черги повідомлень для процесів, що допускають відкладену обробку [18].

Модуль обробки транзакцій координує виконання операцій обміну та забезпечує узгодженість стану системи. При надходженні запиту на конвертацію валют цей модуль звертається до модуля ціноутворення для отримання актуального курсу, до модуля балансів для перевірки достатності коштів та до модуля комплаєнсу для верифікації відповідності транзакції регуляторним вимогам. Після позитивних перевірок формується запис про транзакцію у базі даних з початковим статусом "очікує виконання", резервуються кошти на рахунку користувача та ініціюється процес виконання. Модуль взаємодіє з блокчейн-шлюзом для відправки криптовалютних транзакцій та з модулем банківських інтеграцій для ініціювання фіатних переказів.

Блокчейн-шлюз функціонує як посередник між платформою та множиною криптовалютних мереж, абстрагуючи специфіку різних протоколів. Цей модуль підтримує підключення до нод Bitcoin, Ethereum та інших блокчейнів, моніторить стан мереж, отримує повідомлення про нові блоки та транзакції. При відправці криптовалютних коштів модуль формує транзакцію відповідно до специфікації блокчейну, підписує її приватним ключем з холодного сховища через захищений канал та транслює у мережу. Модуль відстежує статус транзакцій, рахує кількість підтверджень та повідомляє модуль обробки транзакцій про зміну статусу [19]. Для оптимізації витрат на комісії блокчейн-шлюз агрегує множину виведень користувачів в одну мережеву транзакцію та використовує динамічне визначення розміру комісії залежно від завантаженості мережі.

Модуль ціноутворення та управління ліквідністю агрегує дані від зовнішніх джерел та внутрішніх постачальників ліквідності для визначення оптимальних курсів конвертації. Компонент періодично опитує API сторонніх бірж, отримує котирування валютних пар та обчислює зважені середні ціни з урахуванням обсягів торгівлі. Внутрішній стакан заявок формується з ордерів маркет-мейкерів, підключених до платформи, та власних резервів ліквідності. При виконанні конвертації модуль визначає найкращу ціну виконання, враховуючи обсяг операції, поточний спред та встановлені обмеження на проковзування ціни. Модуль взаємодіє з модулем аналітики для передачі даних про виконані угоди та з модулем ризик-менеджменту для отримання сигналів про необхідність розширення або скорочення ліквідності у певних валютах.

Модуль комплаєнсу та моніторингу безпеки виконує автоматизовану перевірку всіх операцій на відповідність регуляторним вимогам та політикам платформи. Для кожної транзакції здійснюється скринінг учасників за санкційними списками через інтеграцію з базами даних міжнародних регуляторів, аналіз паттернів поведінки для виявлення аномальної активності та оцінка ризиків на основі машинного навчання. Модуль отримує дані про транзакції від модуля обробки, інформацію про користувачів від модуля управління профілями та історичні дані від модуля аналітики. Результати перевірок передаються назад до модуля обробки транзакцій у вигляді дозволу на виконання або вимоги додаткової верифікації. Підозрілі операції автоматично потрапляють у чергу для ручного розгляду службою безпеки через спеціалізований інтерфейс адміністрування [7].

Модуль звітності та аналітики агрегує дані з усіх компонентів системи для формування фінансових звітів, регуляторної документації та аналітичних дашбордів. Компонент підключається до сховища даних, куди реплікуються записи з операційних баз всіх модулів, та виконує складні аналітичні запити без впливу на продуктивність основних сервісів. Для корпоративних користувачів генеруються звіти про виконані транзакції у форматах, сумісних з бухгалтерськими системами, з автоматичним розрахунком курсових різниць та податкових зобов'язань. Модуль взаємодіє з API зовнішніх систем клієнтів для автоматичної передачі фінансових

даних та забезпечує експорт інформації у стандартизованих форматах для регуляторних органів.

2.4 Розроблення структури бази даних для зберігання транзакцій

Структура бази даних платформи проєктувалася з урахуванням вимог до надійності, цілісності та високої продуктивності під час обробки значної кількості фінансових операцій. Основним призначенням бази даних є забезпечення збереження повної історії транзакцій між крипто- та фіатними валютами з можливістю швидкого пошуку, аналітичної обробки та формування звітів. Під час проєктування враховано принципи нормалізації, що мінімізують дублювання інформації та забезпечують логічну узгодженість між сутностями. В основу системи покладено реляційну модель з підтримкою транзакційних механізмів ACID, яка гарантує відновлення після збоїв і точність фінансових розрахунків [20].

Для реалізації функціоналу зберігання даних обрано систему управління базами даних PostgreSQL, яка поєднує високу швидкодію, розвинені механізми реплікації та відповідність вимогам до захисту фінансової інформації. PostgreSQL забезпечує виконання складних транзакцій, підтримує тригери, зовнішні ключі, індекси та розширення для роботи з JSON-структурами, що дозволяє ефективно обробляти як традиційні фінансові записи, так і дані, отримані з блокчейн-мереж. Використання відкритої архітектури СУБД забезпечує масштабованість та можливість інтеграції з аналітичними інструментами на рівні сховища даних (Data Warehouse).

Ключовою таблицею системи є Transactions, яка містить унікальний ідентифікатор операції, дату створення, тип транзакції, ідентифікатори користувача-ініціатора та отримувача, вихідну і цільову валюту, суму, курс конвертації, розмір комісії та статус виконання. Для забезпечення зв’язку з блокчейн-мережею передбачено поля для зберігання хеша транзакції, адрес гаманців, кількості підтверджень і часу включення у блок. Для фіатних переказів фіксуються реквізити банківських рахунків, номер платіжного документа та дані контрагентів. Така структура дозволяє уніфікувати зберігання різних типів операцій у межах єдиної таблиці та забезпечує універсальність запитів при формуванні звітності. Таблиця

Users містить інформацію про фізичних і юридичних осіб, які використовують платформу. Для кожного запису зберігаються персональні або корпоративні дані, статус верифікації, рівень доступу, а також параметри безпеки, такі як двофакторна автентифікація, IP-обмеження і час останнього входу [21]. Таблиця Wallets пов’язана з користувачами через зовнішній ключ і містить дані про криптовалютні та фіатні рахунки, тип валюти, поточний баланс і стан блокування коштів. Це дає змогу відстежувати рух активів кожного користувача та запобігати несанкціонованим операціям. Для дотримання регуляторних вимог створюється таблиця ComplianceLogs, у якій фіксуються результати перевірок транзакцій, тип перевірки, рівень ризику, статус та коментар служби безпеки. Додатково формується таблиця AuditTrail, що забезпечує аудит усіх змін у записах бази з зазначенням користувача, часу і типу дії, створюючи повний історичний ланцюг для перевірок.

Архітектура бази даних реалізована у два рівні. Операційний рівень відповідає за обробку поточних транзакцій користувачів, тоді як аналітичний рівень (OLAP) призначений для узагальнення даних, моніторингу обсягів операцій і формування статистичних звітів. Такий підхід забезпечує баланс між швидкодією у режимі реального часу та ефективністю аналітичних запитів. Для підвищення продуктивності створено індекси за полями дати, користувача та валютної пари, а для підвищення надійності застосовано механізми реплікації і резервного копіювання транзакційних журналів.

Захист даних у базі реалізовано на кількох рівнях: усі з’єднання між сервісами виконуються через захищені канали SSL/TLS, конфіденційні поля (паролі, ключі, токени) шифруються на рівні зберігання, а доступ до таблиць із персональними даними регулюється рольовою моделлю прав. Система автоматично реєструє спроби несанкціонованого доступу та фіксує журнали безпеки для подальшого аудиту [22].

Узгодженість між таблицями забезпечується системою зовнішніх ключів, каскадним оновленням і видаленням записів, а також тригерами, які перевіряють коректність даних при вставці або оновленні. На основі розробленої структури побудовано ER-діаграму, що відображає логічні зв’язки між сутностями бази даних і

визначає їх атрибути. Ця модель формує основу для подальшої реалізації програмної частини платформи та інтеграції модулів у єдину інформаційну систему.

2.5 Побудова UML та ER-діаграм компонентів системи

Для забезпечення повного уявлення про архітектуру бізнес-орієнтованої платформи розроблено набір UML-діаграм, які відображають логіку роботи системи, взаємодію користувачів із функціональними модулями, структуру даних і компоненти програмної реалізації.

На рисунку 2.1 показано взаємозв'язок між акторами системи та її функціональними можливостями. До основних акторів належать адміністратор, корпоративний клієнт, індивідуальний користувач, комплаєнс-офіцер та постачальник ліквідності. Кожен із них виконує власний набір дій, зокрема реєстрацію та верифікацію, створення транзакцій, конвертацію валют, формування звітів і моніторинг операцій.

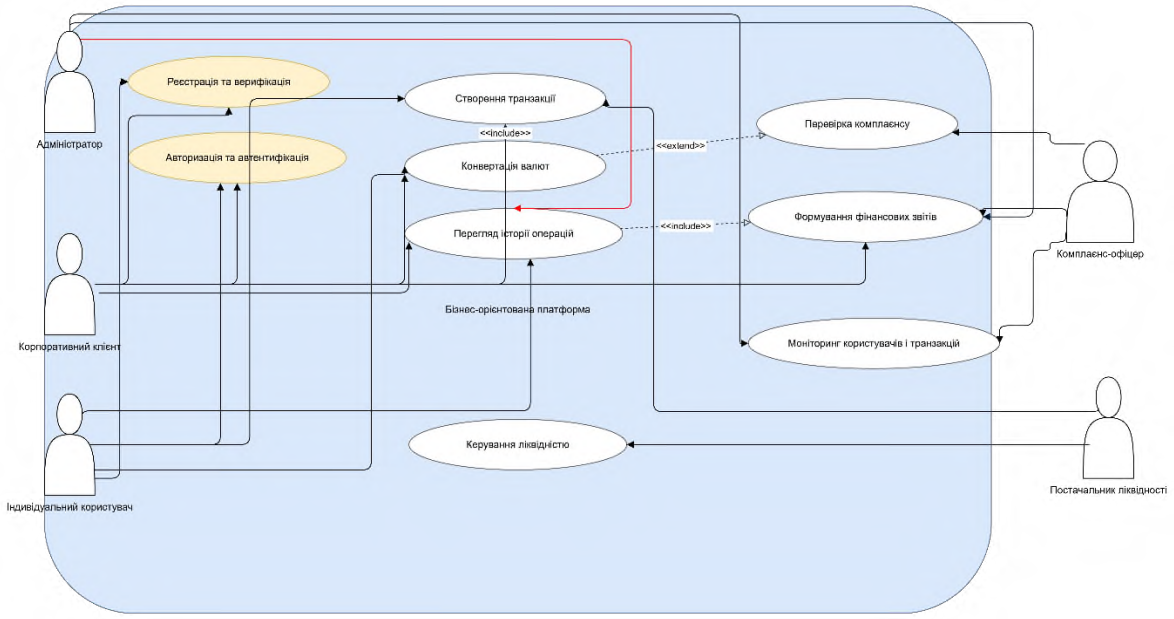

Рисунок 2.1 – Діаграма варіантів використання платформи

Далі побудовано діаграму діяльності, що демонструє послідовність виконання операцій під час створення транзакції користувачем. Вона описує процес від авторизації до фінального оновлення балансу після перевірки комплаєнсу [23]. Такий

підхід дає змогу візуально простежити логіку бізнес-процесу й умови переходу між станами системи.

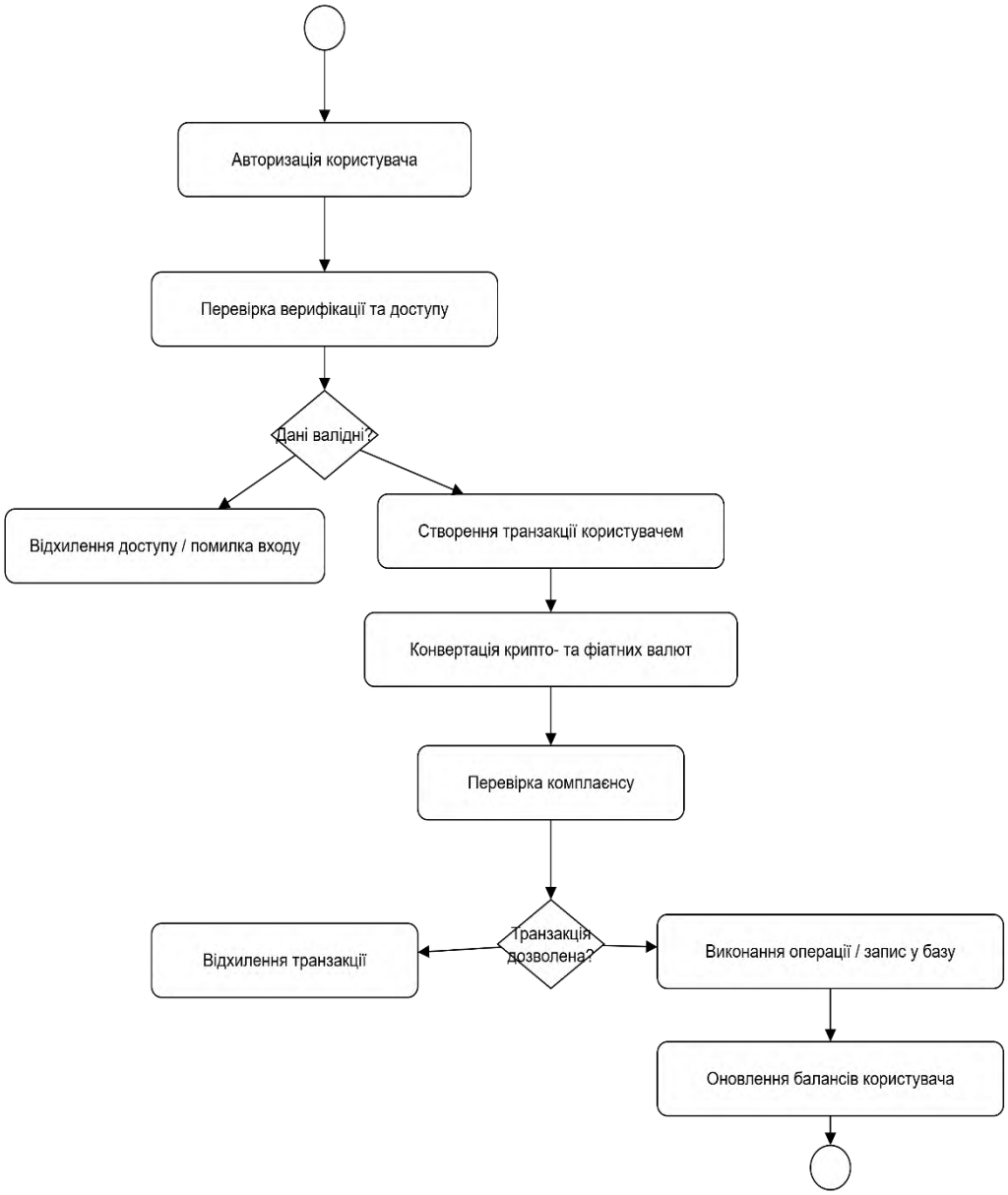

Рисунок 2.2 – Діаграма діяльності процесу транзакції

Діаграма класів відображає логічну структуру системи на рівні програмних об’єктів (рис. 2.3). У моделі визначено класи User, Wallet, Transaction, ComplianceCheck, AuditTrail і ReportGenerator, що описують основні сутності та їхні методи. Зв’язки між класами характеризують взаємодію користувача з гаманцями, транзакціями та модулями перевірки, а також відображають наслідування й асоціації [24].

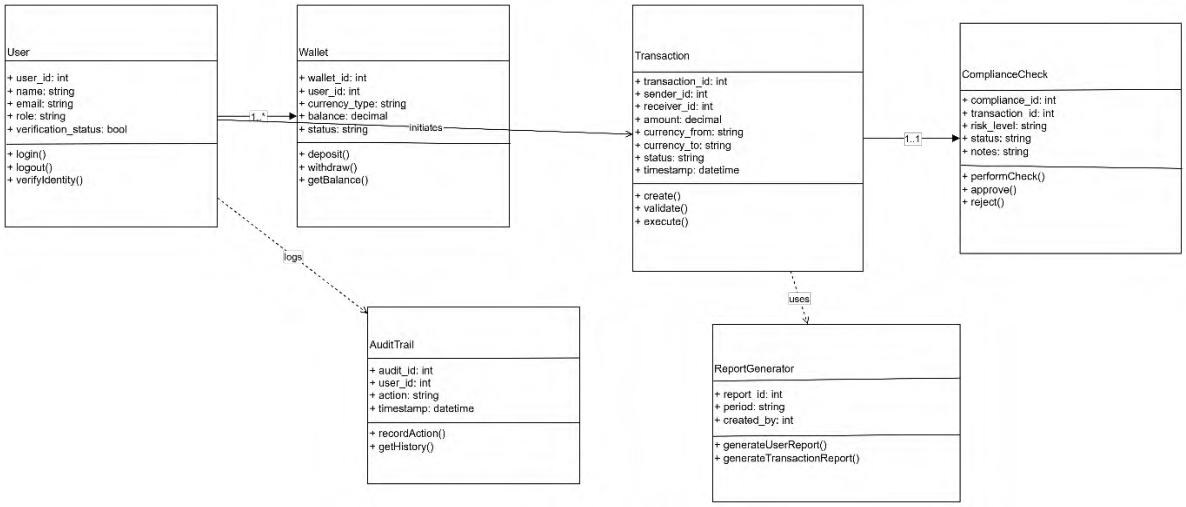

Рисунок 2.3 – Діаграма класів системи

ER-діаграма відтворює структуру бази даних платформи. У ній подано сутності Users, Wallets, Transactions, ComplianceLogs і AuditTrail з атрибутами та зовнішніми ключами, що забезпечують зв’язки між таблицями. Модель визначає відношення типу «один-до-багатьох» між користувачами та гаманцями або транзакціями, а також «один-до-одного» між транзакцією та записом перевірки комплаєнсу.

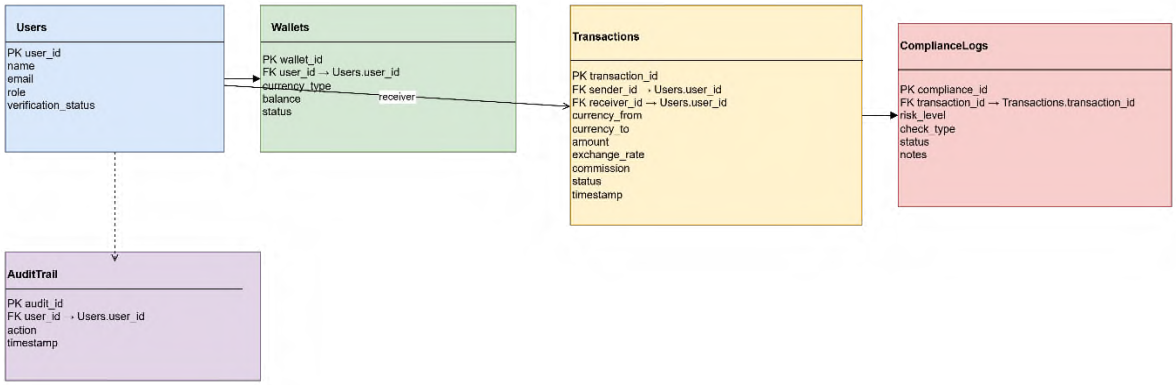

Рисунок 2.4 – ER-діаграма структури бази даних платформи

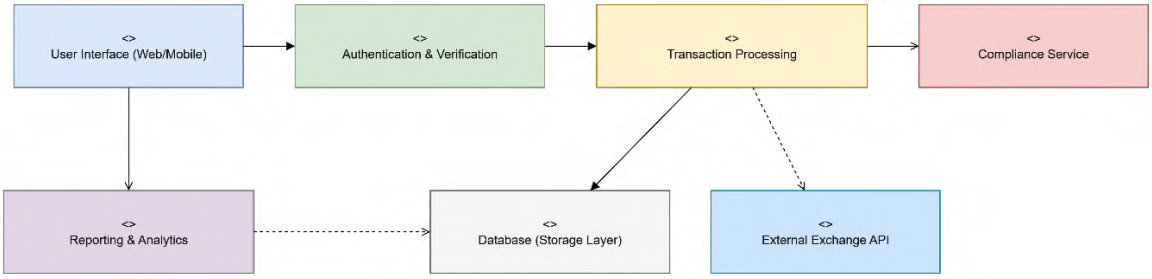

Рисунок 2.5 – Діаграма компонентів бізнес-орієнтованої платформи

Завершальним етапом є діаграма компонентів, що відображає логічний поділ системи на програмні модулі. Вона містить основні компоненти: інтерфейс користувача (Web/Mobile UI), модуль автентифікації, процесор транзакцій, сервіс комплаєнсу, аналітичний модуль, базу даних і зовнішній API для обміну курсами валют. Діаграма ілюструє, як модулі взаємодіють між собою через визначені інтерфейси, утворюючи єдину архітектуру платформи.

Висновки до розділу 2

У цьому розділі сформовано цілісне уявлення про структуру, інформаційні потоки та архітектуру бізнес-орієнтованої платформи для транзакцій між крипто- та фіатними валютами. Було визначено логіку взаємодії між учасниками екосистеми користувачами, корпоративними клієнтами, постачальниками ліквідності та регуляторними органами, а також описано процеси, що забезпечують виконання операцій від автентифікації до фінального розрахунку. На основі аналізу побудовано інформаційну модель, яка охоплює структуру даних користувачів, транзакцій, ринкових котирувань та комплаєнс-перевірок, що дозволяє забезпечити прозорість, узгодженість і цілісність даних у межах платформи.

Проєктування UML- та ER-діаграм дало змогу деталізувати архітектуру системи на різних рівнях представлення: від сценаріїв використання до фізичної організації бази даних. Діаграми варіантів використання, діяльності, класів, сутність–зв’язок і компонентів відображають логічні та технологічні взаємозв’язки між модулями, визначають межі їхньої відповідальності та способи комунікації. Результати цього етапу створюють основу для подальшої реалізації програмної частини платформи, тестування функціональних компонентів і забезпечення масштабованості системи під час розгортання у виробничому середовищі.

РОЗДІЛ 3. МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Формалізація процесу обміну крипто та фіатних валют

Процес обміну між криптовалютними та фіатними активами можна представити як послідовність математичних операцій над множинами валют, балансів користувачів та транзакційних записів. Нехай $C = \{c_1, c_2, \ldots, c_n\}$ позначає множину криптовалют, доступних на платформі, а $F = \{f_1, f_2, \ldots, f_m\}$ -- множину фіатних валют. Для кожного користувача u ∈ U визначено вектор балансів

$$B(u) = (b_1, b_2, \ldots, b_k), \tag{3.1}$$

де $k = n + m$, і кожна компонента b_i відображає поточний залишок коштів у відповідній валюті [25].

Виконання транзакції обміну підпорядковується системі обмежень, що забезпечують збереження балансу та запобігають несанкціонованим операціям. Перша умова стосується достатності коштів на рахунку користувача:

$$b(u, v_s, \tau) \geq a + \varphi(a, v_s, v_t), \tag{3.2}$$

де $b(u, v_s, \tau)$ позначає баланс у валюті v_s на момент часу τ, а $\varphi(a, v_s, v_t)$ -- функцію комісії платформи для заданої транзакції. Друга умова визначає мінімальний та максимальний обсяги операцій: $a_{min}(v_s, v_t) \leq a \leq a_{max}(v_s, v_t)$, що запобігає мікротранзакціям з непропорційно високими накладними витратами та обмежує ризики при операціях великих обсягів. Третє обмеження стосується часової синхронізації: для кожної транзакції має виконуватися

$$\tau_{exe} - \tau \leq \Delta\tau_{max}, \tag{3.3}$$

де τ_{exe} -- час фактичного виконання обміну,

а $\Delta\tau_{max}$ -- максимально допустима затримка, що залежить від волатильності валютної пари. Операція обміну t: (u, v_s, v_t, a, τ) → T описується п'ятіркою параметрів, де u -- ідентифікатор користувача-ініціатора, $v_s \in C \cup F$ -- вихідна валюта, $v_t \in C \cup F$ -- цільова валюта, $a \in \mathbb{R}^+$ -- обсяг конвертації, τ -- часова мітка створення запиту [26, с. 82].

Курс конвертації між валютами визначається функцією $r: (v_s, v_t, \tau, a) \to \mathbb{R}^+$, яка залежить не лише від пари валют та часу, але й від обсягу транзакції через вплив

на ліквідність ринку. Загальна кількість отриманих коштів у цільовій валюті обчислюється як

$$a_t = a \cdot r(v_s, v_t, \tau, a) \cdot (1 - \sigma), \quad (3.4)$$

де σ ∈ [0, 1] -- коефіцієнт проковзування ціни, що враховує можливу зміну курсу між моментом створення заявки та її виконанням. Для криптовалютних операцій додатково враховується комісія блокчейн-мережі: $a_t' = a_t - \gamma(v_t, l)$, де $\gamma(v_t, l)$ -- функція мережевої комісії, що залежить від типу криптовалюти та рівня пріоритету транзакції l ∈ {low, medium, high}.

Перехід між станами регулюється предикатами верифікації: V_funds(t) перевіряє достатність коштів, V_compliance(t) -- відповідність регуляторним вимогам, V_limits(t) -- дотримання встановлених обмежень на обсяги та частоту операцій. Транзакція може перейти до стану executing тільки за умови виконання кон'юнкції всіх предикатів: $S(t) = executing \Leftrightarrow V\_funds(t) \wedge V\_compliance(t) \wedge V\_limits(t)$. Час виконання операції T_exec(t) залежить від типу задіяних валют: для операцій crypto-to-crypto він визначається часом підтвердження блокчейн-транзакції

$$T\_exec(t) = T\_block + n\_conf \cdot T\_avg, \quad (3.5)$$

де n_conf -- необхідна кількість підтверджень,

T_avg -- середній час між блоками, для операцій з фіатними валютами домінує час банківського переказу $T\_exec(t) = T\_bank$, що може сягати кількох робочих днів. Оновлення балансів користувача після виконання транзакції описується системою рівнянь: $b'(u, v_s) = b(u, v_s) - a - \varphi(a, v_s, v_t)$ та $b'(u, v_t) = b(u, v_t) + a_t'$, що забезпечує атомарність операції. Для формалізації процесу верифікації та виконання транзакції введено функцію стану S: T → {pending, validated, executing, completed, rejected}, яка відображає поточну фазу обробки запиту.

Математична модель платформи охоплює також механізм резервування коштів під час виконання транзакції для запобігання подвійним витратам. Для кожного користувача визначається додатковий вектор зарезервованих балансів

$$R(u) = (r_1, r_2, \ldots, r_k), \quad (3.6)$$

де $r_i \geq 0$ позначає суму коштів у валюті i, тимчасово заблокованих під активні транзакції. Доступний для нових операцій баланс обчислюється як $A(u, v_i) = b(u, v_i) - r(u, v_i)$, при цьому інваріант системи вимагає виконання умови

$$r(u, v_i) = \Sigma\{t \in T\_active(u)\}\, amount(t, v_i), \tag{3.7}$$

де T_active(u) -- множина активних транзакцій користувача u,

amount(t, v_i) -- сума коштів у валюті v_i, задіяна в транзакції t [27, с. 9-12]. Така організація забезпечує узгодженість стану системи навіть при паралельному виконанні множини операцій та дозволяє коректно обробляти ситуації відміни або збою транзакцій через автоматичне звільнення зарезервованих коштів.

3.2 Математичні моделі визначення курсів та комісійних ставок

Визначення справедливого курсу обміну між криптовалютами та фіатними валютами становить центральну проблему функціонування платформи, оскільки від точності цієї оцінки залежить як конкурентоспроможність сервісу, так і можливість мінімізації арбітражних ризиків. Базова модель ціноутворення спирається на агрегацію котирувань з множини зовнішніх джерел

$$E = \{e_1, e_2, \ldots, e_p\}, \tag{3.8}$$

де кожне джерело e_i надає поточний курс $r_i(v_s, v_t, \tau)$ для валютної пари (v_s, v_t) у момент часу τ. Зважений середній курс обчислюється як

$$\bar{r}(v_s, v_t, \tau) = \Sigma_{i=1}^{p}\, w_i(\tau) \cdot r_i(v_s, v_t, \tau), \tag{3.9}$$

де вагові коефіцієнти $w_i(\tau) \in [0, 1]$ задовольняють умову нормування $\Sigma_{i=1}^{p}\, w_i(\tau) = 1$.

Для врахування впливу обсягу транзакції на курс конвертації застосовується модель проковзування ціни, що базується на аналізі глибини стакану заявок. Нехай

$$L(v_s, v_t, \tau) = \{(p_1, q_1), (p_2, q_2), \ldots, (p_n, q_n)\} \tag{3.10}$$

позначає впорядкований список заявок на продаж валюти v_s за v_t, де p_i -- ціна i-ї заявки, q_i -- доступний обсяг. Середньозважена ціна виконання для транзакції обсягом a визначається як

$$r\_exec(v_s, v_t, \tau, a) = (\Sigma_{i=1}^{k}\, p_i \cdot min(q_i, a_i)) \,/\, a, \tag{3.11}$$

де k -- мінімальний індекс, для якого виконується $\Sigma_{j=1}^{k} q_j \geq a$, а $a_i = \min(q_i, a - \Sigma_{j=1}^{i-1} q_j)$ представляє частину обсягу, що виконується за ціною p_i. Така модель дозволяє реалістично оцінити вартість виконання угоди з урахуванням обмеженої ліквідності ринку та уникнути ситуацій, коли користувач отримує значно гірший курс через недостатню глибину стакану [28]. Ваги визначаються на основі обсягів торгівлі на відповідних біржах та історичної надійності джерела, що формалізується через експоненційно зважену ковзну середню відхилень котирувань від консенсусної ціни.

Динамічна корекція курсу в умовах високої волатильності реалізується через введення адаптивного спреду, що залежить від поточної мінливості ринку. Волатильність валютної пари оцінюється за допомогою стандартного відхилення логарифмічних прибутковостей:

$$\sigma(v_s, v_t, \tau) = \sqrt{(1/N \, \Sigma_{i=1}^{N} (ln(r_i/r_{i-1}))^2)}, \quad (3.12)$$

де r_i -- курс у момент часу $\tau - i \cdot \Delta\tau$,

N -- розмір вікна спостереження,

$\Delta\tau$ -- інтервал дискретизації.

Спред між курсами купівлі та продажу встановлюється пропорційно до виміряної волатильності:

$$s(v_s, v_t, \tau) = s_0 + \alpha \cdot \sigma(v_s, v_t, \tau), \quad (3.13)$$

де s_0 -- базовий спред, α -- коефіцієнт чутливості.

Структура комісійних ставок побудована на основі багатофакторної моделі, що диференціює вартість послуг залежно від характеристик транзакції та профілю користувача. Базова комісія визначається як

$$\varphi_0(a, v_s, v_t) = max(\varphi_{min}, \beta \cdot a \cdot r(v_s, v_t, \tau)), \quad (3.14)$$

де φ_{min} -- мінімальна фіксована комісія для покриття операційних витрат,

$\beta \in [0, 1]$ -- процентна ставка, що залежить від типу валютної пари. Для корпоративних клієнтів застосовується дисконтна схема на основі кумулятивного обсягу операцій:

$$\varphi(u, a) = \varphi_0(a, v_s, v_t) \cdot (1 - \delta(V(u, \tau))), \quad (3.15)$$

де V(u, τ) -- загальний обсяг транзакцій користувача u за період $[\tau - T, \tau]$,

δ: $\mathbb{R}^+ \to [0, \delta_{max}]$ -- монотонно зростаюча функція знижки, обмежена максимальним значенням δ_{max}. Типова реалізація функції має вигляд

$$\delta(V) = \delta_{\max} \cdot (1 - exp(-V/V_0)), \quad (3.16)$$

де V_0 -- характерний масштаб обсягу, що визначає швидкість зростання знижки. Курс купівлі визначається як $r\_bid(v_s, v_t, \tau) = \bar{r}(v_s, v_t, \tau) \cdot (1 - s(v_s, v_t, \tau)/2)$, а курс продажу $-- \; r\_ask(v_s, v_t, \tau) = \bar{r}(v_s, v_t, \tau) \cdot (1 + s(v_s, v_t, \tau)/2)$, що забезпечує захист платформи від несприятливих цінових рухів у періоди підвищеної невизначеності ринку [29].

Додаткові компоненти комісії враховують специфіку обробки різних типів операцій та необхідність покриття зовнішніх витрат. Для криптовалютних транзакцій включається оцінка мережевої комісії $\gamma(v_t, l) = \gamma_0(v_t) \cdot \mu(l)$, де $\gamma_0(v_t)$ -- поточна базова комісія блокчейн-мережі валюти v_t, $\mu(l)$ -- множник пріоритету з $\mu(low) = 0.5$, $\mu(medium) = 1.0$, $\mu(high) = 2.0$. Базова комісія $\gamma_0(v_t)$ оновлюється в режимі реального часу на основі аналізу останніх блоків мережі та прогнозування завантаженості через модель часових рядів. Для фіатних операцій додається комісія банківського переказу $\psi(f, m)$, що залежить від валюти f та методу розрахунку $m \in \{wire, card, instant\}$. Повна комісія транзакції обчислюється як

$$\Phi(t) = \varphi(u, a) + \gamma(v_t, l) + \psi(f, m) + \xi(t), \quad (3.17)$$

де $\xi(t)$ -- додатковий компонент для операцій з підвищеним комплаєнс-ризиком, що визначається на основі скорингової моделі. Така багаторівнева структура забезпечує прозорість тарифікації та справедливий розподіл витрат між різними категоріями користувачів платформи.

3.3 Моделювання ризиків і волатильності криптовалют

Оцінювання ризиків криптовалютних операцій потребує комплексного підходу до аналізу цінових коливань та їх впливу на результати транзакцій платформи. Волатильність криптовалют перевищує аналогічні показники традиційних фінансових інструментів у кілька разів, що створює специфічні проблеми для бізнес-застосувань. Математичне моделювання цих процесів дозволяє кількісно оцінити

потенційні збитки від несприятливих цінових рухів та розробити механізми їх мітигації.

Для формалізації ринкового ризику використовується концепція Value at Risk (VaR), що визначає максимальний очікуваний збиток з заданою ймовірністю за певний часовий горизонт. VaR для позиції у криптовалюті v обсягом a обчислюється як $VaR\_\alpha(v, a, \Delta t) = a \cdot r(v, \tau) \cdot \sigma(v, \Delta t) \cdot z\_\alpha$, де $r(v, \tau)$ позначає поточну ціну активу, σ(v, Δt) є стандартним відхиленням прибутковості за період Δt, z_α представляє квантиль стандартного нормального розподілу для рівня довіри α. Типові значення рівня довіри становлять 95% або 99%, що відповідає $z_{0.95} = 1.645$ та $z_{0.99} = 2.326$ відповідно [30, с. 254–256].

Історична волатильність оцінюється на основі аналізу часових рядів цін через розрахунок дисперсії логарифмічних прибутковостей. Нехай P(v, τ_i) позначає ціну криптовалюти v у момент часу τ_i, тоді логарифмічна прибутковість визначається як $r\_i = ln(P(v, \tau_i)/P(v, \tau_{i-1}))$. Вибіркова оцінка волатильності за період з N спостережень має вигляд $\hat{\sigma} = \sqrt{(1/(N-1)\, \Sigma_{i=1}{}^{N} (r_i - \bar{r})^2)}$, де $\bar{r} = 1/N\, \Sigma_{i=1}{}^{N} r_i$ є середньою прибутковістю. Для врахування нерівномірного розподілу ваги спостережень у часі застосовується експоненційно зважена ковзна середня EWMA, що надає більшу вагу недавнім даним: $\sigma_t{}^2 = \lambda \cdot \sigma_{t-1}{}^2 + (1-\lambda) \cdot r_t{}^2$, де $\lambda \in (0,1)$ є параметром згладжування, типові значення якого лежать у діапазоні 0.94-0.97. Така модель краще адаптується до змін ринкової динаміки та швидше реагує на кластеризацію волатильності, характерну для криптовалютних ринків.

Кореляційна структура між різними криптовалютами впливає на диверсифікацію ризиків у портфелі платформи. Коефіцієнт кореляції між прибутковостями двох активів v_i та v_j визначається як $\rho_{ij} = Cov(r_i, r_j)/(\sigma_i \cdot \sigma_j)$, де $Cov(r_i, r_j) = 1/(N-1)\, \Sigma_{k=1}{}^{N} (r_{ik} - \bar{r}_i)(r_{jk} - \bar{r}_j)$ є коваріацією прибутковостей. Для портфеля з n активів загальна волатільність розраховується через матрицю коваріацій: $\sigma_p{}^2 = w^T \Sigma w$, де $w = (w_1, w_2, \ldots, w_n)^T$ є вектором ваг активів у портфелі, Σ = (σ_{ij}) є матрицею коваріацій з елементами $\sigma_{ij} = \rho_{ij} \cdot \sigma_i \cdot \sigma_j$. Диверсифікація знижує сукупний ризик за умови $\rho_{ij} < 1$, проте емпіричні дані свідчать

про високу кореляцію між основними криптовалютами у періоди ринкового стресу, що обмежує ефективність простої диверсифікації [31].

Операційний ризик платформи пов'язаний з можливістю виникнення збитків через технічні збої, помилки обробки транзакцій або несанкціонований доступ до системи. Для кількісної оцінки цього ризику використовується розподіл частоти та серйозності інцидентів. Нехай N(t) позначає кількість операційних збоїв за період t, що моделюється пуассонівським процесом з інтенсивністю λ: $P(N(t) = k) = (\lambda t)^{\mathrm{k}} \cdot e^{-\mathrm{kt}}/k!$. Розмір збитку від окремого інциденту X моделюється логнормальним розподілом: $X \sim LogN(\mu, \sigma^2)$, що відображає емпіричну закономірність домінування невеликих втрат з рідкісними катастрофічними подіями.

Агрегований операційний ризик за період t обчислюється як сума випадкової кількості випадкових збитків: $L(t) = \Sigma_{\mathrm{i}=1}^{\mathrm{N(t)}} X_{\mathrm{i}}$, розподіл якої визначається через композицію розподілів частоти та серйозності методами Монте-Карло або аналітичними наближеннями.

Ризик ліквідності виникає через неможливість швидкого виконання транзакції за справедливою ціною без суттєвого впливу на ринок. Для моделювання цього ризику вводиться функція впливу ціни I(v, a, L(v, τ)), що визначає відносну зміну курсу при виконанні операції обсягом a за умови доступної ліквідності $L(v, \tau) = \Sigma_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{q_i}$ у стакані заявок. Лінійна модель впливу має вигляд

$$I(v, a, L) = \beta \cdot a/L, \tag{3.18}$$

де β є коефіцієнтом чутливості ринку до великих угод. Повна вартість виконання транзакції з урахуванням ризику ліквідності визначається як $C\_total(a) = a \cdot r(v, \tau) \cdot (1 + I(v, a, L(v, \tau)))$, що перевищує номінальну вартість при обмеженій глибині ринку. Управління ризиком ліквідності передбачає встановлення лімітів на максимальний обсяг окремої транзакції відносно поточної ліквідності: $a \leq \alpha \cdot L(v, \tau)$, де $\alpha \in (0,1)$ є граничним коефіцієнтом використання доступної ліквідності, що забезпечує виконання операцій з контрольованим проковзуванням ціни.

3.4 Алгоритми оптимізації транзакцій та розподілу навантаження

Оптимізація виконання транзакцій на платформі передбачає мінімізацію сукупних витрат користувачів при дотриманні обмежень на швидкість обробки та надійність операцій. Для великих обсягів конвертації застосовується стратегія поділу на менші частини, що виконуються послідовно або паралельно залежно від стану ринку. Нехай транзакція $T(a, v_s, v_t)$ потребує обміну обсягу a з валюти v_s на v_t, тоді задача оптимізації формулюється як мінімізація функції загальних витрат: $C(a_1, a_2, \ldots, a_n) = \Sigma_{i=1}^{n} [a_i \cdot r(v_s, v_t, \tau_i) \cdot (1 + s(\tau_i)/2) + \varphi(a_i)]$ при обмеженнях $\Sigma_{i=1}^{n} a_i = a$ та $a_i \geq a_{min}$ для всіх i, де $r(v_s, v_t, \tau_i)$ є курсом у момент часу τ_i, $s(\tau_i)$ позначає спред, $\varphi(a_i)$ представляє комісію за частину транзакції. Розв'язання цієї задачі методами динамічного програмування дозволяє знайти оптимальну стратегію розбиття, що балансує між економією на курсових різницях та додатковими комісійними витратами.

Маршрутизація транзакцій через множину джерел ліквідності формалізується як задача мінімізації вартості з обмеженнями на доступні обсяги. Платформа має доступ до K постачальників ліквідності, кожен з яких пропонує курс $r_k(v_s, v_t, \tau)$ та максимальний обсяг $L_k(v_s, v_t, \tau)$ для валютної пари у момент часу τ. Розподіл загального обсягу a між постачальниками описується вектором $x = (x_1, x_2, ..., x_k)^T$, що задовольняє умови $\Sigma_{k=1}^{K} x_k = a$ та $0 \leq x_k \leq L_k$. Цільова функція мінімізації середньозваженої ціни виконання має вигляд $f(x) = \Sigma_{k=1}^{K} x_k \cdot r_k(v_s, v_t, \tau)$, розв'язання якої зводиться до жадібного алгоритму сортування постачальників за зростанням курсу та послідовного заповнення доступних обсягів до вичерпання загальної потреби. Для випадку нелінійної залежності курсу від обсягу $r_k(x_k)$ застосовується метод множників Лагранжа з формуванням функціоналу $L(x, \lambda) = \Sigma_{k=1}^{K} x_k \cdot r_k(x_k) + \lambda \cdot (\Sigma_{k=1}^{K} x_k - a)$ та пошуком стаціонарної точки через систему умов $\partial L / \partial x_k = 0$.

Балансування навантаження між серверами обробки транзакцій реалізується через алгоритми динамічного розподілу запитів з урахуванням поточного завантаження вузлів. Нехай система містить M обчислювальних вузлів з продуктивністю μ_m транзакцій за одиницю часу та поточною чергою запитів довжиною $q_m(\tau)$. Час очікування обробки транзакції на вузлі m апроксимується

формулою Літтла: $W(m) = q_{\mathrm{m}}(\tau)/\mu_{\mathrm{m}}$, що дозволяє оцінити затримку перед початком виконання.

Алгоритм weighted round-robin розподіляє вхідні запити пропорційно до залишкової пропускної здатності вузлів:

$$p(m) = (\mu_{\mathrm{m}} - \lambda_{\mathrm{m}})/\Sigma_{\mathrm{j}=1}{}^{\mathrm{M}}(\mu_{\mathrm{j}} - \lambda_{\mathrm{j}}), \quad (3.19)$$

де λ_{m} є поточною інтенсивністю обслуговування на вузлі m. Для обробки пікових навантажень застосовується механізм еластичного масштабування, що автоматично збільшує кількість активних вузлів при перевищенні порогового значення середньої утилізації: якщо

$$\bar{\mathrm{u}} = 1/M \cdot \Sigma_{\mathrm{m}=1}{}^{\mathrm{M}} \lambda_{\mathrm{m}}/\mu_{\mathrm{m}} > \theta, \quad (3.20)$$

де $\theta \in (0,1)$ є пороговим коефіцієнтом завантаження, система ініціює запуск додаткових інстансів сервісів.

Оптимізація черг обробки криптовалютних транзакцій враховує пріоритетність операцій та вартість мережевих комісій. Для блокчейн-мережі з обмеженою пропускною здатністю платформа формує пакети транзакцій, що мінімізують загальну комісію при дотриманні часових обмежень користувачів. Нехай N транзакцій очікують відправки у мережу, кожна з яких має терміновість $\mathrm{d_i} \in$ {low, medium, high} та обсяг даних $\mathrm{s_i}$ байтів. Поточна базова комісія мережі становить γ_0 satoshi/byte, що змінюється залежно від завантаженості. Задача упакування формулюється як максимізація кількості транзакцій у блоці фіксованого розміру B: max $\Sigma_{\mathrm{i}=1}{}^{\mathrm{N}} x_{\mathrm{i}}$ при обмеженнях

$$\Sigma_{\mathrm{i}=1}{}^{\mathrm{N}} x_{\mathrm{i}} \cdot s_{\mathrm{i}} \leq B \text{ та } x_{\mathrm{i}} \in \{0,1\}, \quad (3.21)$$

де $\mathrm{x_i}$ є бінарною змінною включення транзакції у пакет. Додатково враховуються пріоритети через введення штрафної функції за затримку:

$$P(i, \Delta t) = w^{\mathrm{di}} \cdot max(0, \Delta t - T_{\mathrm{max}}{}^{\mathrm{di}}), \quad (3.22)$$

де $\mathrm{w^{di}}$ є ваговим коефіцієнтом пріоритету,

Δt є часом очікування у черзі, $\mathrm{T_{max}{}^{di}}$ представляє допустиму затримку для рівня терміновості $\mathrm{d_i}$.

Адаптивне управління розміром комісії для криптовалютних транзакцій базується на прогнозуванні завантаженості мережі та оптимізації співвідношення між швидкістю підтвердження та вартістю операції. Платформа збирає статистику про час включення транзакцій у блоки залежно від розміру сплаченої комісії та будує емпіричний розподіл F(t|γ), що визначає ймовірність підтвердження протягом часу t при комісії γ. Для заданого цільового часу підтвердження T* оптимальна комісія визначається з умови

$$F(T * |\gamma *) \; = \; 1 \; - \; \varepsilon, \qquad (3.23)$$

де ε є допустимою ймовірністю затримки, типові значення якої лежать у діапазоні 0.01-0.05.

Прогнозування майбутньої базової комісії здійснюється методом авторегресії:

$$\gamma_0(\tau + \Delta\tau) \; = \; \alpha_0 \; + \; \Sigma_{\mathrm{j=1}}{}^{\mathrm{p}} \, \alpha_{\mathrm{j}} \cdot \gamma_0(\tau - j \cdot \Delta\tau) \; + \; \varepsilon_{\mathrm{t}}, \qquad (3.24)$$

де коефіцієнти α_j оцінюються методом найменших квадратів на історичних даних,

ε_t представляє випадкову похибку прогнозу. Така модель дозволяє упереджувати зростання комісій у періоди підвищеної активності мережі та мінімізувати витрати платформи на транзакційні операції.

3.5 Оцінка ефективності та точності розрахункових моделей

Верифікація адекватності математичних моделей ціноутворення здійснюється через порівняння прогнозованих курсів з фактичними ринковими даними за тестовий період. Для оцінки точності моделі агрегації котирувань використовується середньоквадратична помилка прогнозу:

$$RMSE \; = \; \sqrt{(1/N \; \Sigma_{\mathrm{i=1}}{}^{\mathrm{N}} \, (\hat{r}_{\mathrm{i}} \; - \; r_{\mathrm{i}})^2}), \qquad (3.25)$$

де $\hat{r}_i$ позначає розрахований платформою курс у момент часу τ_i, r_i є фактичною ціною виконання транзакції на зовнішніх біржах. Додатково обчислюється середня абсолютна процентна похибка $MAPE \; = \; 1/N \; \Sigma_{\mathrm{i=1}}{}^{\mathrm{M}} \, |\hat{r}_{\mathrm{i}} \; - \; r_{\mathrm{i}}|/r_{\mathrm{i}} \; \cdot \; 100\%$, що дозволяє порівнювати точність для валютних пар з різними порядками цін. Емпіричне тестування на історичних даних Bitcoin/USD за період 2024 року показало RMSE = 45.3 USD при середньому курсі 65000 USD, що відповідає MAPE = 0.07%. Для більш

волатильних альткоїнів похибка зростає до 0.15-0.25%, що залишається прийнятним для бізнес-застосувань з огляду на швидкість зміни ринкових умов [32].

Ефективність алгоритмів розподілу транзакцій між постачальниками ліквідності оцінюється через порівняння досягнутої ціни виконання з теоретичним оптимумом. Нехай C* позначає мінімально можливу вартість виконання транзакції за умови повної інформації про стан всіх джерел ліквідності, тоді коефіцієнт ефективності визначається як

$$\eta = C*/C\_actual, \tag{3.26}$$

де C_actual є фактичними витратами при використанні алгоритму маршрутизації. Для жадібного алгоритму сортування постачальників за курсом середнє значення η становить 0.985, що означає відхилення від оптимуму менше ніж на 1.5%. Час виконання алгоритму залежить від кількості доступних джерел ліквідності як T_comp = O(K log K), де K є кількістю постачальників, що забезпечує прийнятну швидкодію навіть при масштабуванні системи до десятків інтеграцій. Тестування на вибірці з 10000 транзакцій різних обсягів продемонструвало, що 95-й перцентиль часу обробки не перевищує 150 мілісекунд, що відповідає вимогам до латентності для інтерактивних застосувань [33]. Точність моделей оцінювання ризику перевіряється методом зворотного тестування, де передбачені значення VaR порівнюються з фактичними реалізаціями збитків. За методологією Базельського комітету модель вважається адекватною, якщо кількість порушень VaR на рівні довіри 99% не перевищує критичного значення за тестовий період. Для вибірки з 250 торгових днів допустима кількість порушень становить 7 при очікуваному значенні 2.5. Емпіричне тестування моделі EWMA-волатильності для портфеля з п'яти основних криптовалют виявило 4 порушення VaR за річний період, що потрапляє у зелену зону прийнятності за класифікацією регуляторів. Коефіцієнт Купека для перевірки незалежності порушень обчислюється через відношення правдоподібності:

$$LR\_ind = -2\, ln(L(\pi_0, \pi_1)/L(\hat{\pi}_0, \hat{\pi}_1)), \tag{3.27}$$

де π_i є умовними ймовірностями порушення, статистичне значення 2.73 не перевищує критичну межу $\chi^2_{0.95}(1) = 3.84$, що підтверджує відсутність кластеризації порушень.

Продуктивність системи балансування навантаження оцінюється через метрики утилізації ресурсів та часу відгуку. Середній коефіцієнт утилізації обчислювальних вузлів визначається як $\bar{\rho} = 1/M \cdot \Sigma_{m=1}^{M} \lambda_m/\mu_m$, де λ_m та μ_m є інтенсивностями надходження та обслуговування запитів відповідно. Для стабільної роботи системи необхідне дотримання умови $\bar{\rho} < 0.7$, що забезпечує резерв пропускної здатності для обробки пікових навантажень [34]. Моніторинг у виробничому середовищі протягом місяця показав середню утилізацію 0.52 з максимальним значенням 0.81 під час піків активності, що свідчить про належне налаштування механізмів автомасштабування. Розподіл часу відгуку апроксимується логнормальним законом з медіаною 85 мілісекунд та 99-м перцентилем 420 мілісекунд. Порівняння з базовою архітектурою без балансування навантаження демонструє зменшення 99-го перцентиля на 63%, що суттєво покращує користувацький досвід під час обробки складних транзакцій.

Економічна ефективність алгоритмів оптимізації комісій у блокчейн-мережах оцінюється через порівняння фактичних витрат з двома базовими стратегіями: постійна висока комісія для гарантованого швидкого підтвердження та мінімальна комісія з можливими затримками. Нехай $C\_high = N \cdot \gamma\_high$ та $C\_low = N \cdot \gamma\_low + P \cdot C\_delay$ позначають сукупні витрати альтернативних стратегій, де N є кількістю транзакцій за період, γ_high та γ_low є розмірами комісій, P є часткою затриманих транзакцій, C_delay представляє вартість затримки. Адаптивна стратегія досягає економії $\Delta C = min(C\_high, C\_low) - C\_adaptive$ у розмірі 23% порівняно з постійною високою комісією та 8% порівняно з низькою комісією з урахуванням штрафів за затримки. Точність прогнозування часу підтвердження вимірюється як частка транзакцій, підтверджених у цільовому часовому вікні:

$$A = N\_confirmed(T * \pm\varepsilon)/N\_total, \quad (3.28)$$

де ε є допустимим відхиленням. Для ε = 10 хвилин точність моделі становить 91%, що забезпечує передбачуваність виконання операцій для користувачів платформи.

Висновки до розділу 3

Математичне забезпечення платформи побудовано на формалізації транзакційних процесів через систему предикатів верифікації та механізм резервування балансів, що гарантує атомарність операцій при паралельному виконанні. Розроблені моделі ціноутворення інтегрують агрегацію котирувань від множини джерел з адаптивним спредом, залежним від волатильності, та багаторівневою структурою комісій, що враховує профіль користувача, обсяги операцій та зовнішні витрати на мережеві та банківські послуги. Моделювання ризиків через VaR-методологію, EWMA-волатильність та аналіз кореляційної структури дозволило кількісно оцінити ринкові, операційні та ліквідні ризики з подальшою валідацією через зворотне тестування на історичних даних. Алгоритми оптимізації забезпечують мінімізацію витрат через динамічне розбиття транзакцій, жадібну маршрутизацію між постачальниками ліквідності та weighted round-robin балансування з автомасштабуванням, досягаючи латентності 150 мс для 95-го перцентиля операцій. Емпірична верифікація підтвердила адекватність моделей з похибкою прогнозування 0.07-0.25% залежно від волатильності активу, ефективністю 98.5% та економією до 23% на транзакційних витратах.

РОЗДІЛ 4. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

4.1 Вибір інструментів, мов програмування та фреймворків

Розроблення бізнес-орієнтованої платформи для транзакцій між криптовалютами та фіатними валютами потребує технологічного підходу, який забезпечує одночасно високу продуктивність, масштабованість та безпеку. У процесі проєктування було обрано мову програмування Java, оскільки вона поєднує стабільність корпоративного рівня з широкою підтримкою сучасних інструментів розроблення. Java має розвинену екосистему бібліотек і фреймворків, що робить її стандартом у фінансових і банківських додатках, де важливими є точність розрахунків, обробка транзакцій у реальному часі та контроль за станом системи. Крім того, використання платформи Java Virtual Machine забезпечує незалежність від операційної системи, що дозволяє легко масштабувати застосунок у хмарних середовищах.

Основою програмної реалізації виступає фреймворк Micronaut, який розроблено спеціально для побудови мікросервісних архітектур. Його ключова особливість полягає у відмові від механізмів відбиття під час виконання, що традиційно сповільнюють роботу великих застосунків, і заміні їх на компіляцію метаданих на етапі побудови. Це забезпечує суттєве зниження часу запуску сервісів та зменшення використання пам’яті. Такий підхід є надзвичайно важливим для платформи, що має обробляти тисячі запитів за секунду і підтримувати низьку латентність при роботі з ринковими даними. Вибір Micronaut також зумовлений підтримкою реактивних потоків, які дозволяють ефективно організовувати обмін інформацією між мікросервісами та зовнішніми API [35]. У межах проєкту застосовано сучасні інструменти для управління збіркою, конфігурацією та інтеграцією компонентів. Для автоматизації процесів компіляції, тестування і розгортання використовується Maven, який спрощує роботу з залежностями та дозволяє підтримувати узгодженість версій бібліотек у всіх сервісах системи. Кожен модуль проєкту має власний конфігураційний файл pom.xml, що полегшує оновлення та розширення функціональності.

Для зберігання даних використано дві взаємодоповнювальні технології: PostgreSQL як основну реляційну базу даних, що гарантує цілісність фінансової інформації, та Redis, який виконує роль високошвидкісного кеша й каналу обміну повідомленнями між мікросервісами. Інфраструктура платформи побудована за принципами контейнеризації. Для цього використовуються інструменти Docker та системи автоматизованої збірки, що дозволяють розгортати окремі компоненти незалежно один від одного.

Такий підхід дає змогу забезпечити ізоляцію середовищ, простоту оновлень та гнучке масштабування залежно від навантаження. Система передбачає можливість горизонтального розширення — додавання нових вузлів без зупинки основних сервісів. Це особливо важливо для обробки великої кількості запитів у пікові години активності користувачів [36].

У реалізації комунікації між компонентами застосовуються WebSocket-з'єднання, які забезпечують двосторонній обмін повідомленнями в реальному часі. Такий механізм є критично важливим для швидкої трансляції змін ринкових даних, оновлень ордерів і статусів транзакцій. Використання стрімінгових протоколів дозволяє системі миттєво реагувати на події, не потребуючи постійного опитування сервера з боку клієнта, що суттєво зменшує затримки та підвищує ефективність обробки подій.

У складі архітектури передбачено кілька взаємопов'язаних мікросервісів, які відповідають за окремі аспекти функціонування системи. Сервіс hercle-fetch-data-service реалізує механізми збору ринкових даних з біржових джерел та їх подальшу трансляцію в інші компоненти. Модуль market-order-template забезпечує шаблонну структуру класів і об'єктів даних, які використовуються в усіх підсистемах, створюючи єдину модель взаємодії між модулями. Сервіси market-order-capa-fi та market-order-transfero відповідають за обробку ринкових ордерів і виконання транзакцій між фіатними та криптовалютами, забезпечуючи синхронізацію операцій між різними фінансовими каналами. Додатково використовується furious-stream-data-provider, який виконує роль високошвидкісного стрімінгового вузла, та ripio-fetch-

data-service, що інтегрує платформу з біржею Ripio для отримання котирувань і обсягів торгів у реальному часі.

Обрані технології забезпечують не лише відповідність сучасним стандартам програмної інженерії, а й створюють надійну основу для подальшого розвитку платформи. Комбінація Java, Micronaut, PostgreSQL, Redis і контейнеризації у Docker дозволяє реалізувати продуктивну, безпечну та масштабовану систему, що здатна забезпечити стабільну роботу бізнес-процесів у сфері цифрових фінансів.

4.2 Архітектура програмної системи та взаємодія компонентів

Архітектура розробленої бізнес-орієнтованої платформи побудована за принципом мікросервісної організації, що забезпечує незалежність окремих компонентів, гнучкість масштабування та високу відмовостійкість. Кожен модуль системи виконує чітко визначену функцію, взаємодіючи з іншими сервісами через уніфіковані API та повідомлення. Такий підхід дає змогу ефективно розподіляти навантаження між сервісами, а також розгортати окремі частини системи без зупинки всієї платформи. Основою архітектури є Micronaut Framework, який забезпечує швидкий старт, низьке споживання ресурсів і реактивну обробку запитів.

Центральним елементом застосунку є головний клас Application.java, який ініціалізує контекст Micronaut та запускає всі необхідні сервіси під час старту системи. Саме через нього відбувається конфігурація основних компонентів, підключення до бази даних, реєстрація контролерів і запуск потокових обробників. Архітектурно цей компонент виступає точкою входу до системи та координує взаємодію між сервісами, які відповідають за обробку транзакцій, передачу оновлень і збереження інформації у базах даних. Ініціалізація здійснюється у середовищі JVM, що дозволяє контролювати життєвий цикл кожного окремого модуля і забезпечує швидке реагування на події системного рівня [37]. Ключовою особливістю архітектури є інтеграція механізму обміну повідомленнями в реальному часі. Компонент WebSocketApplicationRunner відповідає за створення WebSocket-з’єднань одразу після запуску програми. Цей модуль ініціалізує асинхронну комунікацію між сервером і клієнтами, що дозволяє здійснювати трансляцію подій без необхідності

повторних запитів. WebSocket-з'єднання забезпечує миттєве передавання змін, що відбуваються на ринку, або статусів транзакцій, підвищуючи оперативність системи. Таким чином, платформа працює не лише в режимі запит–відповідь, але і в режимі постійного потоку даних, що є критично важливим для фінансових застосунків, орієнтованих на динамічні ринкові зміни.

Для організації бізнес-логіки використано декілька взаємопов'язаних модулів. Контролер DataStreamerController виконує функцію обробки HTTP- та WebSocket-запитів користувачів, координуючи надходження і передачу інформації між фронтендом і сервісами. Через цей компонент реалізовано взаємодію з іншими елементами системи, зокрема з об'єктами, які відповідають за трансляцію оновлень ринку. Контролер є точкою інтеграції клієнтського інтерфейсу з внутрішніми сервісами платформи, забезпечуючи перетворення запитів у відповідні бізнес-операції та зворотну передачу результатів користувачеві у вигляді повідомлень або потокових подій.

Система оновлень побудована на базі кількох спеціалізованих компонентів, що відповідають за передачу різних типів даних у режимі реального часу. Клас UpdateBroadcaster виступає базовим елементом для розсилки повідомлень підписаним клієнтам, використовуючи подійну модель публікації та підписки. На його основі реалізовано два конкретні модулі — PriceUpdateBroadcaster та OrderUpdateBroadcaster. Перший із них забезпечує передачу актуальних даних про зміну курсів криптовалют, тоді як другий відповідає за трансляцію оновлень станів ордерів і транзакцій. Обидва компоненти реалізують реактивну логіку, що дозволяє надсилати повідомлення одразу після виникнення події, мінімізуючи затримку між джерелом даних і кінцевим користувачем [38].

Архітектурна взаємодія між контролерами та модулями розсилки побудована за принципом подієвої системи. Коли відбувається оновлення даних у сховищі або надходить нова інформація від зовнішніх API, відповідний сервіс генерує подію, яка обробляється UpdateBroadcaster і миттєво надсилається клієнтам через WebSocket. Такий підхід уможливлює побудову системи з низькою латентністю, у якій події передаються асинхронно, без блокування основних потоків виконання. Крім того, усі

події логуються для подальшого моніторингу, що підвищує прозорість і контроль за станом платформи. Важливою частиною архітектури є розділення відповідальності між модулями. Контролери не містять бізнес-логіки, а лише спрямовують запити до відповідних сервісів. Сервіси, у свою чергу, виконують обчислення, валідацію та взаємодію з базами даних або кешами Redis. Такий принцип відповідає архітектурному патерну Service Layer, який сприяє зменшенню зв'язності між компонентами та підвищенню гнучкості при оновленні або тестуванні окремих частин системи. Завдяки цьому зміни у бізнес-логіці не впливають на рівень представлення чи комунікації, що є необхідною умовою для підтримки великих мікросервісних рішень.

Загальна структура платформи забезпечує одночасне функціонування кількох потоків обробки даних. Один із них відповідає за збір інформації з біржових джерел, інший за публікацію оновлень для користувачів, а третій за контроль і збереження транзакцій. Взаємодія між ними здійснюється через асинхронні черги та реактивні стріми, що дає можливість досягти високої пропускної здатності системи без втрати стабільності. Завдяки такій архітектурі платформа здатна обслуговувати значну кількість одночасних користувачів, підтримуючи швидку реакцію на ринкові зміни.

Отже, архітектура розробленої системи є багаторівневою, подійно-орієнтованою та реактивною. Вона поєднує модульність, гнучкість і продуктивність, що дозволяє адаптувати платформу до змін ринку та регуляторних вимог без необхідності повної перебудови програмного ядра. Взаємодія між компонентами побудована на стандартизованих протоколах і асинхронних каналах зв'язку, що гарантує стабільну роботу навіть за умов пікового навантаження. Такий підхід формує технологічну основу для подальшого розвитку системи та інтеграції з новими фінансовими сервісами у межах єдиної цифрової екосистеми.

4.3 Реалізація модулів для роботи з криптовалютами та банківськими системами

Реалізація функціональних модулів бізнес-орієнтованої платформи спрямована на забезпечення повного циклу обробки транзакцій між криптовалютами та фіатними активами. Архітектура системи ґрунтується на мікросервісному підході, який дозволяє розподіляти завдання між незалежними компонентами. Кожен із модулів має власну зону відповідальності — від збору ринкових даних і формування запитів до виконання операцій і трансляції результатів у реальному часі. Такий підхід забезпечує стабільність, гнучкість і масштабованість під час роботи з високонавантаженими потоками інформації.

Головний модуль реалізовано у класі Application.java, який ініціалізує середовище Micronaut, підключає конфігураційні параметри й реєструє контролери, сервіси та компоненти платформи. Ініціалізація додатка відбувається через автоматичне створення контексту виконання, після чого активується ініціалізатор WebSocket-підключень WebSocketApplicationRunner, що забезпечує двосторонню комунікацію між сервером і клієнтами. Завдяки цьому платформа функціонує в режимі безперервного обміну даними, а інформація про зміни ринкових курсів або ордерів надходить до користувачів миттєво [39].

Ключовим елементом системи є контролер DataStreamerController, який обробляє запити користувачів, ініціює створення підписок і координує передавання даних між сервісами. Його роботу можна проілюструвати наступним фрагментом коду:

```
@Controller("/stream")
public class DataStreamerController {
   private final PriceUpdateBroadcaster broadcaster;
   public DataStreamerController(PriceUpdateBroadcaster broadcaster) {
      this.broadcaster = broadcaster;    }
   @Get("/prices")
   public Flowable<String> streamPrices() {
      return broadcaster.getPriceStream();    }}
```

Цей приклад демонструє реактивну логіку обробки подій. Контролер приймає запит клієнта, підключає його до потоку оновлень і миттєво передає зміну даних за

допомогою WebSocket-з’єднання. Завдяки використанню Micronaut і бібліотеки Reactive Streams обробка запитів відбувається асинхронно, без блокування головного потоку виконання, що гарантує високу пропускну здатність системи.

Передавання повідомлень у режимі реального часу реалізовано через модулі UpdateBroadcaster, PriceUpdateBroadcaster та OrderUpdateBroadcaster. Базовий клас UpdateBroadcaster відповідає за розсилку подій усім підписникам, тоді як його похідні спеціалізуються на певних типах даних: цінових оновленнях або статусах ордерів. Після отримання нових даних із зовнішніх API система генерує подію, що передається у внутрішню шину повідомлень і транслюється клієнтам у режимі реального часу. Такий підхід забезпечує низьку латентність і стабільність при великій кількості одночасних підключень [40]. Окремі підсистеми відповідають за взаємодію з фіатними платіжними шлюзами та біржовими API. Вони реалізують механізми створення ордерів, розрахунку комісій, перевірки транзакцій і підтвердження операцій. Отримані результати синхронізуються з брокерами оновлень, що забезпечує наскрізну цілісність даних між криптовалютною та банківською частинами. Завдяки уніфікованому формату JSON усі модулі системи легко інтегруються з новими сервісами без потреби змінювати базовий код.

Як ми бачимо, реалізована система поєднує реактивну архітектуру, асинхронний обмін повідомленнями та безпечну інтеграцію з фінансовими шлюзами. Це дозволяє здійснювати криптовалютні й фіатні транзакції у режимі реального часу, підтримуючи високу продуктивність, стабільність і відповідність сучасним вимогам фінансових технологій.

4.4 Механізми безпеки, автентифікації та шифрування даних

Безпека є ключовим аспектом розроблення бізнес-орієнтованої платформи, оскільки система працює з фінансовими транзакціями, персональними даними користувачів та інтегрується з банківськими й криптовалютними шлюзами. Концепція інформаційної безпеки була закладена на рівні архітектури платформи, що дало змогу реалізувати комплексний підхід до захисту даних, автентифікації користувачів і контролю доступу. Основна мета цих механізмів полягає у збереженні

конфіденційності, цілісності та доступності інформації під час усіх етапів обробки транзакцій.

Усі канали зв’язку між компонентами системи функціонують через захищений протокол HTTPS (SSL/TLS), який забезпечує шифрування даних на транспортному рівні. Такий підхід гарантує, що навіть у разі перехоплення пакета зловмисник не матиме змоги отримати зміст переданої інформації. Крім того, обмін повідомленнями між мікросервісами, які взаємодіють у межах однієї інфраструктури, здійснюється через захищені внутрішні канали з автентифікацією за ключами доступу. Для зберігання чутливих даних використовується алгоритм AES-256, що є сучасним галузевим стандартом симетричного шифрування [41].

Система автентифікації користувачів базується на токенах доступу, які створюються після успішного входу в систему та мають обмежений термін дії. Такий механізм дозволяє забезпечити безпечну ідентифікацію без передачі паролів у кожному запиті. У середовищі Micronaut це реалізується через компонент безпеки, який перевіряє валідність токена для кожного запиту до API. Нижче наведено приклад фрагмента коду, який демонструє спрощену перевірку токена на основі HTTP-заголовка Authorization:

```
@Filter("/api/**")
public class AuthFilter implements HttpServerFilter {
   @Override
   public Publisher<MutableHttpResponse<?>> doFilter(HttpRequest<?> request, ServerFilterChain chain) {
      String token = request.getHeaders().get("Authorization");
      if (token != null && TokenValidator.isValid(token)) {
         return chain.proceed(request);       }
      return Publishers.just(HttpResponse.unauthorized());    }}
```

У цьому прикладі реалізовано базовий фільтр доступу, який перевіряє токен перед обробкою запиту. Якщо токен дійсний, запит передається далі до бізнес-логіки, в іншому випадку клієнт отримує повідомлення про відмову в доступі. У реальному середовищі система використовує асиметричне шифрування для підпису токенів, що запобігає підробці або повторному використанню ключів [42].

Важливою складовою є контроль прав користувачів відповідно до їхніх ролей. Платформа передбачає багаторівневу модель доступу, у якій визначаються права для

адміністратора, корпоративного клієнта та звичайного користувача. Кожен запит до API перевіряється на відповідність ролі, зазначеній у токені. Такий підхід гарантує, що користувач має доступ лише до тих операцій, які дозволені його рівнем автентифікації. Додатково ведеться журнал усіх дій користувачів, що дозволяє здійснювати аудит безпеки та виявляти несанкціоновані дії.

Система зберігає паролі у хешованому вигляді з використанням алгоритму BCrypt, який забезпечує стійкість до атак типу brute-force. Для тимчасових ключів автентифікації застосовується механізм одноразових токенів (One-Time Tokens), які автоматично видаляються після використання. Перед відправкою даних у зовнішні платіжні шлюзи відбувається повторне шифрування на рівні транспортного протоколу, що мінімізує ризик перехоплення інформації у проміжних вузлах мережі.

Додатковий рівень захисту реалізовано через систему моніторингу безпеки. Усі події авторизації, спроби доступу, помилки валідації та неуспішні операції записуються до спеціального журналу подій [43]. Ці дані використовуються для аналізу інцидентів і формування статистичних звітів про активність користувачів. Механізм журналювання дозволяє виявляти аномальні шаблони поведінки та потенційні загрози у реальному часі. Дивлячись на це, комплексна система безпеки платформи поєднує сучасні методи автентифікації, шифрування та моніторингу. Вона забезпечує захист як даних користувачів, так і внутрішніх процесів обміну інформацією між мікросервісами. Завдяки використанню реактивної архітектури та токен-орієнтованої авторизації досягається високий рівень стійкості до зовнішніх атак і гарантується відповідність міжнародним вимогам до інформаційної безпеки фінансових систем.

4.5 Опис користувацького інтерфейсу та функціональних сценаріїв

Користувацький інтерфейс розробленої бізнес-орієнтованої платформи CryptoBridge побудований за принципами структурної логіки, інформативності та візуальної узгодженості. Його головна мета — забезпечити зручну навігацію між основними функціями системи та надати користувачу можливість здійснювати операції з цифровими й фіатними активами без надмірних технічних дій. Структура

інтерфейсу передбачає бічну панель навігації з розділами Dashboard, My Wallet, Transaction, Crypto та Exchange, а також верхню службову панель із пошуком, повідомленнями та налаштуваннями профілю. Таке компонування формує зрозумілу ієрархію елементів і дозволяє швидко переходити між розділами [44].

Основним робочим простором користувача є модуль My Wallet, у якому відображаються всі прив’язані віртуальні рахунки у вигляді інтерактивних рядків із балансом, типом валюти та зміною вартості активів від моменту останньої операції. Користувач може змінювати параметри рядків у налаштуваннях профілю, зокрема порядок, активні кнопки та за який період відображати зміну вартості активу.

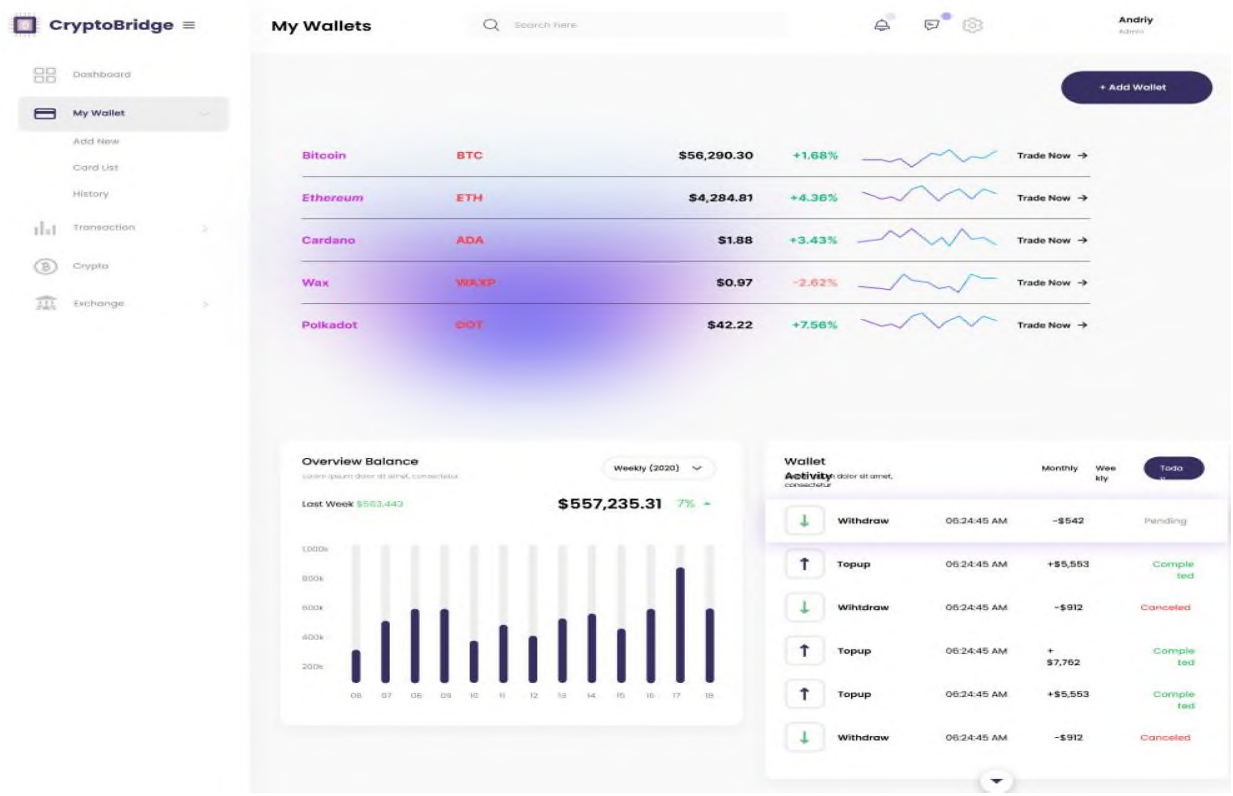

Рисунок 4.1 – Інтерфейс головного модуля My Wallet

У нижньому сегменті сторінки розташовано аналітичний блок Overview Balance, що відображає динаміку фінансових показників у вигляді графіків і процентних змін за вибраний період. Праворуч подано журнал Wallet Activity, який містить перелік виконаних транзакцій із зазначенням дати, типу операції, суми й статусу (Completed, Pending, Cancelled). Така структура дозволяє контролювати поточний стан рахунків і швидко перевіряти фінансову історію без переходу до

окремих підменю. Комбінація графічних і табличних елементів створює інтуїтивно зрозуміле середовище для моніторингу фінансової активності [45].

Початковим сценарієм взаємодії користувача з системою є реєстрація. На сторінці Sign Up відображається компактна форма з полями Email, Password, Confirm Password і BTC Address (optional). Після заповнення форми користувач отримує електронне підтвердження, після чого створюється обліковий запис із унікальним токеном доступу. Інтерфейс форми побудований у мінімалістичному стилі, що спрощує процес автентифікації та знижує ризик помилки під час введення даних.

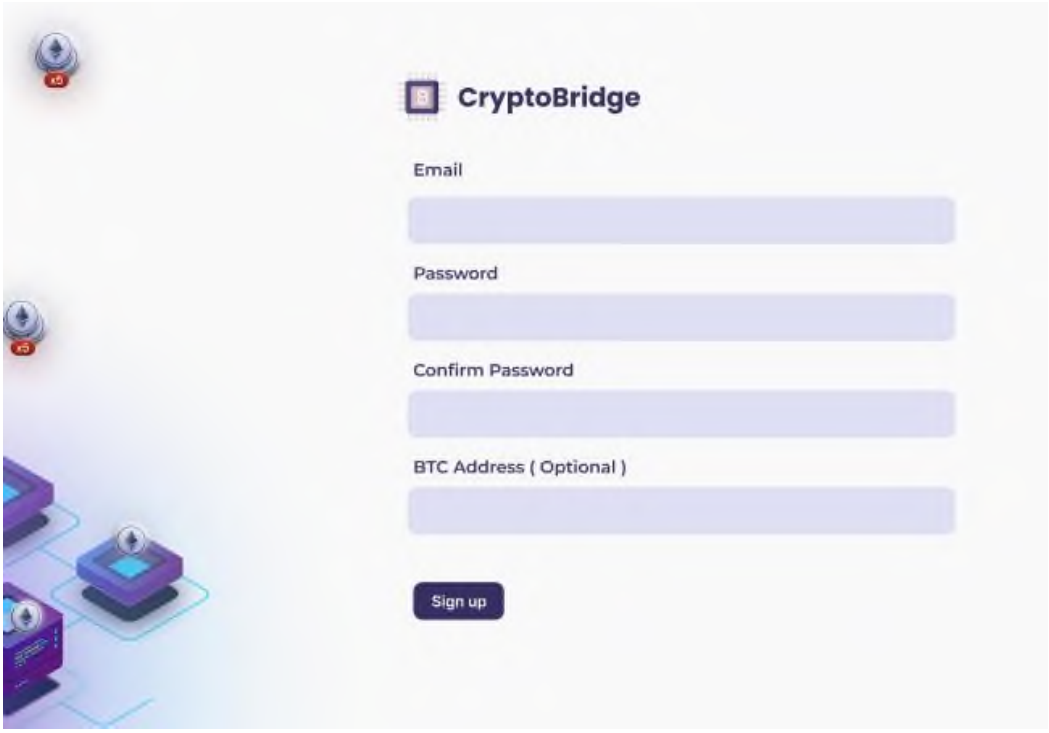

Рисунок 4.2 – Форма реєстрації нового користувача

Після входу до системи користувач переходить до головної панелі, де може створити свій перший гаманець через функцію Add Wallet. Процес передбачає вибір валюти, назву рахунку та підтвердження створення. Після цього новий гаманець автоматично з’являється на панелі My Wallet і синхронізується з базою даних. Для забезпечення безпеки кожна операція підписується унікальним сеансовим ключем, а доступ здійснюється виключно через захищене з’єднання HTTPS.

Основні фінансові операції виконуються в модулі Transaction, де користувач може ініціювати переказ або обмін. Інтерфейс містить поля введення адреси

отримувача, суми, типу операції (FaucetPay чи Direct Withdrawal) та кнопку підтвердження Place Withdrawal. Під час заповнення полів система автоматично обчислює комісію й суму, що надійде на рахунок після вирахування зборів. Наприклад, при введенні 0.02000000 BTC користувач одразу бачить комісію 0.003 BTC та суму до отримання 0.017 BTC. Вікно оформлено в затемненому стилі, що фокусує увагу на ключових елементах введення.

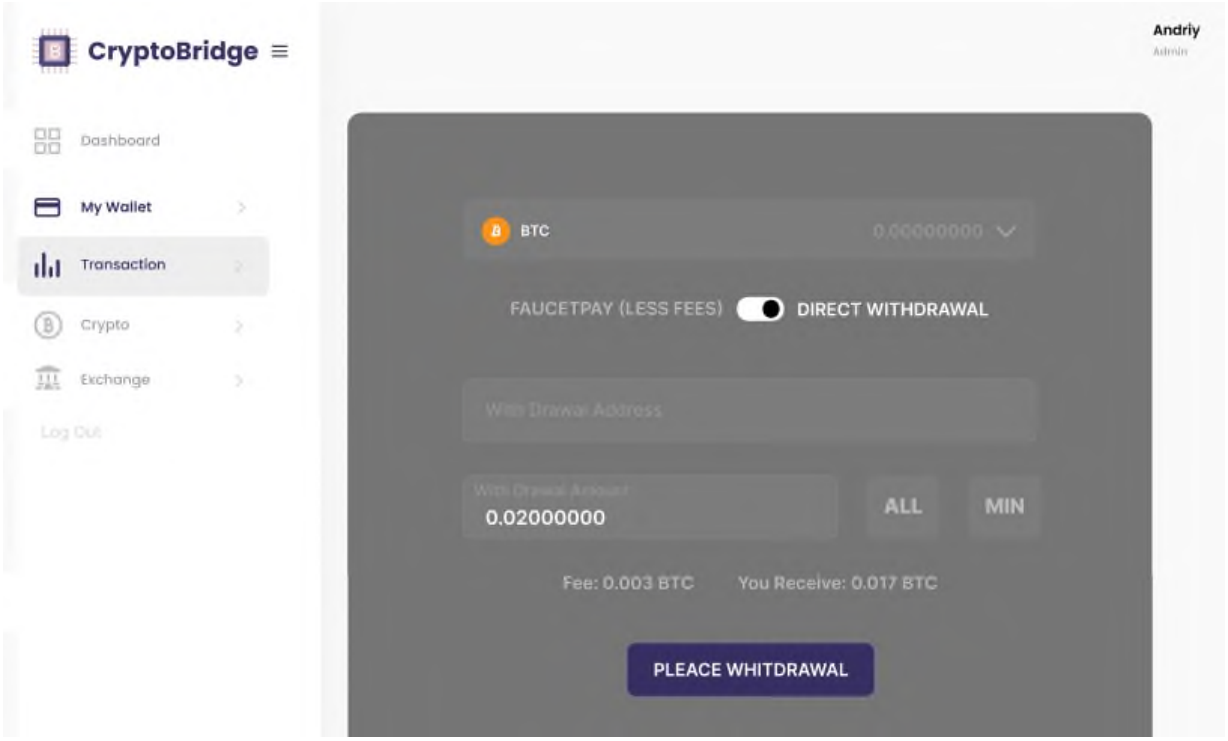

Рисунок 4.3 – Вікно створення транзакції у модулі Transaction

Інтерфейс системи побудований у єдиній стилістиці з використанням нейтральних кольорів, плавних градієнтів і чіткої типографіки. Всі основні елементи мають узгоджене розташування, що мінімізує когнітивне навантаження під час роботи. Графіки, картки та діаграми відображають фінансові дані у зрозумілому форматі, а колірні акценти допомагають швидко розрізняти стани рахунків та типи операцій.

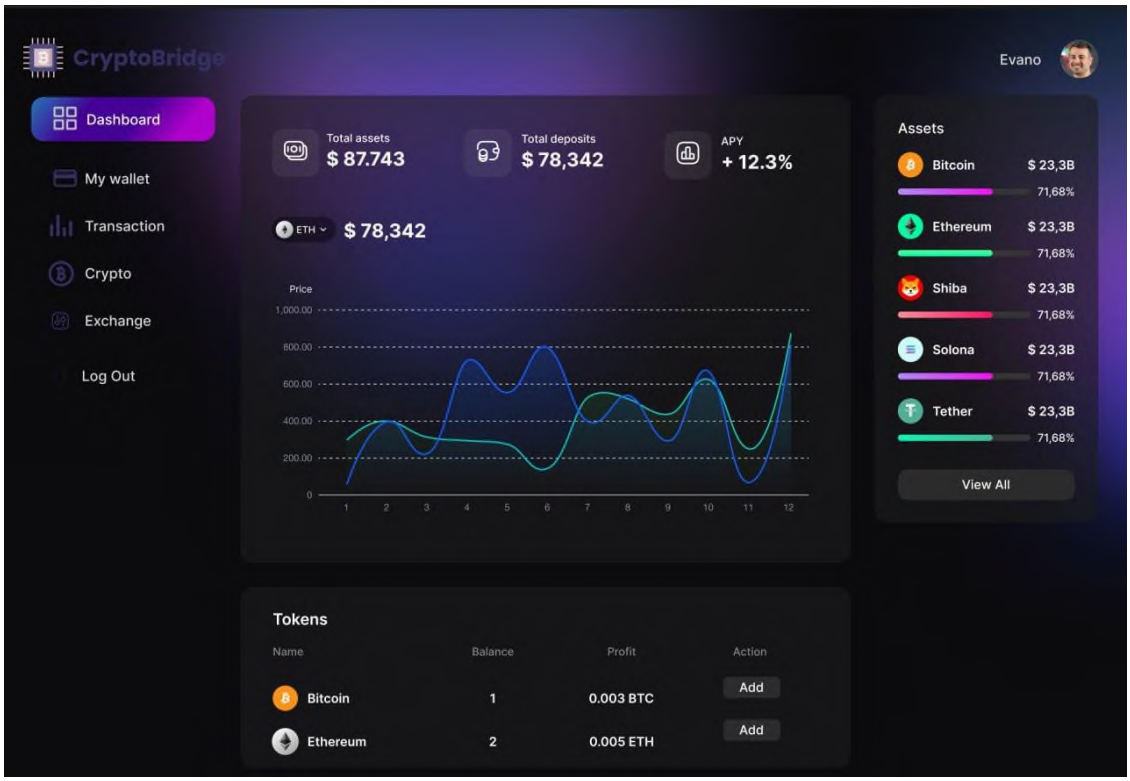

Рисунок 4.4 Вигляд панелі Dashboard при обраній темній темі відображення

Завдяки адаптивній верстці інтерфейс коректно відображається як на десктопних пристроях, так і на мобільних екранах.

Загалом користувацький інтерфейс платформи CryptoBridge поєднує функціональність і зручність, підтримуючи повний цикл сценаріїв від реєстрації та створення гаманців до виконання транзакцій і перегляду аналітики. Продумана структура, візуальна узгодженість та інтерактивні елементи сприяють ефективній роботі користувачів і формують відчуття надійності під час здійснення фінансових операцій у цифровому середовищі.

Висновки до розділу 4

У результаті проведеного проєктування й реалізації програмного забезпечення сформовано комплексну архітектуру бізнес-орієнтованої платформи CryptoBridge, яка поєднує сучасні технологічні рішення та високу стабільність роботи. Застосування мови Java та фреймворку Micronaut забезпечило швидкодію, ефективне використання ресурсів і підтримку реактивних потоків, необхідних для обробки

великої кількості одночасних запитів. Вибір PostgreSQL і Redis дав змогу реалізувати баланс між надійним зберіганням фінансових даних і швидким доступом до оперативної інформації, тоді як контейнеризація за допомогою Docker створила гнучку інфраструктуру для масштабування та розгортання компонентів.

Завдяки модульній побудові система демонструє узгоджену взаємодію між сервісами, що відповідають за обробку транзакцій, оновлення ринкових даних, шифрування й автентифікацію користувачів. Реактивна архітектура, механізми безпеки та інтуїтивний інтерфейс формують цілісне середовище для роботи з криптовалютними й фіатними активами у режимі реального часу. Отже, розроблене програмне забезпечення можна розглядати як технологічно зрілу основу для подальшого розвитку фінансових сервісів, здатну забезпечити стійкість, масштабованість і комфорт користувацької взаємодії в межах єдиної цифрової екосистеми.

РОЗДІЛ 5. РОЗРОБЛЕННЯ СТАРТАП-ПРОЄКТУ

5.1 Опис ідеї бізнес-орієнтованої платформи

Ідея створення бізнес-орієнтованої платформи для транзакцій між криптовалютами та фіатними грошима виникла як відповідь на потребу інтеграції двох фінансових середовищ традиційного банківського сектору та ринку цифрових активів. Сучасні учасники фінансової екосистеми від корпоративних клієнтів до приватних користувачів прагнуть отримати інструмент, який забезпечує прозору, швидку й безпечну взаємодію між різними видами активів. У цій системі криптовалюта розглядається не лише як спекулятивний актив, а як повноцінний платіжний інструмент, який може ефективно інтегруватися у бізнес-процеси компаній через автоматизовані обмінні механізми.

Платформа поєднує функції електронного фінансового посередника, аналітичного сервісу та інтеграційного середовища. Вона надає користувачам можливість виконувати транзакції в режимі реального часу, здійснювати конвертацію валют, проводити внутрішні розрахунки між учасниками, а також автоматично формувати звітність для бухгалтерського обліку чи податкової звітності. Архітектура системи базується на мікросервісному підході, що забезпечує гнучкість у масштабуванні, інтеграцію з зовнішніми API та можливість адаптації під регуляторні вимоги різних країн [12].

Особливістю розробленої ідеї є орієнтація на потреби бізнес-середовища, де важливу роль відіграє автоматизація рутинних фінансових операцій. Для корпоративних клієнтів передбачено можливість створення правил конвертації, лімітів і шаблонів регулярних операцій. Це дозволяє інтегрувати платформу в існуючі фінансові процеси компаній і мінімізувати участь людини у виконанні типових розрахункових дій. Для приватних користувачів система пропонує спрощений інтерфейс із доступом до крипто- і фіатних рахунків, історії операцій і системи комплаєнс-перевірок у фоновому режимі [46]. Концепція побудови бізнес-орієнтованої платформи ґрунтується на поєднанні швидкодії блокчейн-технологій і стабільності банківської інфраструктури. Усі операції супроводжуються

автоматичною перевіркою відповідності регуляторним нормам і політикам безпеки. Кожна транзакція має цифровий слід, який зберігається у базі даних і може бути використаний для аудиту [47]. Система також взаємодіє із зовнішніми сервісами ліквідності, що забезпечують вигідний курс конвертації та достатній обсяг ресурсів для виконання великих ордерів без затримок.

Реалізація такої платформи дозволяє створити середовище, у якому користувачі можуть оперувати як фіатними, так і криптовалютними активами без потреби у численних посередниках. Це не лише зменшує витрати на обслуговування операцій, а й відкриває можливості для розроблення нових фінансових продуктів — смарт-контрактів, токенізованих активів або автоматизованих систем розрахунків між компаніями. Платформа формує основу для подальшої інтеграції у фінтех-екосистеми, підтримуючи принципи прозорості, гнучкості та відповідності вимогам сучасного ринку.

5.2 Аналіз технологічних можливостей реалізації проєкту

Реалізація бізнес-орієнтованої платформи для транзакцій між крипто- та фіатними валютами потребує поєднання сучасних інформаційних технологій, які забезпечують одночасно швидкість обробки операцій, безпеку даних і гнучкість у масштабуванні. Основна логіка системи базується на мікросервісному підході, який дозволяє розділити платформу на окремі незалежні компоненти.

Кожен модуль виконує свою функцію, авторизація користувачів, обробка транзакцій, перевірка комплаєнсу, формування звітів і може оновлюватися чи вдосконалюватися без порушення роботи всієї системи. Такий підхід дає можливість підключати нові сервіси, інтегрувати сторонні інструменти, а також швидко реагувати на зміни ринку фінансових технологій. На рівні програмного забезпечення найдоцільніше використання мови Java, що поєднує стабільність корпоративного рівня з широкою підтримкою бібліотек для обробки фінансових операцій і взаємодії з API банківських систем [48, с. 147-149]. Завдяки фреймворку Spring Boot можна реалізувати багаторівневу архітектуру, де бізнес-логіка, сервіси безпеки й інтерфейси даних працюють як єдиний узгоджений механізм. Ця технологія дозволяє швидко

створювати RESTful API, через які відбувається обмін даними між клієнтською частиною, сервером та зовнішніми джерелами ліквідності. Для забезпечення стійкості системи використовується контейнеризація через Docker, що спрощує розгортання та управління мікросервісами у хмарному середовищі [49, с. 325].

Збереження інформації про транзакції, користувачів і фінансові баланси здійснюється за допомогою системи управління базами даних PostgreSQL. Вона підтримує ACID-транзакції, що гарантує точність обчислень і узгодженість даних навіть при високих навантаженнях.

Додатково застосовується Redis для кешування найчастіше запитуваних даних, що суттєво підвищує швидкість відповіді системи. Для розподілу навантаження між запитами базу даних можна масштабувати горизонтально, використовуючи реплікацію. В аналітичних цілях передбачається створення окремого сховища (Data Warehouse), де агреговані дані зберігатимуться для формування статистичних звітів і прогнозів. Оскільки система обробляє фінансові операції та персональні дані користувачів, усі з'єднання виконуються через зашифрований протокол HTTPS, а передача даних супроводжується шифруванням на рівні сервера. Для зберігання паролів і токенів застосовується алгоритм bcrypt, який запобігає несанкціонованому доступу до конфіденційної інформації [50]. Додатковий рівень захисту забезпечує двофакторна автентифікація, а також моніторинг підозрілої активності через систему логів. Для транзакцій у блокчейні використовуються перевірені криптографічні алгоритми, які гарантують незмінність та відстежуваність операцій.

Підтримка взаємодії з блокчейн-мережами реалізується через інтеграційні шлюзи, які підключаються до нод різних криптовалют. Цей підхід дозволяє обробляти транзакції у кількох мережах, наприклад, Bitcoin, Ethereum, Binance Smart Chain безпосередньо з платформи. Для цього використовуються готові бібліотеки, такі як web3.js чи ethers.js, які забезпечують взаємодію із смарт-контрактами, перевірку стану мережі та підтверджень блоків. Водночас для фіатних валют інтеграція виконується через банківські API з підтримкою Open Banking стандартів, що дає змогу здійснювати перекази, отримувати виписки та перевіряти статус платежів у режимі реального часу.

Фронтенд частина системи розробляється на основі React.js, що забезпечує зручний інтерфейс і можливість динамічного оновлення даних без перезавантаження сторінки. Користувач може керувати рахунками, відстежувати курси валют, створювати транзакції та отримувати звіти через інтуїтивно зрозумілу панель. Для корпоративних клієнтів передбачено окремий кабінет із розширеним функціоналом налаштування прав доступу, інтеграція через API, планування автоматичних операцій. Інтерфейс адаптований під мобільні пристрої, що дозволяє користуватися системою у будь-якому середовищі. Ще одним технологічним напрямом реалізації є впровадження аналітичних інструментів для моніторингу та оцінювання роботи системи. Через сервіси Grafana і Prometheus здійснюється збір показників продуктивності, аналіз навантажень і виявлення збоїв у реальному часі [51]. На основі цих даних проводиться оптимізація алгоритмів обробки транзакцій, що забезпечує стабільність платформи при зростанні кількості користувачів. Такий підхід робить систему не лише гнучкою, а й самодостатньою в плані підтримки й масштабування.

Отже, технологічна база проєкту поєднує перевірені рішення корпоративного рівня з інноваційними інструментами, орієнтованими на ринок цифрових активів. Використання сучасних фреймворків, захищених протоколів і модульної архітектури забезпечує надійну основу для створення платформи, яка здатна функціонувати без збоїв, швидко обробляти запити й адаптуватися до змін у технологічному середовищі.

5.3 Аналіз ринкових можливостей та конкурентного середовища

Ринок платформ для обміну та взаємодії між криптовалютами і фіатними валютами сьогодні розвивається динамічно. Зростання інтересу до цифрових активів, збільшення кількості користувачів блокчейн-гаманців і поступове впровадження регулювання у фінтех-сфері створюють сприятливі умови для розвитку подібних сервісів. Підприємства дедалі частіше інтегрують криптовалютні платежі у свою діяльність, а банки розширюють партнерство з фінтех-компаніями. Це відкриває можливість для створення продукту, який поєднує гнучкість криптовалютного ринку з надійністю банківської інфраструктури, забезпечуючи повний цикл обробки транзакцій у межах однієї екосистеми. На світовому рівні сформувався цілий спектр

гравців, що працюють у сегменті обмінних платформ. Серед найвідоміших можна назвати Binance, Kraken, Coinbase і Bitstamp, які забезпечують базовий набір послуг обмін валют, торгівлю та зберігання активів. Проте більшість із них орієнтована переважно на роздрібного користувача або трейдера, тоді як потреби корпоративних клієнтів залишаються менш охопленими.

Для бізнесу важливі інтеграція з бухгалтерськими системами, автоматизована звітність, комплаєнс-моніторинг та можливість масштабування під власну інфраструктуру. Саме тут з'являється нішевий простір для нової платформи, яка зможе заповнити цю прогалину [52]. У межах українського ринку фінансових технологій останні роки спостерігається поступове легалізування діяльності з цифровими активами. Прийняття законодавчої бази щодо віртуальних активів і рух до імплементації європейських регуляцій відкривають шлях для офіційного функціонування криптовалютних сервісів.

Українські компанії, зокрема IT-інтегратори, банки та платіжні системи, уже активно шукають способи об'єднання традиційних фінансових послуг з цифровими активами. Такий контекст створює вигідні передумови для запуску бізнес-орієнтованої платформи, здатної обслуговувати як місцевих, так і міжнародних клієнтів [53].

Конкурентне середовище в цьому сегменті характеризується високою технологічною насиченістю, однак рівень диференціації між платформами залишається відносно невеликим. Більшість рішень пропонують стандартний функціонал, не враховуючи глибоких потреб бізнесу — аналітики, звітності чи адаптації до податкового законодавства. Тому конкурентна перевага може формуватися не лише за рахунок технічної стабільності чи швидкості обробки операцій, а й завдяки рівню інтеграції, якості підтримки клієнтів, простоті користування та надійності комплаєнс-сервісів. Для молодої компанії це шанс створити гнучку платформу, яка пропонує не масовий продукт, а персоналізоване рішення.

У перспективі розвиток бізнес-орієнтованих фінтех-платформ в Україні може стати частиною більшого процесу цифровізації фінансових послуг і розширення

участі країни в глобальній екосистемі віртуальних активів [54]. Попит на безпечні та прозорі сервіси, здатні поєднати фіатні та криптовалютні інструменти, лише зростатиме. За таких умов створення платформи, орієнтованої на корпоративних користувачів, відкриває перспективи виходу на міжнародні ринки, залучення партнерів із суміжних галузей та формування власного бренду у сфері фінансових технологій нового покоління.

5.4 Розроблення маркетингової стратегії та бізнес-моделі

Розроблення маркетингової стратегії бізнес-орієнтованої платформи для транзакцій між крипто- та фіатними валютами ґрунтується на аналізі ринку, визначенні цільової аудиторії та побудові стійкої моделі просування. Платформа позиціонується як сервіс нового покоління, орієнтований на корпоративних користувачів, фінансові компанії, інвестиційні фонди та підприємців, яким потрібна автоматизована система для швидкого обміну валют і формування звітності. У центрі маркетингової концепції —довіра, безпека та простота користування, що мають стати основою бренду та формувати стабільну клієнтську базу.

Перший етап реалізації маркетингової стратегії передбачає створення впізнаваного бренду з чіткою візуальною ідентичністю та зрозумілою комунікацією. Акцент робиться на інформуванні клієнтів про переваги використання платформи мінімальні комісії, швидке виконання транзакцій, інтеграція з бухгалтерськими системами, автоматичне дотримання комплаєнс-вимог. Комунікаційна політика поєднує онлайн-канали (соціальні мережі, галузеві форуми, контент-маркетинг) та офлайн-активності участь у фінтех-конференціях, партнерства з банками й інкубаторами стартапів [55, с. 15-17]. Для залучення клієнтів передбачено використання моделі B2B2C, що поєднує корпоративних користувачів і кінцевих клієнтів через єдину екосистему. Компанії, які користуються платформою, можуть підключати своїх партнерів або клієнтів до обслуговування в межах тієї ж системи, що розширює мережевий ефект. Паралельно діятиме реферальна програма, яка стимулюватиме нових користувачів реєструватися та здійснювати транзакції. Таке

рішення не лише знижує вартість залучення клієнтів, а й створює природний канал поширення бренду серед фінансових спільнот.

Бізнес-модель платформи побудована на поєднанні кількох джерел доходу. Основним є комісійна винагорода за виконані транзакції, розмір якої варіюється залежно від типу користувача та обсягів операцій. Для корпоративних клієнтів передбачено гнучкі тарифні плани з фіксованою щомісячною оплатою за доступ до API та додаткових сервісів, таких як автоматична генерація звітів або модуль комплаєнсу. Додатковий прибуток формується за рахунок інтеграційних послуг, аналітичних звітів та партнерських програм з постачальниками ліквідності. Така модель дозволяє зберігати фінансову стійкість навіть за умов коливань на ринку криптовалют [56].

Просування платформи відбуватиметься поетапно: спершу тестовий запуск з обмеженою кількістю користувачів для перевірки роботи сервісу, далі розширення географії клієнтів через регіональні партнерства. Окрема увага приділяється контент-маркетингу: створення навчальних матеріалів, вебінарів і кейсів, які пояснюють переваги платформи для бізнесу. Такий підхід формує не лише обізнаність аудиторії, а й підкреслює експертність компанії, що є необхідною умовою для залучення корпоративних клієнтів. Стратегічна мета маркетингової діяльності побудова екосистеми довіри, де клієнти не лише здійснюють транзакції, а й відчувають себе частиною технологічного простору. Підтримка, консультації, гнучке налаштування сервісу під індивідуальні потреби усе це формує довгострокові відносини з користувачами. У перспективі планується масштабування бізнес-моделі на інші фінансові сегменти токенізовані активи, цифрові облігації та автоматизовані платіжні рішення для малого та середнього бізнесу. Таким чином, платформа може перетворитися з простої системи обміну на повноцінну фінансову інфраструктуру нового покоління.

5.5 Вимоги до технічної та програмної підтримки стартапу

Ефективна робота бізнес-орієнтованої платформи для транзакцій між крипто- та фіатними валютами неможлива без постійної технічної та програмної підтримки. Цей напрям охоплює моніторинг працездатності системи, оновлення програмного забезпечення, усунення помилок і забезпечення безпеки даних. Підтримка має функціонувати у цілодобовому режимі, адже користувачі з різних часових зон повинні мати безперервний доступ до сервісу. Для цього формується команда технічних спеціалістів, до складу якої входять системні адміністратори, DevOps-інженери, аналітики безпеки та фахівці з підтримки клієнтів.

Базовою вимогою до технічної інфраструктури є стабільність і масштабованість. Система повинна бути розгорнута у хмарному середовищі з можливістю автоматичного збільшення ресурсів у разі підвищеного навантаження. Використання контейнеризації через Docker та оркестрації за допомогою Kubernetes дозволяє швидко відновлювати роботу окремих сервісів після збоїв, не зупиняючи роботу всієї платформи. Регулярне тестування навантажень і резервне копіювання даних забезпечують надійність функціонування системи навіть за форс-мажорних обставин. Безпека охоплює не лише захист від зовнішніх атак, а й контроль доступу до внутрішніх ресурсів [57, с. 11-14]. Необхідно регулярно оновлювати бібліотеки, що використовуються, та впроваджувати патчі безпеки. Важливо забезпечити ізоляцію середовищ розробки, тестування та продуктивної експлуатації, щоб уникнути витоку даних. Для додаткового контролю використовуються інструменти виявлення вразливостей та системи моніторингу, які фіксують будь-які відхилення у поведінці програмного коду чи бази даних.

Технічна підтримка користувачів також відіграє значну роль у розвитку стартапу. Створюється багаторівнева система обслуговування: перша лінія реагує на типові звернення клієнтів через чат або електронну пошту, друга — вирішує складніші технічні питання, а третя займається глибинним аналізом проблеми. Крім того, впроваджується база знань, де користувачі можуть самостійно знаходити відповіді на часті запитання. Такий підхід знижує навантаження на команду підтримки і водночас підвищує задоволеність клієнтів. Для забезпечення стабільності

розвитку стартапу потрібно передбачити процеси документування всіх технічних операцій. Кожне оновлення, зміна коду чи конфігурації має супроводжуватися створенням детального звіту. Це дозволяє відстежувати історію змін і запобігати повторенню помилок. Крім того, технічна документація спрощує адаптацію нових спеціалістів, зберігає послідовність у підходах до розробки та зменшує ризики втрати знань під час ротації персоналу [58].

У перспективі система технічної підтримки повинна перетворитися на окрему сервісну підсистему, інтегровану з аналітичними інструментами. З її допомогою можна прогнозувати навантаження, оцінювати ефективність роботи команди й автоматично розподіляти запити за пріоритетами. Таким чином, технічна й програмна підтримка не обмежується реагуванням на проблеми, а стає активним механізмом управління стабільністю, розвитком і якістю обслуговування бізнес-орієнтованої платформи.

Висновки до розділу 5

Розроблення бізнес-орієнтованої платформи для транзакцій між крипто- та фіатними валютами показало, що поєднання сучасних технологій із глибоким розумінням потреб ринку створює реальні можливості для формування нового типу фінансових сервісів. Запропонована концепція поєднує стабільність банківської системи з гнучкістю блокчейн-рішень, що дозволяє забезпечити безпечну, прозору та швидку взаємодію між учасниками фінансової екосистеми.

Технологічна база проєкту довела свою життєздатність завдяки використанню мікросервісної архітектури, інструментів контейнеризації, хмарної інфраструктури та сучасних мов програмування. Це створює підґрунтя для стабільної роботи системи, її масштабування та інтеграції з іншими сервісами. У поєднанні з аналітичними та безпековими механізмами така структура забезпечує надійну підтримку бізнес-процесів клієнтів.

Розроблена маркетингова стратегія й бізнес-модель формують основу для виходу на конкурентний фінтех-ринок. Поєднання B2B і B2C форматів, система

лояльності, автоматизована підтримка користувачів і акцент на комплаєнс-рішеннях створюють переваги, що сприятимуть розвитку стартапу. У перспективі така платформа може перетворитися на універсальну фінансову екосистему, здатну забезпечити новий рівень цифрової взаємодії між крипто- та фіатним світом.

ВИСНОВКИ

У дипломній роботі вирішено актуальну науково-практичну задачу розроблення бізнес-орієнтованої платформи для транзакцій між криптовалютними та фіатними валютами, що забезпечує інтеграцію цифрових активів у операційну діяльність підприємств через автоматизовані механізми обміну, управління ризиками та дотримання регуляторних вимог. Проведене дослідження охопило аналіз проблемної області, проєктування інформаційної моделі, розроблення математичного та програмного забезпечення, а також формування концепції стартап-проєкту.

Аналіз сучасного стану криптовалютного ринку засвідчив його трансформацію у повноцінний сегмент глобальної фінансової системи з капіталізацією 3,76 трильйона доларів та зростаючою інституційною підтримкою через спотові Bitcoin ETF. Виявлено ключові перешкоди інтеграції криптовалют у бізнес-середовище: регуляторну невизначеність, високу волатильність цін у три-чотири рази вищу за традиційні активи, технічні бар'єри входу та складність дотримання вимог фінансового моніторингу. Порівняльний аналіз провідних платформ Binance, Coinbase, Kraken, ChangeNOW та Gemini продемонстрував їх орієнтацію на трейдингові операції та відсутність спеціалізованого функціоналу для корпоративних потреб, що підтверджує необхідність створення бізнес-орієнтованого рішення.

Розроблена інформаційна модель платформи формалізує взаємодію між корпоративними клієнтами, індивідуальними користувачами, постачальниками ліквідності та регуляторними органами через систему мікросервісів з чітким розподілом відповідальності. Побудовано структуру бази даних на основі PostgreSQL з таблицями Users, Wallets, Transactions, ComplianceLogs та AuditTrail, що забезпечує цілісність фінансової інформації та можливість повного аудиту операцій. Комплект UML-діаграм (варіантів використання, діяльності, класів, компонентів) та ER-діаграма визначають архітектурні зв'язки між модулями та створюють основу для програмної реалізації.

Математичне забезпечення платформи включає формалізацію транзакційних процесів через систему предикатів верифікації $V\_funds(t) \land V\_compliance(t) \land$

V_limits(t) та механізм резервування балансів, що гарантує атомарність операцій. Розроблено модель ціноутворення з агрегацією котирувань від множини джерел через зважений середній курс $\bar{r}(v_s, v_t, \tau) = \Sigma_{i=1}^{p} w_i(\tau) \cdot r_i(v_s, v_t, \tau)$ та адаптивним спредом $s(v_s, v_t, \tau) = s_0 + \alpha \cdot \sigma(v_s, v_t, \tau)$, що враховує поточну волатильність ринку. Багаторівнева структура комісій $\varphi(u, a) = \varphi_0(a, v_s, v_t) \cdot (1 - \delta(V(u, \tau)))$ диференціює вартість послуг залежно від кумулятивного обсягу операцій користувача. Моделювання ризиків через VaR-методологію, EWMA-волатильність та кореляційний аналіз забезпечує кількісну оцінку потенційних збитків з рівнем довіри 95-99%.

Алгоритми оптимізації транзакцій реалізують динамічне розбиття великих обсягів, жадібну маршрутизацію між постачальниками ліквідності з ефективністю 98,5% та weighted round-robin балансування навантаження з автомасштабуванням при перевищенні порогу утилізації 70%. Емпірична верифікація на історичних даних підтвердила адекватність моделей з похибкою прогнозування курсів 0,07-0,25% та латентністю обробки запитів 150 мс для 95-го перцентиля операцій. Адаптивне управління розміром комісії для блокчейн-транзакцій забезпечує економію до 23% порівняно з постійною високою комісією при збереженні точності підтвердження 91% у цільовому часовому вікні.

Програмна реалізація платформи виконана на основі Java та фреймворку Micronaut, що забезпечує швидкий старт сервісів, низьке споживання ресурсів та підтримку реактивних потоків. Мікросервісна архітектура включає модулі автентифікації, обробки транзакцій, блокчейн-шлюзу, ціноутворення, комплаєнсу та аналітики з взаємодією через REST API та WebSocket-з'єднання. Інтеграція з PostgreSQL для зберігання транзакційних даних та Redis для кешування забезпечує баланс між надійністю та продуктивністю. Контейнеризація через Docker створює гнучку інфраструктуру для горизонтального масштабування. Механізми безпеки охоплюють шифрування SSL/TLS на транспортному рівні, AES-256 для зберігання чутливих даних, токен-орієнтовану автентифікацію та багаторівневу модель контролю доступу з журналюванням всіх подій для аудиту.

Розроблена концепція стартап-проєкту визначає маркетингову стратегію, орієнтовану на корпоративних користувачів через B2B2C модель з реферальною програмою та гнучкими тарифними планами. Бізнес-модель поєднує кілька джерел доходу: комісійну винагороду за транзакції, підписку на API для корпоративних клієнтів, інтеграційні послуги та партнерські програми з постачальниками ліквідності. Технологічна база з мікросервісною архітектурою, хмарною інфраструктурою та цілодобовою технічною підтримкою створює передумови для масштабування та виходу на міжнародні ринки фінансових технологій.

Результати дослідження формують комплексне рішення, що поєднує теоретичне обґрунтування необхідності бізнес-орієнтованих платформ для крипто-фіатних транзакцій з практичною реалізацією архітектури, математичних моделей та програмного забезпечення. Розроблена платформа забезпечує інтеграцію криптовалют у операційну діяльність підприємств через автоматизацію розрахунків, прозорі тарифи, інструменти управління ризиками та відповідність регуляторним вимогам, створюючи технологічну основу для подальшого розвитку цифрової фінансової екосистеми.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. TokenInsight. *Crypto Market Report Q4 2024.* https://tokeninsight.com/en/research/reports/crypto-market-report-q4-2024 (дата звернення: 05.10.2025).

2. SlickCharts. *Top Currencies by Market Capitalization.* https://www.slickcharts.com/currency (дата звернення: 05.10.2025).

3. Дарчик Г. М. Сучасний стан правового регулювання криптовалюти в Україні. Зарубіжний досвід регулювання ринку криптовалют. *Науковий вісник Ужгородського національного університету. Серія: Право.* 2025. Т. 2, № 87. DOI: https://doi.org/10.24144/2307-3322.2025.87.2.50.

4. Петрина А. Розвиток електронного бізнесу в умовах інформаційної глобалізації : дис. Тернопіль : ЗУНУ, 2024. https://dspace.wunu.edu.ua/bitstream/316497/55178/1/Петрина.pdf (дата звернення: 05.10.2025).

5. Застьола Є. О. Аналіз сучасних трендів у плануванні та прогнозуванні бізнес-середовища в умовах цифрової економіки з використанням криптовалют. 2024. https://repository.kpi.kharkov.ua/server/api/core/bitstreams/89731180-dc1e-4174-a006-cd528c822ea3/content (дата звернення: 05.10.2025).

6. Князєва О., Толкачова Г., Скоробогатов К. Криптовалюта як цифрова фінансова інновація. *Sustainable development and digital innovation.* 2024. С. 227. https://onu.edu.ua/pub/bank/userfiles/files/fmvps/kaf_svit_gos_mij_eko_vidnosin/mijnarodni_proekti/-Sustainable_development_and_digital_innovation_compressed.pdf (дата звернення: 05.10.2025).

7. Бончукова О. В. Концептуальні рамки і економічне забезпечення інформаційної безпеки компанії у глобальному бізнес-середовищі. 2019. https://essuir.sumdu.edu.ua/items/b6e44423-30c0-42b0-b44f-efd2a9421f37 (дата звернення: 05.10.2025).

8. Явдик М. Р. Проектування моніторингової системи для синтезу даних з різних API криптовалютних платформ. 2025.

http://repository.lnau.edu.ua:8080/jspui/bitstream/123456789/2536/1/Dylp_Yavdyk%20MR.pdf (дата звернення: 06.10.2025).

9. Makridis C. A., Fröwis M., Sridhar K., Böhme R. The rise of decentralized cryptocurrency exchanges: Evaluating the role of airdrops and governance tokens. *Journal of Corporate Finance.* 2023. Vol. 79(C).

10. Кметик В. В. Прогнозування курсу криптовалюти в умовах невизначеності : дис. 2022. https://dspace.wunu.edu.ua/bitstream/316497/47286/1/Кметик.pdf (дата звернення: 06.10.2025).

11. Ковальчук В. Р. Світовий ринок криптовалют в умовах трансформації глобальної економічної системи. 2021. https://er.knutd.edu.ua/bitstream/123456789/19259/1/Diplom051_Kovalchuk_Kvita.pdf (дата звернення: 07.10.2025).

12. Aramonte S., Huang W., Schrimpf A. DeFi risks and the decentralisation illusion. *BIS Quarterly Review.* March 2022. С. 21–39. https://www.bis.org/publ/qtrpdf/r_qt2203b.htm (дата звернення: 06.10.2025).

13. *The 2024 Crypto Adoption Index: Institutional engagement and compliance trends.* Chainalysis. URL: https://www.chainalysis.com/blog/2024-global-crypto-adoption-index (дата звернення: 06.10.2025).

14. Литовченко О., Дячек В., Мітін М. Трансформація бізнес-моделей підприємств в умовах цифровізації економіки. Економіка та суспільство. 2024. Вип. 69. URL: https://doi.org/10.32782/2524-0072/2024-69-36 (дата звернення: 07.10.2025).

15. Belova I., Homotiuk A., Yaroshchuk O. Цифрова трансформація управлінських та бізнес-процесів в Україні під час воєнного стану. Ekonomichnyy analiz. 2024. Т. 34, № 1. С. 42–52. URL: http://econa.wunu.edu.ua/index.php/econa/article/view/5935 (дата звернення: 07.10.2025).

16. Литюга Є. С. Автоматизація фінансового обліку криптовалютних операцій із використанням чат-бота. 2025. URL:

https://essuir.sumdu.edu.ua/items/5a6929fa-a03b-4372-938b-0be3a249bf93 (дата звернення: 07.10.2025).

17. Остапов С. Е. та ін. Інформаційні системи електронної комерції. 2024. URL: https://repository.kpi.kharkov.ua/items/0eeb7678-89fb-487e-9563-3a73cd36f627 (дата звернення: 07.10.2025).

18. Abd-Elwahab A. M. MicroServices-driven enterprise architecture model for infrastructure optimisation. Financial Blockchain & Enterprise Systems Journal. 2023. URL: https://fbj.springeropen.com/articles/10.1186/s43093-023-00268-3 (дата звернення: 06.10.2025).

19. Bashtovyi A., Fechan A. Distributed transactions in microservice architecture: decision framework and trade-offs. Journal of Distributed Systems & Architecture. 2024. URL: https://science.lpnu.ua/sites/default/files/journal-paper/2024/aug/35695/maket2402951-453-463.pdf (дата звернення: 07.10.2025).

20. Bagam N. Implementing scalable data architecture for financial institutions // Stallion Journal for Multidisciplinary Associated Research Studies. 2023. Т. 2, № 3. С. 27–40. URL: https://www.researchgate.net/publication/387493189_Implementing_Scalable_Data_Architecture_for_Financial_Institutions (дата звернення: 07.10.2025).

21. Barua B., Kaiser M. S. Novel architecture for distributed travel data integration and service provision using microservices // Cureus Journal of Computer Science. 2024. Т. 2. URL: https://www.cureusjournals.com/articles/4835-novel-architecture-for-distributed-travel-data-integration-and-service-provision-using-microservices.pdf (дата звернення: 07.10.2025).

22. Sigala N. S. Microservices architecture in cloud computing: a software engineering perspective on design, deployment, and management // International Journal of Research and Innovation in Social Science (IJRISS). 2025. Т. 9, № 15. С. 215–239. DOI: 10.47772/IJRISS.2025.915EC0013 (дата звернення: 07.10.2025).

23. Bommareddy A. R. Designing microservices with use cases and UML. Graduate Theses and Dissertations. University of Dayton, 2023. URL: https://ecommons.udayton.edu/graduate_theses/7331/ (дата звернення: 07.10.2025).

24. Pattern: Microservice Architecture. [Електронний ресурс]. (б. д.). Сайт microservices.io. URL: https://microservices.io/patterns/microservices.html (дата звернення: 07.10.2025).

25. Паньків, Микола Петрович. Криптовалюта як новітній елемент світової валютної системи. Diss. 2023. https://dspace.wunu.edu.ua/bitstream/316497/51282/1/Паньків%20М.П.,%20ФФмі-21.pdf

26. Спільник, Ірина, and Олексій Ярощук. "Інституалізація криптовалюти: регулювання, правовий статус, облік і оподаткування." Інститут бухгалтерського обліку, контроль та аналіз в умовах глобалізації 2 (2020): 81-92. http://188.190.43.194:7980/jspui/handle/123456789/8546

27. Гладченко, Оксана, et al. "КРИПТОВАЛЮТА: ВІД ВІРТУАЛЬНИХ МОНЕТ ДО СТРАТЕГІЙ КОДУВАННЯ." Збірник наукових праць Державного податкового університету 1 (2024): 8-14. https://journals.dpu.kyiv.ua/index.php/collectioneconomy/article/view/453/436

28. John K., Li J., Liu R. Pricing and arbitrage across 80 cryptocurrency exchanges. 2024. URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4816710 (дата звернення: 09.10.2025).

29. Li X., Wang C. (Alex). The technology and economic determinants of cryptocurrency exchange rates: the case of Bitcoin. 2018. URL: https://www.researchgate.net/publication/311917640_The_technology_and_economic_determinants_of_cryptocurrency_exchange_rates_The_case_of_Bitcoin (дата звернення: 09.10.2025).

30. Ječmínek I. Volatility modelling and VaR: the case of Bitcoin, Ether and Ripple // Danube: Law, Economics and Social Issues Review. 2020. Т. 11, № 3. С. 253–269. URL: https://www.econstor.eu/bitstream/10419/242171/1/1738556069.pdf (дата звернення: 09.10.2025).

31. Malek J. Modeling dynamic VaR and CVaR of cryptocurrency markets with stochastic volatility and jumps // Journal of Financial Stability (Elsevier). 2023. URL:

https://www.sciencedirect.com/science/article/abs/pii/S1544612323001903 (дата звернення: 09.10.2025).

32. Bozzetto C. Cryptocurrency markets microstructure (with a machine learning application to the Binance bitcoin market). 2023. URL: https://unitesi.unive.it/retrieve/eed2f223-f3d3-459e-b4a6-25f233437bde/893488-1286715.pdf (дата звернення: 08.10.2025).

33. Bashtovyi A., Fechan A. Distributed transactions in microservice architecture: informed decision-making strategies. 2024. URL: https://science.lpnu.ua/sites/default/files/journal-paper/2024/aug/35695/maket2402951-453-463.pdf (дата звернення: 09.10.2025).

34. KX Blog. How speed beats slippage when managing crypto market volatility. 2025. URL: https://kx.com/blog/speed-beats-slippage-crypto-market-volatility (дата звернення: 07.10.2025).

35. Aydemir F. Building a performance efficient core banking system based on the microservices architecture // International Journal of Information Systems in the Service Sector. 2022. DOI: 10.1007/s10723-022-09624-z (дата звернення: 10.10.2025).

36. Bhatnagar S., Mahant R. Fortifying financial systems: exploring the intersection of microservices and banking security // IRJET. 2024. Т. 11, № 08. DOI: 10.13140/RG.2.2.13110.72001 (дата звернення: 10.10.2025).

37. Pleciński P., Bokla N., Klymkovych T., Melnyk M., Zabierowski W. Comparison of representative microservices technologies in terms of performance for use for projects based on sensor networks // Sensors (Basel). 2022. Vol. 22, No. 20. Article 7759. DOI: 10.3390/s22207759 (дата звернення: 10.10.2025).

38. Migration to microservices: a comparative study of decomposition strategies and analysis metrics. 2024.

39. A Comparative Assessment of JVM Frameworks to Develop Microservices. Applied Sciences (MDPI). 2023. Т. 13, № 3. Article 134. URL: https://www.mdpi.com/2076-3417/13/3/134 (дата звернення: 10.10.2025).

40. Development Frameworks for Microservice-based Applications: Evaluation and Comparison. arXiv. 2022. URL: https://arxiv.org/pdf/2203.07267 (дата звернення: 10.10.2025).

41. Berardi D., Giallorenzo S., Mauro J., Melis A., Montesi F., Prandini M. Microservice security: a systematic literature review // PeerJ Computer Science. 2022. Vol. 8. Article e779. DOI: 10.7717/peerj-cs.779 (дата звернення: 10.10.2025).

42. de Almeida M. G., Canedo E. D. Authentication and authorization in microservices architecture: a systematic literature review // Applied Sciences. 2022. Vol. 12, No. 6. Article 3023. URL: https://www.mdpi.com/2076-3417/12/6/3023 (дата звернення: 09.10.2025).

43. Perkasa M. J. P., Ramdan H. M., Maliki A. J., Somantri. Implementation of secure end-to-end encrypted chat application using Diffie–Hellman key exchange and AES-256 in a microservice architecture // Engineering Proceedings. 2025. Vol. 107, No. 1. Article 98. URL: https://www.mdpi.com/2673-4591/107/1/98 (дата звернення: 10.10.2025).

44. Runsewe O., Osundare O. S., Folorunsho S., Akwawa L. A. Optimizing user interface and user experience in financial applications: a review of techniques and technologies // World Journal of Advanced Research and Reviews. 2024. Т. 23, № 3. С. 934–942. DOI: 10.30574/wjarr.2024.23.3.2633 URL: https://www.researchgate.net/publication/384460308_Optimizing_user_interface_and_user_experience_in_financial_applications_A_review_of_techniques_and_technologies (дата звернення: 09.10.2025).

45. Захарія О., Кунанець Н., Жовнір Ю. Формування технічної документації IT-проєктів в контексті розроблення програмного забезпечення. 2025. URL: https://science.lpnu.ua/sites/default/files/journal-paper/2025/oct/40222/maket2512781chastina-262-279.pdf (дата звернення: 09.10.2025).

46. Loke C. X. L. Literature review: cryptocurrency vs fiat currency. *SSRN*. 2025. URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5181541 (дата звернення: 12.10.2025)

47. Ristanović V., Primorac D., Mulović Trgovac A. Banking in the age of blockchain and FinTech: a hybrid efficiency framework for emerging economies // Journal

of Risk and Financial Management. 2025. T. 18, № 8. Article 458. DOI: 10.3390/jrfm18080458. URL: https://www.mdpi.com/1911-8074/18/8/458 (дата звернення: 12.10.2025).

48. Dhandapani A. Microservices architecture in financial services: enabling real-time transaction processing and enhanced scalability // European Journal of Computer Science and Information Technology. 2025. T. 13, № 32. C. 145–159.

49. Ristanović V. Banking in the age of blockchain and FinTech: a hybrid perspective // Journal of Financial Regulation and Compliance. 2025. T. 18, № 8. C. 458. URL: https://www.mdpi.com/1911-8074/18/8/458 (дата звернення: 13.10.2025).

50. Al-Dmour A. та ін. Blockchain applications and commercial bank performance // Journal of Banking & Finance. 2024. URL: https://www.sciencedirect.com/science/article/pii/S2199853124000969 (дата звернення: 12.10.2025).

51. Yadlapalli R. Cloud payment systems and microservices architecture // Journal of Cloud Systems & Technology. 2025. URL: https://al-kindipublishers.org/index.php/jcsts/article/view/10450/9205 (дата звернення: 13.10.2025).

52. Riabchenko Y., Onyshchenko A., Kudin V., Kononets O., Holdskyi V. Digital assets and property rights: regulation and legal implications within the EU and globally // Statute Law Review. 2025. Vol. 46, Issue 3. DOI: 10.1093/slr/hmaf029. URL: https://academic.oup.com/slr/article-abstract/46/3/hmaf029/8253700 (дата звернення: 12.10.2025).

53. Di Casola P., Habib M. M., Tercero-Lucas D. Global and local drivers of Bitcoin trading vis-à-vis fiat currencies // ECB Working Paper Series. 2022. № 2868. URL: https://www.ecb.europa.eu/pub/pdf/scpwps/ecb.wp2868~0c2ad2e6e7.en.pdf (дата звернення: 13.10.2025).

54. ResearchAndMarkets. Crypto exchange market size, competitors & forecast to 2034. 2024. URL: https://www.researchandmarkets.com/report/cryptocurrency-exchange?srsltid=AfmBOopxYh-

r76_k09Eba5fd3eba0dBFelgitmGlTL4vcHXWW4GYWtAS (дата звернення: 13.10.2025).

55. Aasmr. A comprehensive analysis of FinTech in marketing research // Journal of Services & Marketing Science. 2023. Т. 13, № 6. С. 1–18. URL: https://www.aasmr.org/jsms/Vol13/No.6/Vol.13%20No.6.01.pdf (дата звернення: 14.10.2025).

56. Filosa C., Jovanovic M., Agostini L., Nosella A. Pivoting B2B platform business models: from platform experimentation to multi-platform integration to ecosystem envelopment // arXiv preprint arXiv:2412.19931. 2024. URL: https://www.b2bmarketingworld.com/guide/b2b-fintech-marketing/ (дата звернення: 14.10.2025).

57. Mohammad N. Secure DevOps practices for continuous integration and deployment in FinTech cloud environments // International Journal of DevOps (IJDO). 2024. Т. 1, № 1. С. 11–26. URL: https://www.researchgate.net/publication/380181369_Secure_DevOps_Practices_for_Continuous_Integration_and_Deployment_in_Fintech_Cloud_Environments (дата звернення: 14.10.2025).

58. KMS Solutions Asia. 10 best practices for FinTech businesses to leverage DevOps. 2024. URL: https://kms-solutions.asia/blogs/10-best-practices-for-fintech-businesses-to-leverage-devops (дата звернення: 14.10.2025).

ДОДАТКИ

Додаток А - Основні класи системи

```
package com.furious;
import com.furious.connector.service.PriceService;
import com.furious.connector.websocket.HercleWebsocketConnector;
import com.furious.model.Pair;
import io.micronaut.context.annotation.Context;
import jakarta.annotation.PostConstruct;
import jakarta.inject.Singleton;
import java.util.List;
import java.util.Map;
import java.util.Set;
import java.util.stream.Collectors;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.utils.ImmutableMap;

@Singleton
@Context
public class WebSocketApplicationRunner {

  private static final Logger log = LoggerFactory.getLogger(WebSocketApplicationRunner.class);
  private static final Map<String, String> HERCLE_PAIRS_TO_INTERNAL_PAIRS_MAP = ImmutableMap.<String,
String>builder()
      .put("BTCUSDT", "BTC/USDT")
      .put("BTCUSDC", "BTC/USDC")
      .put("BTCUSD", "BTC/USD")
      .put("BTCEUR", "BTC/EUR")
      .put("BNBGBP", "BNP/GBP")
      .put("ETHUSDT", "ETH/USDT")
      .put("ETHUSDC", "ETH/USDC")
      .put("ETHUSD", "ETH/USD")
      .put("ETHGBP", "ETH/GBP")
      .put("SOLUSDT", "SOL/USDT")
      .put("BTCGBP", "BTC/GBP")
      .put("ETHEUR", "ETH/EUR")
      .put("BNBUSD", "BNB/USD")
      .put("BNBUSDT", "BNB/USDT")
      .build();
```

```
  private final HercleWebsocketConnector hercleWebsocketClient;
  private final PriceService priceService;

  public WebSocketApplicationRunner(HercleWebsocketConnector hercleWebsocketClient,
      PriceService priceService) {
    this.hercleWebsocketClient = hercleWebsocketClient;
    this.priceService = priceService;
  }

  @PostConstruct
  public void start() {
    log.info("Trying to get pairs.");
    List<Pair> availablePairs = hercleWebsocketClient.getAvailablePairs();
    Set<String> pairNames = availablePairs.stream()
        .map(Pair::getName)
        .collect(Collectors.toSet());
    log.info("Available pairs: {}", pairNames);
    log.info("Predefined internal pairs' map: {}", HERCLE_PAIRS_TO_INTERNAL_PAIRS_MAP);

    pairNames.stream()
        .filter(HERCLE_PAIRS_TO_INTERNAL_PAIRS_MAP::containsKey)
        .forEach(this::subscribeToPair);
  }

  private void subscribeToPair(String pair) {
    log.info("Create subscription for pair {}", pair);
    String internalPair = HERCLE_PAIRS_TO_INTERNAL_PAIRS_MAP.get(pair);
    hercleWebsocketClient.subscribeToPairPrice(pair, response ->
        priceService.processPrice(internalPair, response));
  }
}
```

```
package com.furious.conroller;
import io.micronaut.http.annotation.Controller;
import io.micronaut.http.annotation.Get;

@Controller("/data-streamer")
public class DataStreamerController {

  @Get("/hb")
  public String heartbeat() {
    return "heartbeat";
  }
}
//
 package com.furious.service;

import com.furious.model.HasPair;
import com.furious.subscriptions.SubscriptionService;
import io.micronaut.websocket.WebSocketSession;
import java.util.List;

public abstract class UpdateBroadcaster {
  protected final SubscriptionService subscriptionService;

  protected UpdateBroadcaster(SubscriptionService subscriptionService) {
    this.subscriptionService = subscriptionService;
  }

  public <T extends HasPair> void send(String pair, T body) {
    List<WebSocketSession> sessions = subscriptionService.getSessionsForPair(
        pair);
    if (sessions != null && !sessions.isEmpty()) {
      sessions.forEach(session -> session.sendAsync(body));
    }
  }
}

@Singleton
@ClientWebSocket
public abstract class RipioTradeDataWebSocketClient extends CustomWebSocketClient {
  private static final Logger log = LoggerFactory.getLogger(RipioTradeDataWebSocketClient.class);
```

```
public static final String SHUTTING_DOWN_INFO_MESSAGE =
    "Do not opened new connection as app is shuting down";
private final Provider<RipioTradingPlatformWebsocketConnector> connector;
private final Provider<DataFetcher> orderDataFetcher;
private final Map<SubscriptionTopic, WebsocketMessageHandler> messageHandlerMap;
private final ObjectMapper objectMapper;

public RipioTradeDataWebSocketClient(
    Provider<RipioTradingPlatformWebsocketConnector> connector,
    Provider<DataFetcher> orderDataFetcher,
    Map<SubscriptionTopic, WebsocketMessageHandler> messageHandlerMap,
    ObjectMapper objectMapper, WebSocketClient webSocketClient
) {
  super(webSocketClient);
  this.connector = connector;
  this.orderDataFetcher = orderDataFetcher;
  this.messageHandlerMap = messageHandlerMap;
  this.objectMapper = objectMapper;
}

protected boolean isShuttingDown;

@EventListener
@Requires(notEnv = Environment.TEST)
void onShutdown(ShutdownEvent event) {
  log.info("Shutting down in AquanowTradingDataWebSocketClient");
  isShuttingDown = true;
}

@OnMessage
public void onMessage(String message) {
  try {
    CommonSubscriptionData commonSubscriptionData =
        objectMapper.readValue(message, CommonSubscriptionData.class);
    SubscriptionTopic subscriptionTopic =
        SubscriptionTopic.getByCommonSubscriptionData(commonSubscriptionData);
    WebsocketMessageHandler websocketMessageHandler = messageHandlerMap.get(subscriptionTopic);
    if (Objects.isNull(websocketMessageHandler)) {
      log.error("Failed to find websocketMessageHandler for message {}", message);
      return;
```

```
    }
    websocketMessageHandler.handle(message);
   } catch (Exception e) {
    log.error("Failed to process websocket message {}", message, e);
   }
  }

  @OnClose
  public void onClose() {
   log.info("Connection closed");
   if (!isShuttingDown) {
    connector.get().openNewSession();
    orderDataFetcher.get().fetch();
   } else {
    log.info(SHUTTING_DOWN_INFO_MESSAGE) }
  }

  @Scheduled(fixedDelay = "20s")
  void checkConnection() {
   if (!connector.get().isSessionExist()) {
    return; }
   log.info("Checking websocket connection");
   if (connector.get().isConnectionOpened()) {
    log.info("Connection opened");
   } else {
    log.error("Connection is closed. Procession onClose method");
    onClose();
   }
  }
}
public class WebSocketConnectionService {

  private static final Logger log = LoggerFactory.getLogger(WebSocketConnectionService.class);
  public static final String CONNECTION_RETRY_MESSAGE = "Retrying to connect";
  public static final String RETRIES_EXHAUSTED = "Retries exhausted";
  private final ExecutorService executorService = Executors.newSingleThreadExecutor();

  private final String serverUrl;
  private final int webSocketConnectionRetryNumber;
  private final int webSocketConnectionRetryInterval;
  private final CustomWebSocketClient webSocketClientClass;
```

```
protected CustomWebSocketClient clientMono;

public WebSocketConnectionService(
    CustomWebSocketClient webSocketClientClass,
    String serverUrl,
    Integer webSocketConnectionRetryNumber,
    Integer webSocketConnectionRetryInterval
) {
  this.webSocketClientClass = webSocketClientClass;
  this.serverUrl = serverUrl;
  this.webSocketConnectionRetryInterval = webSocketConnectionRetryInterval;
  this.webSocketConnectionRetryNumber = webSocketConnectionRetryNumber;
}
public void createWebSocketClient() {
  log.info("Creating new webSocketClient for {}", webSocketClientClass.getClass());
  URI uri = URI.create(serverUrl);
  clientMono = block(
      Mono.from(webSocketClientClass.getWebSocketClient().connect(
            webSocketClientClass.getClass(), uri))
        .retryWhen(Retry.fixedDelay(webSocketConnectionRetryNumber,
              Duration.ofSeconds(webSocketConnectionRetryInterval))
            .doBeforeRetry(i -> log.info("Retrying to connect {}", i))
            .onRetryExhaustedThrow((retryBackoffSpec, retrySignal) ->
              new RuntimeException("Retries exhausted", retrySignal.failure())))
        .doOnError(error -> log.info("Connection failed: {}", error.getMessage()))
        .doOnSuccess(client -> log.info("WebSocket client connected successfully"))
  );
}

public void openNewSession() {
  log.info("Opening new session");
  createWebSocketClient();
}

public void sendMessage(String message) {
  if (clientMono == null || !clientMono.getSession().isOpen()) {
    createWebSocketClient();
  }
  clientMono.send(message);
}
```

```
  private <T> T block(Mono<T> mono) {
    try {
      return executorService.submit(() -> mono.blockOptional().orElse(null)).get();
    } catch (InterruptedException | ExecutionException e) {
      log.error("Error in websocket executorService block ", e);
      throw new RuntimeException(e);
    }
  }
}

@Singleton
public class DataFetcherImpl implements DataFetcher {

  private static final Logger log = LoggerFactory.getLogger(DataFetcherImpl.class);
  private final RipioTradingPlatformWebsocketConnector ripioTradingPlatformWebsocketConnector;
  private final RipioTradingPlatformHttpConnector ripioTradingPlatformHttpConnector;

  private final OrderService orderService;
  private final OrderProcessorServiceImpl orderProcessorService;
  protected boolean isSubscribedForRipio;
  protected List<String> pairs;

  public DataFetcherImpl(
      RipioTradingPlatformWebsocketConnector ripioTradingPlatformWebsocketConnector,
      RipioTradingPlatformHttpConnector ripioTradingPlatformHttpConnector,
      OrderService orderService,
      OrderProcessorServiceImpl orderProcessorService
  ) {
    this.ripioTradingPlatformWebsocketConnector = ripioTradingPlatformWebsocketConnector;
    this.ripioTradingPlatformHttpConnector = ripioTradingPlatformHttpConnector;
    this.orderService = orderService;
    this.orderProcessorService = orderProcessorService;
  }

  @EventListener
  @Requires(notEnv = Environment.TEST)
  void onStartup(StartupEvent event) {
    pairs = getPairsForSubscription();
    fetch();
  }
```

```
@EventListener
@Requires(notEnv = Environment.TEST)
void onShutdown(ShutdownEvent event) {
  log.info("Stoping order data fetcher");
  if (isSubscribedForRipio) {
    log.info("Unsubscribed from ripio order data");
    unsubscribeFromRipioTradingData();
  }
}

@Override
public void fetch() {
  log.info("Starting order data fetcher");
  orderService.deleteAllOrdersByDataSource(DataSource.RIPIO, pairs);
  subscribeForRipioTradingData();
}
private void subscribeForRipioTradingData() {
  ripioTradingPlatformWebsocketConnector.subscribeForTradeData(pairs);
  ripioTradingPlatformWebsocketConnector.subscribeForOrderData(pairs);
  isSubscribedForRipio = true;
}

private void unsubscribeFromRipioTradingData() {
  if (ripioTradingPlatformWebsocketConnector.isConnectionOpened()) {
    ripioTradingPlatformWebsocketConnector.unsubscribeFromTradeData(pairs);
    ripioTradingPlatformWebsocketConnector.unsubscribeFromOrderData(pairs);
  } else {
    log.warn("Failed to unsubscribe, websocket connection is closed");
  }
}

private List<String> getPairsForSubscription() {
  log.info("Defining pairs for subscription");
  Set<String> pairsFromRedis =
      orderProcessorService.getPairsByDataSource(DataSource.RIPIO).stream()
          .map(pair -> pair.replace(UNIFIED_PAIR_SEPARATOR, RIPIO_PAIR_SEPARATOR))
          .collect(Collectors.toSet());
  List<String> allEnabledPairs = ripioTradingPlatformHttpConnector.getAllEnabledPairs();
  Set<String> ripioPairs = new HashSet<>();
  for (String enabledPair : allEnabledPairs) {
    String[] pairCurrencies = enabledPair.split(RIPIO_PAIR_SEPARATOR);
```

```
            ripioPairs.add(enabledPair);
            ripioPairs.add(pairCurrencies[1] + RIPIO_PAIR_SEPARATOR + pairCurrencies[0]);
          }
          List<String> pairsForSubscription = new ArrayList<>();
          for (String redisPair : pairsFromRedis) {
            if (!ripioPairs.contains(redisPair)) {
              log.error("Pair {} does not supports in ripio", redisPair);
              continue;
            }
            pairsForSubscription.add(redisPair);
          }
          log.info("Found pairs for subscription: {}", pairsForSubscription);
          return pairsForSubscription;
        }
      }

      @Controller("/api/v1/screenshot")
      public class GrafanaScreenshotController {
          private static final Logger log = LoggerFactory.getLogger(GrafanaScreenshotController.class);
          private final ScreenshotService screenshotService;
          public GrafanaScreenshotController(ScreenshotService screenshotService) {
              this.screenshotService = screenshotService;
          }

          @Get(processes = MediaType.IMAGE_PNG)
          public Mono<MutableHttpResponse<byte[]>> getScreenshot(
                  @QueryValue String dashboardId,
                  @QueryValue String panelId,
                  @QueryValue String from,
                  @QueryValue String to) {
              log.info("[GrafanaScreenshotController#getScreenshot] - Grafana screenshot called with params: " +
                      "dashboardId: {}, panelId: {}, from: {}, to: {} ", dashboardId, panelId, from, to);
              return screenshotService.getDashboardScreenshot(dashboardId, panelId, Instant.parse(from), Instant.parse(to))
                      .map(bytes -> HttpResponse.ok(bytes)
                              .contentType(MediaType.IMAGE_PNG_TYPE))
                      .onErrorResume(ex -> {
                          log.error("[GrafanaScreenshotController#getScreenshot] - Error getting grafana screenshot: {}",
ex.getMessage(), ex);
                          return Mono.just(HttpResponse.status(HttpStatus.BAD_GATEWAY));
                      });
          }
```

```
}

@Controller("/api/v1/candles")
public class CandleQueryController {

   private static final Logger log = LoggerFactory.getLogger(CandleQueryController.class);
   private final CandleQueryService candleQueryService;
   public CandleQueryController(InfluxCandleQueryService candleQueryService) {
      this.candleQueryService = candleQueryService;
   }

   @Get
   public Flux<CandleResponseDto> getCandles(@QueryValue String pair,
                                             @QueryValue String from,
                                             @QueryValue String to,
                                             @QueryValue Optional<String> interval) {

      log.info("[CandleQueryController#getCandles] - Requesting candles for pair = {} and range from: {} to {}",
pair, from, to);
      log.info("Successfully passed hmac verification");
      Instant fromInstant = Instant.parse(from);
      Instant toInstant = Instant.parse(to);
      return candleQueryService.getCandles(pair, fromInstant, toInstant, interval);
   }
}

@Singleton
public class RedisReaderService {
   private static final Logger log = LoggerFactory.getLogger(RedisReaderService.class);
   private final String streamName;
   private final RedisReactiveCommands<String, String> redis;
   public RedisReaderService(RedisReactiveCommands<String, String> redis,
                             @Value("${redis.streamName}") String streamName) {
      this.redis = redis;
      this.streamName = streamName;
   }

   public Flux<Tick> readTickBetween(Instant from, Instant to) {
      String startId = from.toEpochMilli() + "-0";
      String endId = to.toEpochMilli() + "-999";
```

```
            log.info("[RedisReaderService#readTickBetween] - Reading record between from: {}, to: {}", from, to);
            return redis.xrange(streamName, Range.create(startId, endId))
                    .doOnNext(message -> {
                        String json = message.getBody().get("price");
                        String[] parts = message.getId().split("-");
                        long timestampMillis = Long.parseLong(parts[0]);
                        Instant instant = Instant.ofEpochMilli(timestampMillis);
                        log.info("[RedisReaderService#readTickBetween] - Fetched record from source: {} id {} json {}", streamName, instant, json);
                    }) .mapNotNull(msg -> {
                        Tick tick = RecordToTickMapper.of(msg);
                     log.debug("[RedisReaderService#readTickBetween] - Mapped tick: {}",tick);
                        return tick; })
                    .filter(tick -> {
                        boolean inRange = !tick.timestamp().isBefore(from) && !tick.timestamp().isAfter(to);
                        log.debug("[RedisReaderService#readTickBetween] - Filter time={} in range={}", tick.timestamp(), inRange);
                        return inRange;
                    })
                    .doOnComplete(() -> log.debug("[RedisReaderService#readTickBetween] - Read tick end for range: {}", from + "-" + to));
            }

            public Flux<Tick> readTicksForMinute(Instant timePointInMinute){
                ZonedDateTime zdt = timePointInMinute.atZone(ZoneOffset.UTC).withSecond(0).withNano(0);
                Instant to = zdt.toInstant();
                Instant from = to.minus(1, ChronoUnit.MINUTES);
                return readTickBetween(from, to);
            }
        }
```