Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра системного аналізу та інформаційних технологій

Бакалаврська кваліфікаційна робота на тему:

«АРБІТРАЖНИЙ БОТ НАЙПОПУЛЯРНІШИХ БІРЖ КРИПТОВАЛЮТИ»

Виконав: студент 4 курсу, групи 2ІСТ-21б спеціальності 126 – «Інформаційні системи та технології»
_____ Павло ГОЛОВІН
Керівник: к.т.н., доц. каф. САІТ
_____ Георгій ГОРЯЧЕВ
«06» 06 2025 р.
Рецензент: к.т.н., доц. каф. КН
_____ Олександр ШЕВЧУК
«08» 06 2025 р.

Допущено до захисту

Завідувач кафедри САІТ
_____ д.т.н., проф. Віталій МОКІН
«06» 06 2025 р.

Вінниця ВНТУ – 2025 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра системного аналізу та інформаційних технологій
Рівень вищої освіти – перший (бакалаврський)
Галузь знань – 12 Інформаційні технології
Спеціальність 126 – «Інформаційні системи та технології»
Освітньо-професійна програма – Прикладні інформаційні технології

ЗАТВЕРДЖУЮ
Завідувач кафедри САІТ
_____ д.т.н., проф. Віталій МОКІН
"18" 03 2025 року

ЗАВДАННЯ
НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Головіну Павлу Павловичу

1. Тема роботи: «Арбітражний бот найпопулярніших бірж криптовалюти»
керівник роботи: Горячев Г.В., к.т.н., доц. каф. САІТ
затверджені наказом ВНТУ від «20» 03 2025 року № 97
2. Термін подання студентом роботи 31.05.25
3. Вихідні дані до роботи:
1) Дані про ринкові ціни та ліквідність криптовалютних бірж;
2) Дані API криптовалютних бірж;
4. Теоретична частина:
1) Аналіз проблематики арбітражної торгівлі на криптовалютних біржах.
2) Вибір оптимальних інформаційних технологій.
3) Проєктування та реалізація бота для виявлення арбітражних можливостей.
4) Тестування ефективності інформаційної системи на ринкових даних.
5. Перелік ілюстрацій до роботи:
1) Блок-схема архітектури Telegram-бота;
2) Діаграма модулів системи;
3) Блок-схема алгоритму обробки різниць цін і сповіщень.
4) Use Case діаграма інформаційної системи.
5) IT-інфраструктура бота арбітражу криптовалют.
6) Приклади візуалізації аналізу даних та сповіщень.

6. Консультанти розділів роботи:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 28.03.25

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1	Аналіз предметної області	*01.02.2025*	*24.04.2025*	вик
2	Порівняльний аналіз проблематики	*24.04.2025*	*03.05.2025*	вик
3	Вибір оптимальних інформаційних технологій	*03.05.2025*	*10.05.2025*	вик
4	Розроблення інформаційної системи автоматизації роботи користувачів	*10.05.2025*	*28.05.2025*	вик
5	Оформлення матеріалів до захисту БКР	*28.05.2025*	*04.06.2025*	вик

Студент _____ Павло ГОЛОВІН

Керівник роботи _____ Георгій ГОРЯЧЕВ

АНОТАЦІЯ

Бакалаврська кваліфікаційна робота складається з 74 сторінок формату А4, на яких є 23 рисунка, 3 таблиці, список використаних джерел містить 33 найменування**.**

Метою роботи є розроблення інформаційної системи для збирання та аналізу даних для пошуку арбітражних можливостей у найпопулярніших біржах криптовалют.

Розроблено Telegram-бот із використанням мови програмування Python, що забезпечує збір ринкових даних з криптобірж, їх обробку, виявлення арбітражних можливостей з надсиланням сповіщень користувачеві та архівацію даних для подальшого аналізу. Джерела даних - API криптовалютних ринків.

Ключові слова: криптовалюта, арбітраж, Telegram-бот, Python, автоматизація.

ABSTRACT

The bachelor's qualification work consists of 74 pages of A4 format, which include 23 figures, 3 tables, and the list of sources used contains 33 names.

The method of work is to develop an information system for collecting and analyzing data to find arbitrage opportunities on the most popular cryptocurrency exchanges.

A Telegram bot has been developed using the Python programming language, which provides collection of market data from crypto exchanges, their processing, detection of arbitrage opportunities with sending notifications to the user and archiving of data for further analysis. Data sources - API of cryptocurrency markets.

Keywords: cryptocurrency, arbitrage, Telegram bot, Python, automation.

ЗМІСТ

ВСТУП

Актуальність теми: у сучасному світі цифрові активи відіграють важливу роль у світовій економіці. Зростання кількості криптовалютних бірж, токенів, платформ, мереж, блокчейнів створює безліч можливостей для стратегій у трейдингу, зокрема арбітражу — можливості отримання прибутку за рахунок різниці в цінах одного й того самого активу у різних торгових середовищах.

Одним із перспективних напрямів є створення автоматизованих систем, які в режимі реального часу відстежують такі можливості та повідомляють про них користувачів. Застосування месенджерів як платформ для надсилання сповіщень щодо арбітражних можливостей робить систему зручною, швидкою та доступною.

Таким чином, розробка бота для арбітражу криптовалют є актуальною як з прикладної (отримання прибутку, підвищення ефективності торгівлі), так і з науково-дослідної точки зору (вивчення алгоритмів автоматизованого трейдингу, API інтеграцій, аналізу ринку тощо).

Об'єкт дослідження: інформаційне забезпечення підтримки прийняття рішень у сфері арбітражної стратегії торгівлі криптовалютами.

Предмет дослідження: алгоритми видобутку та аналізу даних для виявлення можливостей для арбітражної стратегії на криптовалютних ринках.

Мета дослідження: Розроблення інформаційної системи для збирання та аналізу даних для пошуку арбітражних можливостей у найпопулярніших біржах криптовалют.

Для досягнення мети в роботі поставлено та вирішено такі завдання:

- аналіз ринку криптовалют та механізмів арбітражної торгівлі;
- дослідження API криптовалютних бірж для збирання ринкових даних;
- розробка алгоритмів виявлення арбітражних ситуацій;
- створення системи для моніторингу та сповіщення користувача про виявлені можливості щодо арбітражної стратегії;

- реалізація програмної архітектури та функціоналу бота;
- тестування ефективності розробленого рішення на реальних прикладах.

Практичне значення одержаних результатів: результати дослідження мають на меті підвищення ефективності прийняття рішень у сфері криптовалютної торгівлі. Розроблена система повинна автоматизувати найбільш трудомісткі процеси зі збирання, обробки та аналізу актуальних даних з різних криптовалютних бірж. Система забезпечує швидкий пошук арбітражних можливостей, оперативне інформування користувача та розрахункову аналітику ринкової ситуації. Отримані результати можуть бути використані трейдерами, аналітиками криптовалютної галузі, розробниками інформаційних систем у сфері фінансових технологій та розробниками власних автоматизованих торгових ботів.

1 ТЕОРЕТИЧНІ ЗАСАДИ КРИПТОВАЛЮТНОГО РИНКУ ТА АРБІТРАЖНОЇ ТОРГІВЛІ

1.1 Базові поняття та структура криптовалютного ринку

Криптовалюта – це цифровий актив, який використовує криптографію для захисту транзакцій і базується на децентралізованій технології блокчейн, що забезпечує прозорість і безпеку операцій [1-2]. Основними прикладами є Bitcoin, Ethereum та Tether, які домінують на ринку за ринковою капіталізацією.

Криптовалюти, подібно до фіатних грошей, виконують функцію засобу фінансового обміну, їх використання з часом почало охоплювати все більше аспектів сьогоденного життя. Цифрові валюти застосовуються також й у різноманітних сферах, від децентралізованих фінансів та ігрової індустрії до штучного інтелекту.

Блокчейн [3] - це новаторська технологія зберігання даних, яка працює як послідовний ланцюг цифрових блоків. Кожен блок містить зашифровану інформацію, а всі записи захищені від змін і зберігаються на великій кількості пристроїв одночасно, без єдиного центру керування. Наведено приклад блоку в мережі Bitcoin(рис. 1.1).

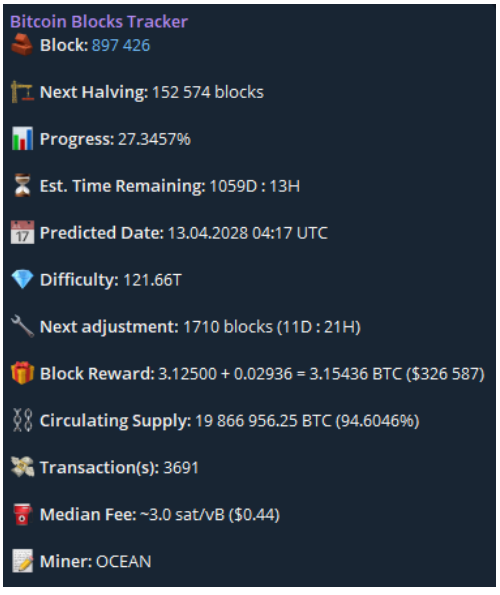

Рисунок 1.1 - Блок у мережі Bitcoin, статистика одного з актуальних блоків

Смартконтракт [4] – це цифрова угода, зображена як автоматизований код що збережений у блокчейні. Головна функція - забезпечити транзакції безпосередньо між 2 сторонами, не залучаючи 3 сторону. Наприклад, необхідно купити актив BTC за актив USDT. Смартконтракт виконуватиме роль посередника.

У блокчейні зберігається вся історія транзакцій та актуальні баланси кожного криптогаманця. Самі ж гаманці лише зберігають пари криптографічних ключів: публічний ключ (адресу) та приватний ключ, який забезпечує доступ до активів власнику гаманця. Завдяки приватному ключу користувач може підписувати транзакції підтвердження права власності на відповідні активи у мережі, тобто підтвердити право володіння гаманцем, передавати активи та виконувати інші операції, наприклад конвертація активів або приєднання до застосунків.

Головні переваги блокчейну:

- Прозорість та незмінність - усі транзакції записуються у прозорий реєстр, який доступний кожному учаснику мережі для перегляду. Після підтвердження транзакцій, внесені дані неможливо змінити або видалити.
- Захищеність - блокчейн використовує складні криптографічні алгоритми, які гарантують цілісність та незмінність інформації, дані зберігаються на величезній кількості пристроїв водночас, що посилює надійність блокчейну.
- Автоматизований - смартконтракти дозволяють виконувати операції автоматично, за визначеними умовами. Це знижує ризик людської помилки, прискорює взаємодію між сторонами шляхом вилучення 3 сторони.
- Децентралізована структура - система працює без єдиного центру управління. Дані зберігаються на величезній кількості вузлів по всьому світу, що робить мережу стійкою до контролю та збоїв у роботі одного або кількох учасників і спробах пошкодити роботу блокчейну з одного із вузлів.
- Надійність - якщо частина мережі виходить з ладу, інші вузли продовжують її підтримувати, що забезпечує безперебійну роботу.

- Відсутність кордонів - криптоактиви можливо надсилати та отримувати з будь-якої точки світу де це урегульовано законом.
- Обмежена кількість активів - більшість криптоактивів має передбачувану емісію, що допомагає запобігти інфляції з часом та напроти, збільшити цінність активу, наприклад у разі знецінення фіатних грошей.

Головні недоліки блокчейну:

- Правова невизначеність - у багатьох державах відсутні чіткі правила щодо використання блокчейну та цифрових валют, що може сприяти поширенню шахраїв та нелегальних оборотів коштів.
- Загроза з боку зловмисників - хоча система добре захищена, випадки успішного зламу чи шахрайства все ж трапляються. Слід зауважити, що технологія блокчейну вважається надійнішою ніж сучасні системи обліку фіатних грошей.
- Великі витрати електроенергії - при видобутку криптовалют, що вимагає спеціалізованого обладнання та систем охолодження.
- Проблеми з продуктивністю - зі зростанням кількості учасників мережа може працювати повільніше, особливо у моменти пікового навантаження.
- Технічна складність - для новачків користування блокчейном може викликати труднощі навіть для простого користування. Для розробників, розгортання блокчейн-рішень потребує спеціальних знань і часу, потрібен швидкісний інтернет, сучасні пристрої та технічні навички.

1.2 Класифікація криптовалютних бірж: централізовані та децентралізовані

Централізовані біржі (CEX) - це онлайн-платформи для купівлі, продажу та обміну криптовалют, якими керує одна компанія або організація. Вони виступають посередником між користувачами, зберігаючи кошти на власних гаманцях з підтвердженим резервом 1:1 та деяким надлишком для випадків втрати коштів.

Децентралізовані біржі (DEX) - це платформи для обміну криптовалют, які працюють без посередників або централізованого управління. Усі операції виконуються між користувачами (peer-to-peer) через підпис смартконтрактів з двох сторін, а кошти зберігаються на особистих гаманцях користувачів.

Виділено основні (табл. 1.1) [5] характеристики порівняння CEX і DEX. Ці параметри допоможуть зрозуміти принципи роботи кожного типу платформ, їх переваги та обмеження. Звертається увага на різницю в способах управління активами, безпеці, вимогах до верифікації користувачів, швидкості транзакцій та ліквідності. Розуміння цих відмінностей є важливим для вибору оптимальної платформи залежно від потреб і цілей користувача. У контексті арбітражної стратегії можуть використовуватись обидва типи платформ у форматі DEX - DEX, CEX - CEX, CEX - DEX.

Таблиця 1.1 Відмінності між централізованими та децентралізованими біржами.

Характеристика	Централізована біржа	Децентралізована біржа
Управління	Керується однією компанією	Працює без централізованого контролю
Зберігання коштів	Активи зберігаються на біржі	Кошти залишаються на гаманці користувача
KYC/AML	Обов'язкова процедура	Не потрібно
Безпека	Активи біржі розташовані по декількох гаманцях, при зломі одного, біржа відкупляє зі своїх коштів активи та повертає користувачам.	Вразлива до зломів та вірусних програм. При втраті контролю над гаманцем втрачаються всі кошти

Таблиця 1.1 Відмінності між централізованими та децентралізованими біржами.

Характеристика	Централізована біржа	Децентралізована біржа
Простота використання	Зручний та простий інтерфейс	Може бути значно складнішою для новачків
Швидкість транзакцій	Транзакції між однією біржою моментальні, між різними біржами займає часу стільки ж, як децентралізовані транзакції	Залежить від завантаження блокчейну (1-20 хв).
Ліквідність	Залежить від біржі, але в основному висока	Залежить від обсягу внесених активів у пули ліквідності. Зазвичай менша за CEX
Управління коштами	Користувач передає контроль біржі	Повний контроль у користувача
Підтримка користувачів	Наявна служба підтримки 24/7	Відсутня централізована підтримка
Комісії	Низькі	Відносно великі, залежать від мережі
Доступність	Може бути обмежена за регіоном проживання, віком.	Доступна без обмежень по регіонах
Відновлення доступу	Можливе через підтримку	Неможливе без приватних ключів
Підтримка торгових інструментів	Широкий вибір: спот, ф’ючерси, опціони, маржинальна торгівля	Обмежений вибір: в основному пули ліквідності чи AMM

Централізовані та децентралізовані біржі мають свої переваги та недоліки. CEX підходять користувачам, які цінують простоту, швидкість і технічну підтримку, тоді як DEX більше орієнтовані на тих, хто прагне максимальної безпеки та контролю над своїми коштами без посередників. Вибір платформи залежить від індивідуальних потреб користувача, рівня досвіду та пріоритетів щодо безпеки й свободи та обмежень у використанні своїх криптоактивів.

1.3 Арбітраж у криптовалютній торгівлі: види та принципи

1.3.1 Визначення та види арбітражної торгівлі

Арбітраж – це стратегія, що передбачає купівлю активу за нижчою ціною на одній біржі та продаж за вищою на іншій, використовуючи різницю цін, відому як спред [6-7]. Цей підхід є основою для багатьох торгових стратегій на фінансових ринках, включаючи криптовалютні

Ключові види арбітражу:

1. Чистий арбітраж - це стратегія, що базується на одночасному придбанні та продажі однакових фінансових інструментів на різних ринках для отримання прибутку з різниці цін. Такий підхід використовує неефективність ринку, коли однаковий актив тимчасово торгується за різними цінами на різних платформах.
2. Арбітраж злиття - це вид арбітражу пов’язаний з угодами злиття або поглинання компаній. Після оголошення угоди ринкова ціна акцій компанії-цілі часто відрізняється від пропонованої ціни купівлі. Зазвичай ця ціна нижча через ризики, що угода не відбудеться.

Арбітраж може статися за наступних умов:

– На різних ринках один і той самий актив продається за різними цінами.
– Активи зі схожим грошовим потоком торгуються за різними цінами.
– Актив має відому майбутню ціну в невеликому діапазоні часу.

1.3.2 Ключові фінансові показники для виявлення арбітражних можливостей

Різниця цін є основною передумовою та причиною виникнення арбітражних можливостей, без суттєвої різниці цін арбітраж не має економічного сенсу. Зазвичай різниця цін вимірюється в відсотках та рахується за формулою (1.1).

$$Spread\ \% = \left(\frac{P_{sell} - P_{buy}}{P_{sell}}\right) * 100\% \,, \qquad (1.1)$$

де:

P_{buy}— ціна активу на ринку, де його купують,

P_{sell} — ціна активу на ринку, де його продають

Всі комісії та витрати на виконання операції (комісії брокерів, плата за торгівлю, податки, комісія за переказ) мають бути меншими за різницю цін. Якщо комісії занадто великі, арбітраж стає збитковим та неактуальним. З врахуванням комісій формула різниці цін (1.2) виглядатиме так:

$$Spread\ \% = \left(\frac{P_{sell} - P_{buy}}{P_{sell}}\right) * 100\% - (C_{sell} + C_{buy}) \,, \qquad (1.2)$$

де:

P_{buy} - ціна активу на ринку, де його купують,

P_{sell} - ціна активу на ринку, де його продають.

C_{sell} - комісія за продаж.

C_{buy} - комісія за покупку.

Мережі (табл. 1.2) для переказу криптовалют між біржами також відіграють важливу роль в арбітражній торгівлі. Арбітражні операції повинні виконуватися швидко і майже одночасно, інакше різниця в ціні може зникнути через ринкові

коливання [8]. Висока швидкість транзакцій є обов'язковою частиною стратегії та покращує вірогідність на успішне завершення стратегії та саму економічну вигоду. При низькій швидкості транзакцій виду переказ, можливе невчасне постачання активу на ринок, у якому ціна активу вища за ціну покупки. Перед початком операції, необхідно перевірити зазначену приблизну швидкість переказу, зачислення на біржі - отримувач та відповідні комісії за отримання та переказ.

Таблиця 1.2 Основні мережі для переказу криптовалют між біржами

Мережа	Комісії за транзакцію	Швидкість переказу	Доступність на біржах
Ethereum (Mainnet)	$10–$50	1–5 хвилин	Binance, Kraken, Bybit, Coinbase. Bitget, Gate, OKX, Mexc
Binance Smart Chain (BSC)	$0.01–$0.50	3–5 хвилин	Binance, KuCoin, Bybit. Bitget, Gate, OKX, Mexc
Solana	$0.0001–$0.01	~1 хвилина	Binance, Bybit, FTX, Kraken, Bitget, Gate, OKX, Mexc
Polygon (Layer 2)	$0.01–$0.10	5–10 хвилин	Binance, KuCoin, Coinbase, Bitget, Gate, OKX, Mexc
Tron	$0.01–$0.50	3–10 хвилин	Binance, Huobi, KuCoin, Gate, OKX, Mexc

Актив повинен мати достатній рівень ліквідності на обох ринках, щоб була можливість швидко купити та продати необхідний обсяг без значного впливу на ціну. Низька ліквідність підвищує ризики і знижує ефективність. Для прикладу наведемо 2 активи на 2 біржах, найпопулярніший криптоактив Bitcoin на біржі Binance та майже нікому невідомий актив WCT на біржі GATE. При спробі придбати BTC на платформі

Binance на 10000$ ніяк не вплинемо на ціну активу, оскільки загальна ринкова капіталізація активу станом на сьогодні 2.2 Т. доларів, а купуючи WCT на ту ж саму суму, можемо вплинути на ціну активу, купивши його частинами по різним цінам драбинкою вверх, на декілька відсотків. Як результат арбітражна стратегія може стати неактуальною.

Книга ордерів (рис 1.2) - це електронний список ордерів на купівлю (bids) і продаж (asks) активу на біржі, упорядкований за ціною, саме за ним можливо визначити який вплив на ціну матиме купівля чи продаж активу.

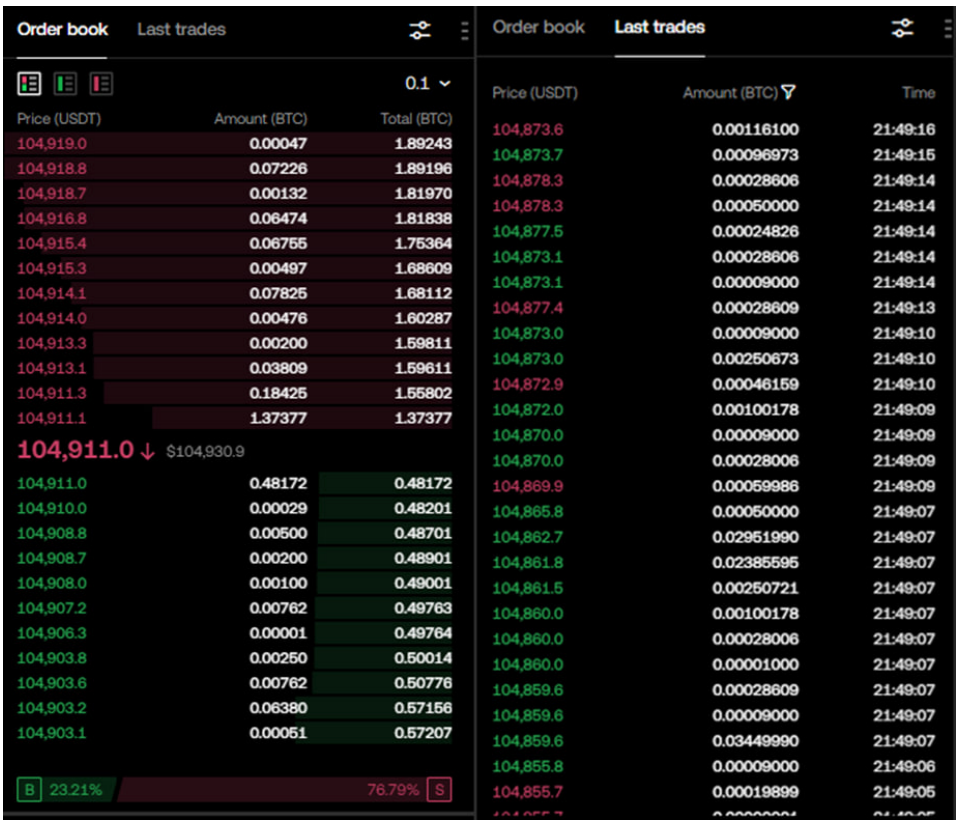

Рисунок 1.2 - Книга ордерів біржі OKX активу BTC [9]

Книга ордерів відображає які обсяги активу доступні для торгівлі та по яких цінах, також показує прозору історію виконаних ордерів в реальному часі платформи представника активу. Користувачі можуть розміщувати свої ордери за цікавими для

них цінами. Біржа також розміщує свої ордери для надання ліквідності користувачам та запобігання штучного впливу на ціну недобросовісними користувачами.

Арбітраж вважається стратегією з малим ризиком, але на відміну від теорії на практиці існують ризики:

- Волатильність ринків: якщо ціни на ринку мають різкі зміни, операції можуть не виконатись відносно одночасно.
- Технічні ризики: технічні збої, затримки, помилки біржі.
- Регуляторні ризики: обмеження на операції виводу, депозиту, ліміт на покупку або продаж активу.

Врахування цих факторів та розрахунок можливих ризиків є важливою частиною планування арбітражної стратегії.

Наведено наступні приклади арбітражної торгівлі:

1. Виявлено різницю в ціні Bitcoin (BTC) між двома централізованими біржами (CEX). На Binance BTC коштує $60000, на KuCoin BTC коштує $66000, комісія за ордер на Binance 0.1% від вартості ордера, на KuCoin 0.2. Вираховуємо різницю в ціні за формулою (1.3).

$$Spread\ \% = \left(\frac{P_{sell} - P_{buy}}{P_{sell}}\right) * 100\% - (C_{sell} + C_{buy}), \qquad (1.3)$$

де:

P_{buy} - ціна активу на ринку, де його купують,

P_{sell} - ціна активу на ринку, де його продають.

C_{sell} - комісія за продаж.

C_{buy} - комісія за покупку.

Обидві біржі мають високі обсяги торгів (перевірено через книгу ордерів), що дозволяє виконати ордер на 1 BTC без значного впливу на ціну. Переказ з біржі на біржу через мережу BSC займає приблизно 1 хвилину, а волатильність не перевищує

руху ціни більш ніж на 5% за хвилину. Як результат маємо чудову можливість для відпрацювання арбітражної стратегії.

2. Виявлено різницю в ціні Solana (SOL) між двома централізованими біржами (CEX). На Bybit BTC коштує $120, на Mexc SOL коштує $125, комісія за ордер на Bybit 0.3% від вартості ордера, на Mexc 0.2. Коливання цін за останню добу були у межах 2%. Вираховувавши різницю цін за тією ж самою формулою, отримаємо 3.5%.

Mexc має невелику ліквідність, що може мати значний вплив при покупці активу, також коливання цін може вплинути на економічну вигоду після завершення операцій. Переказ з біржі на біржу в мережі SOL триває приблизно 30 хвилин.

Отже, арбітражна стратегія є ризиковою через високу волатильність, значний час переказу активів та малу ліквідність на стороні одної з бірж. Найкращим рішенням буде почекати кращої можливості.

1.4 Деталізація завдань розробки інформаційної системи

Метою даної кваліфікаційної роботи є розробка інформаційної системи для арбітражу найпопулярніших бірж криптовалют, спрямованої на автоматизацію сповіщень щодо можливостей арбітражної торгівлі на криптовалютному ринку шляхом збирання, обробки та аналізу ринкових даних із різних бірж. Система має забезпечувати моніторинг цін активів, ліквідності, спредів, а також торгових і мережевих комісій на централізованих біржах.

Проведено аналіз інформації та відгуків щодо надійності платформ:

- Binance [10]: Найбільша CEX за обсягом торгів (~$7.35 трлн у 2024, 38% ринку). Підтримує 400+ активів (BTC, ETH, USDT). Висока ліквідність ($1 млрд/день для BTC/USDT). API з REST і WebSocket: для цін, стаканів, статусу виведення тощо. Смарт контракт виконуватиме роль посередника.

- Bybit [11]: Популярна CEX, фокус на деривативи, обсяг торгів $1.2 трлн у 2024. Підтримує 300+ пар. Висока ліквідність для BTC, ETH ($500 млн/день). Швидкий API для цін, стаканів, виводі та іншого.
- KuCoin [12]: Глобальна CEX із 200+ активами, обсяг торгів $700 млрд у 2024. Середня ліквідність ($100 млн/день для BTC/USDT). Середня по швидкості API інтеграція.
- OKX [13]: Велика CEX, обсяг торгів $1.5 трлн у 2024, 350+ активів. Висока ліквідність ($600 млн/день для BTC/USDT). Швидкий API.
- Gate [14]: CEX із широким вибором не ліквідних активів, обсяг торгів $500 млрд у 2024, 1000+ пар. Середня ліквідність ($50 млн/день для BTC/USDT). Повільний API.
- Mexc [15]: MEXC - CEX із фокусом на нові токени, обсяг торгів $400 млрд у 2024, 1500+ пар. Середня ліквідність ($30 млн/день для BTC/USDT). Присутній швидкий API з WebSocket.

Окрім збирання даних, невіднятною частиною роботи є автоматична аналітика щодо можливостей міжбіржевого арбітражу, із розрахунком потенційного прибутку та рекомендаціями щодо оптимальних мереж для переказів (Ethereum, BSC, Solana).

Зі зростанням криптовалютного ринку та конкуренцією між трейдерами потреба в автоматизованих інструментах для швидкого виявлення арбітражних можливостей стає дедалі актуальнішою.

Бот має надавати користувачам візуалізовано зображені дані у вигляді текстових повідомлень, таблиць або графіків (наприклад, порівняння спредів між біржами), що спрощують сприйняття інформації. Усі дані повинні фільтруватися за критеріями ліквідності, розміру спреду та витрат на перекази, щоб уникнути збиткових угод.

Розроблено план розробки Telegram-бота:

1. Аналіз і визначення вимог:

- Визначено функціональні вимоги: інтеграція з API криптовалютних бірж для збирання даних про ціни, глибину ринку (bid/ask), статус виведення коштів.
- Визначено нефункціональні вимоги: підтримка мультимереж, швидкодія, стійкість до помилок, масштабованість для подальших покращень.
- Встановлення критеріїв фільтрації арбітражних можливостей.

2. Розробка модулів збирання даних :
   - Отримання актуальних цін з API бірж.
   - Перетворення отриманих даних у єдиний уніфікований формат.
3. Обробка та аналіз даних:
   - Розрахунок ліквідності на основі глибини стакана.
   - Перевірка доступності мереж та визначення комісій на виведення і транзакції.
   - Порівняння цін між біржами з урахуванням комісій і доступних мереж.
4. Технічна реалізація:
   - Асинхронізація коду для забезпечення швидкодії та паралелізації запитів.
5. Тестування та експериментальні дослідження:
   - Проведення експериментів з метою перевірки точності розрахунків, швидкодії та ефективності бота у виявленні арбітражних можливостей.

1.5 Висновки

Розглянуто базові відомості про криптовалютні біржі: їх класифікацію, принципи роботи, мережі для переказів. Досліджено арбітражну торгівлю, її види, переваги та ризики. Проаналізовано інструменти аналітики та їх недоліки. Деталізовано вимоги до Telegram-бота і порядок його розробки, обрано API бірж (Binance, Bybit, KuCoin, OKX, Gate.io, MEXC) як джерела даних та ринки для роботи.

2 ВИБІР ТЕХНОЛОГІЧНИХ ІНСТРУМЕНТІВ

2.1 Огляд мов програмування, платформ та середовищ розробки

PyCharm [16] - це середовище розробки для роботи з різними мовами програмування та величезним набором інструментів, розроблене компанією JatBrains для Windows, Linux та maxOS.

PyCharm пропонує зручний інтерфейс з інструментами для підвищення продуктивності: вбудований дебагер для пошуку помилок, підтримка тестування коду, інтегрований термінал із функціоналом, аналогічним системним терміналам, можливості для роботи з даними що дозволяють керувати інформацією у середовищі.

Для реалізації Telegram-бота обрано мову програмування Python [17] завдяки її простоті, зрозумілому синтаксису та широкому набору бібліотек, які підтримують роботу з API бірж, асинхронне програмування та обробку даних. Налаштування інтерпретатора Python і вибір бібліотек (рис 2.1) здійснюється через налаштування проєкту в PyCharm, що забезпечує гнучкість і адаптивність розробки.

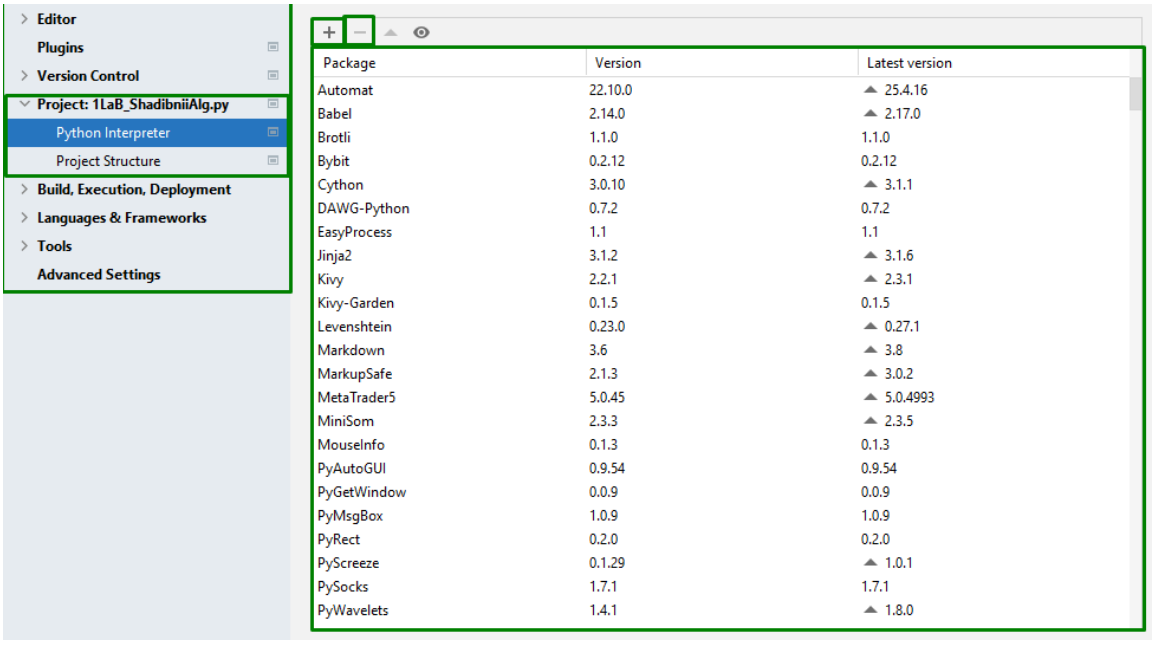

Рисунок 2.1 - Інструменти інтерпретатора PyCharm

Бібліотеки Python являють собою колекцію готових модулів, класів і функцій, які значно спрощують виконання різноманітних завдань програмування. Вони розширюють функціональні можливості мови, дозволяючи розробникам ефективно розв'язувати складні задачі без необхідності створення додаткового коду з нуля.

2.2 Обґрунтування вибору бібліотек, API криптобірж і сервісів даних

2.2.1 Вибір та обґрунтування бібліотек для розробки бота

Requests [18] - це популярна бібліотека Python для роботи з HTTP-запитами. Дозволяє легко надсилати GET, POST, PUT, DELETE та інші запити до вебсерверів. Неможливо представити отримання даних без запитів до REST API криптобірж для отримання актуальних даних, отже бібліотека є обов'язковою для створення системи.

Time [19] – бібліотека Python, яка використовується майже всюди для роботи з часом і датами. Вона надає функції для отримання поточного часу, вимірювання тривалості виконання коду, створення затримок у програмі та форматування часових даних.

Приклад використання бібліотеки time та requests для виміру швидкості запиту до API Binance (рис 2.2):

```
4   symbol= "BTCUSDT"
5   url = f'https://api.binance.com/api/v3/depth?symbol={symbol}&limit=100'
6   time_start= time.time()
7   response = requests.get(url) Запит до API біржі
8   time_end=time.time()
9   print(f"Time per request: {time_end-time_start}") Таймер
10  data = response.json()
11  time.sleep(10)
12  print(data)
```

Рисунок 2.2 - Приклад виміру швидкості запиту до API Binance

Результат : Time per request: 0.8062715530395508. Що свідчить про недостатню швидкість для арбітражної торгівлі, де затримки можуть призводити до втрати можливостей через волатильність ринку.

Hmac [20] - дозволяє генерувати хеш на основі ключа та повідомлення, використовуючи алгоритми хешування, такі як SHA-256 або SHA-512, що підтримуються бібліотекою hashlib. У тематиці кваліфікаційної роботи hmac застосовується для підпису запитів до API бірж, які вимагають авторизації. Це забезпечує захист від несанкціонованого доступу, оскільки підпис підтверджує, що запит надіслано від авторизованого користувача. Авторизація є обов'язковою для отримання даних щодо статусу роботи мереж, частково обов'язковою для інших даних таких як ліміти позицій, глибина стакан, ордери й так далі.

Asyncio [21] - це вбудований модуль Python, призначений для асинхронного програмування, що дозволяє виконувати операції паралельно без блокування основного потоку виконання. У контексті інформаційної системи для арбітражної торгівлі asyncio відіграє ключову роль, пришвидшення запитів до REST API.

Base64 [22] - це вбудований модуль Python, який використовується для кодування та декодування даних у форматі Base64. Може бути використана для обробки даних, що надсилаються до API бірж або месенджерів, таких як API-ключі.

Json - це вбудований модуль, який використовується для роботи з даними у форматі json. Майже всі відповіді від API бірж надходять у цьому форматі, тому вона являється необхідною.

Threading [23] - це вбудований модуль Python, який використовується для створення та керування потоками виконання в програмах. Він дозволяє виконувати кілька задач одночасно в межах одного процесу, що є корисною функцією для паралельної обробки. У контексті системи отримання даних бірж є важливим модулем для забезпечення швидкодії через розпаралелювання запитів до API. Наведено приклад розпаралелювання та його ефективності (рис. 2.3).

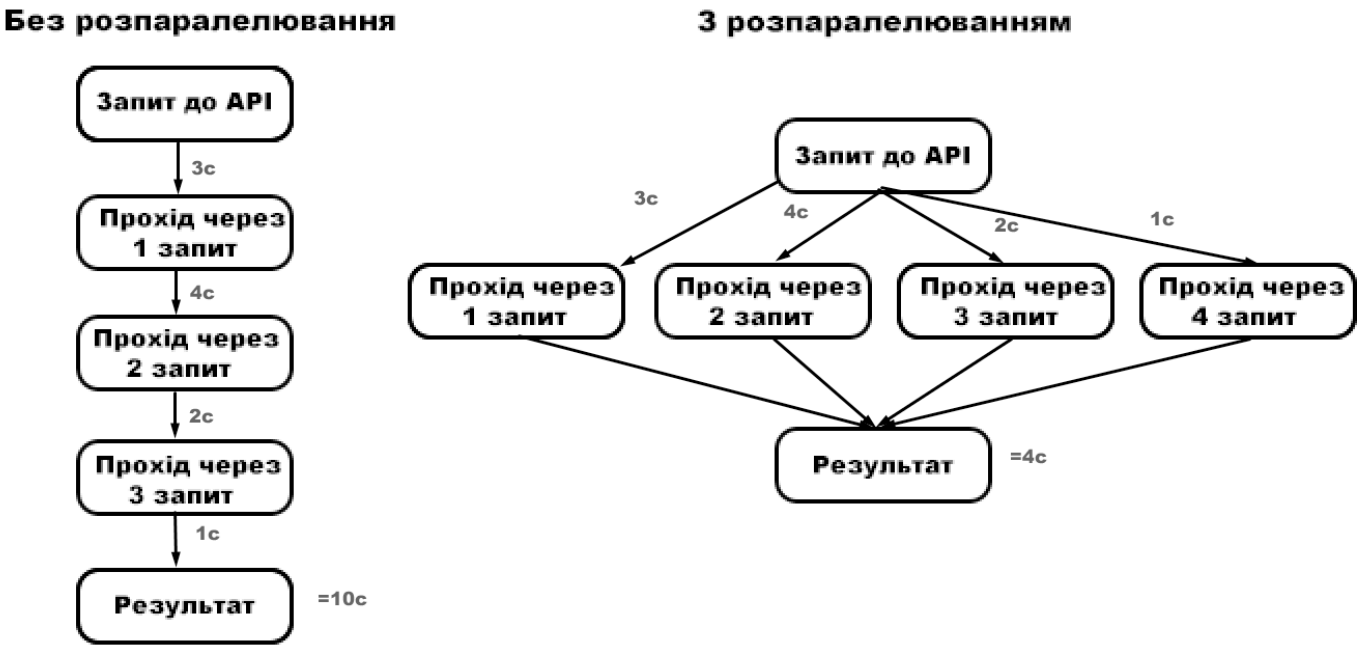

Рисунок 2.3 - Принцип розпаралелювання запитів API

Hashlib [24] - це вбудований модуль, який надає інструменти для створення криптографічних хеш-функцій, таких як MD5, SHA-1, SHA-256, SHA-512 та інших. Вона використовується для генерації хешів із даних, що забезпечує їх цілісність і безпеку, основною функцією у роботі є автентифікація запитів до API.

Aiohttp [25] - це популярна асинхронна бібліотека для роботи з HTTP-запитами, яка дозволяє виконувати запити до вебсерверів і обробляти відповіді у неблокуючому поток режимі. Вона чудово підходить для задач, що потребують високої швидкості та паралельної обробки.

2.2.2 Аналіз API криптовалютних бірж і сервісів даних

Перед виконанням практичної частини, проведено ручну оцінку API відібраних криптовалютних бірж (Binance, Bybit, KuCoin, OKX, Gate.io, MEXC), щоб забезпечити швидкий доступ до ринкової інформації, такої як ціни активів, глибина ринку, статус транзакцій і комісії мережі. Основна мета - визначити, які API найкраще відповідають потребам бота у швидкості, надійності та функціональності для аналізу арбітражних можливостей у реальному часі, визначити можливі недоліки або ж переваги.

Binance [26] виділяється як лідер завдяки широкому функціоналу API. Він надає доступ даних активів через REST-ендпоінт та WebSocket-канали. WebSocket забезпечує оновлення даних із затримкою менше 0.1 секунди, що ідеально для арбітражу, хоча REST-запити займають близько 0.7 секунди. Ліміт запитів — 1200 за хвилину, що дозволяє досить швидко оновлювати дані, але потребує асинхронного підходу для уникнення перевантаження.

Bybit [27] забезпечує стабільний доступ до даних через API з REST-ендпоінтами та WebSocket. Затримка WebSocket становить приблизно 0.15 секунди, що робить її ефективною для моніторингу цін і стаканів у реальному часі. REST-запити займають 0.6 секунди, а ліміт становить 600 запитів за хвилину, що теж дозволяє швидко оновлювати дані.

KuCoin [28] також пропонує API з REST-ендпоінтами та WebSocket. Затримка WebSocket — близько 0.3 секунди, що прийнятна, але REST-запити займають до 0.9 секунди, що уповільнює роботу системи. Ліміт запитів — 300 за хвилину, що в купі може створювати затримки при одночасному моніторингу всіх активів.

OKX [29] має швидке API з REST-ендпоінтом та WebSocket. Затримка WebSocket становить 0.2 секунди, а REST-запитів — 0.5 секунди. Ліміт 600 запитів за хвилину дозволяє ефективно працювати з всіма торговими парами. Для роботи системи показники більш ніж задовільнять потреби.

Gate.io [30] пропонує API з REST-ендпоінтом та WebSocket. Проте затримка REST-запитів досягає 1.8 секунди, а WebSocket — 0.5 секунди, що робить її менш конкурентною. Ліміт запитів — лише 200 за хвилину, що значно обмежує можливості моніторингу, але все ще задовольняє потреби системи. Платформа має низьку ліквідність, тому в подальшій розробці Telegram-бота можливе виключення або заміна.

MEXC [31] має швидке API з REST-ендпоінтами та WebSocket, із затримкою WebSocket 0.25 секунди та REST 0.4 секунди. Ліміт запитів — 400 за хвилину, що дозволяє обробляти достатню кількість даних, але низька ліквідність на платформі може ускладнити виконання великих угод.

Представлення таблиця з порівнянням швидкості API бірж (табл. 2.1):

Таблиця 2.1 Порівняння швидкості API бірж

Біржа	REST-ендпоінт	Затримка REST (сек)	WebSocket-ендпоінт	Затримка WebSocket (сек)
Binance	/api/v3/ticker/price	0.7	wss://stream.binance.com:9443/ws	0.01
Bybit	/v5/market/ticker	0.6	wss://stream.bybit.com/v5/public	0.015
KuCoin	/api/v1/market/orderbook	0.9	wss://push.kucoin.com/spot	0.03
OKX	/api/v5/market/ticker	0.5	wss://ws.okx.com:8443/ws/v5	0.02

Таблиця 2.1 Порівняння швидкості API бірж

Біржа	REST-ендпоінт	Затримка REST (сек)	WebSocket-ендпоінт	Затримка WebSocket (сек)
Gate.io	/api/v4/spot/tickers	1.2	wss://api.gateio.ws/ws/v4	0.05
MEXC	/api/v3/ticker/price	0.4	wss://ws.mexc.com/ws	0.025

Також, проведено повне дослідження (табл. 2.2) майже усіх криптовалютних бірж щодо доступності вебсокетів та REST API, багато з них були відсіяні через негативні відгуки або інші причини. Для подальшої роботи обрані 7 найкращих платформ.

Таблиця 2.2 Порівняння доступності REST та WebSocket API

Exchange	Websocket	REST API Official	Docs
Binance	Yes	Yes	https://developers.binance.com/docs/
Bybit	Yes	Yes	https://bybit-exchange.github.io/docs/v5/spread/websocket/
Bitget	Yes	Yes	https://www.bitget.com/api-doc/contract/websocket/
OKX	Yes	Yes	https://www.okx.com/docs-v5/en/#order-book-trading-trade-ws-place-order
HTX	Yes	Yes	https://www.htx.com/en-us/opend/newApiPages
Gate	Yes	Yes	https://www.gate.io/docs/developers/

Таблиця 2.2 Порівняння доступності REST та WebSocket API

Exchange	Websocket	REST API Official	Docs
OrangeX	Yes	Yes	https://openapi-docs.orangex.com/
MEXC	No	Read-only	---
LBANK	No	Yes, after request	---
Bitmart	No	Yes, after request	https://developer-pro.bitmart.com/en/
Hashkey	Read-only	Read-only	---
BingX	No	Yes	https://bingx-api.github.io/docs/
WEEX	No	No	---
Kucoin	Read-only	Yes	https://www.kucoin.com/docs/rest/
Bitunix	Read-only	Yes	https://openapidoc.bitunix.com/doc/
Coinbase	Yes	Yes	https://docs.cdp.coinbase.com/
Toobit	Read-only	Yes	https://toobit-docs.github.io/apidocs/
CoinW	No	Yes, after request	https://github.com/Coinw-Api/websocket
BloFin	Read-only	Yes	https://docs.blofin.com/
CoinEx	Read-only	Yes	https://docs.coinex.com/api/v2/
XT	Read-only	Yes	https://doc.xt.com
Hyperliquid	Yes	Yes	https://hyperliquid.gitbook.io/hyperliquid-docs/for-developers/api/websocket/post-requests
Kcex	No	No	----

Таблиця 2.2 Порівняння доступності REST та WebSocket API

Exchange	Websocket	REST API Official	Docs
Deepcoin	Read-only	Yes	https://www.deepcoin.com/docs/
Hotcoin	Read-only	Yes	https://hotcoinex.github.io/en/
Whitebit	Read-only	Yes	https://docs.whitebit.com/
Kraken	Read-only	Yes	https://docs.kraken.com/api/docs
OurBit	No	No	---
Bitrue	No	Yes	https://github.com/Bitrue-exchange

Отже, для потреб інформаційної системи обрано 7 бірж REST та WebSocket API яких є оптимальними завдяки низьким REST та WebSocket-затримкам і достатнім лімітам запитів. Для оптимізації застосунку обрано REST-запити, тому що вони забезпечують простоту реалізації, легкість налагодження та достатню швидкість для задач, які не потребують оновлення даних у реальному часі. Наприклад, для періодичного збирання цін (кожні 10–15 секунд) або перевірки статусу виводів і депозитів REST-запити з затримками 0.4–0.7 секунди є достатніми. WebSocket, попри швидкість, ускладнює логіку коду через необхідність управління постійним з’єднанням і обробки потоку даних, що менш критично для періодичних задач. Для прикладу автоматизації наведено результати дослідження над іншим проєктом. Час отримання цін та відправки ордера для спотової купівлі на GATE можливо скоротити до 0.04 секунд через постійне підтримування відкритих WebSockets, в той час, REST API має затримку біля секунди на виконання ордера та ще секунду на оновлення цін. Тобто, при подальшому переході на автоматизовану систему без участі людини, перехід на WebSocket буде критичним, оскільки він забезпечить миттєве реагування на арбітражні ситуації та автоматизацію всіх процесів.

2.3 Вибір платформи для реалізації системи

Для реалізації сповіщень від інформаційної системи арбітражної торгівлі обрано Telegram як основну платформу завдяки його популярності серед криптовалютних користувачів, простоті інтерфейсу і швидкості сповіщень.

Проведено дослідження медіаплатформ аналогів для реалізації сповіщень від інформаційної системи (табл. 2.3).

Таблиця 2.3 Порівняння аналогів Telegram, переваги та недоліки.

Платформа	Переваги	Недоліки
Telegram	Просте API, швидкі сповіщення, кросплатформність, популярність серед трейдерів	Обмеження на масові розсилки (30 msg/s), потреба в управлінні чатами
Discord	Потужне API, підтримка складних ботів, активна криптоспільнота	Складніший інтерфейс для залучення нової аудиторії
Slack	Інтеграція з робочими інструментами, стабільне API	Менш популярний серед користувачів, платний контент
WhatsApp	Простота використання	Обмежене API, затримки сповіщень, мала цільова аудиторія

Telegram обрано через швидкість, простоту інтеграції та популярність у криптоспільноті, що переважає над розглянутими альтернативами. Наявна можливість створення ботів замість використання реальних акаунтів користувачів, що могло б привести до блокування акаунту через спам чи неправомірне користування акаунтом. Рішення Telegram як основного засобу повідомлення користувачів забезпечує боту ефективне доставлення сповіщень про різницю цін і зручність для користувачів.

2.4 Висновки

Проведено аналіз вимог до Telegram-бота, визначено технологічні інструменти та обґрунтовано їх вибір. Для розробки обрано Python і обґрунтовано список допоміжних бібліотек. Переглянута та оцінена ефективність всіх API до всіх бірж, обґрунтований вибір Telegram як платформи для виводу корисної інформації.

3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ TELEGRAM-БОТА

3.1 Загальна архітектура застосунку

Розроблена система складається з модулів, які взаємодіють між собою (рис. 3.1): модуль збирання даних із бірж, модуль аналізу, модуль перевірки доступності мереж і модуль сповіщень через Telegram. Кожен модуль працює незалежно, обмінюючись даними через центральний компонент, основний цикл asyncio. Кожен модуль додатково поділений на частини щодо кожної криптовалютної платформи.

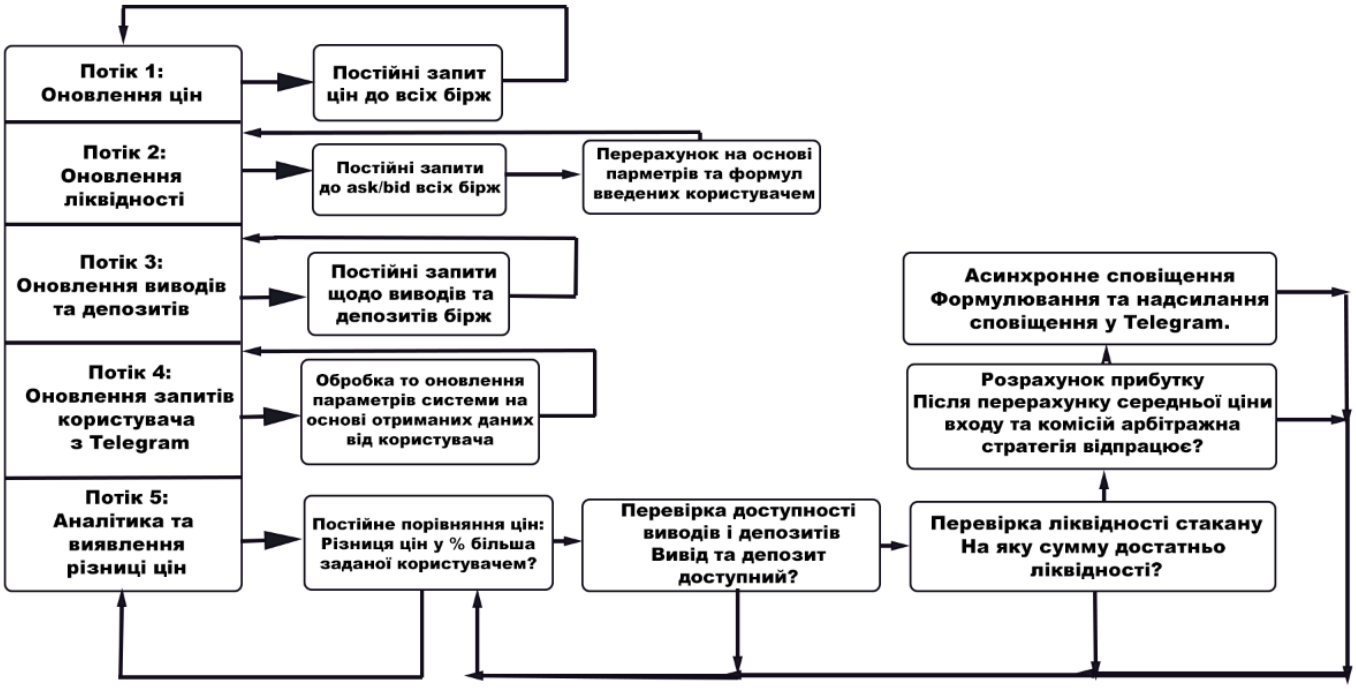

Рисунок 3.1 - Архітектура інформаційної системи

Надана блок схема ілюструє загальну архітектуру роботи Telegram-бота для виявлення арбітражних можливостей між криптовалютними біржами. Система поділена на п’ять основних потоків, кожен з яких виконує окрему функцію:

- Оновлення цін: Постійно надсилаються REST API запити до всіх бірж для отримання актуальних цін на активи, дані зберігаються в глобальних змінних щодо кожної платформи.
- Оновлення ліквідності: Здійснюються запити до стаканів лімітних ордерів (bid/ask), що необхідно для оцінки глибини ринку та можливості виконання угоди без достатнього проковзування, щоб анулювати економічну вигоду арбітражної стратегії.
- Оновлення виводів та депозитів: Контролюється доступність переказів між біржами, що є критичним для завершення арбітражної операції. Кожні 10 хвилин та перед відправкою сповіщення, щоб не загружати мережу.
- Обробка запитів з Telegram: Обробляються параметри, введені користувачем, включаючи бажаний мінімальний спред, доступні мережі, комісії тощо.
- Аналітика та виявлення різниці цін: Відбувається постійне порівняння цін між біржами в окремому потоці. Якщо знайдено спред, який перевищує заданий користувачем поріг, система перевіряє доступність депозиту/виводу, ліквідність стакана та розраховує можливий прибуток з врахування всіх витрат.

У разі підтвердження всіх умов, бот надсилає асинхронне повідомлення користувачу в Telegram із деталями можливої арбітражної угоди. Такий підхід дозволяє автоматизовано, швидко та ефективно виявляти економічно вигідні можливості арбітражу між біржами та вчасно надіслати сповіщення.

3.2 Реалізація логіки збору, обробки та аналітики ринкових даних

3.2.1 Збір даних щодо біржових активів

Поетапно розглянуто на прикладах функцій як саме збираються дані. Для кожної біржі реалізовано окремі функції, що здійснюють запити до відкритих REST

API. Дані збираються асинхронно (через aiohttp), що дозволяє зменшити затримку в обробці великої кількості запитів.

Наведено приклад функції збирання глибини ринку Binance (рис. 3.2) на основі даних щодо поточних ордерів користувачів і самої біржі:

```
96  # Order_books
97  def buy_stakan_binance(symbol):
98      try:
99          global stakan_value
100         url = f'https://api.binance.com/api/v3/depth?symbol={symbol}&limit=100'
101         response = requests.get(url)
102         data = response.json()
103         total_usdt = 0
104         total_quantity=0
105         price_mas=[]
106         quantity_mas=[]
107         for ask in data['asks']:
108             price, quantity = ask
109             if total_usdt<stakan_value:
110                 total_usdt+=float(price)*float(quantity)
111                 price_mas.append(price)
112                 quantity_mas.append(quantity)
113         for i in range(len(quantity_mas)):
114             total_quantity+=float(quantity_mas[i])
115         averageprice=total_usdt/total_quantity
116         mas=[]
117         mas.append(averageprice)
118         mas.append(total_usdt)
119         return mas
120
121     except Exception as e:
122         print(e)
```

Рисунок 3.2 - Збір глибини на ринку Binance

Функція *buy_stakan_binance(symbol)* здійснює запит до глибини ринку (order book) для заданого торгового активу. Вона розраховує середню ціну покупки з урахуванням заданого обсягу USDT (stakan_value) на стороні *ask*. Наприклад, нам

необхідно розрахувати вплив на ціну при покупці активу на $8000. Функція повертає середню ціну по якій придбаємо актив та максимально можливу суму. Аналогічна функція створена як для сторони *bid,* так і для кожної біржі з урахуванням специфічності API посилань та формату відповіді.

Для знаходження актуальних тікерів та інформації про них використовується ряд функцій *get_назва_біржі_data(session)*. Наведено приклад, отримання тікерів з OKX, однієї з таких функцій (рис. 3.3)

```
579 async def get_okx_data(session):
580     global okx_data
581     url = "https://www.okx.cab/api/v5/market/tickers?instType=SPOT"
582     async with session.get(url) as response:
583         if response.status == 200:
584             data = await response.json()
585             for entry in data['data']:
586                 symbol = entry['instId'].replace('-', '')
587                 try:
588                     okx_data[symbol] = {
589                         'last': float(entry['last']),
590                         'askPrice': float(entry['askPx']),
591                         'bidPrice': float(entry['bidPx'])
592                     }
593                 except Exception as e:
594                     print(e)
595                     continue
596         else:
597             print(f"get_okx_data: {response.status}")
598 
599         #print(okx_data)
```

Рисунок 3.3 - Функція отримання тікерів з платформи OKX

Дана функція має простіший характер, без додаткових розрахунків. Використовує бібліотеку aiohttp для отримання актуальних тікерів з OKX. Вона зберігає останню ціну, а також останні ціни купівлі та продажу (bidPrice, askPrice). Функція необхідна для знаходження Telegram - ботом різниці цін покупки та продажу між різними біржами та для формування повного списку тікерів усіх бірж.

Наступним прикладом наведено функцію get_kukoin_networks() (рис. 3.4), яка виконує запит до API біржі KuCoin з метою отримання списку доступних криптовалют разом з інформацією про підтримувані мережі виводу/депозиту.

```
686  #KUCOIN
687  async def get_kukoin_networks(session):
688      api_key = ''
689      api_secret = ''
690      api_passphrase = ''
691
692      url = 'https://api.kucoin.com/api/v3/currencies'
693
694      timestamp = str(int(time.time() * 1000))
695      method = 'GET'
696      data = f"{timestamp}{method}/api/v1/withdrawals/quotas"
697
698      signature = base64.b64encode(hmac.new(api_secret.encode('utf-8'), data.encode('utf-8'), hashlib.sha256).digest())
699
700      headers = {
701          'KC-API-KEY': api_key,
702          'KC-API-SIGN': signature,
703          'KC-API-TIMESTAMP': timestamp,
704          'KC-API-PASSPHRASE': api_passphrase,
705      }
706
707      response_kucoin_networks = await session.get(url, headers=headers)
708      data=response_kucoin_networks.json()
709      return data['data']
```

Рисунок 3.4 - Функція для отримання доступності мереж на платформи Kukoin

Для цієї функції використовується аутентифікований запит з підписом, сформованим на основі секретного ключа, мітки часу та HTTP-методу. Кожна біржа вимагає обов'язкову аутентифікацію за допомогою *api_key* і *api_secret.* Аутентифікація на кожній біржі має свої особливості шифрування та підпису для

підтвердження особи, тому важливо підтримувати свої API ключі у безпеці та оновлювати кожні пів року (залежить від платформи).

Функція update_prices() (рис. 3.5) відповідає за регулярне та паралельне отримання актуальних ринкових даних (цін) з кількох криптовалютних бірж. Вона запускається в нескінченному циклі з затримкою 1 секунда після кожного циклу, що дозволяє не перевищити ліміти запитів до API та підтримувати свіжість даних у швидкому режимі для подальшого аналізу арбітражних можливостей.

```
1163  async def update_prices():
1164      async with aiohttp.ClientSession() as session:
1165          while True:
1166              try:
1167                  print("PRICE UPDATE")
1168                  time_start=time.time()
1169                  tasks = [
1170                      get_binance_data(session),
1171                      get_bybit_data(session),
1172                      get_gate_data(session),
1173                      get_kukoin_data(session),
1174                      get_okx_data(session),
1175                      get_mexc_data(session)
1176                  ]
1177                  await asyncio.gather(*tasks)
1178                  time_end = time.time()
1179                  print("Time per cycle: ", time_end-time_start)
1180              except Exception as e:
1181                  print(f"{e}")
1182
1183              await asyncio.sleep(1)
```

Рисунок 3.5 - Функція асинхронного оновлення ринкових цін з бірж

У функції асинхронного оновлення ринкових цін створюється асинхронна HTTP-сесія, яка використовується всіма підфункціями, а з допомогою asyncio.gather() всі запити до API різних бірж виконуються одночасно. Для зручності й виявлення

неполадок додано таймер, який виводить час проходження повного круга. У випадку помилок (через втрату з'єднання, інші причини майже неможливі) дані продовжують оновлюватись при пере підключенню до інтернету. Аналогічні функції створені для асинхронного оновлення ліквідності стаканів, доступності мереж та інше.

3.2.2 Механізм обробки отриманих даних

Досить часто бажаний формат отриманих даних не збігатися з очікуваним, тому розроблено підфункції які час від часу викликаються в коді. Наведено приклад функції (рис. 3.6) для обробки посилання на актив у необхідній біржі, для швидкого доступу та економії часу при пошуку.

```
1122    def get_url_echange(symbol, exchange):
1123        try:
1124
1125            if exchange == 'BINANCE':
1126                symbol = symbol[:-4] + '_' + symbol[-4:]
1127                url=f"https://www.binance.com/en/trade/{symbol}?type=spot"
1128                return url
1129            elif exchange == 'BYBIT':
1130                symbol = symbol[:-4] + '/' + symbol[-4:]
1131                url = f"https://www.bybit.com/ru-RU/trade/spot/{symbol}"
1132                return url
1133            elif exchange == 'GATE':
1134                symbol = symbol[:-4] + '_' + symbol[-4:]
1135                url = f"https://www.gate.io/ru/trade/{symbol}"
1136                return url
1137            elif exchange == 'KUKOIN':
1138                symbol = symbol[:-4] + '-' + symbol[-4:]
1139                url = f"https://www.kucoin.com/ru/trade/{symbol}"
1140                return url
1141            elif exchange == 'OKX':
1142                symbol = symbol[:-4] + '-' + symbol[-4:]
1143                url = f"https://www.okx.com/ru/trade-spot/{symbol}"
1144                return url
1145            elif exchange == 'MEXC':
1146                symbol = symbol[:-4] + '_' + symbol[-4:]
1147                url = f"https://www.mexc.com/ru-RU/exchange/{symbol}?_from=search_spot_trade"
1148                return url
1149        except:
1150            return None
```

Рисунок 3.6 - Підфункція обробки посилання на актив

Наведено ще один приклад (рис. 3.7) функції обробки отриманих даних, проблема полягає в небажаному форматі тікерів зі сторони майданчика Kukoin, де замість відповіді в форматі BTCUSDT отримуємо BTC_DAI або щось подібне.

```
342  def format_symbol_okx_kukoin(symbol):
343      if symbol.endswith("USDT"):
344          symbol = symbol[:-4] + '-' + 'USDT'
345      elif symbol.endswith("GBP"):
346          symbol = symbol[:-3] + '-' + 'GBP'
347      elif symbol.endswith("BTC"):
348          symbol = symbol[:-3] + '-' + 'BTC'
349      elif symbol.endswith("ETH"):
350          symbol = symbol[:-3] + '-' + 'ETH'
351      elif symbol.endswith("TUSD"):
352          symbol = symbol[:-4] + '-' + 'TUSD'
353      elif symbol.endswith("USTC"):
354          symbol = symbol[:-4] + '-' + 'USTC'
355      elif symbol.endswith("USDT"):
356          symbol = symbol[:-4] + '-' + 'USDT'
357      return symbol
```

Рисунок 3.7 - Обробка тікерів біржі Kukoin

У відповідях API щодо доступності мереж можуть повертатися однакові мережі, але з різними назвами. Наприклад, на біржі MEXC мережа вказана як BNB Smart Chain, тоді як на Binance вона позначається як BEP20. Для обробки таких розбіжностей були реалізовані окремі допоміжні функції, які нормалізують назви мереж до єдиного формату. Таким чином, у системі використовується набір невеликих підфункцій, що забезпечують уніфікацію та коректну інтерпретацію отриманих даних.

3.2.3 Аналітика отриманих даних

На цьому етапі розробки система переходить до аналізу готової зібраної ринкової інформації з всіх ззалучених криптовалютних бірж. Основна мета полягає у виявленні арбітражних можливостей ситуацій, коли одна й та ж криптовалюта має різні ціни у книзі ордерів на різних платформах.

Для цього дані кожної біржі групуються за торговими парами. Система формує набір усіх унікальних символів або тікерів, як заведено їх називати у криптоспільноті. Якщо тікер є хоча б на одній з бірж, він додається в список та проводиться попарне порівняння між кожною парою бірж для кожного символу. Якщо виявляється, що ціна купівлі на одній біржі нижча за ціну продажу на іншій, розраховується різниця цін. У разі, якщо цей спред перевищує встановлене мінімальне значення, виконується оцінка актуальності арбітражної стратегії з урахуванням об’ємів, доступності мереж, швидкості мереж, максимальних об'ємів та інше. За умови досягнення потенційної вигоди, що перевищує нуль, формується повідомлення з усіма деталями угоди та надсилається через телеграм-бот або інший інформаційний канал.

Реалізація подібного підходу дозволяє системі кожну секунду оновлювати ціна та шукати вигідні точки входу в арбітражні стратегії, спираючись на дані з декількох джерел. Завдяки цьому можлива оперативна реакція на зміни ринку з мінімальною затримкою.

3.2.4 База даних для архівації ринкових даних

Отримані дані під час запитів до REST API платформ зберігання для подальшого аналізу ринкових даних. Розроблена спеціалізована база даних, призначена для архівації цінових показників активів. Розроблена база даних дозволяє накопичувати інформацію про ціни активів на різних біржах, інформацію про мережі та інше. База даних забезпечує можливість подальшого аналізу виникнувших

ситуацій, перегляду застарілих даних для аналізу ситуації на той час і тому подібне. Структура бази даних включає таблиці (рис. 3.8), які відображають як основні довідники, так і деталізовані дані про ціни та статуси мереж.

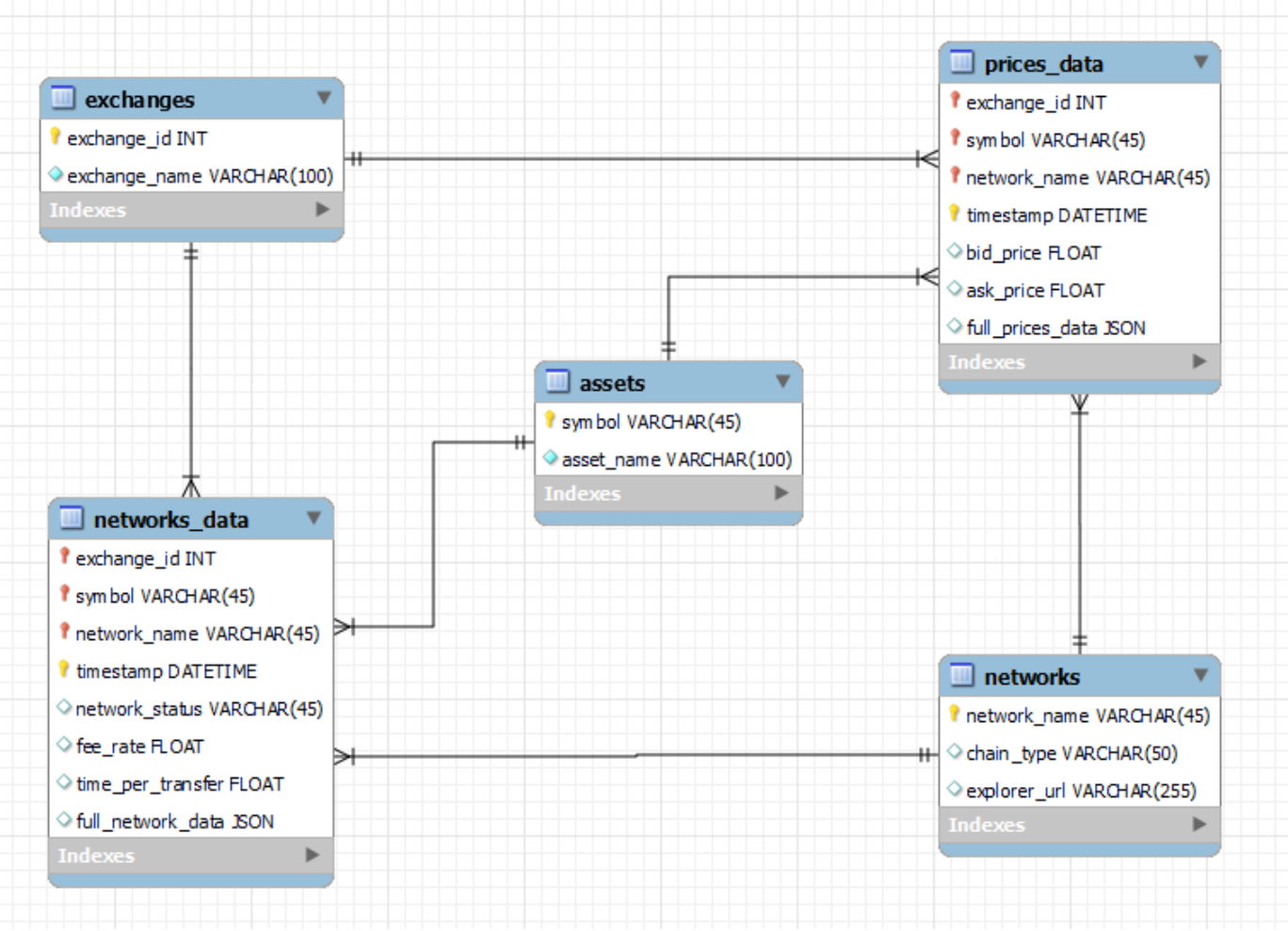

Рисунок 3.8 - База даних для архівації ринкових даних

Дані щоразу додаються до таблиці під час асинхронних запитів до REST API криптоплатформ. Таким чином, під час посекундного оновлення даних, створюємо масивну базу даних, яка не надається у бірж. Наприклад, нам необхідно проаналізувати волатильність цін на 2 біржах під час арбітражної можливості на секундному графіку. Виникає проблема, біржі зазвичай не представляють дані у секундному таймфреймі, але як раз для такого випадку можливо використати базу

даних, відмотати час трохи назад та візуально відобразити хід ціни у потрібний момент часу. Зображено у вигляді 10-секундного графіка (рис. 3.9) хід ціни активу CETUS на Binance.

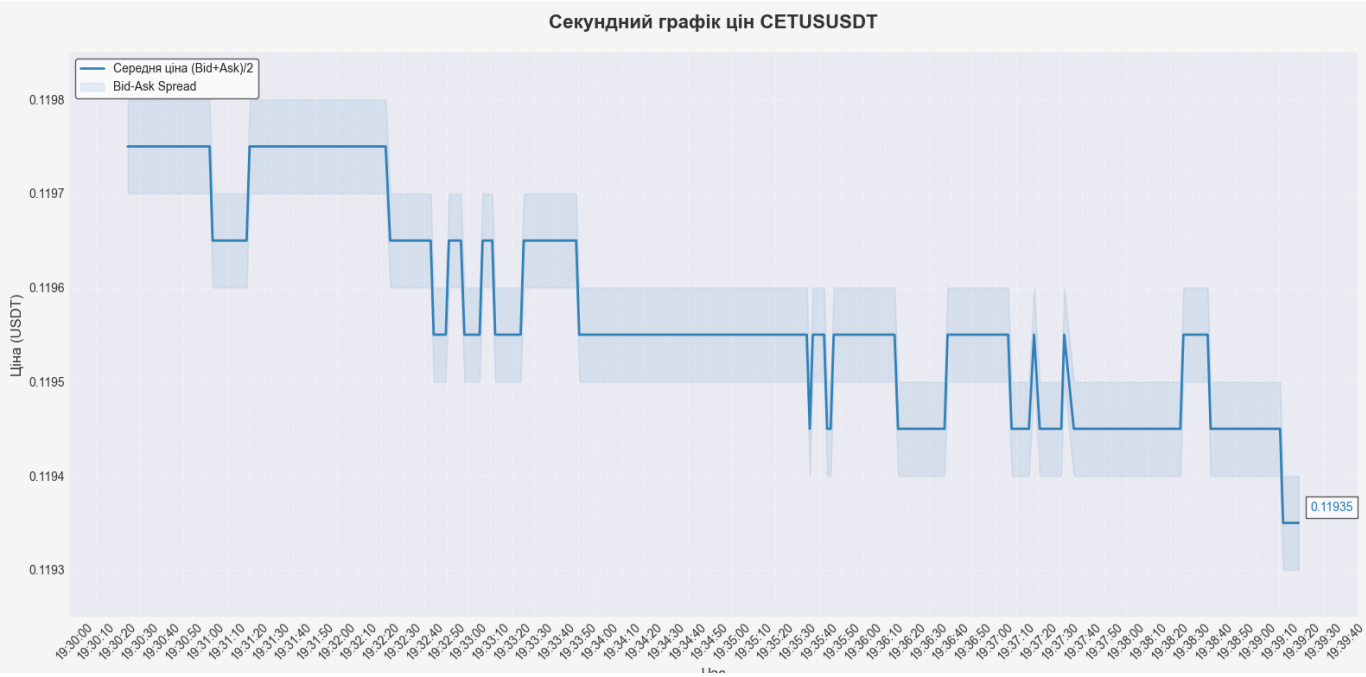

Рисунок 3.9 - 10-ти секундний графік ціни активу CETUS на Binance

Аналогічно й зі статусом мереж, подібна інформація дозволить зрозуміти їхню стабільність і продуктивність у реальному часі, що є критичним для оцінки ризиків під час арбітражних операцій. Зберігаючи дані про статус мереж разом із часом, комісіями та тривалістю трансферів, база даних забезпечує можливість аналізу залежності ринкових коливань від технічних факторів інформаційної системи. Це дає змогу виявляти потенційні перешкоди, такі як затримки в обробці транзакцій, і оптимізувати стратегію трейдингу на основі об’єктивних даних. Наприклад, не рідкість коли під час арбітражної стратегії неочікувано відключають вивід з біржі певного активу, для розв'язання проблеми необхідно зрозуміти на скільки швидко проходить цей процес.

3.3 Внутрішня логіка Telegram-бота: реалізація, сповіщення

У разі успішного виконання всіх умов для арбітражної можливості, генерується та надсилається повідомлення, яке включає:

- Назву торгового символу (тікер, наприклад BTC_USDT).
- Значення різниці цін у відсотках.
- Назви двох бірж, між якими виникла арбітражна операція.
- Ціну купівлі на першій біржі та ціну продажу на другій.
- Об'єми стаканів на кожній платформі.
- Середні ціни купівлі та продажу з урахуванням глибини ринку.
- Розрахований потенційний прибуток в еквіваленті долару США, криптоактиву USDT який приєднаний до його курсу.
- Статус доступності введення та виведення активу на обох біржах.

Наведено приклад генерації повідомлення у коді (рис. 3.10):

```
potential_profit=min(round(volume_a[1],2),round(volume_b[1],2)) * (
message = (
    f"🚀 #{symbol} Spread: {spread:.2f}%\n"  Різниця цін
    f"————————————\n"
    f"🏦 Exchange: {name_a}\n" Назва 1 біржі
    f"🟢 Buy: {ask_a:.6f}$\n" Остання ціна покупки
    f"                ⬇️⬆️     \n"
    f"🔴 Sell: {ask_b:.6f}$\n" Остання ціна продажу
    f"🏦 Exchange: {name_b}\n" Назва 2 біржі
    f"————————————\n" Середня ціна покупки
    f"Average price buy: {format_4(volume_a[0])}$\n"
    f"💰 Volume: {round(volume_a[1],2)}$\n" Можливий buy об'єм
    f"💎 Potential Profit: {round(potential_profit,2)}$\n" Прибуток
    f"💰 Volume: {round(volume_b[1],2)}$\n" Можливий sell об'єм
    f"Average price sell: {format_4(volume_b[0])}$\n"
    f"————————————\n" Середня ціна продажу
    f"🔄 Status:\n"
    f"      📥 Deposit: 🟢\n" Статус депозитів
    f"      📤 Withdrawal: 🟢" Статус виводів
)
if potential_profit > 0: Якщо прибуток не збитковий
    send_message(chat_id=-1002514744243,message=message,button_text
```

Рисунок 3.10 - Генерація повідомлення

Функція *send_message* (рис. 3.11) призначена для надсилання повідомлень до Telegram-каналу або чату через Telegram Bot API, з можливістю додавання інтерактивних кнопок для прискорення процесу переходу на біржу.

```python
def send_message(chat_id, message, button_text=None, button_url=None, button_text2=None, button_url2=None):
    try:
        ##############################################################################################
        bot_token = ""
        ##############################################################################################

        url = f"https://api.telegram.org/bot{bot_token}/sendMessage"
        params = {
            "chat_id": chat_id,
            "text": message,
        }

        inline_keyboard = []
        if button_text and button_url:
            inline_keyboard.append({"text": button_text, "url": button_url})
        if button_text2 and button_url2:
            inline_keyboard.append({"text": button_text2, "url": button_url2})


        if inline_keyboard:
            params["reply_markup"] = json.dumps({
                "inline_keyboard": [inline_keyboard]
            })

        response = requests.get(url, params=params, verify=False)

        if response.status_code == 200:
            return response.json().get("result", {}).get("message_id")
        else:
            print("Failed to send message:", response.text)
            return None

    except Exception as e:
        print(e)
```

Рисунок 3.11 - Функція для відправлення сповіщень Telegram

У функцію надходять наступні параметри:

- *chat_id* — ID чату або каналу, куди буде надіслане повідомлення.
- *message* — текст самого повідомлення.

- *button_text, button_url* — текст і посилання першої кнопки (якщо задано).
- *button_text2, button_url2* — текст і посилання другої кнопки (якщо задано).

У функції вказується токен бота, після чого формується HTTP-запит до офіційного REST API Telegram з параметрами, які включають ідентифікатор чату, текст повідомлення, за наявності кнопки з відповідними посиланнями до торгових платформ. Якщо обидві пари параметрів button_text та button_url задані, формується вбудована клавіатура з однією або двома кнопками. Надсилання запиту здійснюється через бібліотеку requests, а результат перевіряється за HTTP-статусом. У разі успішного запиту функція повертає message_id, який Telegram надає повідомленню.

3.4 Інфраструктура розгортання бота

Обрано IT-інфраструктуру (рис. 3.12), яка забезпечить стабільну, безпечну, масштабовану та безперебійну роботу системи.

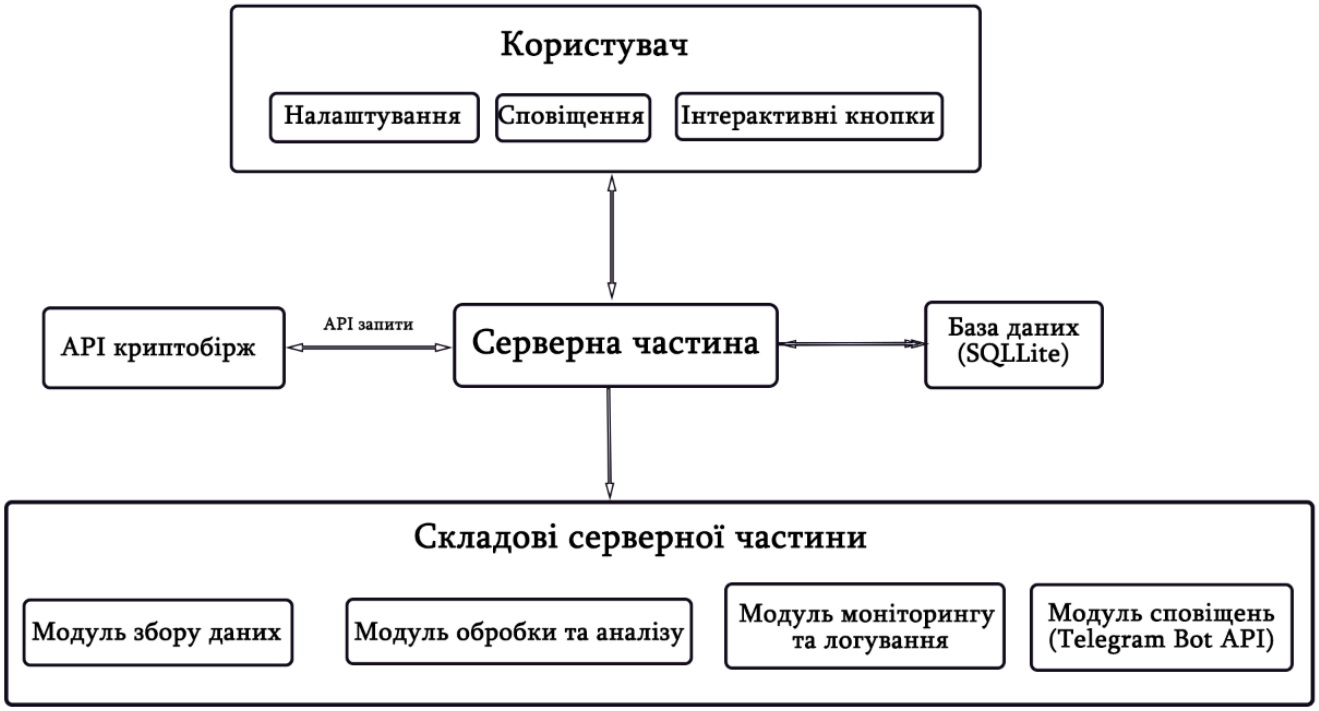

Рисунок 3.12 - IT-інфраструктура бота арбітражу криптовалют

Система базується на модульній архітектурі, кожен компонент якої виконує окрему функцію: збір даних з API, обробку та аналітику, перевірку мережевих статусів, логування, сповіщення тощо.

На віртуальному сервері виконуватимуться основні алгоритми бота. Наприклад, після успішного тестування роботи системи на локальних пристроях, доцільно орендувати віртуальний сервер на ОС Linux з характеристиками 2 CPU, 4GB RAM та SSD диск від 1000 ГБ. Великий обсяг пам'яті SSD диску пов'язаний зі збереженням логів і бази даних. Основні характеристики обчислювальної системи можуть мати не високі показники, оскільки бот не вимагає значних обчислювальних ресурсів. До модулів серверної частини належать: модуль збирання даних, обробки та аналізу, бази даних, сповіщень та управління.

Користувач взаємодіє з ботом через Telegram, що також робить систему абсолютно безпечною. Клієнт використовує інтерфейс месенджера для сповіщень, зміни параметрів пошуку різниці цін, виводу результатів аналітики і кнопок для швидкого переходу на біржі. Додаткові панелі не доцільні та будуть заважати.

Стійкість системи забезпечується асинхронною архітектурою, тобто, в разі збою одного компоненту, мінімалізується вплив на роботу всієї системи шляхом тимчасового виключення з алгоритму. Фонові задачі автоматично перезапускаються у разі винятків або інших неполадок, у разі розриву інтернет з'єднання з API платформ система повторює спробу перепідключення, забезпечуючи не тільки стабільність у роботі внутрішніх алгоритмів, а також безперебійну роботу з зовнішніми компонентами.

Зображена Use Case діаграма (рис. 3.13) для автоматизації роботи між користувачем та ботом для арбітражу криптовалют. Діаграма відображає основні сценарії використання. Актор "Користувач" взаємодіє з ботом через такі процеси, як налаштування параметрів, отримання аналітичних даних та доступ до сповіщень через авторизацію. Бот, у свою чергу, віддає дані щодо аналітики та надсилає сповіщення.

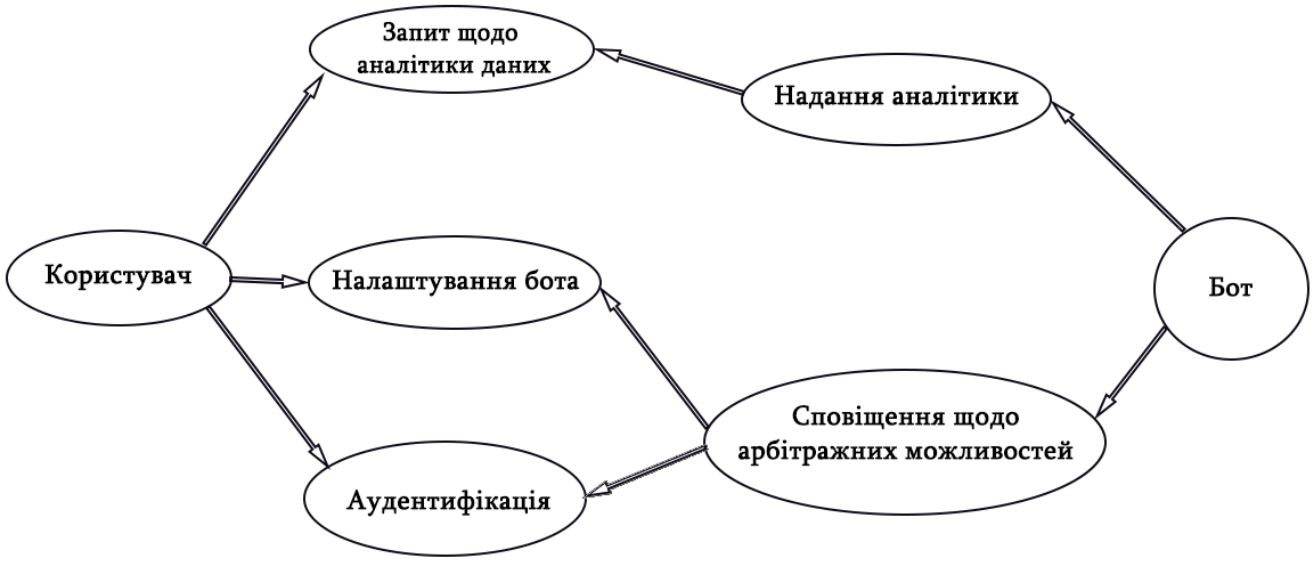

Рисунок 3.13 - Use Case діаграма інформаційної системи

3.5 Висновки

У третьому розділі було розроблено архітектуру інформаційної системи для виявлення арбітражних можливостей. Реалізовано модулі для збирання ринкових даних, оцінки ліквідності, перевірки доступності мереж, розроблена база даних для подальшої аналітики та надсилання повідомлень через Telegram. Застосовано асинхронний підхід на основі бібліотеки asyncio, що забезпечує паралельну роботу всіх потоків і мінімізує затримки при обробці запитів. Усі отримані дані проходять попередню обробку та нормалізацію, що дозволяє забезпечити коректний аналіз і точне виявлення прибуткових арбітражних угод. Telegram-бот виконує роль інтерфейсу взаємодії з користувачем, своєчасно надсилаючи повідомлення про можливі арбітражні стратегії на основі заданих параметрів. Реалізована система дозволяє оперативно реагувати на ринкові зміни та приймати рішення з високим рівнем автоматизації.

4 ТЕСТУВАННЯ TELEGRAM-БОТА

4.1 Інструкція користувача

Інструкція надана для допомоги у використанні інформаційної системи арбітражу на біржах Binance, Bybit, KuCoin, OKX, Gate.io та MEXC у режимі спот-спот торгівлі. Бот аналізує ціни, ліквідність, виводи/депозити, мережеві витрати та надсилає сповіщення у Telegram-канал.

Крок 1: Запуск бота

Надішліть */start* у чаті. Бот відповість: “Бот активовано. Очікуйте сповіщень.” Моніторинг цін — кожні 5 сек, виводів/депозитів — кожні 30 сек.

Крок 3: Налаштування

Використовуйте /set:

- /set spread 2 — спред 2%.
- /set volume 1000 — обсяг ліквідності 1000 USDT.
- /set network BEP20 — тільки зв'язки у мережі BEP20.

Крок 4: Сповіщення

Бот надсилає повідомлення у канал, наприклад: “ #TRIASUSDT Spread: 2.64% | Buy: $7.56 (MEXC) → Sell: $7.76 (KuCoin)……

Крок 5: Виконання угоди

- Протестуйте зв’язок: перекажіть малу суму (наприклад, $5), щоб перевірити доступність виводів/депозитів.
- Після успішного переказу збільште суму (наприклад, $500) і надішліть ще одну транзакцію слідом.
- Аналізуйте стакан, купуйте/продавайте частинами (не більше $200 за раз), щоб уникнути проковзування та погіршення ціни.

– Приклад: купіть TRIAS на MEXC за $500.28, перекажіть на KuCoin через BEP20 (витрати $2.97), продайте 3 частинами за $530.92, отримавши приблизний прибуток $25.

4.2 Тестування на практичних прикладах

4.2.1 Тестування арбітражної стратегії за допомогою розробленого бота

Тестування бота проводилося на реальних даних бірж, отриманих через API, що дозволило виявити арбітражні ситуації з прибутковістю від 0,5% до 5%. Дослідження показують, що такі можливості часто виникають через низьку ліквідність або затримки в оновленні цін на різних платформах [32].

Знайдено реальну арбітражну ситуацію з використанням даних із бірж MEXC і KuCoin для торгової пари TRIAS/USDT. Бот виявив вигідну можливість для арбітражу (рис. 4.1), проаналізувавши ринкові дані в реальному часі.

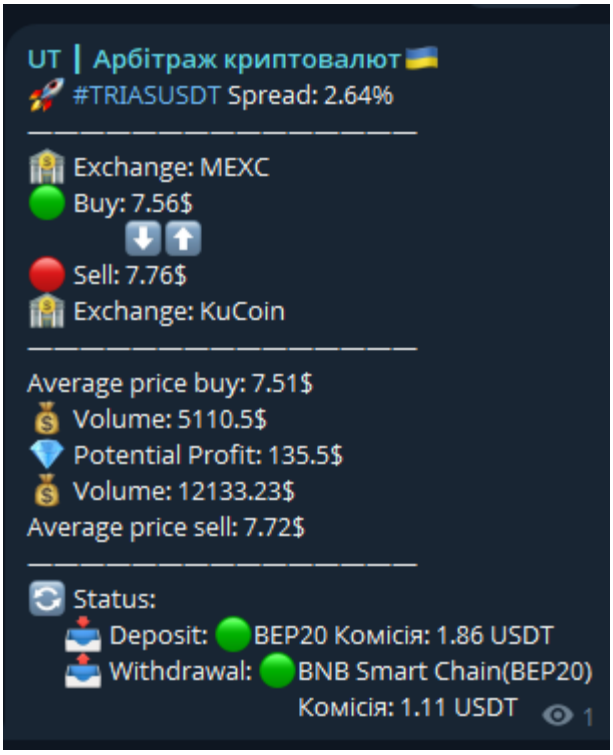

Рисунок 4.1 - Приклад вхідного сповіщення про арбітражну ситуацію

Бот отримав актуальні ціни: на біржі MEXC ціна покупки TRIAS/USDT склала $7.56, тоді як на KuCoin ціна продажу була $7.76, що створило спред у 2.64%. Середня ціна покупки на MEXC з урахуванням глибини ринку становила $7.51, а середня ціна продажу на KuCoin — $7.72. Обсяг торгів для покупки на MEXC склав $5110.5 (еквівалентно 677.8 TRIAS), а для продажу на KuCoin — $12133.23 (1565.7 TRIAS). Потенційний прибуток при обсязі угоди 677.8 TRIAS та обмеженій ліквідності MEXC, може скласти ($7.72 - $7.51) × 677.8 = $142.34.

Далі бот перевірив доступність депозитів і виводів. На MEXC депозити через мережу BEP20 (BNB Smart Chain) були доступні з комісією $1.86, а на KuCoin виводи через BEP20 — також доступні з комісією $1.11. Загальні мережеві витрати склали $1.86 + $1.11 = $2.97. Біржові торгові комісії (припустимо, 0.1% на обох біржах) додали ще $5.11 для покупки (($5110.5 × 0.001)) і $5.23 для продажу (($12133.23 × 0.001)). Таким чином, чистий прибуток після всіх витрат склав би $142.34 - $2.97 - $5.11 - $5.23 = $129.03.

Переконавшись, що ліквідність на обох біржах достатня і прибуток позитивний, бот надіслав сповіщення у Telegram-канал. Отримавши повідомлення, розпочато арбітражну операцію. Спочатку придбано 677.8 TRIAS на MEXC за $5090.28, сплативши торгову комісію $5.09. Потім я вивів TRIAS із MEXC на KuCoin через мережу BEP20, сплативши $1.86 за транзакцію; переказ зайняв близько 5 хвилини. Після зарахування депозиту на KuCoin, я продав 677.8 TRIAS за $5200.92, сплативши торгову комісію $5.23 що трохи відрізняється від очікуваного результату. Загальні витрати склали $5.09 + $1.86 + $1.11 + $5.23 = $13.29. Чистий прибуток від угоди склав $5200.92 - $5090.28 - $13.29 = $97.35, що близько до прогнозованих $142.34. Операція завершилася за 7 хвилини, і підтвердила ефективність бота у виявленні та реалізації арбітражних можливостей із мінімальними ризиками.

4.2.2 Теоретичний аналіз на основі отриманих даних

Для аналітиків ринку криптовалют велику користь відіграє інформація, яку можливо використати в подальшій аналітиці різноманітними способами для пошуку нових алгоритмів, закономірностей, причин невдач та іншого.

Для прикладу, проведено теоретичний аналіз ринкової поведінки активу XYZ на основі даних, зібраних у базі даних за допомогою запитів до REST API. Архівні дані з секундним таймфреймом дозволили детально дослідити волатильність активу та її вплив на арбітражну стратегію. Аналіз базується на датафреймі від 01:14:00 до 01:15:00 на біржі OKX. За допомогою створеної інформаційної системи зібрано та відображено у вигляді графіка отримані дані (рис. 4.2).

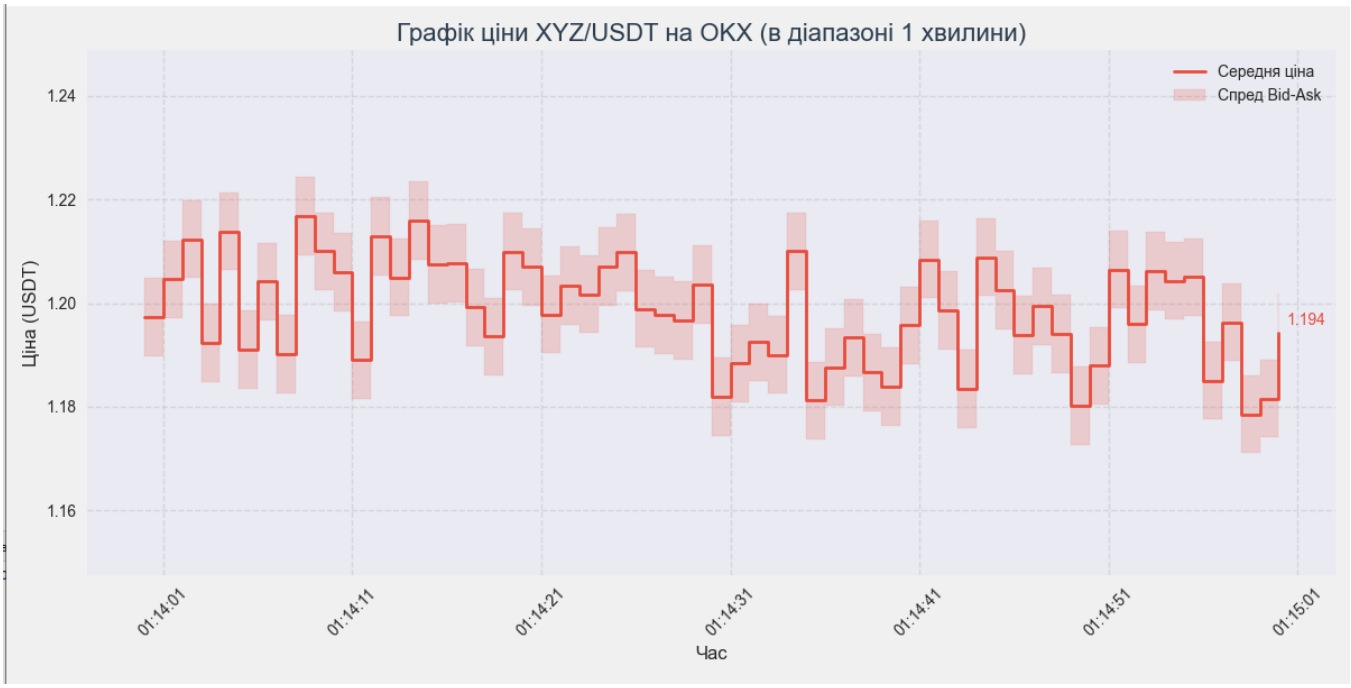

Рисунок 4.2 - Графік XYZ у діапазоні 1 хвилини

Отримані дані щодо ціни активу показали надзвичайно високу волатильність протягом однієї хвилини. Секундний таймфрейм виявив сходинчасті коливання ціни у межах від 1.180 до 1.220. Такі нестабільні стрибки, вказують на значний ризик у разі

використання арбітражної стратегії, зумовлений непередбачуваною активністю на біржі OKX, або ж недостатньою ліквідністю та порожнім простором між ордерами покупки та продажу. На основі цих даних можлива рекомендація щодо виключення цього активу із подальших арбітражних стратегій, віддаючи перевагу активам із більш стабільною поведінкою.

Наведено наступний приклад, знову зібрані дані у межах 1 хвилини для візуалізації графіка WIF з біржі GATE на основі ціни покупки та ціни продажу окремо. Протягом хвилини спред між ціною bid та ціною ask коливався в межах приблизно 2% від середньої ціни (рис. 4.3).

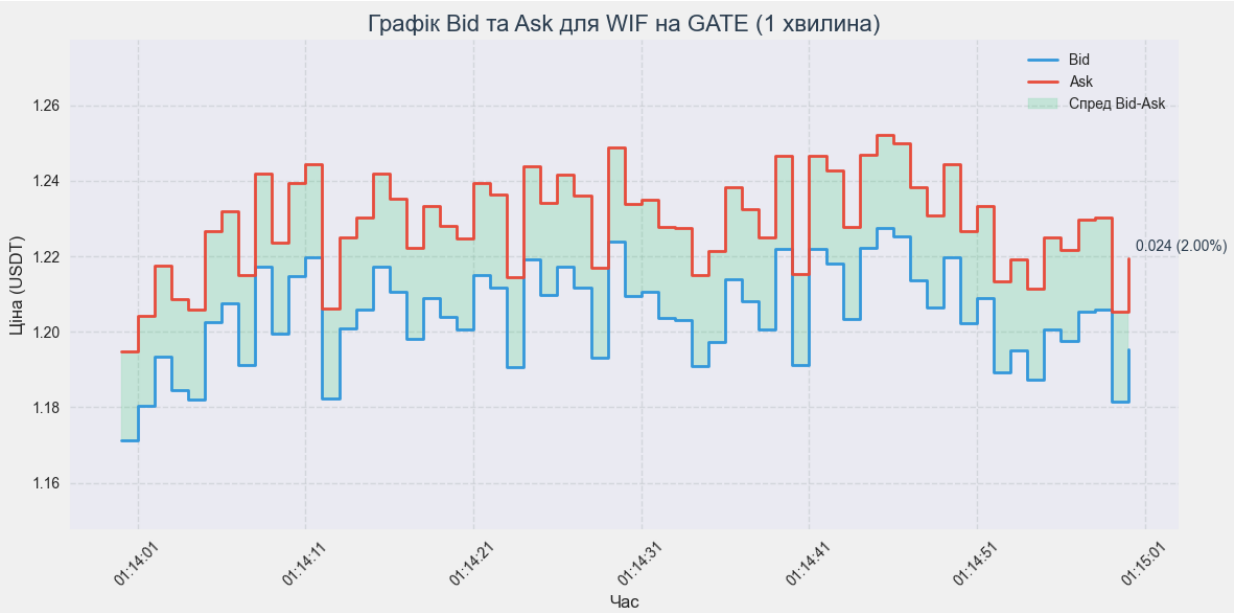

Рисунок 4.3 - Графік WIF для bid та ask у діапазоні 1 хвилини

Зазвичай, така різниця цін на одній платформі часто виникає саме під час абітражних можливостей, оскільки інші учасники ринку також активізуються. Такий широкий спред лише з однієї сторони (у вигляді платформи) арбітражної стратегії вказує на низьку ліквідність активу WIF на біржі GATE, що може бути спричинено недостатньою глибиною книги ордерів від біржі, неполадками, недостатньою зацікавленістю користувачів або обмеженою торгівельною активністю. При

моментальній купівлі активу за ціною ask і негайному продажу за ціною bid трейдер втрачає -2% від ціни через некомпетентність біржі. Це робить арбітражну стратегію ризиковою, оскільки при неполадках і неможливості вивести актив на іншу платформу при спробі продати актив назад, одразу втратимо ці 2%. Слід враховувати втрати на спреді які можуть перевищити потенційний прибуток від малих цінових рухів. Отже, під час арбітражу схожої торгових пар необхідно враховувати простір між цінами продажу та покупки який створюється під час арбітражної ситуації.

У наступному прикладі зображена (рис. 4.4) поширена ситуація, коли ціна на транзакцію (вивід активу з біржі) збільшується у декілька разів від початку арбітражної ситуації.

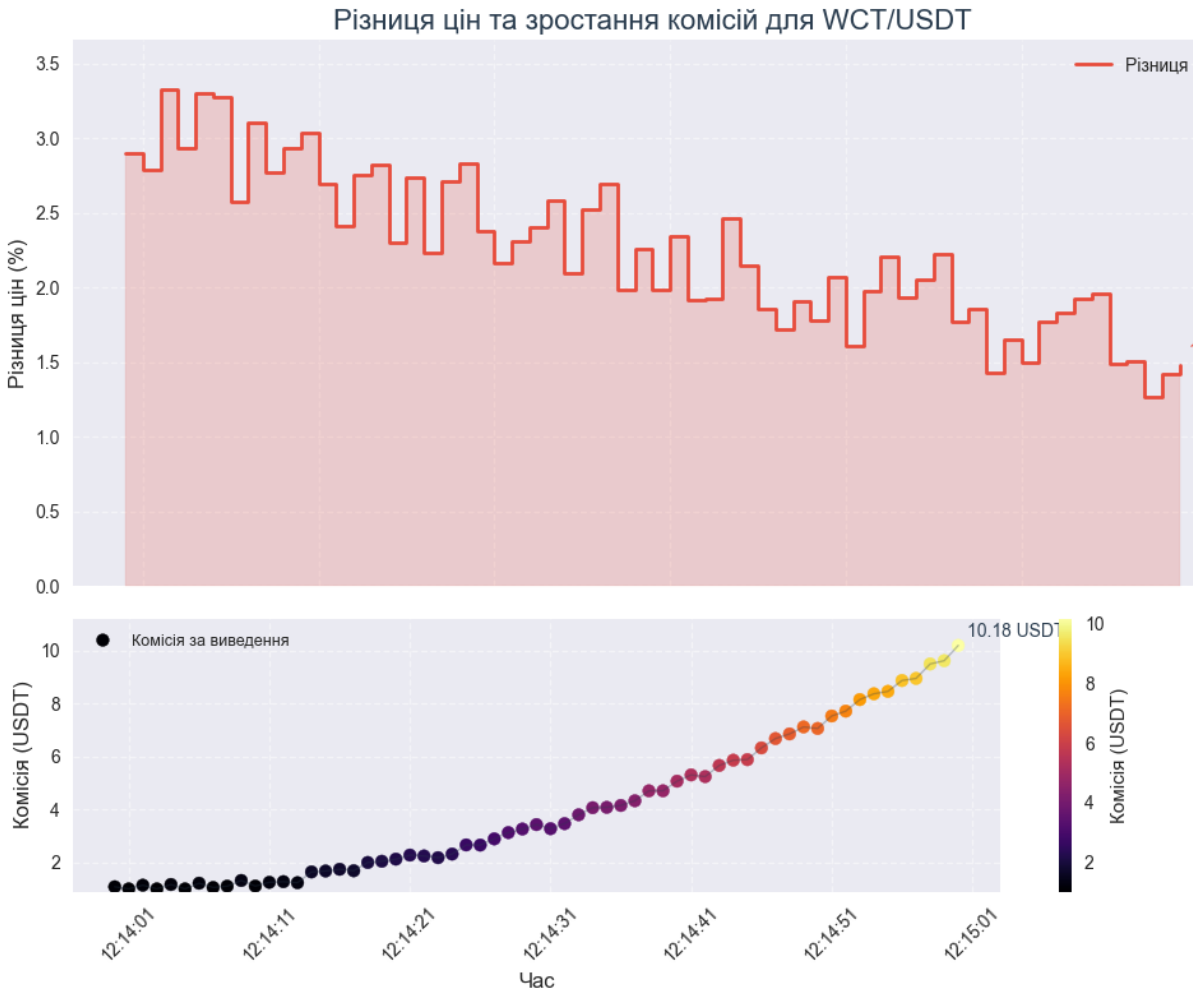

Рисунок 4.4 - Різниця цін та комісія за транзакцію у діапазоні хвилини

Хоча ціновий розрив і зберігався, зростання комісії на виведення могло зробити арбітраж економічно невигідним, особливо при роботі з фіксованими сумами.

Наприклад, розглянемо двох учасників ринку. Перший - початківець із невеликим стартовим капіталом у 100 доларів. Другий - досвідчений трейдер із капіталом у 10000 доларів. За середньої різниці цін у 1.5% прибуток першого складе лише 1.5 долара, тоді як другого — 150 доларів.

Однак якщо комісія на виведення зростає до 10 доларів, то для початківця арбітражна операція виявляється глибоко збитковою, адже витрати значно перевищують потенційний дохід. У випадку досвідченого трейдера із більшим капіталом витрати на комісію майже не впливають на економічний прибуток. Це підкреслює, що при високих мережевих комісіях арбітраж стає можливим лише для операцій з високим обсягом.

4.3 Порівняння результатів з чинними арбітражними сервісами

Оскільки більшість схожих сервісів мають платну основу, безкоштовних конкурентів одиниці й розроблений Telegram-бот для арбітражної торгівлі порівнювався з українським сервісом UaInvest [33] (рис. 4.5), який також пропонує безкоштовний доступ до аналізу арбітражних можливостей між криптовалютними біржами. Оцінка проводилась за швидкістю, зручністю, точністю та функціональністю.

Розроблений у кваліфікаційній роботі бот оновлює ціни кожну секунду, що дозволяє швидко виявляти спред. UaInvest оновлює ціни лише раз на 30 секунд, що значно уповільнює реакцію на ринкові зміни, особливо у волатильні періоди.

Бот враховує мережеві витрати й торгові комісії, прогнозуючи прибуток. UaInvest не уточнює витрати в реальному часі через затримки в оновленні даних, що може призводити до неточних прогнозів.

Бот працює через Telegram із командами, надсилаючи автоматичні сповіщення із деталями угоди. UaInvest не підтримує автоматичні сповіщення, а моніторинг через сайт незручний. Користувачу доводиться тримати сторінку відкритою та дивитись та оновлювати дані самому, що забирає час і знижує ефективність.

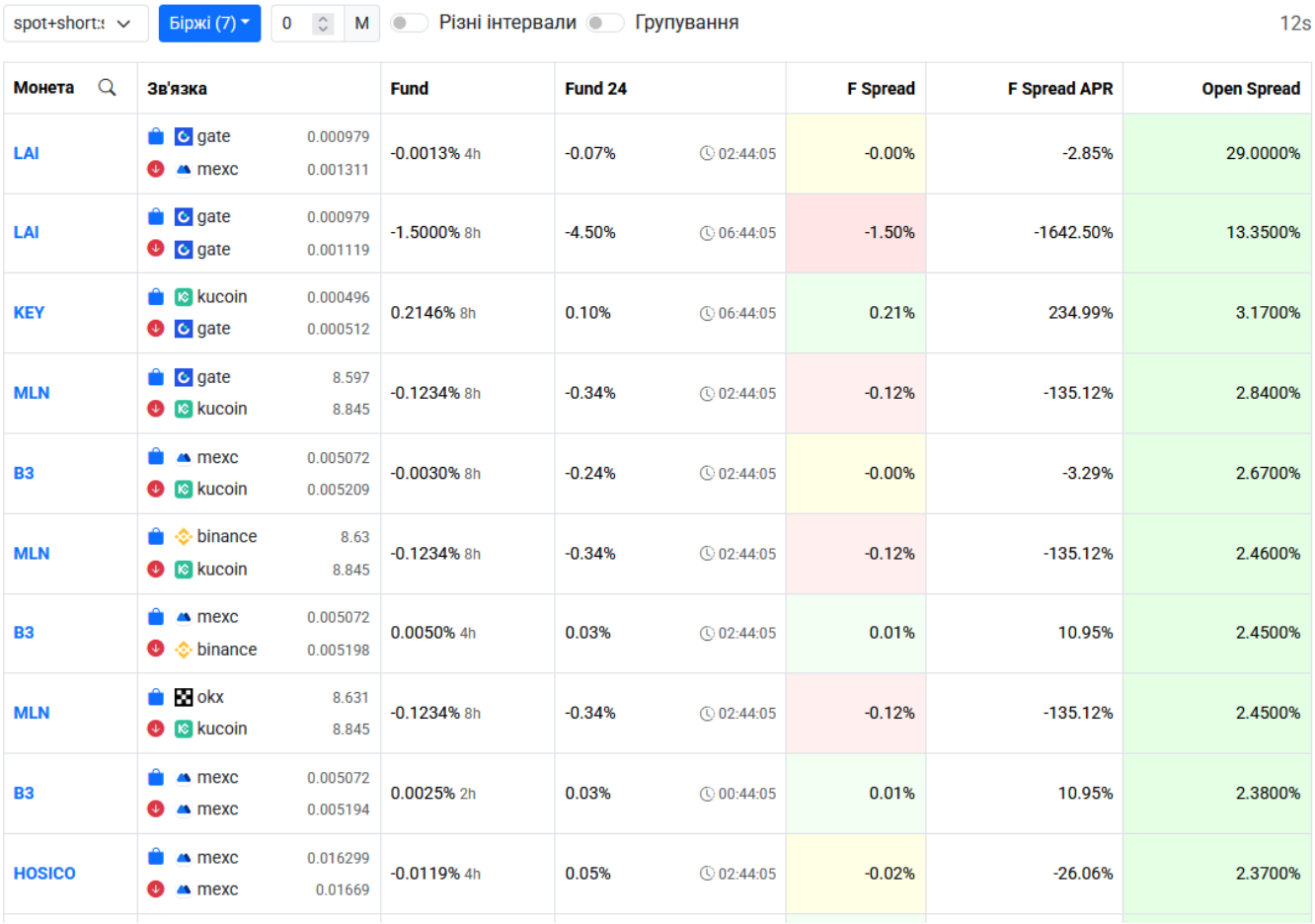

Рисунок 4.5 - Український сервіс UaInvest для арбітражу [33]

Бот перевіряє ліквідність, доступність виводів/депозитів і розраховує чистий прибуток, а також зберігає логи у JSON. UaInvest пропонує базовий аналіз спреду, але не враховує ліквідність і не надає логів чи історії операцій.

UaInvest має перевагу в дизайні: його веб інтерфейс виглядає сучасно, із графіками та кольоровим виділенням спредів, що приємно для ока. Наш бот, який працює через Telegram, поступається у візуальному оформленні, адже сповіщення

надсилаються у текстовому форматі. Проте цей недолік несуттєвий, оскільки на екрані користувача відображається лише необхідна інформація, без надлишкових даних, що забезпечує швидке сприйняття інформації.

Розроблений бот перевершує UaInvest за швидкістю зручністю і функціональністю, програє лише у візуалізації. UaInvest. Хоч платформа й безкоштовна, поступається через повільне оновлення цін, відсутність сповіщень, що робить його менш ефективним для арбітражу.

4.4 Висновки

У розділі 4 створено інструкцію для Telegram-бота. Проведено тестування на практичному прикладі з розрахунками та тестування архівованих даних на зразку аналітики різноманітних аспектів арбітражної стратегії. Виконано порівняння з конкурентом у вигляді веб сервісу UaInvest, встановлено переваги та недоліки.

ВИСНОВКИ

Кваліфікаційна робота зосереджена на розробці інформаційної системи для автоматизації пошуку арбітражних можливостей на криптовалютних платформах, що сприяє підвищенню ефективності прийняття торгових рішень та є актуальним аналітичним інструментом. Виконання поставлених завдань дозволило отримати наступні результати:

- Проведено аналіз криптовалютного ринку та механізмів арбітражної торгівлі, що дало змогу визначити ключові принципи та види арбітражу, а також визначити основні фінансові показники для виявлення можливостей для арбітражної стратегії.
- Досліджено API провідних криптовалютних бірж, що забезпечило розуміння технічних аспектів збирання ринкових даних у реальному часі.
- Розроблено алгоритми для виявлення арбітражних ситуацій, які базуються на аналізі різниці цін активів на різних біржах із урахуванням транзакційних витрат та доступності мереж.
- Реалізовано програмну архітектуру бота з модульною структурою, що забезпечує гнучкість, масштабованість та інтеграцію з API криптовалютних бірж.
- Створено бота, який виконує функції моніторингу ринкових даних, обробки інформації та оперативного сповіщення користувачів про виявлені арбітражні можливості. Бот має простий інтерфейс для користувача і забезпечує повну аналітику ринкової ситуації.
- Додатково впроваджено архівування отриманих даних для подальшого аналізу ефективності арбітражної стратегії під час її виникнення.
- Проведено тестування розробленого рішення на реальних ринкових даних, що підтвердило його працездатність та здатність виявляти арбітражні можливості.

У результаті розроблено бота, мета якого полягає в автоматизації трудомістких процесів збирання та аналізу ринкових даних. Розроблений Telegram-бот може бути використаний як трейдерами, так і аналітиками і розробниками торгових систем.

Подальший розвиток проєкту може включати інтеграцію додаткових бірж, удосконалення алгоритмів за рахунок WebSockets та урахування волатильності ринку. Можлива інтеграція ф'ючерсної торгівлі, а також розширення функціоналу бота, наприклад, додавання автоматичного виконання торгових операцій.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Larry Ericson, Solar Powered Charging Infrastructure for Electric Vehicles, Larry Ericson, USA, Washington: CRC PRESS, 2019. – 168 с.
2. Що таке криптовалюта. URL: https://academy.binance.com/uk-UA/articles/what-is-a-cryptocurrency (дата звернення: 01.03.2025).
3. Блокчейн: що це за технологія та як вона змінює сучасний бізнес. URL: https://hub.kyivstar.ua/articles/blokchejn-shho-cze-za-tehnologiya-ta-yak-vona-zminyuye-suchasnij-biznes (дата звернення: 04.03.2025).
4. Що таке смартконтракти і як вони працюють? URL: https://academy.binance.com/uk-UA/articles/what-are-smart-contracts (дата звернення: 07.03.2025).
5. Централізовані та децентралізовані біржі: різниця. URL: https://galka.if.ua/tsentralizovani-ta-detsentralizovani-birzhi-riznytsia/ (дата звернення: 10.03.2025).
6. Makarov P., Schoar R. Trading and Arbitrage in Cryptocurrency Markets. Journal of Financial Economics. 2020. V. 135, no. 2. pp. 293–319. DOI: 10.1016/j.jfineco.2019.07.001
7. Арбітраж (економіка). URL: https://uk.wikipedia.org/wiki/%D0%90%D1%80%D0%B1%D1%96%D1%82%D1%80%D0%B0%D0%B6_(%D0%B5%D0%BA%D0%BE%D0%BD%D0%BE%D0%BC%D1%96%D0%BA%D0%B0) (дата звернення: 13.03.2025).
8. Antonopoulos A.M. Mastering Bitcoin: Programming the Open Blockchain. 2nd ed. Sebastopol, USA: O'Reilly Media, 2017. – 408 с.
9. Книга ордерів на OKX. URL: https://www.okx.com/trade-spot/btc-usdt (дата звернення: 16.03.2025).
10. Офіційний сайт Binance. URL: https://www.binance.com/ua-UA (дата звернення: 18.03.2025).

11. Офіційний сайт Bybit. URL: https://www.bybit.com/ (дата звернення: 18.03.2025).
12. Офіційний сайт KuCoin. URL: https://www.kucoin.com/ (дата звернення: 18.03.2025).
13. Офіційний сайт OKX. URL: https://www.okx.com/ (дата звернення: 18.03.2025).
14. Офіційний сайт OKX. URL: https://www.okx.com/ (дата звернення: 18.03.2025).
15. Офіційний сайт MEXC. URL: https://www.mexc.com/uk-UA/ (дата звернення: 18.03.2025).
16. PyCharm: інтегроване середовище розробки. URL: https://www.jetbrains.com/pycharm/ (дата звернення: 27.03.2025).
17. Офіційна документація Python 3. URL: https://docs.python.org/3/ (дата звернення: 27.03.2025).
18. Приклади використання бібліотеки Requests у Python. URL: https://www.it-notes.wiki/python/requests-python-usage-examples/ (дата звернення: 29.03.2025).
19. Документація бібліотеки time у Python 3.13. URL: https://docs.python.org/uk/3.13/library/time.html (дата звернення: 30.03.2025).
20. Документація бібліотеки hmac у Python 3.13. URL: https://docs.python.org/3/library/hmac.html (дата звернення: 31.03.2025).
21. Документація бібліотеки asyncio у Python 3.13. URL: https://docs.python.org/uk/3.13/library/asyncio.html (дата звернення: 01.04.2025).
22. Документація бібліотеки base64 у Python 3.13. URL: https://docs.python.org/uk/3.13/library/base64.html (дата звернення: 03.04.2025).
23. Документація бібліотеки threading у Python 3.13. URL: https://docs.python.org/uk/3.13/library/threading.html (дата звернення: 05.04.2025).
24. Документація бібліотеки hashlib у Python 3.13. URL: https://docs.python.org/uk/3.13/library/hashlib.html (дата звернення: 09.04.2025).
25. Документація бібліотеки aiohttp. URL: https://docs.aiohttp.org/en/stable/ (дата звернення: 09.04.2025).

26. Документація API Binance Spot. URL: https://developers.binance.com/docs/binance-spot-api-docs (дата звернення: 11.04.2025).
27. Документація API Bybit v5. URL: https://bybit-exchange.github.io/docs/v5/intro (дата звернення: 13.04.2025).
28. Вступ до документації KuCoin. URL: https://www.kucoin.com/docs/beginners/introduction (дата звернення: 12.03.2025). (дата звернення: 15.04.2025).
29. Документація API OKX v5. URL: https://www.okx.com/docs-v5/en/#overview (дата звернення: 17.04.2025).
30. Документація API Gate.io v4. URL: https://www.gate.io/docs/developers/apiv4/ (дата звернення: 19.04.2025).
31. Документація API MEXC Spot v3. URL: https://mexcdevelop.github.io/apidocs/spot_v3_en/#introduction (дата звернення: 21.04.2025).
32. Alden B. Trading and Exchanges: Market Microstructure for Practitioners. Oxford, UK: Oxford University Press, 2002. – 656 с.
33. Арбітраж на UaInvest. URL: https://uainvest.com.ua/ (дата звернення: 23.04.2025).

Додаток А

(обов’язковий)

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації

ЗАТВЕРДЖУЮ

Завідувач кафедри САІТ

_____ д.т.н., проф. Віталій МОКІН

“____”_____ 2025 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на бакалаврську кваліфікаційну роботу

АРБІТРАЖНИЙ БОТ НАЙПОПУЛЯРНІШИХ БІРЖ КРИПТОВАЛЮТИ

08-34.БКР.002.00.000 ТЗ

Керівник: к.т.н., доц.

_____ Георгій ГОРЯЧЕВ

«__» _____ 2025 р.

Розробив студент гр. 2ІСТ-21б

_____ Павло ГОЛОВІН

«__» _____ 2025 р.

Вінниця 2025

1. Підстава для проведення робіт.

Підставою для виконання роботи є наказ №___по ВНТУ від «__» _____2025р., та індивідуальне завдання на БКР, затверджене протоколом №_ засідання кафедри САІТ від «__» _____2025р.

2. Джерела розробки:
    1. Larry Ericson, Solar Powered Charging Infrastructure for Electric Vehicles, Larry Ericson, USA, Washington: CRC PRESS, 2019.
    2. Makarov P., Schoar R. Trading and Arbitrage in Cryptocurrency Markets. Journal of Financial Economics. 2020. V. 135, no. 2. pp. 293–319. DOI: 10.1016/j.jfineco.2019.07.001 (дата звернення: 16.03.2025).
3. Мета і призначення роботи.

Метою роботи є розроблення інформаційної системи для збирання та аналізу даних для пошуку арбітражних можливостей у найпопулярніших біржах криптовалют.

4. Вихідні дані для проведення робіт:

1) Дані про ринкові ціни та ліквідність криптовалютних бірж;

2) Дані API криптовалютних бірж;

5. Методи дослідження:

1) Аналіз криптовалютного ринку та принципів арбітражу, вивчення API бірж для збирання даних, розробка алгоритмів для пошуку арбітражних можливостей, структурний аналіз архітектури інформаційних систем, розроблення UML діаграм, тестування бота на реальних даних.

6. Етапи роботи і терміни їх виконання:

a) Аналіз предметної області _____–_____

b) Огляд аналогів _____–_____

c) Вибір оптимальних інформаційних технологій _____–_____

d) Розроблення інформаційної системи збирання та аналізу даних для пошуку

арбітражних можливостей у найпопулярніших біржах криптовалют _____–_____

е) Оформлення матеріалів до захисту БКР _____–_____

7. Очікувані результати та порядок реалізації

1. Збирання ринкових даних з бірж, виявлення арбітражних можливостей, надсилання сповіщень користувачу, зберігання даних для подальшого аналізу.

8. Вимоги до розробленої документації

Текстова та ілюстративна частини роботи оформлені у відповідності до вимог «Методичних вказівок до виконання бакалаврських кваліфікаційних робіт для студентів спеціальностей: 124 «Системний аналіз», 126 «Інформаційні системи та технології» (освітня програма «Прикладні інформаційні технології»)».

9. Порядок приймання роботи

Публічний захист «___» _____ 2025 р.

Початок роботи «___» _____ 2025 р.

Граничні терміни виконання БКР «___» _____ 2025 р.

Розробив студент групи 2ІСТ-21б _____ Павло ГОЛОВІН

Додаток Б
(обов'язковий)

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Назва роботи: «Абітражний бот найпопулярніших бірж криптовалюти»

Тип роботи: бакалаврська кваліфікаційна робота

Підрозділ: кафедра САІТ, ФІІТА, гр. 2ІСТ-21б

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism 0,83 %

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне):

- ■ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту
- □ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.
- □ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

Віталій МОКІН, зав. каф. САІТ _____ (підпис)

Євгеній КРИЖАНОВСЬКИЙ, доц. каф. САІТ _____ (підпис)

Особа, відповідальна за перевірку _____ (підпис) Сергій ЖУКОВ

З висновком експертної комісії ознайомлений(-на)

Керівник _____ (підпис) Георгій ГОРЯЧЕВ, к.т.н., доц. каф. САІТ

Здобувач _____ (підпис) Павло ГОЛОВІН

Додаток В
(довідниковий)
Фрагмент лістингу програми

```
import requests
import time
import hmac
from pybit.unified_trading import HTTP
import asyncio
import threading
import okx.Funding as Funding
import base64
import hashlib
import json
import aiohttp
from dotenv import load_dotenv
import os

stakan_value=100
minimal_spread = 1

# Order_books
def buy_stakan_binance(symbol):
    try:
        global stakan_value
        url = f'https://api.binance.com/api/v3/depth?symbol={symbol}&limit=100'
        response = requests.get(url)
        data = response.json()
        total_usdt = 0
        total_quantity=0
        price_mas=[]
```

```
        quantity_mas=[]
        for ask in data['asks']:
            price, quantity = ask
            if total_usdt<stakan_value:
                total_usdt+=float(price)*float(quantity)
                price_mas.append(price)
                quantity_mas.append(quantity)
        for i in range(len(quantity_mas)):
            total_quantity+=float(quantity_mas[i])
        averageprice=total_usdt/total_quantity
        mas=[]
        mas.append(averageprice)
        mas.append(total_usdt)
        return mas

    except Exception as e:
        print(e)

# Get last prices bid ask

binance_data = {}
bybit_data = {}
gate_data = {}
kucoin_data = {}
okx_data = {}
mexc_data = {}

async def get_binance_data(session):
    try:
        global binance_data
```

```
        url = "https://api.binance.com/api/v3/ticker/bookTicker"
        async with session.get(url) as response:
            if response.status == 200:
                data_binance = await response.json()
                for entry in data_binance:
                    symbol = entry['symbol']
                    binance_data[symbol] = {
                        'askPrice': float(entry['askPrice']),
                        'bidPrice': float(entry['bidPrice'])
                    }
            else:
                print(f"Binance API error: {response.status}")

    except Exception as e:
        print(e)

    print(data_binance)

get_binance_data()
time.sleep(1000)
#KUCOIN
async def get_kukoin_networks(session):
    api_key = ''
    api_secret = ''
    api_passphrase = ''

    url = 'https://api.kucoin.com/api/v3/currencies'

    timestamp = str(int(time.time() * 1000))
    method = 'GET'
```

```
    data = f"{timestamp}{method}/api/v1/withdrawals/quotas"

    signature = base64.b64encode(hmac.new(api_secret.encode('utf-8'), data.encode('utf-8'), hashlib.sha256).digest())

    headers = {
        'KC-API-KEY': api_key,
        'KC-API-SIGN': signature,
        'KC-API-TIMESTAMP': timestamp,
        'KC-API-PASSPHRASE': api_passphrase,
    }

    response_kucoin_networks = await session.get(url, headers=headers)
    data=response_kucoin_networks.json()
    return data['data']
def binance_deposit(x,price):
    mas=[]
    for coin_info in response_binance_networks.json():
        coin_name = coin_info['coin']
        if coin_name == x:

            networks = coin_info.get('networkList', [])
#           print(f"Coin: {coin_name}")
            for network_info in networks:
                network_name = network_info['name']
                is_busy = network_info.get('busy', False)
                withdrawal_enabled = network_info.get('withdrawEnable', False)
                deposit_enabled = network_info.get('depositEnable', False)
                withdrawFee = float(network_info.get('withdrawFee', False))*float(price)
                if deposit_enabled==True:
                    mas.append(network_name+" Комиссия: " + str((round(withdrawFee,2)))+ " USDT")
```

```
    return mas
async def update_prices():
    async with aiohttp.ClientSession() as session:
        while True:
            try:
                print("PRICE UPDATE")
                time_start=time.time()
                tasks = [
                    get_binance_data(session),
                    get_bybit_data(session),
                    get_gate_data(session),
                    get_kukoin_data(session),
                    get_okx_data(session),
                    get_mexc_data(session)
                ]
                await asyncio.gather(*tasks)
                time_end = time.time()
                print("Time per cycle: ", time_end-time_start)
            except Exception as e:
                print(f"{e}")

            await asyncio.sleep(1)

def run_price_updates():
    loop = asyncio.new_event_loop()
    asyncio.set_event_loop(loop)
    loop.run_until_complete(update_prices())

price_thread = threading.Thread(target=run_price_updates)
price_thread.start()
```

Додаток Г
(обов’язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

АРБІТРАЖНИЙ БОТ НАЙПОПУЛЯРНІШИХ БІРЖ КРИПТОВАЛЮТИ

Нормоконтроль: к.т.н., доцент
_____Сергій ЖУКОВ
«___» _____2025 р.

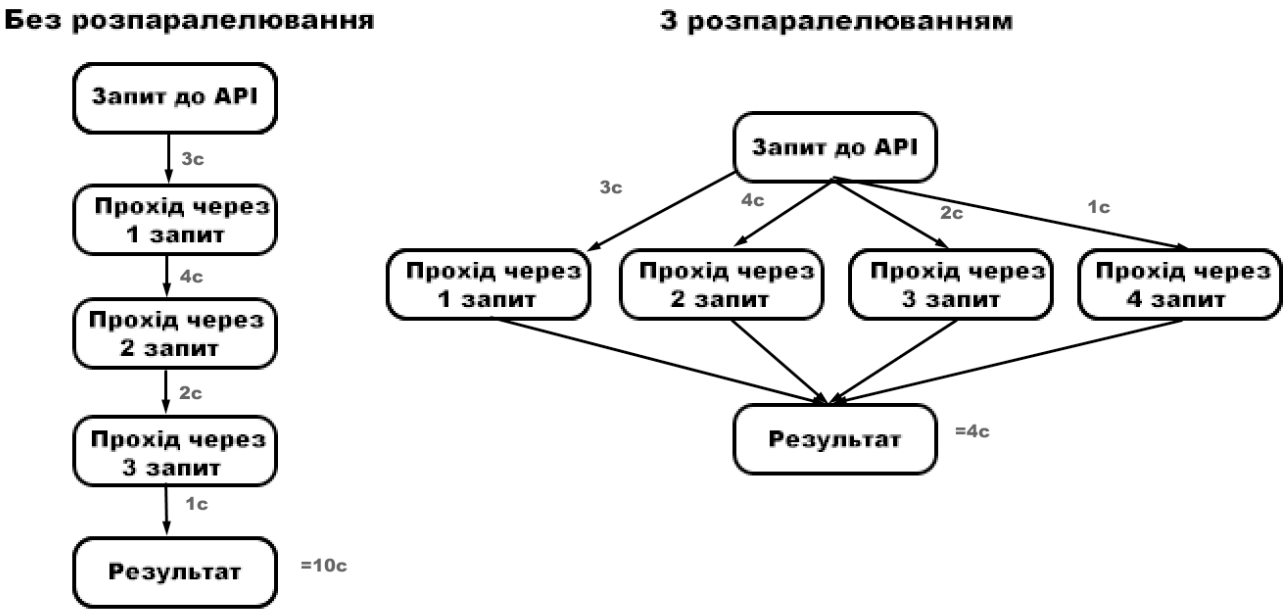

Рисунок Г.1 - Принцип розпаралелювання запитів API

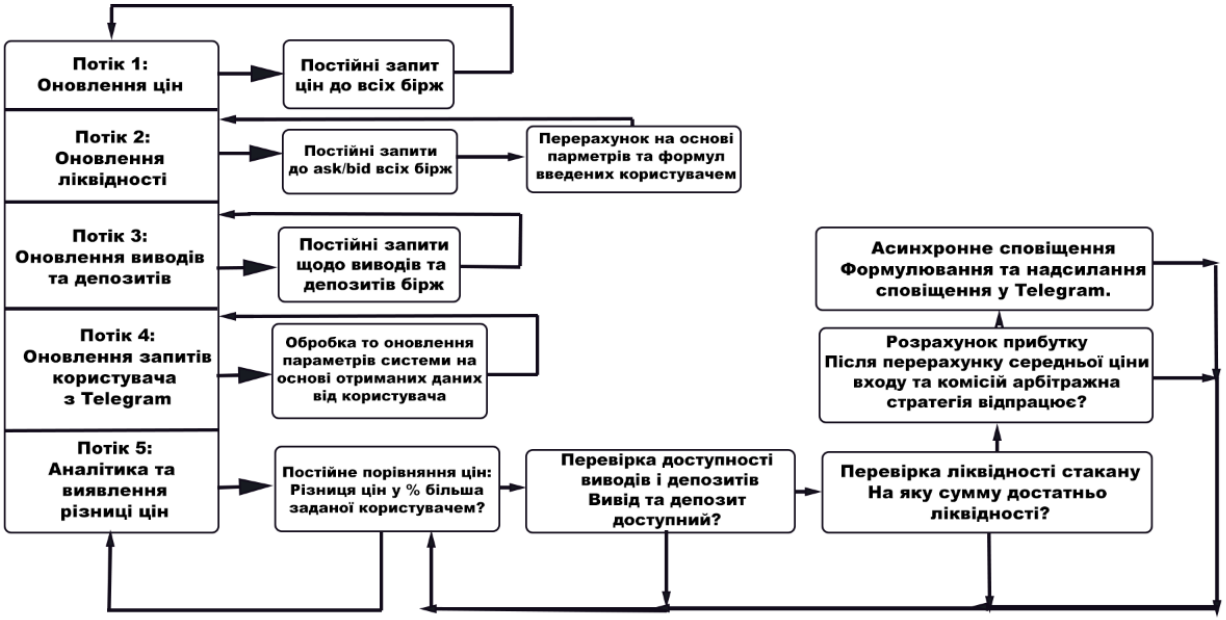

Рисунок Г.2 - Архітектура інформаційної системи

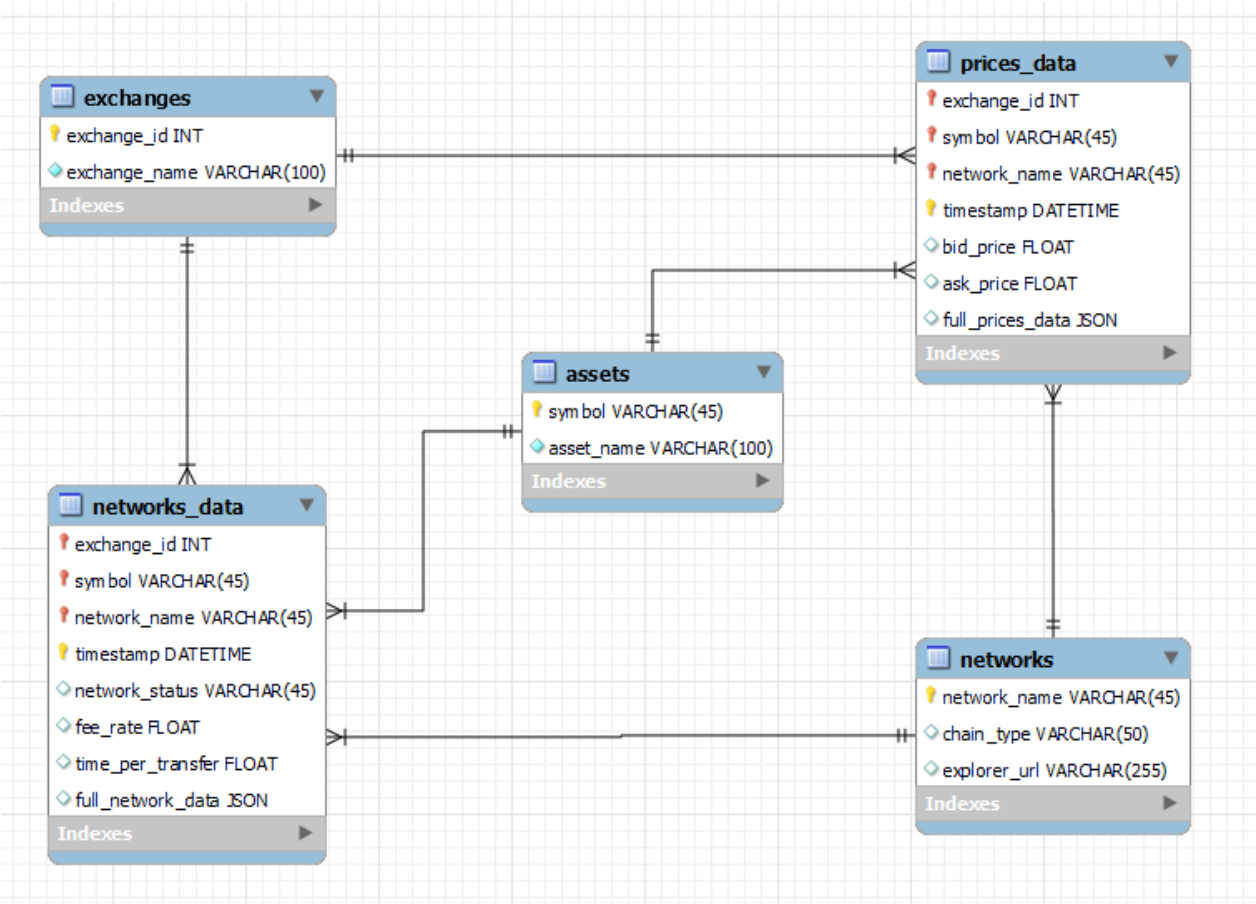

Рисунок Г.3 - База даних для архівації ринкових даних

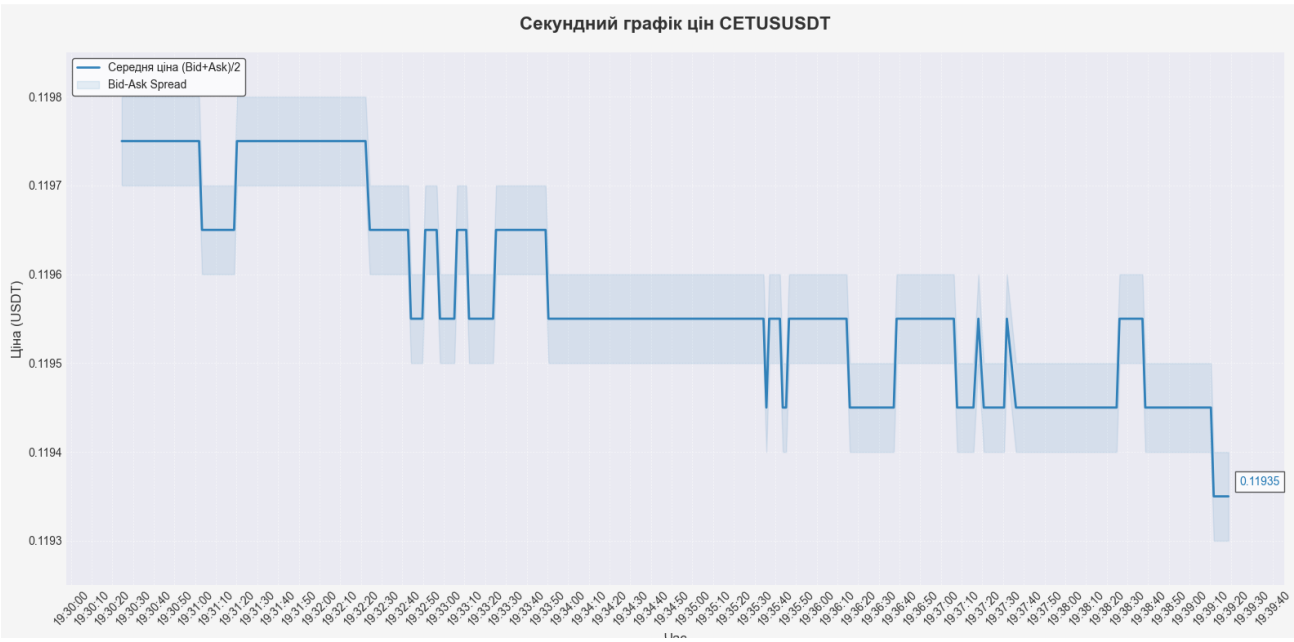

Рисунок Г.4 - 10-ти секундний графік ціни активу CETUS на Binance

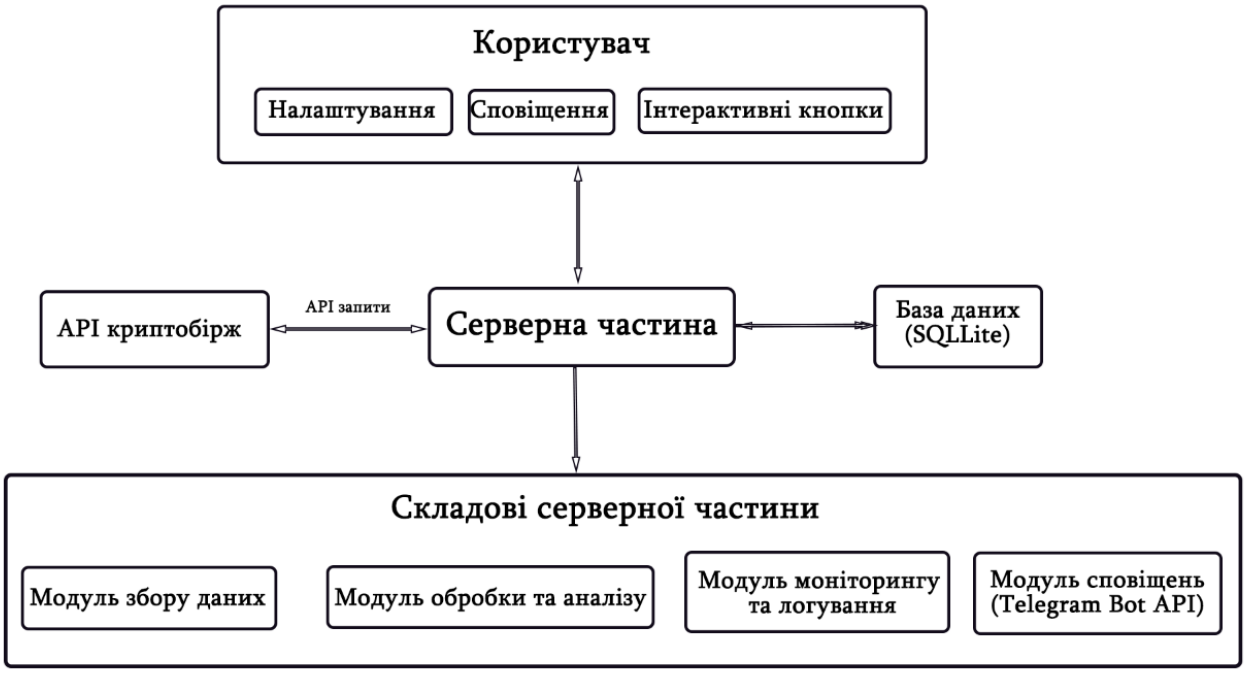

Рисунок Г.5 - ІТ-інфраструктура бота арбітражу криптовалют

Рисунок Г.6 - Use Case діаграма інформаційної системи

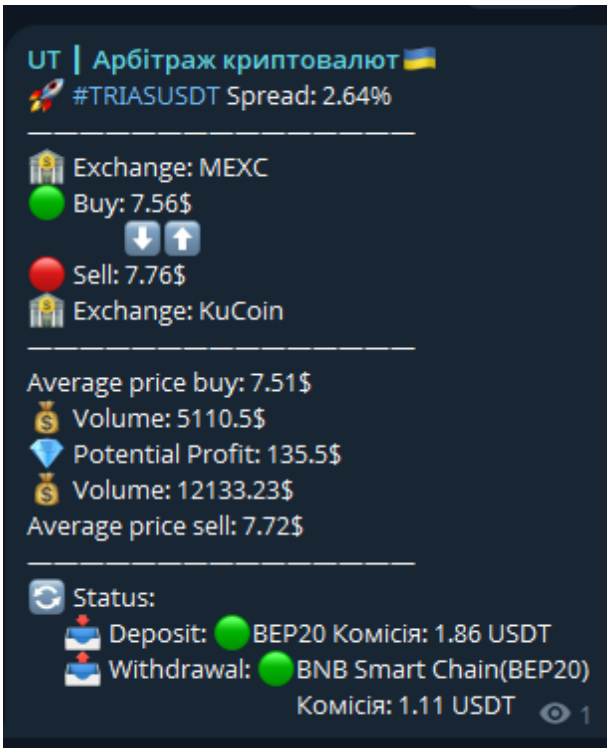

Рисунок Г.7 - Приклад вхідного сповіщення про арбітражну ситуацію

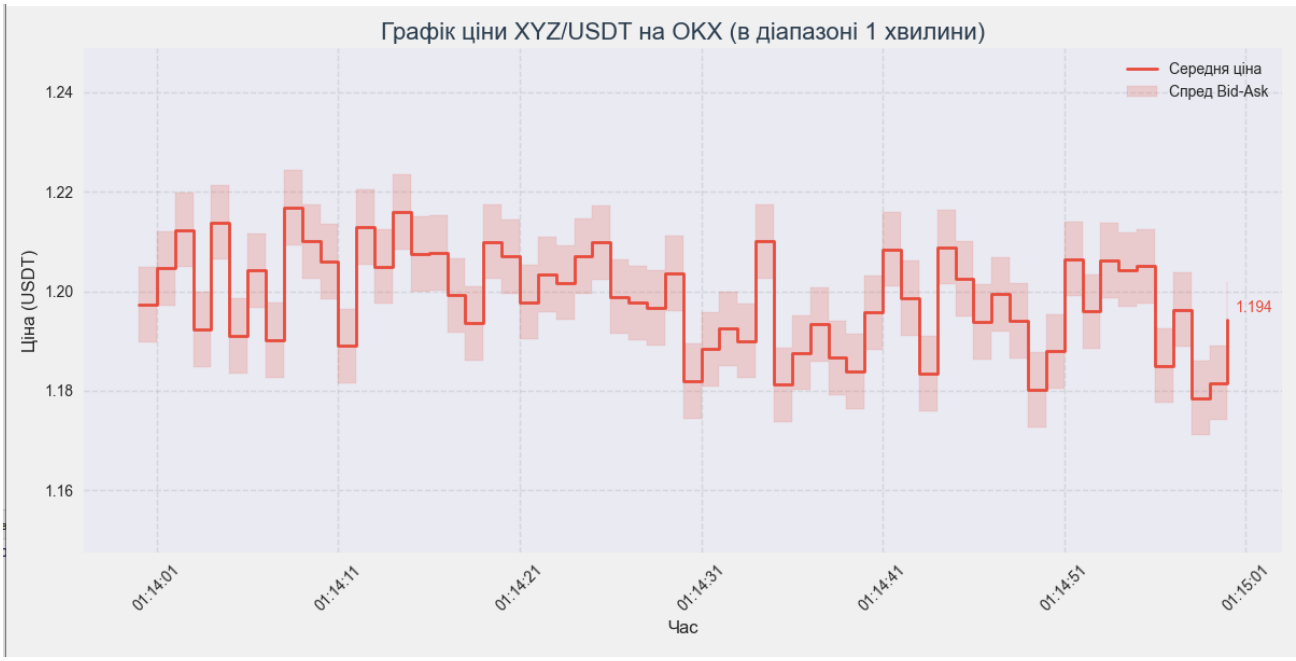

Рисунок Г.8 - Графік XYZ у діапазоні 1 хвилини

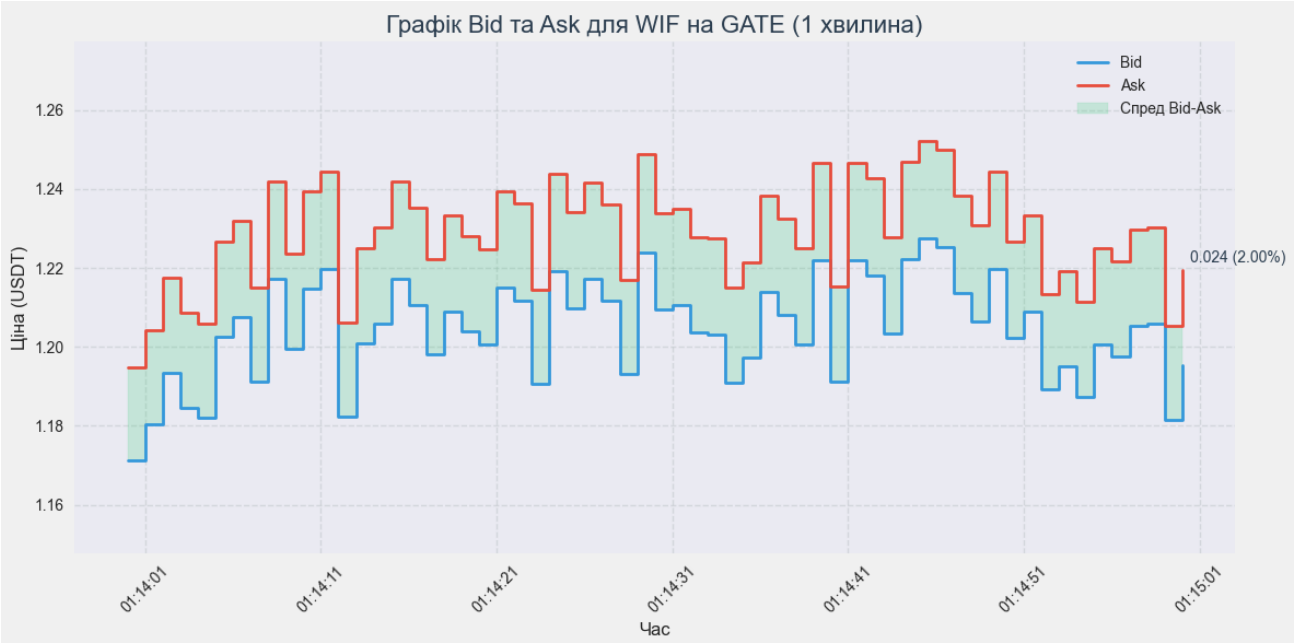

Рисунок Г.9 - Графік WIF для bid та ask у діапазоні 1 хвилини

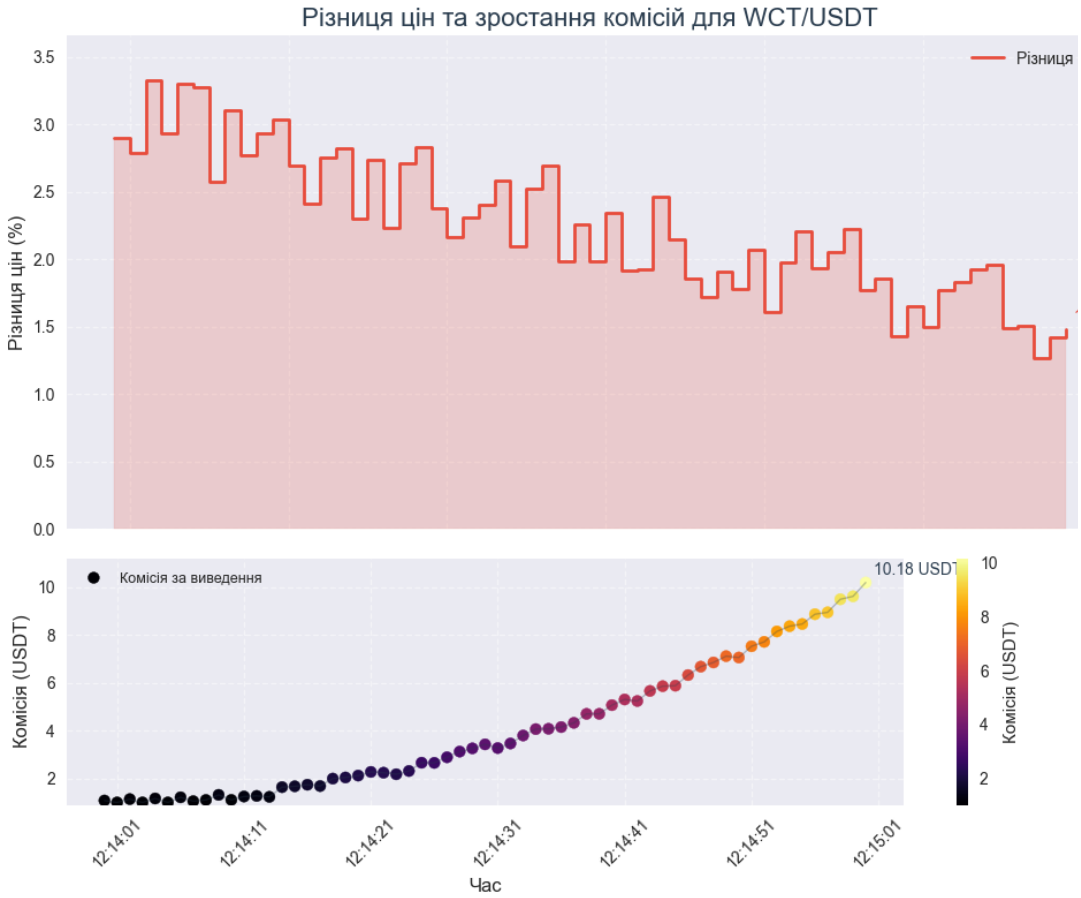

Рисунок Г.10 - Різниця цін та комісія за транзакцію у діапазоні хвилини