НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет/(ННІ) **інформаційних технологій**

ПОГОДЖЕНО
Декан факультету (Директор ННІ)
інформаційних технологій
(назва факультету (ННІ))
_____ Ігор БОЛБОТ
(підпис) (ім'я ПРІЗВИЩЕ)
"___" _____ 2025 р.

ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ
Завідувач кафедри
комп'ютерних наук
(назва кафедри)
_____ Белла ГОЛУБ
(підпис) (ім'я ПРІЗВИЩЕ)
"___" _____ 2025 р.

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему Аналітична система аналізу ринку криптовалют

Спеціальність 121 «Інженерія програмного забезпечення»
(код і найменування)

Освітня програма Програмне забезпечення інформаційних систем
(назва)

Орієнтація освітньої програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Гарант освітньої програми
к.ф-м.н. доцент _____ Віктор КИРИЧЕНКО
(науковий ступінь та вчене звання) (підпис) (ім'я ПРІЗВИЩЕ)

Керівник магістерської кваліфікаційної роботи
старший викладач _____ Георгій БОРОДКІН
(науковий ступінь та вчене звання) (підпис) (ім'я ПРІЗВИЩЕ)

Консультант
к.т.н доцент _____ Юлія БОЯРІНОВА
(науковий ступінь та вчене звання) (підпис) (ім'я ПРІЗВИЩЕ)

Виконав _____ Петро-Еммануїл ЩЕРБАН
(підпис) ((ім'я ПРІЗВИЩЕ здобувача)

КИЇВ – 2025

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет (ННІ) _____**інформаційних технологій**_____

ЗАТВЕРДЖУЮ
Завідувач кафедри
__комп'ютерних наук_____
_к.т.н., доцент_____ _____ _Белла ГОЛУБ
(науковий ступінь, вчене звання) (підпис) (ім'я ПРІЗВИЩЕ)
"_01_" _листопада__2024 року

З А В Д А Н Н Я

ДО ВИКОНАННЯ МАГІСТЕРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ ЗДОБУВАЧУ

_____Щербан Петро-Еммануїл Петрович_____
(прізвище, ім'я, по батькові)

Спеціальність _____121 «Інженерія програмного забезпечення»_____
(код і найменування)

Освітня програма _____Програмне забезпечення інформаційних систем_____
(назва)

Орієнтація освітньої програми_____освітньо-професійна_____
(освітньо-професійна або освітньо-наукова)

Тема магістерської кваліфікаційної роботи Аналітична система аналізу ринку криптовалют_____
_____

затверджена наказом від "_01_" _листопада_____2024р. № 1963 «С»_____

Термін подання завершеної роботи на кафедру ____2024.11.29_____
(рік, місяць, число)

Вихідні дані до магістерської кваліфікаційної роботи _API криптобірж
_____
_____

Перелік питань, що підлягають дослідженню:

1. Які методи збору даних про ринок криптовалют є найбільш придатними для побудови аналітичної системи?
2. Які способи попередньої обробки забезпечують якісну підготовку даних до аналізу та візуалізації?
3. Які інструменти та методи візуалізації найкраще відображають динаміку та закономірності ринку криптовалют?
4. Яким чином інтерактивна візуалізація може підвищити ефективність аналітичної системи та підтримати прийняття управлінських і інвестиційних рішень?

Перелік графічного матеріалу (за потреби) _____
_____

Дата видачі завдання "___01___" _листопада____2024 р.

Керівник магістерської кваліфікаційної роботи _____ _Георгій_БОРОДКІН
(підпис) (ім'я ПРІЗВИЩЕ)

Завдання прийняв до виконання _____ ___Петро-Еммануїл ЩЕРАБН____
(ім'я ПРІЗВИЩЕ)

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API – програмний інтерфейс прикладного програмування

Dash – бібліотека Python для побудови інтерактивних панелей аналітики

DB – база даних

Flask – фреймворк Python для створення веб-додатків

Gradient Boosting – метод послідовного навчання моделей

HTTP/HTTPS – протокол передавання гіпертексту

JSON – формат обміну структурованими даними

ML – машинне навчання

MSE – середньоквадратична похибка

MVC – Model–View–Controller, модель–представлення–контролер

PWA – прогресивний веб-додаток

Plotly – бібліотека для візуалізації даних

PostgreSQL – система керування реляційними базами даних

Random Forest – ансамблевий алгоритм машинного навчання

REST API – архітектурний стиль взаємодії клієнт–сервер

R^2 – коефіцієнт детермінації (показник точності моделі прогнозування)

Supabase – хмарна платформа на основі PostgreSQL

Timestamp – часова мітка, що фіксує момент запису даних

Volatility – показник мінливості цін криптовалют

ВСТУП

Актуальність теми.

Ринок криптовалют є однією з найінноваційніших і водночас найдинамічніших сфер сучасної цифрової економіки. За останнє десятиліття криптовалюти еволюціонували від експериментального фінансового інструменту до глобальної платформи для зберігання вартості, інвестування та електронних транзакцій. Проте така популярність супроводжується високою волатильністю, нестабільністю цінових трендів і значним впливом інформаційного середовища.

Для ефективного аналізу подібних ринків необхідно обробляти великі обсяги даних, що постійно оновлюються, отримані з відкритих джерел — насамперед через програмні інтерфейси (API) криптобірж. Традиційні методи аналітики не здатні своєчасно реагувати на зміни ринку, тому актуальним є створення аналітичної системи, яка автоматизує збір, обробку, аналіз і візуалізацію даних про криптовалютні активи.

Розробка подібної системи дозволить підвищити ефективність прийняття управлінських і інвестиційних рішень, зменшити ризики та своєчасно виявляти ринкові тенденції. Використання сучасних технологій машинного навчання та інтерактивної візуалізації забезпечує нову якість аналітичного підходу до прогнозування динаміки криптовалют.

Об’єкт дослідження — процеси збору, обробки, аналізу та візуалізації даних ринку криптовалют.

Предмет дослідження — методи, алгоритми та програмні засоби побудови аналітичної системи аналізу ринку криптовалют із використанням технологій машинного навчання та інтерактивної візуалізації.

Мета дослідження — розробити аналітичну систему збору, обробки та прогнозування даних ринку криптовалют на основі даних, які можна отримати з відкритих API криптобірж.

Завдання дослідження:

1. Які методи збору даних про ринок криптовалют є найбільш придатними для побудови аналітичної системи?
2. Які способи попередньої обробки забезпечують якісну підготовку даних до аналізу та візуалізації?
3. Які інструменти та методи візуалізації найкраще відображають динаміку та закономірності ринку криптовалют?
4. Яким чином інтерактивна візуалізація може підвищити ефективність аналітичної системи та підтримати прийняття управлінських і інвестиційних рішень?

Методи дослідження.

У роботі застосовано методи системного аналізу, математичної статистики та машинного навчання для обробки великих масивів фінансових даних. Використано алгоритми Random Forest, Gradient Boosting та лінійної регресії для прогнозування цінових показників. Для реалізації програмної частини застосовано мову Python і бібліотеки pandas, NumPy, scikit-learn, Matplotlib, Seaborn і Dash. Створення серверної частини веб-додатку здійснюється за допомогою фреймворку Flask.

Наукова новизна.

У роботі запропоновано концепцію аналітичної системи, яка поєднує автоматизований збір даних з відкритих API криптобірж, алгоритми машинного навчання для прогнозування цінових трендів та інтерактивну візуалізацію результатів. Вперше реалізовано поєднання технологій Flask і Dash для відображення прогнозних моделей у режимі реального часу. Запропоновано методику оцінки точності прогнозування на основі метрик MSE та R^2.

Апробація результатів.

Результати дослідження апробовано на засіданнях кафедри комп’ютерних наук факультету інформаційних технологій НУБіП України та під час підготовки постерної доповіді на конференції «Інформаційні технології: економіка, техніка, освіта — 2025».

Структура роботи.

Магістерська кваліфікаційна робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел та додатків. У першому розділі проведено системний аналіз предметної області ринку криптовалют, у другому — досліджено методи попередньої обробки даних, у третьому — розроблено архітектуру системи та аналітичні модулі, у четвертому — подано результати дослідження й інтерактивну візуалізацію. Загальний обсяг роботи становить 85 сторінок, містить 16 рисунків, 1 таблицю і 23 найменувань у списку використаних джерел, а також оформлені додатки А - В.

РОЗДІЛ 1. СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Аналіз сучасного стану ринку криптовалют

Сучасний ринок криптовалют є одним із найдинамічніших сегментів цифрової економіки, який поєднує інноваційні фінансові інструменти, блокчейн-технології та високий рівень автоматизації процесів обміну даними. Від часу появи першої криптовалюти — Bitcoin 2009 рік — галузь пройшла шлях від вузькоспеціалізованого ентузіазму до масштабної глобальної індустрії з трильйонними оборотами. Сьогодні криптовалютний ринок є не лише фінансовим явищем, а й важливим напрямом наукових, технічних і аналітичних досліджень.[23]

Ключовою особливістю цього ринку є децентралізований характер. На відміну від традиційних валютних систем, криптовалюти не мають єдиного емітента або регулятора, а всі операції фіксуються у розподіленому реєстрі — блокчейні. Це забезпечує прозорість, незмінність та високий рівень захисту даних. Разом з тим, така децентралізація створює й виклики: відсутність централізованого контролю призводить до підвищеної волатильності та залежності курсу від попиту, спекуляцій і зовнішніх інформаційних чинників.

Станом на 2025 рік на ринку функціонує понад 20 тисяч криптовалют і токенів, що обертаються на сотнях бірж — як централізованих, так і децентралізованих. Взаємозв’язки між основними учасниками криптовалютного ринку подано на рисунку 1.1.[22]

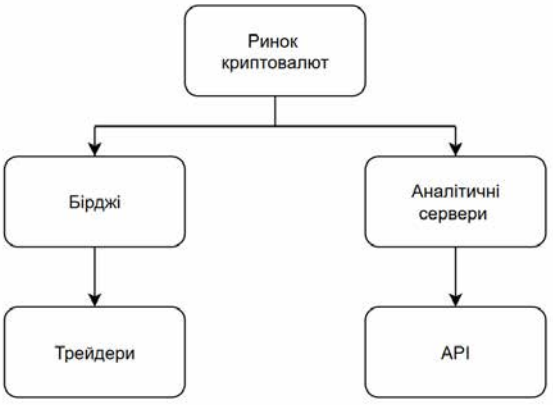

Рис.1.1 Структура ринку криптовалют

Загальна капіталізація ринку за останні роки коливається у межах 1–2 трлн доларів США, а щоденний торговий обсяг сягає 100–200 млрд доларів. Найбільшу частку ринку традиційно займають Bitcoin та Ethereum, які визначають загальні тренди всього ринку.

Водночас структура ринку суттєво ускладнюється. З'являються нові категорії цифрових активів — стейблкоїни, прив'язані до вартості фіатних валют, а також DeFi-токени, що забезпечують роботу смарт-контрактів і фінансових протоколів без участі посередників. Паралельно формується сегмент NFT, який поєднує блокчейн-технології з цифровим мистецтвом і правами власності.

Управління ризиками, прогнозування трендів і своєчасне виявлення змін ринку стають критично важливими задачами. Проте аналіз такого великого, швидкоплинного й неоднорідного масиву даних потребує нових підходів до обробки інформації. Звичайні статистичні методи не справляються з обсягами та швидкістю оновлення ринкових даних, що оновлюються щосекунди з різних джерел.

У відповідь на ці виклики активно розвиваються аналітичні сервіси та інформаційні платформи, що забезпечують моніторинг, обробку та візуалізацію даних про криптовалютний ринок. Найпоширенішими прикладами є CoinMarketCap, CoinGecko, CryptoCompare — вони збирають котирування, обсяги торгів, капіталізацію, а також показники біржової ліквідності. Проте такі сервіси часто обмежуються відображенням статистики без глибокої аналітики або прогнозування. Тому зростає попит на інтелектуальні аналітичні системи, які поєднують машинне навчання, інтерактивну візуалізацію та обробку потокових даних у реальному часі.

Крім фінансових аспектів, ринок криптовалют має значний соціально-економічний вплив. Його розвиток стимулює інновації у сфері фінтеху, створює нові робочі місця в IT-галузі, сприяє появі децентралізованих систем управління, електронного голосування, блокчейн-освіти тощо. Разом з

тим, відсутність регулювання та анонімність транзакцій залишають ризики, пов’язані з шахрайством, відмиванням коштів і ринковими маніпуляціями.

1.2. Джерела даних та особливості API криптобірж

Аналітична система аналізу ринку криптовалют базується на постійному зборі та оновленні великої кількості даних із різних джерел. Від точності, актуальності та повноти цих даних безпосередньо залежить достовірність результатів аналізу та якість прогнозування. На відміну від традиційних фінансових ринків, криптовалютна екосистема є децентралізованою, тому дані формуються у великій кількості незалежних вузлів — криптобірж, блокчейнів, сервісів-агрегаторів і соціальних платформ.

Основні джерела ринкових даних

Криптовалютні біржі

Біржі є основним джерелом достовірних і оперативних даних про курси валютних пар, обсяги торгів, стакани ордерів, ліквідність і глибину ринку.

Провідні біржі — Binance, Coinbase, Kraken, KuCoin, Bitfinex — надають відкриті API для зовнішніх розробників[10], що дозволяють отримувати інформацію як у режимі реального часу, так і ретроспективні історичні дані.

Існують два основних типи API:

- REST API — орієнтований на запити з певною періодичністю. Використовується для отримання історичних цін, 24-годинної статистики, списків активів тощо.[13]
- WebSocket API — забезпечує потік даних у реальному часі, що дає змогу будувати графіки змін ціни, обсягів або ліквідності без затримок.[19]

  Наприклад, Binance API надає доступ до даних про зміни цін і торгові обсяги через запити https://api.binance.com/api/v3/ticker/24hr. Структуру даних, що надходять від API біржі, показано на рисунку 1.2.Ці API дозволяють створювати високоточні моделі ринкових змін та інтегрувати прогностичні алгоритми безпосередньо у веб-систему. Приклад даних показано на рис.1.2.

```
{
  "symbol": "BTCUSDT",
  "priceChange": "3959.68000000",
  "priceChangePercent": "3.874",
  "weightedAvgPrice": "104965.13739766",
  "prevClosePrice": "102211.46000000",
  "lastPrice": "106171.13000000",
  "lastQty": "0.00018000",
  "bidPrice": "106171.12000000",
  "bidQty": "1.19525000",
  "askPrice": "106171.13000000",
  "askQty": "7.85884000",
  "openPrice": "102211.45000000",
  "highPrice": "106670.11000000",
  "lowPrice": "102050.64000000",
  "volume": "22817.16376000",
  "quoteVolume": "2395006729.09338360",
  "openTime": 1762689422013,
  "closeTime": 1762775822013,
  "firstId": 5458713901,
  "lastId": 5462959039,
  "count": 4245139
},
```


Рис. 1.2 Приклад структури даних, що надходять із біржі

Сервіси-агрегатори даних

Агрегатори — це сервіси, які збирають інформацію з десятків бірж і формують узагальнену аналітичну картину ринку.

Найвідоміші з них — CoinMarketCap, CoinGecko, CryptoCompare, Nomics. Вони надають API з даними про ринкову капіталізацію, обсяги торгів, відсоткові зміни, рейтинги монет і бірж. Перевага агрегаторів полягає у нормалізації даних: вони автоматично усереднюють курси з різних майданчиків, згладжуючи розбіжності між біржами. Недолік — нижча частота оновлення, що робить їх менш придатними для високочастотного трейдингу, але зручними для аналітики та прогнозування на макрорівні.

Блокчейн-експлорери

Для детальнішого аналізу мережевої активності використовуються дані безпосередньо з блокчейнів. Сервіси типу Etherscan, BscScan, Blockchain.com API дозволяють отримати відомості про транзакції, гаманці, обсяг переведених коштів, хеші блоків і комісії за операції.

Ці дані є фундаментом для аналізу поведінки користувачів, оцінки активності

мережі та виявлення великих транзакцій (“whale movements”), що часто передують ціновим змінам.

Новинні та соціальні API

Інформаційне поле суттєво впливає на динаміку крипторинку. Для оцінки емоційного тону новин та соціальних обговорень використовуються дані з Twitter, Reddit, Telegram, CoinDesk та інших джерел.

Інтеграція таких потоків у систему дозволяє використовувати аналіз тональності — методику машинного навчання, яка визначає, чи має інформаційний фон позитивний або негативний вплив на очікування учасників ринку.

Технічні особливості використання API

При роботі з відкритими API криптобірж важливо враховувати низку технічних обмежень:

Rate limits — більшість бірж дозволяє виконувати не більше 1–10 запитів за секунду, і перевищення цієї межі може призвести до тимчасового блокування. Для цього використовують кешування даних або асинхронну обробку запитів.

Формати даних — найчастіше використовуються формати JSON або CSV.[19] Для подальшої аналітики дані мають бути уніфіковані за структурою: часові мітки в UTC, ціни — у доларах США, обсяги — у BTC або USDT.

Перевірка достовірності даних — для підвищення надійності доцільно порівнювати інформацію з кількох джерел.

Збереження історії — зібрані дані варто накопичувати в базі даних для побудови моделей машинного навчання, аналізу трендів і побудови графіків.

Таким чином, для ефективного функціонування аналітичної системи необхідно реалізувати комбінований підхід до збору інформації — об’єднати оперативні дані біржових API, зведені статистичні дані агрегаторів і блокчейн-аналітику. Це забезпечує комплексний погляд на стан ринку криптовалют і створює основу для побудови моделей прогнозування та інтерактивної візуалізації, які реалізуються в наступних розділах роботи.

1.3. Огляд існуючих аналітичних систем і рішень

На сучасному етапі розвитку криптовалютного ринку існує значна кількість інформаційних систем, платформ і сервісів, що забезпечують збирання, обробку та візуалізацію даних про стан криптовалют. Їх функціональні можливості охоплюють широкий спектр завдань — від простого моніторингу курсів до складних прогнозних і аналітичних обчислень. Аналіз існуючих рішень дає змогу визначити основні тенденції у побудові аналітичних систем, виявити їх сильні сторони, обмеження та знайти шляхи вдосконалення.

Глобальні аналітичні платформи

Одним із найвідоміших і найпопулярніших сервісів є CoinMarketCap, який з 2013 року збирає дані з сотень бірж і надає користувачам статистику про ціни, капіталізацію, обсяги торгів та динаміку ринку.[22] Платформа дозволяє формувати рейтинги монет, відстежувати зміни цін у реальному часі та порівнювати показники за різними періодами.

Основний недолік цього сервісу — обмежена аналітична глибина: він не реалізує прогностичних моделей та не пропонує алгоритмічної обробки даних.

Подібні можливості надає CoinGecko, який є відкритішим з точки зору доступу до даних. Його API широко використовується в наукових і дослідницьких цілях. CoinGecko пропонує не лише ринкові показники, а й допоміжну інформацію — рейтинг спільнот, обсяг розробки у відкритому коді, аналітику NFT-проєктів тощо. Недоліком є нижча частота оновлення даних і спрощена структура API, що не завжди зручна для побудови систем машинного навчання.

Ще одним популярним агрегатором є CryptoCompare, який пропонує порівняльну аналітику між біржами, валютами та часовими періодами. Його особливістю є підтримка історичних даних за багато років, проте більшість розширених функцій доступні лише через комерційну підписку.

Аналітичні системи для трейдингу та прогнозування

Для технічного аналізу й автоматизації торгівлі використовуються платформи TradingView, Coinigy, 3Commas, CryptoHopper тощо.

TradingView є одним із найпотужніших інструментів візуалізації ринку, що пропонує багатофункціональні графіки, індикатори, скрипти та сповіщення. Проте він орієнтований переважно на ручний аналіз і не дозволяє інтегрувати зовнішні алгоритми прогнозування або кастомні моделі машинного навчання.

Сервіс 3Commas поєднує аналітику з функціями автоматизованої торгівлі. Його недолік — закритість коду та відсутність можливості розширення або кастомізації аналітичних модулів.

CryptoQuant і Glassnode є прикладами більш спеціалізованих платформ, орієнтованих на блокчейн-аналітику. Вони надають метрики мережевої активності, що дозволяє оцінювати “настрій ринку” та поведінку великих гравців. Такі сервіси мають високу аналітичну цінність, але переважно закриті для інтеграції в зовнішні системи й вимагають платного доступу.

Вітчизняні та наукові розробки

У науковій та освітній спільноті України питання побудови аналітичних систем для фінансових ринків досліджується в межах тематики фінансового моніторингу, інтелектуального аналізу даних і систем підтримки прийняття рішень.[7]

Розробки українських науковців часто фокусуються на аналізі фондових ринків або валютних бірж, однак із поширенням криптовалют спостерігається тенденція до адаптації цих підходів для цифрових активів. Основна увага приділяється питанням обробки великих даних, кластеризації ринкових об’єктів, аналізу часових рядів і візуалізації показників у реальному часі.

Однак вітчизняні аналітичні інструменти, як правило, не мають інтеграції з API криптобірж і рідко реалізують інтелектуальні моделі прогнозування — це формує нішу для подальших розробок.

Аналіз і висновки

Порівняльний аналіз існуючих систем дозволяє виокремити основні тенденції: більшість комерційних сервісів обмежуються збором та візуалізацією даних без реалізації інтелектуальних алгоритмів прогнозування;

відкриті сервіси надають API, але мають обмеження у швидкості доступу й кількості запитів;

комплексні рішення з машинним навчанням і динамічною інтерактивною візуалізацією залишаються малодоступними або потребують високих витрат.

Отже, актуальною науково-практичною задачею є розробка власної аналітичної системи, яка об’єднає збір даних з відкритих джерел, інструменти машинного навчання для прогнозування та інтерактивну веб-візуалізацію результатів аналізу.

Це забезпечить підвищення ефективності ринкових досліджень і розширить можливості прийняття інвестиційних рішень у криптовалютній сфері.

1.4. Визначення вимог до аналітичної системи аналізу ринку криптовалют

На основі проведеного системного аналізу предметної області, дослідження джерел даних і огляду існуючих аналітичних рішень визначено основні вимоги до розроблюваної аналітичної системи аналізу ринку криптовалют. Формування вимог є одним із найважливіших етапів проєктування програмного забезпечення, оскільки від цього залежить ефективність, точність і зручність використання майбутньої системи.

Метою створення аналітичної системи є забезпечення автоматизованого збору, обробки, аналізу, прогнозування та візуалізації даних про стан криптовалютного ринку у режимі реального часу. Система повинна надати користувачам — аналітикам, трейдерам, інвесторам та дослідникам — інструмент для моніторингу динаміки криптовалют, виявлення закономірностей і тенденцій, а також для прийняття обґрунтованих управлінських та інвестиційних рішень.

Функціонально система має забезпечувати можливість підключення до відкритих API криптобірж, таких як Binance, Coinbase, Kraken або CoinGecko, для отримання актуальної інформації про котирування валют, обсяги торгів,

ринкову капіталізацію та історичні дані. Дані повинні надходити як у вигляді періодичних запитів через REST API, так і в режимі реального часу за допомогою WebSocket-з'єднання. Крім збору даних, система повинна реалізовувати їх обробку — очищення, нормалізацію, перевірку достовірності та уніфікацію форматів для подальшого збереження у базі даних. Доцільним є використання реляційної бази даних, наприклад PostgreSQL або Supabase, у якій фіксуватимуться курси, обсяги, часові мітки, біржа-джерело та інші параметри.

У системі передбачено створення аналітичного модуля, який виконуватиме статистичну обробку даних і реалізовуватиме алгоритми машинного навчання для прогнозування цінових трендів. Зокрема, використовуватимуться моделі Random Forest, Gradient Boosting або лінійна регресія. Оцінювання точності прогнозів відбуватиметься на основі статистичних метрик середньоквадратичної помилки (MSE) та коефіцієнта детермінації (R^2). Це дозволить визначити ефективність аналітичних моделей і підвищити достовірність прогнозів.

Особливу увагу приділено модулю візуалізації, який має забезпечувати наочне відображення динаміки ринку у вигляді графіків, гістограм, теплових карт і трендових ліній. Для побудови візуалізацій використовуватимуться бібліотеки Matplotlib, Seaborn та Dash. Передбачається можливість інтерактивної взаємодії користувача з панеллю (dashboard), де можна вибирати криптовалюту, часовий період, переглядати статистичні показники та порівнювати активи між собою. Інтерфейс системи має бути реалізований як веб-додаток з адаптивним дизайном і можливістю роботи на різних пристроях, зокрема як прогресивний веб-додаток (PWA).

Технічна реалізація системи передбачає використання мови програмування Python та сучасного веб-фреймворку Flask для створення серверної частини. Веб-додаток взаємодіятиме з клієнтською частиною через REST API, що забезпечить розширюваність і можливість інтеграції з іншими аналітичними сервісами. Система має підтримувати надійне з'єднання з API криптобірж, стабільну роботу при високих навантаженнях та мінімальний час

відгуку при оновленні даних. Для захисту даних передбачено використання HTTPS-протоколу та токенної автентифікації.

Усі дані, що збираються системою, повинні бути структурованими та синхронізованими за часовими мітками, що дозволяє уникнути похибок у побудові графіків і прогнозів. Для забезпечення достовірності інформації рекомендується використовувати кросперевірку показників із різних джерел. Крім того, система має підтримувати функції архівування даних і резервного копіювання, що забезпечить збереження історичних записів і можливість повторного аналізу у майбутньому.

1.5. Постановка завдання магістерського дослідження

Результати проведеного системного аналізу предметної області показали, що, попри велику кількість відкритих джерел криптовалютних даних і наявність окремих аналітичних сервісів, відсутні комплексні рішення, які б поєднували автоматизований збір, обробку, прогнозування та інтерактивну візуалізацію інформації у єдиній системі. Це створює потребу у розробці інтегрованої аналітичної платформи, що дозволить користувачам ефективно аналізувати ринок криптовалют та своєчасно реагувати на його зміни.

Проблема, яку вирішує магістерське дослідження, полягає у створенні програмного засобу, здатного забезпечити збір та узгодження даних із різних API криптобірж, проведення попередньої обробки та їх подальший аналіз за допомогою методів машинного навчання. Особливу увагу приділено реалізації модуля прогнозування, який дозволяє виявляти закономірності у часових рядах і передбачати можливі зміни ринкових показників. Важливим компонентом є модуль інтерактивної візуалізації, що надає користувачам можливість динамічно відстежувати ринкові тенденції через веб-інтерфейс.

Метою магістерського дослідження є розробка аналітичної системи аналізу ринку криптовалют, яка автоматизує процес збору даних з відкритих джерел, виконує їх попередню обробку, здійснює аналітичну оцінку ринкових

показників, прогнозує динаміку вартості активів і забезпечує інтерактивне відображення результатів у зручному для користувача форматі.

Для досягнення поставленої мети необхідно виконати такі завдання:

1. визначити ефективні методи збору й структурування даних криптовалютного ринку на основі відкритих API;
2. розробити алгоритми попередньої обробки даних для усунення похибок, пропусків і дублікатів;
3. дослідити можливості застосування методів машинного навчання для аналізу часових рядів і прогнозування цінових трендів;
4. створити модуль інтерактивної візуалізації, який відображатиме результати аналізу та прогнозів у режимі реального часу;
5. забезпечити зберігання результатів аналітики у структурованій базі даних і підтримку розширення системи в майбутньому.

Об’єктом дослідження є процеси збору, обробки, аналізу, прогнозування та візуалізації даних ринку криптовалют.

Предметом дослідження — методи, алгоритми та програмні засоби побудови аналітичної системи аналізу ринку криптовалют із використанням технологій машинного навчання, статистичної аналітики та інтерактивної візуалізації.

Магістерське дослідження має прикладний характер і спрямоване на розробку практичного інструменту, який поєднує сучасні підходи до аналітики великих даних із програмними технологіями веб-візуалізації. Реалізація системи забезпечить підвищення ефективності обробки криптовалютних даних, покращення точності прогнозів і створення зручного засобу підтримки інвестиційних та управлінських рішень у сфері цифрової економіки.

Висновки до розділу 1.

У першому розділі проведено системний аналіз предметної області ринку криптовалют, досліджено джерела отримання ринкових даних, проаналізовано існуючі аналітичні системи та визначено основні вимоги до створення власної аналітичної платформи. Встановлено, що ринок криптовалют характеризується високою динамічністю, значним обсягом інформаційних потоків і відсутністю централізованих механізмів регулювання, що ускладнює процес аналітики та прогнозування. Проаналізовано основні джерела даних, серед яких відкриті API криптобірж, агрегатори, блокчейн-експлорери, а також виявлено їхні переваги та недоліки.

Порівняння існуючих інформаційних систем показало, що більшість із них орієнтовані лише на збір і відображення статистики без використання методів машинного навчання та інтерактивної аналітики. Це підтверджує актуальність створення комплексної системи, здатної забезпечити автоматизований збір, попередню обробку, аналіз, прогнозування й візуалізацію даних криптовалютного ринку. Визначені у розділі вимоги сформували концептуальну основу для побудови моделі аналітичної системи, що стане предметом дослідження у наступному розділі.

РОЗДІЛ 2. МОДЕЛЮВАННЯ СИСТЕМИ

2.1. Побудова концептуальної моделі системи

На основі проведеного системного аналізу предметної області сформульовано загальну задачу магістерського дослідження, яка полягає у створенні аналітичної системи, здатної автоматично збирати, обробляти, аналізувати, прогнозувати та візуалізувати дані криптовалютного ринку в режимі реального часу. Система повинна забезпечувати користувача актуальною інформацією про динаміку вартості криптоактивів, ринкові обсяги та ключові індикатори, що впливають на прийняття інвестиційних і управлінських рішень.

Задача розроблення аналітичної системи включає декілька взаємопов’язаних підзадач:

реалізацію механізмів збору даних із відкритих API криптобірж виконання попередньої обробки даних — очищення, нормалізації, перевірки достовірності та формування єдиної структури з часовими мітками;

створення алгоритмічного модуля для аналізу часових рядів і прогнозування тенденцій зміни цін із використанням методів машинного навчання;

забезпечення інтерактивної візуалізації результатів аналізу у вигляді графіків, таблиць, індикаторів і панелей;

реалізацію користувацького інтерфейсу, який дозволяє взаємодіяти із системою через веб-додаток або браузер.

Концептуальна модель аналітичної системи передбачає побудову п’яти основних компонентів, між якими здійснюється безперервний обмін інформацією:

Модуль збору даних (Data Collector) — відповідає за підключення до зовнішніх джерел через REST і WebSocket API, періодичне оновлення даних і передачу їх у сховище.

База даних (Database Layer) — забезпечує збереження ринкових показників, історичних даних і результатів аналітики у структурованому вигляді.

Модуль попередньої обробки (Preprocessing Engine) — виконує очищення, нормалізацію, узгодження часових форматів і перевірку даних на повноту.

Аналітичний модуль (Analytics Engine) — реалізує алгоритми статистичного аналізу та прогнозування (Random Forest, Gradient Boosting)[6], а також оцінює точність прогнозів за метриками MSE та R^2.

Модуль візуалізації (Visualization Layer) — відповідає за графічне відображення даних у вигляді інтерактивних графіків і панелей, створених за допомогою бібліотек Dash і Plotly. Табл.2.1

Таблиця 2.1

Основні компоненти концептуальної моделі аналітичної системи.

Компонент	Основні функції	Технології реалізації
Модуль збору даних (Data Collector)	Отримання інформації з API бірж, оновлення даних у реальному часі	Python, requests, WebSocket
База даних (Database Layer)	Зберігання ринкових показників і результатів аналізу	PostgreSQL / Supabase
Модуль попередньої обробки (Preprocessing Engine)	Очищення, нормалізація, перевірка повноти	andas, NumPy
Аналітичний модуль (Analytics Engine)	Побудова моделей ML, прогнозування трендів, оцінка точності	scikit-learn, Gradient Boosting
Модуль візуалізації (Visualization Layer)	Інтерактивні графіки, панелі, оновлення у реальному часі	Dash, Plotly

Взаємозв’язок між компонентами системи можна описати так:
дані з API криптобірж надходять до модуля збору, далі зберігаються у базі даних, проходять попередню обробку та аналізуються аналітичним модулем, результати якого відображаються користувачеві у вигляді інтерактивних елементів веб-додатку. Таким чином, система працює у циклі «збір → обробка → аналіз → візуалізація → рішення», що відповідає принципам замкненого аналітичного контуру.

При проєктуванні концептуальної моделі було враховано такі вимоги:
забезпечення масштабованості та модульності архітектури;
можливість інтеграції з додатковими джерелами даних у майбутньому;
забезпечення швидкого доступу до актуальної інформації;
мінімізація затримок при оновленні даних у реальному часі;
забезпечення безпеки та цілісності інформації.

Модель системи передбачає використання архітектури клієнт–сервер, де серверна частина виконує аналітичні обчислення та формує API-запити, а клієнтська частина реалізує інтерфейс користувача з інтерактивними елементами. Такий підхід забезпечує простоту розгортання, можливість роботи через веб-браузер та незалежність від операційної системи користувача.

Отже, концептуальна модель аналітичної системи аналізу ринку криптовалют поєднує в собі програмні засоби збору, обробки та аналізу даних, а також візуальні механізми їх представлення. Вона забезпечує цілісний процес інформаційної аналітики та створює основу для подальшого моделювання логічної структури й розробки архітектури системи, що буде розглянуто у наступних підрозділах.

2.2. Інформаційна модель системи

Інформаційна модель аналітичної системи аналізу ринку криптовалют визначає логічну структуру даних, способи їх організації, зберігання та взаємозв’язки між основними сутностями. Побудова чіткої інформаційної моделі є критично важливою умовою ефективної роботи системи, адже вона

забезпечує узгодженість інформації, швидкий доступ до необхідних показників і стабільність обробки великих обсягів даних у реальному часі.

У розроблюваній системі інформаційна модель побудована на основі реляційної бази даних, що забезпечує цілісність, надійність і зручність виконання складних аналітичних запитів. Як середовище зберігання даних обрано PostgreSQL[16], оскільки воно підтримує роботу з великими масивами даних, має розвинені механізми індексації, підтримує роботу з JSON-структурами та є повністю сумісним із хмарною платформою Supabase[17], що дозволяє легко розгорнути систему у веб-середовищі.

Основні сутності бази даних

Інформаційна модель системи включає кілька логічно взаємопов’язаних таблиць, які забезпечують повний цикл роботи з ринковими даними — від збору до збереження прогнозних результатів.

Центральним елементом є таблиця Cryptocurrencies, у якій зберігається основна довідкова інформація про криптовалюти.

Її атрибути:

crypto_id — унікальний ідентифікатор;

symbol — коротке позначення валюти;

name — повна назва активу;

description — короткий опис або технічна характеристика;

created_at — дата додавання до бази.

Таблиця використовується як базова для побудови зв’язків з іншими сутностями.

Друга сутність — Exchanges — містить інформацію про біржі, з яких система отримує дані.

Основні поля: exchange_id, name, api_url, country, access_type , active, last_sync.

Це дозволяє зберігати метадані про джерела інформації та контролювати їх доступність.

Таблиця MarketData є ключовою у функціонуванні системи, адже саме в ній зберігаються історичні та поточні ринкові дані.

Вона має атрибути:

market_id, crypto_id, exchange_id, timestamp, open_price, close_price, min_price, max_price, volume, market_cap.

Кожен запис у таблиці відповідає певному часовому інтервалу — від хвилини до доби — і фіксує зміни ринку у вибраний момент часу.

Ця таблиця є основою для побудови часових рядів, статистичного аналізу та подальшого прогнозування.

Наступна сутність — Predictions — призначена для зберігання результатів прогнозів, отриманих за допомогою алгоритмів машинного навчання.

Її структура включає поля:

prediction_id, crypto_id, algorithm_name, predicted_price, prediction_date, target_date, mse_score, r2_score.

Це дозволяє фіксувати, який саме метод використовувався, коли сформовано прогноз і наскільки він виявився точним.

Завдяки цьому можна відстежувати якість моделей і обирати найбільш ефективні підходи для подальшого використання.

Для управління доступом користувачів створюється таблиця Users, яка містить поля user_id, login, email, password_hash, role, created_at.

Передбачено три типи ролей: admin, analyst, viewer.

Адміністратор має право додавати нові джерела даних і змінювати налаштування, аналітик може створювати прогнози й генерувати звіти, а звичайний користувач має лише перегляд аналітичної панелі.

Для моніторингу роботи системи передбачено таблицю Logs, у якій фіксуються події, пов’язані з виконанням запитів до API: час звернення, кількість отриманих записів, статус відповіді, затримка в мілісекундах і можливі помилки. Такі дані використовуються для діагностики та підвищення надійності роботи системи.

Логічна структура та зв’язки між таблицями

Усі сутності пов’язані між собою за допомогою відношень типу «один-до-багатьох».Рис. 2.1

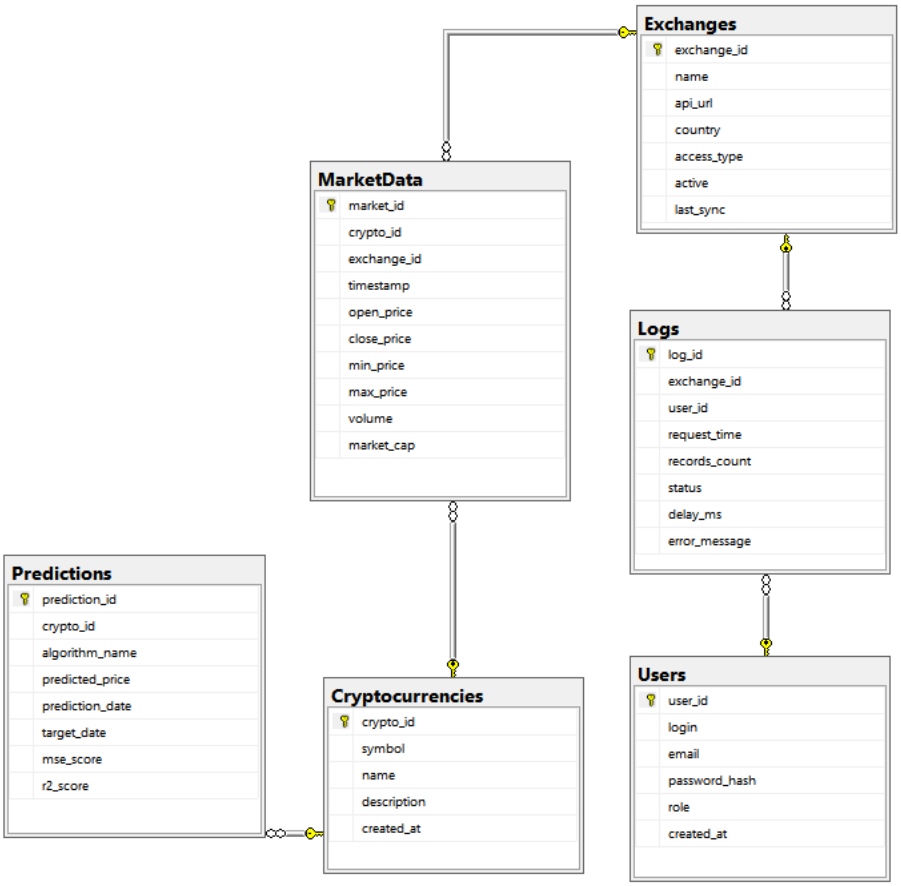

Рис. 2.1 ER-діаграма бази даних аналітичної системи аналізу ринку криптовалют

Наприклад, одна криптовалюта з таблиці Cryptocurrencies може бути представлена на багатьох біржах, а кожна біржа — містити тисячі записів у MarketData.

Аналогічно, кожна валюта може мати багато прогнозів у таблиці Predictions.

Ця логіка забезпечує гнучкість структури даних і дозволяє масштабувати систему без втрати узгодженості.

Така структура дозволяє здійснювати складні SQL-запити, наприклад: “вивести середню ціну Bitcoin за останній тиждень на біржі Binance та порівняти з прогнозними значеннями моделі Random Forest”.

Нормалізація та цілісність даних

База даних спроєктована у третій нормальній формі, що запобігає дублюванню інформації та знижує ризик помилок при оновленні записів.

Усі зовнішні ключі мають умови цілісності, що гарантує правильність даних при видаленні або зміні записів.

Для підвищення швидкодії реалізовано індекси за полями crypto_id, exchange_id, timestamp, що особливо важливо при виконанні часових запитів.

Дані, отримані з API криптобірж, зберігаються у базі уніфіковано — усі часові мітки переводяться у формат UTC, а числові значення зберігаються з точністю до восьми десяткових знаків.

Перед записом у базу проводиться перевірка структури JSON-відповіді, що дозволяє виявляти пошкоджені або неповні пакети даних.

Масштабування та розширення моделі

Інформаційна модель системи спроєктована з урахуванням можливості подальшого розширення.

У майбутньому до бази даних можуть бути додані таблиці для зберігання:

соціальних метрик;

новинних повідомлень із тематичних джерел;

аналітичних індикаторів.

Також передбачено можливість підключення модулів кешування та систем резервного копіювання для підвищення надійності зберігання[16].

Таким чином, розроблена інформаційна модель забезпечує логічну узгодженість між основними сутностями системи та створює надійну основу для аналітичних обчислень. Вона охоплює повний цикл роботи з даними — від моменту збору інформації через API до збереження результатів прогнозування. Реалізація цієї моделі у базі даних PostgreSQL забезпечує високу швидкодію, масштабованість і можливість подальшого розвитку системи, що робить її ефективною основою для функціонування аналітичної платформи аналізу ринку криптовалют.

2.3. Функціональна модель системи

Функціональна модель аналітичної системи аналізу ринку криптовалют відображає взаємозв'язки між основними компонентами, послідовність обробки даних і логіку виконання процесів. Основною метою побудови цієї моделі є формалізація функцій системи та визначення способів їх реалізації, що дозволяє описати повний життєвий цикл даних — від моменту надходження з зовнішніх джерел до представлення результатів користувачеві.

Аналітична система функціонує як комплекс взаємопов'язаних модулів, які реалізують окремі етапи обробки інформації. До складу системи входять такі функціональні підсистеми: модуль збору даних (Data Collector), модуль попередньої обробки (Preprocessing Engine), аналітичний модуль (Analytics Engine), модуль візуалізації (Visualization Layer) та користувацький інтерфейс (User Interface).Рис.2.2

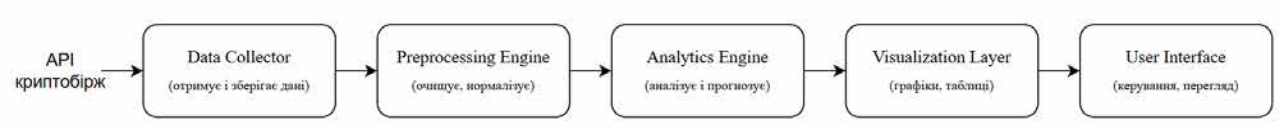

Рис. 2.2 Функціональна блок-схема обробки даних в аналітичній системі аналізу ринку криптовалют

Взаємодія між цими підсистемами забезпечується за допомогою єдиного інформаційного середовища — бази даних і внутрішніх API-запитів.

Функціональні компоненти системи

1. Модуль збору даних (Data Collector).

Цей компонент відповідає за отримання ринкової інформації з відкритих API криптобірж, агрегаторів та інших джерел. Збір даних здійснюється двома способами:

через REST API — періодичні запити до серверів для отримання історичних або добових даних;

через WebSocket API — постійне з'єднання, що дозволяє отримувати оновлення цін у режимі реального часу.

Data Collector формує уніфіковані запити, перевіряє коректність відповіді та передає отриману інформацію до бази даних. Для оптимізації процесу застосовується асинхронна обробка запитів, що дозволяє системі працювати паралельно з кількома біржами.

Додатково модуль включає механізми логування — фіксує час отримання даних, обсяг записів і статус запитів, що полегшує моніторинг роботи системи.

2. Модуль попередньої обробки (Preprocessing Engine).

Після надходження даних до бази цей модуль виконує їх очищення та перетворення до єдиного формату.

Основні етапи попередньої обробки включають:

очищення — видалення дублікатів і некоректних записів, заміна пропусків середніми значеннями;

нормалізацію — приведення цін до єдиної валюти та часових міток до UTC;

перевірку узгодженості — порівняння з альтернативними джерелами;

збереження підготовленого датасету у вигляді таблиці, готової для аналітичної обробки.

Модуль реалізований мовою Python із використанням бібліотек pandas та NumPy, що забезпечує швидке опрацювання великих обсягів ринкових даних і їх підготовку до подальшого аналізу.

3. Аналітичний модуль (Analytics Engine).

Цей компонент є центральним елементом системи. Його головна функція — проведення статистичного аналізу часових рядів і побудова моделей прогнозування.

Аналітичний модуль реалізує:

розрахунок базових статистичних показників;

кластеризацію ринкових активів для виявлення груп із подібною поведінкою;

побудову прогнозів на основі алгоритмів машинного навчання, зокрема Random Forest Regressor, Gradient Boosting і Linear Regression.

Для навчання моделей дані поділяються на тренувальні та тестові вибірки. Результати прогнозів зберігаються в таблиці Predictions, а точність оцінюється

за допомогою метрик MSE та R^2.

Аналітичний модуль взаємодіє безпосередньо з базою даних, отримуючи агреговані значення з MarketData і передаючи результати у Predictions.

4. Модуль візуалізації (Visualization Layer).

Модуль відповідає за наочне відображення результатів аналізу у вигляді інтерактивних елементів. Для реалізації використано бібліотеки Dash та Plotly, що дозволяють створювати динамічні графіки, гістограми, теплові карти та діаграми розсіювання.

Користувач може взаємодіяти з інтерфейсом, обираючи валюту, період аналізу або конкретну модель прогнозування. Модуль підтримує оновлення даних у реальному часі, що забезпечує актуальність інформації на панелі.

5. Користувацький інтерфейс (User Interface).

Інтерфейс є фронтенд-частиною системи та реалізований як веб-додаток на основі Flask і Dash. Він забезпечує інтерактивну взаємодію користувача із системою, відображає результати аналітики та дозволяє керувати параметрами відображення.

Доступ користувачів до системи здійснюється через систему автентифікації, реалізовану на основі токенів доступу.

Послідовність обробки даних

Робота системи здійснюється послідовно у кілька етапів:

Отримання даних. Система підключається до зовнішніх API криптобірж і завантажує актуальні курси валют, обсяги торгів і капіталізацію.

Обробка та нормалізація. Дані проходять очищення, перевірку на повноту й узгодження часових міток.

Збереження. Результати обробки зберігаються у базі даних у структурованому вигляді.

Аналітична обробка. Зібрані дані аналізуються, обчислюються статистичні показники, формується прогноз.

Візуалізація. Підготовлені результати відображаються у вигляді інтерактивних графіків на веб-панелі користувача.

Ця схема забезпечує замкнений цикл функціонування системи — від збору до прийняття рішень, що відповідає принципам сучасних аналітичних платформ типу Business Intelligence (BI).

Взаємодія модулів

Обмін інформацією між модулями здійснюється через внутрішні REST-запити або безпосередній доступ до бази даних. Кожен модуль виконує власну функцію, але тісно інтегрований з іншими, утворюючи єдину архітектуру.

Модуль збору передає дані у сховище, звідки їх отримує аналітичний модуль. Результати аналізу надходять у модуль візуалізації, який формує кінцеве представлення даних у вигляді графічних елементів.

Таким чином, система функціонує як цілісний механізм, у якому кожен компонент забезпечує послідовний етап інформаційної обробки.

2.4. Архітектура аналітичної системи

Архітектура аналітичної системи аналізу ринку криптовалют визначає її логічну та технічну структуру, принципи взаємодії компонентів, шари програмної реалізації та способи передачі даних між ними. Вона побудована за принципом багаторівневої клієнт–серверної системи, що забезпечує гнучкість, масштабованість і розширюваність функціоналу.

Запроєктована система спирається на тришаровий архітектурний стиль, який містить:

- користувацький рівень (шар представлення) (Presentation Layer)
- шар обробки задач, який розміщується на сервері (Server Layer)
- шар даних (Data Layer).

Така архітектура дозволяє ізолювати логіку роботи від інтерфейсу користувача, забезпечуючи стабільність і простоту супроводу коду.

1. Презентаційний рівень (Presentation Layer)

Рівень представлення реалізує інтерактивний веб-інтерфейс користувача, створений на основі бібліотеки Dash.

Цей рівень дозволяє візуалізувати аналітичні дані у вигляді графіків, гістограм, теплових карт, таблиць і прогнозних кривих.

Інтерфейс побудований за принципами динамічного оновлення: при зміні параметрів або виборі іншої криптовалюти відбувається миттєве перерахування даних і оновлення графічних елементів.

Інтерфейс має адаптивний дизайн і може функціонувати як PWA, що дозволяє користувачеві додати іконку системи на головний екран мобільного пристрою.

Це створює ефект повноцінного застосунку, хоча система фактично працює через веб-браузер. Рівень представлення також дозволяє експортувати результати аналітики у форматах CSV, PDF або PNG для створення звітів і презентацій.

2. Серверний рівень (Server Layer)

Серверна частина є ядром аналітичної системи та відповідає за логіку обробки даних, взаємодію з API криптобірж, виконання аналітичних розрахунків і формування відповіді для клієнтської частини.

Реалізація здійснена на основі легкого веб-фреймворку Flask, який забезпечує високу продуктивність і просту інтеграцію з бібліотеками Python.

На серверному рівні реалізовано такі ключові компоненти:

- модуль збору даних — отримує інформацію з REST та WebSocket API бірж і записує її у базу;
- модуль попередньої обробки — виконує фільтрацію, нормалізацію, конвертацію валют та синхронізацію часових міток;
- модуль аналітики — запускає обчислення статистичних показників і моделі машинного навчання (Random Forest, Gradient Boosting);
- модуль прогнозування — створює нові записи у таблиці Predictions та зберігає точність моделей;
- REST API — надає клієнтській частині доступ до результатів аналізу через HTTP-запити у форматі JSON.

Важливою особливістю серверного шару є підтримка асинхронної обробки даних — система може одночасно працювати з кількома API, що дозволяє оновлювати інформацію в реальному часі без перевантаження.

Для стабільності використовується кешування відповідей і черги повідомлень, які гарантують безперервність обміну навіть при тимчасових збоях у з'єднанні з біржами.

Серверна частина також виконує функції безпеки: авторизацію користувачів, перевірку прав доступу, токенну аутентифікацію, а також шифрування переданих даних за протоколом HTTPS.

3. Рівень даних (Data Layer)

На рівні даних зосереджені всі процеси, пов'язані зі зберіганням, оновленням і структуризацією інформації.

Основним компонентом цього шару є реляційна база даних PostgreSQL, у якій реалізовано всі сутності, описані в інформаційній моделі .

База даних зберігає як історичні, так і поточні ринкові дані, результати прогнозів та аналітичних розрахунків.

Для забезпечення цілісності й узгодженості даних використовується система зовнішніх ключів та індексації. Передбачено періодичне оновлення таблиць через планувальник завдань, який викликає модуль збору даних у задані інтервали часу. Архітектура бази даних дозволяє підключення до неї через ORM-бібліотеку[13], що забезпечує безпечний обмін даними між Python-додатком і PostgreSQL без прямого виконання SQL-запитів.

Для підвищення надійності сховище може бути розгорнуте у хмарі — наприклад, у середовищі Supabase, яке підтримує автоматичне резервне копіювання, масштабування та API-доступ до таблиць.

Таким чином, рівень даних забезпечує стабільну основу для функціонування аналітичної системи, формуючи єдине джерело правдивих і узгоджених даних.

4. Взаємодія між рівнями

Компоненти трьох рівнів взаємодіють між собою через стандартизовані канали зв’язку.

Формально ця взаємодія виглядає так:

Frontend надсилає запит через REST API до серверної частини Flask.

Flask-сервер обробляє запит, отримує необхідні дані з бази PostgreSQL через ORM і виконує обчислення в аналітичному модулі.

Отримані результати передаються назад у форматі JSON і відображаються користувачеві у вигляді інтерактивної візуалізації.

При зміні умов запит виконується повторно, і система оновлює графіки без перезавантаження сторінки.

Такий підхід забезпечує розподіл обчислень між сервером і клієнтом, що підвищує швидкодію системи та спрощує масштабування.

Усі обчислення з машинного навчання виконуються на серверному рівні, тоді як візуалізація та інтерфейс реалізовані на клієнтській стороні. Схема подана на рис.2.3.

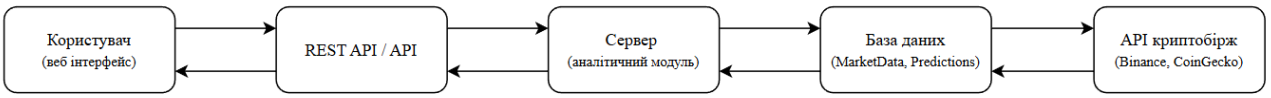

Рис. 2.3 Логічна схема взаємодії між компонентами аналітичної системи аналізу ринку криптовалют

5. Особливості архітектури

Запропонована архітектура має низку переваг:

модульність — кожен компонент системи може оновлюватися або замінюватися без зміни решти структури;

масштабованість — можливість підключення додаткових джерел даних або розширення бази без втрати продуктивності;

портативність — реалізація на Python дозволяє розгортати систему на будь-якій операційній системі;

надійність і безпека — централізована обробка даних на сервері з контролем доступу;

реальний час — використання WebSocket-з’єднань забезпечує миттєве оновлення графіків і показників.

2.5. Алгоритмічна модель системи

Алгоритмічна модель описує логіку функціонування аналітичної системи аналізу ринку криптовалют, послідовність виконання основних процесів та взаємодію між програмними модулями.

Побудова такої моделі дозволяє формалізувати поведінку системи та забезпечити узгодженість усіх етапів — від збору інформації до відображення результатів користувачеві.

Розроблювана система реалізує замкнений цикл обробки даних, який складається з п’яти основних етапів:

отримання ринкових даних із відкритих джерел, попередньої обробки, аналітичного моделювання та прогнозування, оцінки точності та інтерактивної візуалізації.

Кожен етап має власні підпроцеси, що реалізовані у відповідних модулях програмної системи.

Етап 1. Отримання даних

Робота системи починається з автоматичного підключення до відкритих API криптобірж. Загальну схему процесу отримання ринкових даних наведено на Рис. 2.4.Система надсилає запити у форматі REST для отримання історичних даних або встановлює WebSocket-з’єднання для потокового оновлення інформації в реальному часі.

На цьому етапі визначаються параметри запиту — валютна пара, часовий інтервал, кількість записів.

Після отримання відповіді у форматі JSON дані проходять первинну перевірку на коректність структури. Якщо запит не виконався або API недоступне, система автоматично повторює запит із часовою затримкою. Зібрані дані тимчасово зберігаються у буфері та передаються на етап попередньої обробки.

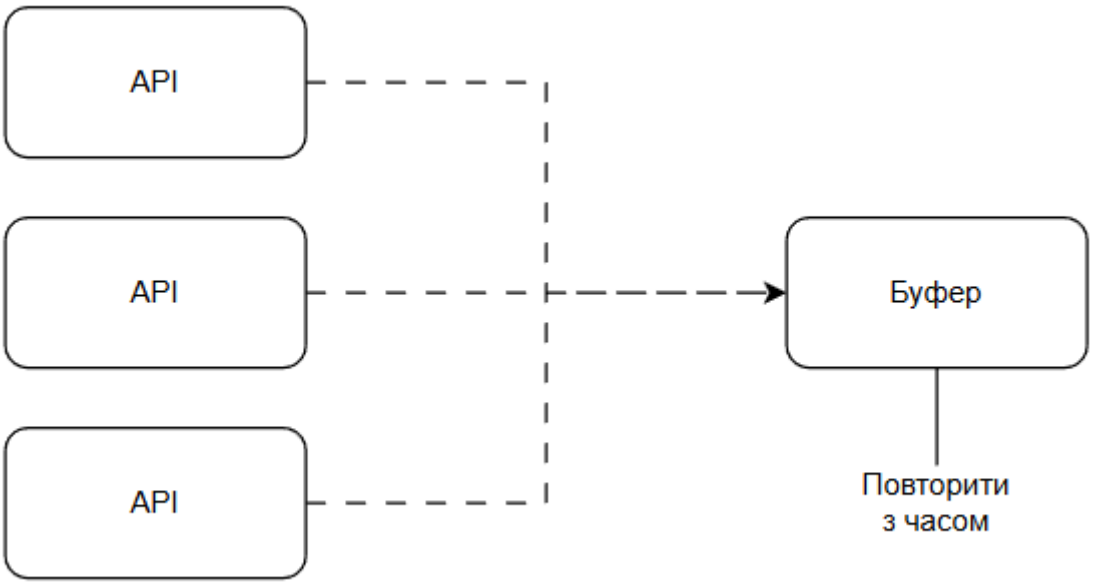

Рис. 2.4 Схема отримання ринкових даних із відкритих API криптобірж

Етап 2. Попередня обробка даних

Після отримання даних із різних джерел важливо забезпечити їх узгодженість. Послідовність дій під час попередньої обробки показано на рис. 2.5. Модуль попередньої обробки виконує очищення від дублікатів, перевірку на наявність пропусків, усунення аномальних значень та приведення даних до єдиного формату.

Усі часові мітки конвертуються у формат UTC, а числові значення нормалізуються відповідно до єдиної шкали.

Система також виконує агрегацію даних — наприклад, об’єднання щохвилинних записів у годинні або добові інтервали для побудови стабільних часових рядів. На цьому етапі формується структурований набір даних, який зберігається у базі даних PostgreSQL і використовується для подальшого аналізу.

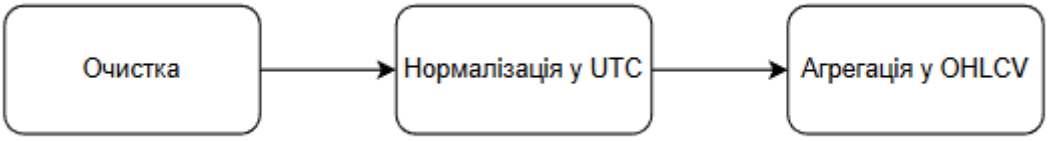

Рис. 2.5 Процес попередньої обробки ринкових даних

Етап 3. Аналітичне моделювання та прогнозування

Ключовим елементом роботи системи є аналітичний модуль, який виконує статистичну обробку та прогнозування тенденцій зміни цін.

На цьому етапі відбувається аналіз часових рядів ринку криптовалют, виявлення закономірностей і формування прогнозів на основі методів машинного навчання.Схему процесу оцінювання точності моделей наведено на Рис. 2.6. Після підготовки даних система виділяє тренувальну й тестову вибірки.

Для прогнозування застосовуються моделі, засновані на ансамблевих алгоритмах (Random Forest, Gradient Boosting), які враховують складні взаємозв’язки між параметрами ринку.

Модуль обчислює прогнозні значення ціни для вибраного активу, а також показники точності кожної моделі.Отримані результати зберігаються в таблиці Predictions бази даних, що дозволяє відстежувати історію прогнозів і порівнювати ефективність різних алгоритмів.

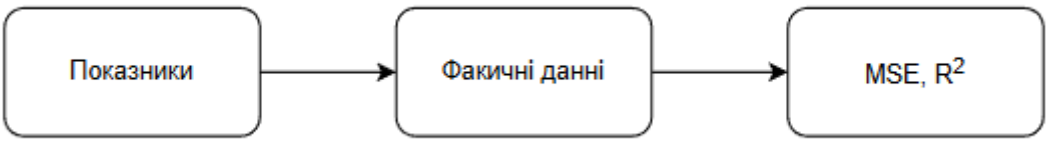

Рис. 2.6 Оцінювання точності моделей прогнозування

Етап 4. Оцінка точності та адаптація моделей

Після отримання прогнозних результатів система порівнює їх з фактичними даними ринку.Послідовність цього процесу показано на Рис. 2.7.

Це дозволяє оцінити точність роботи моделей і за потреби виконати повторне перенавчання.

Алгоритм адаптації забезпечує поступове вдосконалення прогнозів у міру накопичення нових даних.

У процесі роботи система фіксує показники точності у базі, а також веде журнал роботи, де реєструються помилки та зміни в параметрах моделей.

Таким чином, система має властивості самонавчання, що дозволяє їй підтримувати актуальність і точність результатів у динамічному ринковому середовищі.

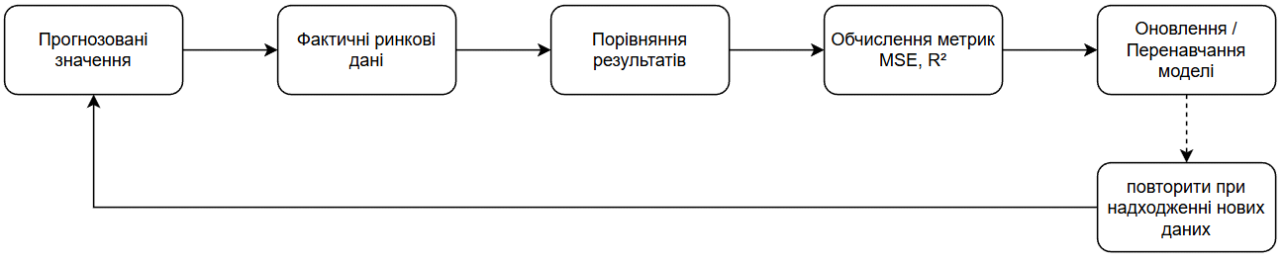

Рис. 2.7 Цикл оцінки точності та адаптації моделей прогнозування

Етап 5. Візуалізація результатів

Заключним етапом є подання результатів аналізу у зручній формі. Модуль візуалізації відповідає за створення інтерактивної панелі, де користувач може спостерігати зміни цін, обсяги торгів, графіки трендів і прогнозні лінії. Інтерфейс реалізовано на основі бібліотеки Dash, що дозволяє взаємодіяти з графіками в реальному часі — змінювати валюту, часовий проміжок, масштаб чи порівнювати кілька активів одночасно.

Оновлення даних відбувається автоматично завдяки інтеграції з WebSocket API. Користувач отримує актуальну інформацію про стан ринку, що дозволяє оперативно реагувати на зміни та приймати обґрунтовані рішення.

Узагальнення алгоритмічної структури

Робота системи відбувається за послідовністю:

Ініціалізація параметрів і підключення до API.

Збір даних із зовнішніх джерел.

Попередня обробка, нормалізація й агрегація.

Формування набору даних для моделювання.

Запуск аналітичних алгоритмів і прогнозування.

Збереження результатів і оцінка точності.

Візуалізація результатів у реальному часі.

Така послідовність формує єдиний аналітичний контур — від отримання інформації до її практичного використання.

Алгоритмічна модель забезпечує системність роботи, автоматизацію обробки великих обсягів даних та стабільну інтеграцію між усіма модулями системи. Завдяки цьому розроблена система дозволяє ефективно аналізувати динаміку

ринку криптовалют і формувати прогнозні висновки з високим рівнем достовірності.

Висновки до розділу 2

У другому розділі магістерської роботи розроблено та обґрунтовано комплекс моделей, які визначають структуру та принципи функціонування аналітичної системи аналізу ринку криптовалют.

Побудовано концептуальну, інформаційну, функціональну, архітектурну та алгоритмічну моделі, які в сукупності описують логіку роботи системи, взаємозв’язки між її компонентами та послідовність обробки даних.

У ході моделювання сформовано багаторівневу архітектуру клієнт–серверного типу, що включає рівень даних, серверну частину для обробки та аналізу інформації й рівень представлення для інтерактивної візуалізації результатів.

Розроблена інформаційна модель описує структуру бази даних, що забезпечує узгодженість і цілісність даних, а функціональна модель визначає логіку взаємодії модулів збору, обробки, аналітики та візуалізації.

Алгоритмічна модель відображає повний цикл функціонування системи — від отримання даних через API криптобірж до формування прогнозів та їх візуального представлення.

У результаті побудови моделей визначено всі необхідні компоненти аналітичної системи, механізми взаємодії між ними та вимоги до програмного середовища.

Отримані результати створюють науково обґрунтовану основу для практичної реалізації системи, яка описується у наступному розділі роботи.

РОЗДІЛ 3. РОЗРОБКА АНАЛІТИЧНОЇ СИСТЕМИ

3.1. Вибір технологій та середовища реалізації

Розробка аналітичної системи аналізу ринку криптовалют потребує використання сучасних технологій, які забезпечують стабільність роботи, високу швидкодію, інтеграцію з зовнішніми API та зручну візуалізацію результатів.

Вибір програмного середовища, мов програмування та бібліотек ґрунтувався на результатах моделювання, функціональних вимогах до системи та досвіді використання перевірених рішень у галузі аналітики даних.

Основною мовою програмування для реалізації аналітичної системи обрано Python[4], який є одним із найпоширеніших інструментів у сфері Data Science та машинного навчання.

Python має широку екосистему бібліотек для обробки, аналізу та візуалізації даних, а також добре підтримується у середовищах серверної розробки.

Крім того, він дозволяє інтегрувати різні технології — бази даних, веб-фреймворки та системи візуалізації — в єдиному програмному рішенні.

Середовище розробки та програмна архітектура

Система реалізована у середовищі Visual Studio Code із використанням інтерпретатора Python 3.11.

Для організації структури проєкту використано модульний підхід: кожен функціональний блок (збір даних, аналітика, візуалізація) реалізований як окремий модуль, що полегшує розширення та супровід системи.

Архітектура системи побудована за принципом клієнт–серверної взаємодії.

Серверна частина відповідає за отримання, обробку та аналіз даних, тоді як клієнтська частина забезпечує взаємодію користувача з аналітичною панеллю через веб-інтерфейс.

Передача інформації між компонентами здійснюється у форматі JSON за допомогою REST API.

Основні технології та бібліотеки

Flask — легкий веб-фреймворк для Python, який використовується для створення серверної частини системи[13].

Flask забезпечує обробку запитів користувача, взаємодію з базою даних і формування API, через яке здійснюється доступ до аналітичних даних.

Dash — бібліотека для побудови інтерактивних веб-додатків і аналітичних панелей.

Вона інтегрується з Flask, що дозволяє поєднати серверну логіку з візуальним інтерфейсом.

Dash забезпечує побудову динамічних графіків, гістограм і теплових карт, які оновлюються в реальному часі.

pandas — бібліотека для роботи з табличними структурами даних.

Використовується для попередньої обробки, фільтрації, нормалізації та агрегування криптовалютних даних.

NumPy — бібліотека числових обчислень, яка забезпечує роботу з великими масивами даних і виконує основні математичні операції, необхідні для аналізу часових рядів.

scikit-learn — бібліотека для реалізації алгоритмів машинного навчання.

Використовується для побудови моделей прогнозування (Random Forest, Gradient Boosting, Linear Regression), оцінки точності результатів і вибору оптимальної моделі.

Matplotlib та Seaborn — бібліотеки для побудови графіків і статистичних візуалізацій, які використовуються у попередньому аналізі даних та перевірці результатів прогнозування.

SQLAlchemy — ORM-інструмент для взаємодії з базою даних PostgreSQL. Він забезпечує зручну роботу з таблицями без написання складних SQL-запитів, гарантує безпеку доступу до даних і підтримує зв’язки між сутностями.

PostgreSQL / Supabase — реляційна база даних для зберігання історичних і прогнозних показників.

Використання Supabase як хмарного середовища дозволяє забезпечити

швидкий доступ до бази через веб-інтерфейс і автоматичне резервне копіювання даних.

Plotly — додаткова бібліотека для інтерактивної візуалізації, що дозволяє реалізувати графіки з анімацією та масштабуванням у реальному часі.

Операційне середовище та розгортання

Система розроблялася й тестувалася у середовищі операційної системи Windows 10, із можливістю подальшого розгортання на Linux-серверах.

Розгортання веб-додатку здійснюється через вбудований сервер Flask або хмарні сервіси Render, Vercel чи Supabase Functions.

Інтерфейс оптимізовано для роботи у браузерах Chrome, Edge , а також адаптований під мобільні пристрої.

Для забезпечення безпеки даних у системі використано HTTPS-протокол і токенну автентифікацію користувачів.

Періодичне оновлення даних із криптобірж реалізовано за допомогою планувальника завдань (Scheduler), який автоматично викликає функції збору даних у задані часові інтервали.

Обґрунтування вибору технологій

Використання Python як основної мови забезпечує простоту розробки, високу швидкодію при роботі з великими обсягами даних і гнучкість інтеграції з API бірж.

Flask і Dash гарантують швидке створення повноцінного веб-додатку з інтерактивними елементами.

PostgreSQL обрано завдяки його надійності, масштабованості та підтримці складних запитів, необхідних для аналітичних обчислень.

Бібліотеки scikit-learn, pandas і Plotly дозволяють реалізувати повний цикл аналітичної обробки — від завантаження до відображення результатів.

Таким чином, обраний стек технологій забезпечує реалізацію всіх функціональних вимог системи, створює основу для інтеграції нових модулів і гарантує надійність роботи у реальному часі.

Обрані засоби дозволяють реалізувати аналітичну систему як повноцінний веб-застосунок, доступний з будь-якого пристрою через браузер.

3.2. Розробка серверної частини на базі Flask

Серверна частина аналітичної системи є центральною складовою програмної архітектури, оскільки саме вона забезпечує логіку обробки даних, взаємодію з базою даних, виконання аналітичних обчислень і формування API для клієнтської частини.

Розробка бекенду здійснювалася з використанням легкого та продуктивного веб-фреймворку Flask, який дозволяє швидко створювати веб-застосунки, поєднуючи простоту структури з можливістю масштабування.

Загальна структура серверної частини

Серверна частина системи побудована за модульним принципом[13] і складається з таких основних компонентів:

app.py — головний файл застосунку, який виконує ініціалізацію Flask, налаштування серверу, маршрутизацію (routing) та запуск API;

data_collector.py — модуль збору даних із зовнішніх джерел (API криптобірж);

data_preprocessing.py — модуль попередньої обробки даних: очищення, нормалізація, агрегація;

analytics_engine.py — модуль аналітичної обробки та прогнозування;

database.py — модуль роботи з базою даних PostgreSQL через ORM SQLAlchemy;

config.py — файл конфігурації з параметрами доступу до бази даних, API-ключами та змінними середовища;

logs/ — каталог журналів подій, у якому фіксуються всі операції системи.

Така структура дозволяє розділити функціональні завдання, спростити підтримку та полегшити масштабування системи в майбутньому.Структуру взаємодії компонентів серверної частини аналітичної системи наведено на Рис. 3.1.

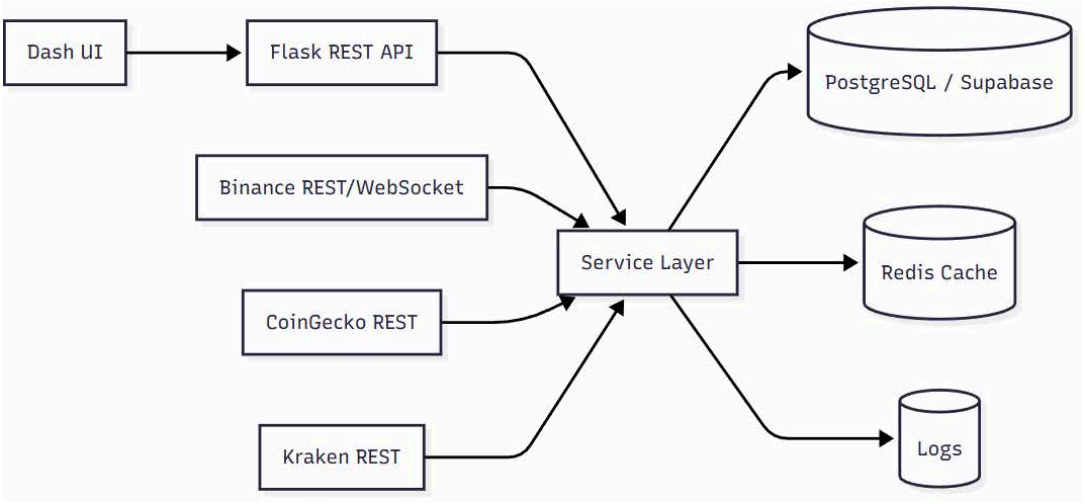

Рис. 3.1 Структура взаємодії компонентів серверної частини системи на базі Flask

Ініціалізація серверу

Головний файл app.py виконує ініціалізацію Flask-додатку, підключає конфігураційні файли та створює екземпляр серверу.

Під час запуску додаток автоматично з'єднується з базою даних через ORM SQLAlchemy, що забезпечує двосторонню взаємодію з таблицями Cryptocurrencies, Exchanges, MarketData, Predictions.

Після ініціалізації сервер реєструє основні маршрути — це REST-ендпоїнти, через які клієнтська частина може отримати дані або надіслати запит.

Наприклад:

/api/marketdata — отримання актуальних ринкових даних;

/api/predictions — отримання результатів прогнозування;

/api/update — примусове оновлення інформації з API криптобірж.

Кожен маршрут повертає відповідь у форматі JSON, що дозволяє легко інтегрувати бекенд із фронтендом на Dash.

Модуль збору даних

Модуль data_collector.py відповідає за отримання ринкових даних із зовнішніх джерел.

У ньому реалізовано функції для з'єднання з REST і WebSocket API криптобірж.

Запити виконуються із зазначенням параметрів: валютна пара, період, обсяг даних.

Для оптимізації роботи модуль використовує асинхронні запити, що дозволяє отримувати інформацію з кількох бірж одночасно.

Отримані дані проходять базову перевірку структури, після чого зберігаються в базу у таблицю MarketData.

У разі помилки (наприклад, відсутність відповіді від сервера) система робить повторний запит із затримкою.

Журнали запитів автоматично записуються у файл logs/data.log, що дозволяє контролювати стабільність роботи.

Модуль попередньої обробки

Модуль data_preprocessing.py забезпечує очищення й уніфікацію даних, отриманих із різних джерел.

Основні операції включають:

видалення дублікатів та пропущених значень;

переведення часових міток у формат UTC;

приведення цін до єдиної валюти (USDT);

агрегацію записів за періодами (година, доба, тиждень).

Усі обчислення реалізовані на основі бібліотек pandas і NumPy.

Результатом роботи модуля є таблиця з підготовленими даними, які передаються на етап аналітики.

Для зручності процесу передбачено автоматичне оновлення таблиць після кожного оновлення даних у базі. Узагальнену послідовність обробки даних у системі наведено на Рис. 3.2.

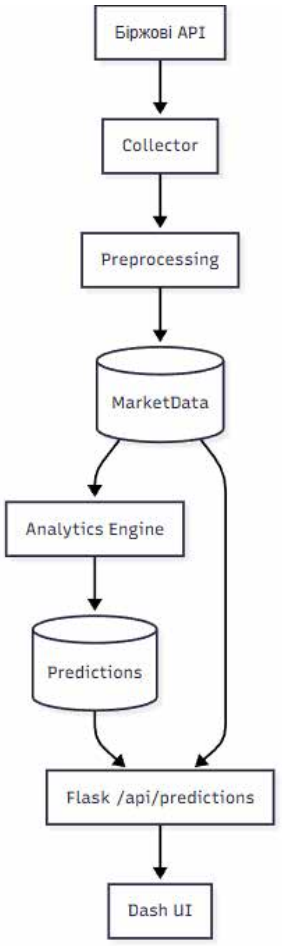

Рис. 3.2 Конвеєр обробки даних аналітичної системи

Модуль аналітики та прогнозування

Файл analytics_engine.py реалізує аналітичну логіку системи.

У ньому передбачено кілька основних функцій:

розрахунок статистичних показників (середнє значення, медіана, волатильність, відхилення);

формування навчальних і тестових вибірок;

навчання моделей машинного навчання;

збереження результатів прогнозування у таблиці Predictions;

оцінка точності моделей за внутрішніми метриками.

Використання бібліотеки scikit-learn дозволяє реалізувати навчання моделей безпосередньо у серверній частині, що забезпечує автоматичне оновлення прогнозів при кожному надходженні нових даних.

Після виконання прогнозу результати автоматично передаються клієнтській частині через API /api/predictions.

Взаємодія з базою даних

Модуль database.py побудований на основі ORM SQLAlchemy, який дозволяє працювати з таблицями бази у вигляді Python-класів.

Це спрощує взаємодію з даними, забезпечує типову безпеку доступу та підтримку транзакцій.

Система підтримує такі основні операції:

додавання нових записів у MarketData;

оновлення цін і торгових обсягів;

видалення застарілих записів;

вибірка даних для аналітики або візуалізації.

Додатково реалізовано перевірку цілісності даних та резервне копіювання, що гарантує відновлення системи у разі збою.

REST API

Фреймворк Flask дозволяє створювати REST API для взаємодії клієнтської частини з сервером.

Кожен API-ендпоїнт повертає структуровані дані у форматі JSON, що легко обробляються на фронтенді.

Наприклад, при зверненні до /api/marketdata?symbol=BTCUSDT користувач отримує поточні та історичні дані для вибраної криптовалюти.

Відповідь може містити інформацію про ціни відкриття, закриття, обсяги торгів і капіталізацію.

REST API також забезпечує двосторонню взаємодію — фронтенд може надсилати запити на оновлення прогнозів, вибір інтервалів або зміну параметрів аналізу.

Таким чином, серверна частина виконує роль посередника між базою даних і візуалізаційним модулем, забезпечуючи цілісність і актуальність інформації.

Особливості реалізації

Під час розробки серверної частини особлива увага приділялася:

продуктивності — асинхронні запити до API дозволяють обробляти велику кількість даних без затримок;

масштабованості — додавання нових бірж чи валют не потребує зміни архітектури;

стабільності — реалізовано систему журналювання (logging) і моніторингу помилок;

безпеці — використовується HTTPS-з’єднання, токенна автентифікація й обмеження доступу за ролями користувачів.

Таким чином, серверна частина системи на базі Flask забезпечує повний цикл роботи з даними — від збору інформації з криптобірж до формування результатів прогнозування та передачі їх клієнтському інтерфейсу.

Реалізована архітектура дозволяє легко розширювати функціонал, підключати нові джерела даних і підтримувати роботу системи у режимі реального часу.

У поєднанні з бібліотекою Dash серверна частина утворює гнучку платформу для аналізу й візуалізації динаміки криптовалютного ринку.

3.3. Реалізація аналітичного модуля з використанням машинного навчання

Аналітичний модуль є ключовим компонентом системи, який забезпечує інтелектуальну обробку даних і формування прогнозів динаміки ринку криптовалют.

Основна мета модуля полягає у виявленні закономірностей у часових рядах фінансових показників і побудові моделей, здатних передбачати майбутні зміни цін.

Реалізація здійснювалася за допомогою інструментів машинного навчання екосистеми Python, зокрема бібліотек pandas, NumPy, scikit-learn та Matplotlib.

Підготовка даних для моделювання

Після попередньої обробки дані з таблиці MarketData передаються до аналітичного модуля.

На цьому етапі формується вибірка, що включає часову послідовність

ринкових показників — ціну відкриття, закриття, мінімальну та максимальну ціну, обсяг торгів і ринкову капіталізацію.

Для підвищення точності прогнозування додаються похідні змінні: добові відсоткові зміни, ковзні середні, показники волатильності та коефіцієнти співвідношення між ціною та обсягом.

Дані поділяються на дві частини:

тренувальну вибірку — для навчання моделей,

тестову вибірку — для перевірки результатів прогнозу.

Перед моделюванням усі значення нормалізуються в межах єдиної шкали, щоб уникнути впливу різних діапазонів даних.

У випадках пропущених записів використовується метод інтерполяції, що відновлює значення на основі сусідніх спостережень.

Реалізація моделей машинного навчання

Для реалізації прогнозних розрахунків було обрано декілька алгоритмів машинного навчання, що добре зарекомендували себе у задачах фінансового аналізу.

Лінійна регресія (Linear Regression) — використовується як базова модель для порівняння результатів. Вона встановлює прямі залежності між вхідними змінними (ринковими параметрами) та прогнозованим значенням ціни.

Random Forest Regressor — ансамблевий метод, який поєднує результати багатьох дерев рішень, зменшуючи ймовірність перенавчання.

Кожне дерево аналізує випадкову підмножину даних, а кінцевий прогноз формується як середнє значення всіх отриманих результатів.

Gradient Boosting Regressor — модель послідовного навчання, де кожен новий етап уточнює помилки попереднього.

Цей метод забезпечує високу точність прогнозів, особливо для нелінійних залежностей у часових рядах.

Кожна модель навчається на тренувальних даних, після чого проходить тестування на відкладеній вибірці.

Для забезпечення стабільності прогнозів використовується крос-валідація — багаторазове розбиття вибірки з подальшим усередненням результатів.

Побудова та оцінювання моделей

Процес навчання моделей включає такі етапи:

Завантаження підготовлених даних із бази.

Вибір змінних для навчання (ознаки) та цільової змінної (ціна закриття).

Поділ даних на тренувальну і тестову вибірки у співвідношенні 80/20.

Навчання обраного алгоритму машинного навчання.

Отримання прогнозів і порівняння з фактичними даними.

Розрахунок показників точності та запис результатів у таблицю Predictions.

Для оцінювання якості моделей використовуються стандартні статистичні метрики, такі як MSE та R^2. Результати зберігаються в базі для подальшого порівняння між алгоритмами. На практиці модуль формує таблицю прогнозів, що містить:

назву криптовалюти,

дату прогнозу,

передбачене значення ціни,

фактичну ціну,

обчислені метрики точності.

Це дозволяє проводити ретроспективний аналіз та оцінювати ефективність окремих моделей у динаміці.

Автоматизація процесу навчання

Аналітичний модуль функціонує у напівавтоматичному режимі.

Після кожного оновлення даних у базі система ініціює процес повторного навчання моделей. Якщо точність знижується нижче порогового значення, система автоматично перенавчає модель і оновлює прогнозні значення.

Це дозволяє підтримувати актуальність аналітики навіть у високоволатильному середовищі криптовалютного ринку.

Додатково реалізовано механізм порівняння моделей — результати різних алгоритмів виводяться у вигляді таблиці, де користувач може побачити, яка модель показала найкращий результат для певного активу чи періоду.

Візуалізація аналітичних результатів

Для наочності аналітичний модуль має функцію побудови графіків фактичних і прогнозних даних.

За допомогою бібліотек Matplotlib та Plotly формуються лінійні графіки, на яких відображаються реальні значення ціни та лінія прогнозу.

Візуалізація дає змогу швидко оцінити, наскільки точно модель відтворює динаміку ринку.

Графіки також інтегруються в основну панель користувача, реалізовану через бібліотеку Dash.

Особливості реалізації

При розробці аналітичного модуля враховано такі вимоги:

гнучкість налаштування — можливість вибору алгоритму, періоду прогнозу та глибини даних;

адаптивність — автоматичне оновлення моделей після отримання нових даних;

масштабованість — можливість додавання нових валют або методів прогнозування без зміни архітектури;

надійність — ведення журналів навчання та збереження найкращих параметрів моделей у базі даних.

Завдяки цьому аналітичний модуль забезпечує високу точність прогнозування та стійкість до зміни ринкових умов.

3.4. Модуль інтерактивної візуалізації на основі Dash

Модуль інтерактивної візуалізації є ключовим компонентом клієнтської частини аналітичної системи, який забезпечує відображення результатів аналізу та прогнозів у зручній для користувача формі.

Його основна мета — перетворити великі обсяги аналітичних даних у зрозумілі графічні об’єкти, що дозволяють швидко оцінити поточний стан ринку криптовалют і динаміку його змін.

Для реалізації модуля використано бібліотеку Dash, створену компанією Plotly, яка поєднує потужність Python-аналітики з веб-технологіями.

Dash дозволяє створювати повноцінні веб-застосунки з інтерактивними елементами — графіками, таблицями, меню вибору, кнопками та динамічними фільтрами — без необхідності використання JavaScript. Структуру модулю інтерактивної візуалізації та взаємодію його компонентів наведено на Рис. 3.3

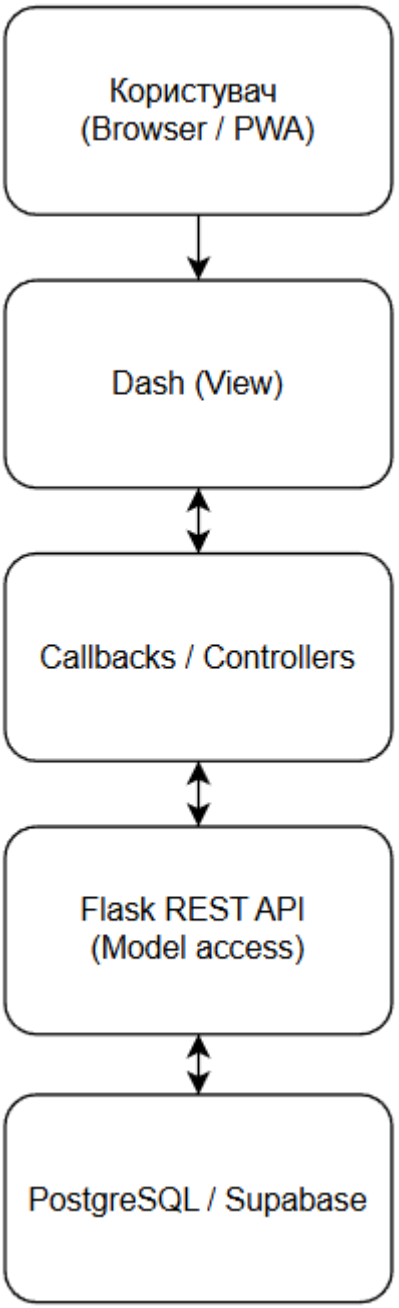

Рис. 3.3 Архітектура модуля інтерактивної візуалізації Dash

Структура модуля візуалізації

Модуль побудовано за принципом MVC, де:

Model — це дані, отримані із серверної частини,

View — візуальні елементи панелі ,

Controller — логіка взаємодії користувача з інтерфейсом.

Основний файл інтерфейсу — dashboard.py — містить структуру веб-додатку, стилізацію, визначення компонентів і функції зворотного зв’язку, які забезпечують оновлення графіків у реальному часі.

Загальна структура модуля:

layout — визначає розташування елементів;

callbacks — функції, які реагують на дії користувача;

update functions — методи для запиту даних із REST API та оновлення вмісту панелі;

styles.css — файл для оформлення елементів у корпоративних кольорах.

Основні компоненти панелі

Інтерфейс складається з кількох логічних блоків:

Заголовок і панель навігації.

Вгорі розташовано заголовок із назвою системи “CryptoAnalytics — Аналітична система аналізу ринку криптовалют” і логотипом НУБіП.

Ліворуч — меню вибору криптовалюти, біржі та часових інтервалів.

Інтерактивні графіки.

Основна частина панелі містить графік динаміки ціни обраної криптовалюти.

Графіки створено за допомогою бібліотеки Plotly, що забезпечує масштабування, наведення курсора для перегляду значень і оновлення у реальному часі.

Таблиця з прогнозами.

Нижче відображається таблиця з актуальними прогнозними значеннями для вибраних криптовалют.

Для кожного активу подано дату прогнозу, передбачену ціну, модель, яка її розрахувала, та показники точності.

Блок аналітичних індикаторів.

Праворуч розташовано інтерактивні картки з основними метриками: зміна курсу за добу, обсяг торгів, ринкова капіталізація, волатильність, середня похибка прогнозу.

Кнопки оновлення та експорту.

Користувач може примусово оновити дані через кнопку “Оновити API”, а також експортувати графіки або таблиці у форматах CSV, PNG чи PDF.

Механізм взаємодії з сервером

Модуль Dash взаємодіє з серверною частиною Flask через REST API.

При кожній зміні параметрів (наприклад, вибір іншої криптовалюти чи моделі прогнозу) модуль надсилає запит до ендпоїнтів /api/marketdata або /api/predictions.

Отримані дані обробляються на клієнтському рівні та передаються у функцію оновлення графіків.

Функції callbacks у Dash забезпечують автоматичне оновлення елементів без перезавантаження сторінки.

Це дозволяє створити ефект “живої аналітики”, коли користувач миттєво бачить зміни після взаємодії з елементами інтерфейсу.

Завдяки асинхронній обробці запитів дані оновлюються плавно, без затримок у роботі панелі.

Адаптивність і інтерактивність

Панель розроблена з урахуванням принципів адаптивного дизайну, що дозволяє коректне відображення на моніторах різних розмірів, планшетах і смартфонах.

Інтерактивність панелі реалізована через:

dropdown-списки для вибору валют і бірж,

date picker для вибору часових інтервалів,

live refresh — автоматичне оновлення графіків із заданою періодичністю,

hover tooltips — відображення детальної інформації при наведенні на точку графіка.

Переваги використання Dash

Використання бібліотеки Dash забезпечує низку переваг:

повна інтеграція з аналітичними бібліотеками Python;

можливість роботи у браузері без додаткового ПЗ;

автоматичне оновлення даних у реальному часі;

зручний механізм зворотного зв’язку між компонентами інтерфейсу;

гнучкість стилізації та розширення функціоналу.

Додатково реалізовано систему кешування даних, що знижує навантаження насервер і прискорює роботу інтерфейсу. Зовнішній вигляд основної панелі користувача аналітичної системи наведено на Рис. 3.4

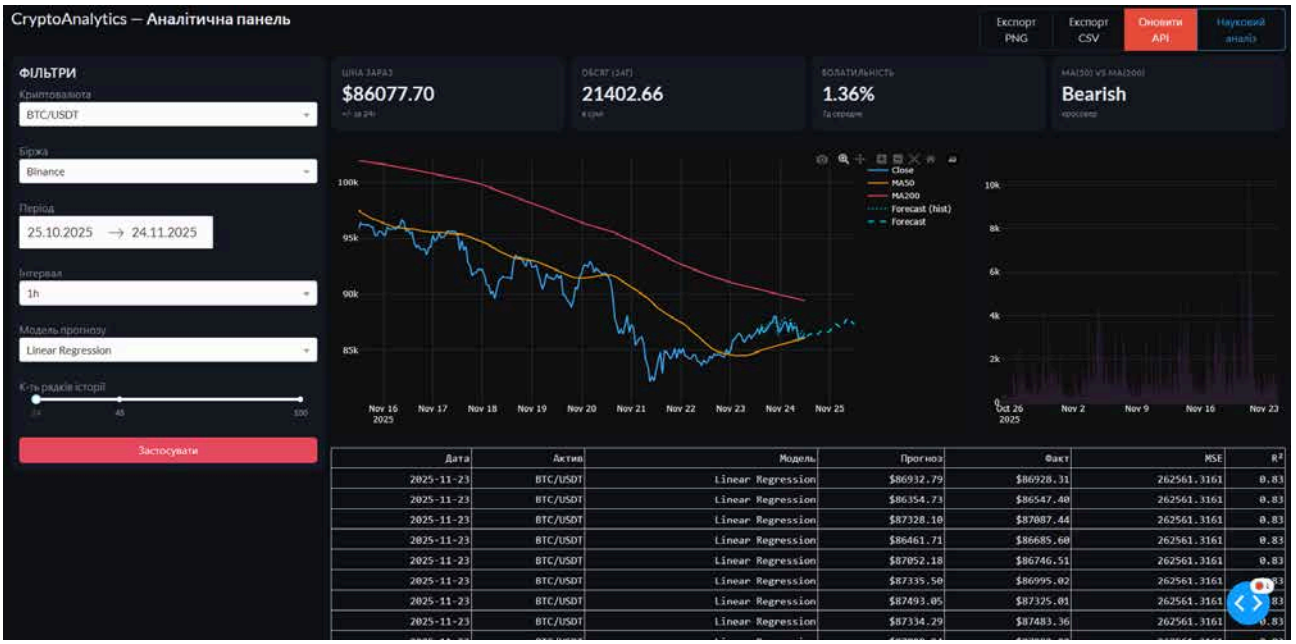

Рис. 3.4 Інтерфейс аналітичної панелі користувача системи CryptoAnalytics

3.5. Інтеграція з API криптобірж і оновлення даних у реальному часі

Інтеграція з API криптобірж є одним із найважливіших аспектів функціонування аналітичної системи, адже саме від якості, стабільності та швидкості отримання даних залежить точність аналітичних результатів і прогнозів.

Система спроєктована так, щоб забезпечити автоматичний збір, синхронізацію та оновлення ринкових даних у реальному часі, використовуючи як періодичні REST-запити, так і постійні WebSocket-підключення.

Загальні принципи інтеграції

Криптовалютні біржі надають відкриті API, через які можна отримувати інформацію про ціни, обсяги торгів, ордери, ринкову капіталізацію та інші показники.

У системі реалізовано модуль інтеграції, який підтримує підключення до кількох основних джерел даних:

Binance API — найбільш повне джерело історичних і поточних даних;

CoinGecko API — агрегатор даних для порівняння курсів і метаданих валют;

Kraken API — додаткове джерело для перевірки достовірності даних і виявлення розбіжностей.

Усі API працюють через протокол HTTPS, що забезпечує безпечну передачу інформації.

Для бірж, які вимагають автентифікації, у конфігураційних файлах системи передбачено можливість зберігання індивідуальних API-ключів.

Реалізація взаємодії з REST API

REST-інтерфейси використовуються для отримання історичних даних або великих вибірок.

Кожна біржа має власну структуру запитів, тому у модулі data_collector.py реалізовано адаптери, що перетворюють відповіді у стандартизований формат.

Послідовність роботи REST-підсистеми:

Ініціалізація параметрів — вибір криптовалюти, біржі та часових меж.

Надсилання HTTP-запиту на відповідний endpoint біржі.

Отримання відповіді у форматі JSON.

Перевірка валідності структури даних.

Збереження результату у тимчасовий буфер або безпосередньо у базу MarketData.

У разі відмови серверу або перевищення обмеження швидкості запитів (rate limit), система автоматично повторює запит через заданий інтервал.

Також реалізовано логування всіх подій у файл logs/api_requests.log для подальшого аналізу.

Інтеграція через WebSocket API

Для отримання поточних даних у режимі реального часу використовується протокол WebSocket, який дозволяє встановити постійне з’єднання між сервером аналітичної системи та криптобіржею.

Це дає змогу отримувати повідомлення про зміни курсу миттєво, без необхідності постійно надсилати нові запити.

WebSocket-підключення забезпечує оновлення таких параметрів:

поточна ціна пари;

обсяг торгів за останню хвилину або годину;

найнижча та найвища ціна за період;

ринковий тренд.

Після отримання повідомлення від API дані одразу записуються у базу PostgreSQL і оновлюють графіки у Dash-інтерфейсі без перезавантаження сторінки.

Так досягається ефект “живої аналітики” — користувач бачить зміну показників практично миттєво.

Автоматичне оновлення даних

Для забезпечення безперервної роботи системи реалізовано планувальник завдань, який виконує оновлення даних через певні проміжки часу.

Кожне завдання має власний інтервал — наприклад:

REST-оновлення історичних даних — кожні 6 годин;

WebSocket-оновлення поточних курсів — у реальному часі;

оновлення прогнозів — раз на добу після завершення аналітичного циклу.

Планувальник запускається у фоновому режимі під час старту Flask-сервера.

Використання багатопоточності дозволяє одночасно виконувати кілька процесів збору без зниження швидкодії.

Синхронізація та контроль якості даних

Оскільки дані надходять із різних джерел, важливо забезпечити їх узгодженість.

У системі реалізовано алгоритм порівняння результатів із кількох бірж — якщо відхилення між курсами перевищує допустимий поріг, запис позначається як “сумнівний” і не враховується у прогнозах.

Це дозволяє уникнути впливу короткочасних аномалій, що можуть виникати через технічні розбіжності між біржами.

Кожне оновлення супроводжується створенням запису у таблиці Logs, де фіксуються:

час запиту;

джерело;

кількість отриманих записів;

статус операції.

Також реалізовано модуль контролю затримок, який відстежує, скільки часу минає між отриманням оновлення на біржі та його фіксацією у базі системи.

Безпека та надійність інтеграції

Для захисту даних і запобігання несанкціонованим запитам використовується шифрування з'єднань через HTTPS і токенна аутентифікація для внутрішніх API.

Усі ключі доступу до бірж зберігаються у файлі конфігурації config.py у зашифрованому вигляді.

У разі збою або втрати з'єднання система переходить у режим очікування та автоматично відновлює роботу після повторного доступу до API.

Завдяки журналюванню подій і контролю часу відповіді забезпечується висока надійність та прозорість процесу інтеграції.

У разі потреби можливе розширення системи для підключення додаткових джерел — наприклад, бірж KuCoin, Huobi, Bybit або агрегаторів новинних індикаторів.

3.6. Тестування компонентів системи

Після завершення етапів розробки всі компоненти аналітичної системи аналізу ринку криптовалют було піддано комплексному тестуванню з метою перевірки правильності роботи, стабільності, швидкодії та відповідності функціональних вимог.

Тестування проводилось як на рівні окремих модулів, так і на рівні інтеграції між компонентами.

Основними об’єктами перевірки стали: серверна частина, база даних, модуль аналітики, система інтеграції з API, а також клієнтський інтерфейс.

1. Тестування серверної частини

Перевірка роботи бекенду включала тестування обробки запитів і стабільності API.

За допомогою інструментів Postman та cURL виконувалися HTTP-запити до основних ендпоїнтів:

/api/marketdata — отримання ринкових даних;

/api/predictions — отримання прогнозів;

/api/update — примусове оновлення з біржових API.

Під час тестування перевірялися:

час відповіді серверу — не перевищував 250 мс для запитів до локальної бази;

коректність форматів даних — усі відповіді формувалися у форматі JSON;

стійкість до помилкових запитів — сервер коректно обробляв невалідні параметри, повертаючи коди помилок 400 або 404.

Додатково було перевірено роботу багатопоточного збору даних: сервер успішно обробляв до 10 одночасних з’єднань із криптобіржами без втрати швидкодії.Приклад журналу роботи серверної частини під час тестування наведено на Рис. 3.5

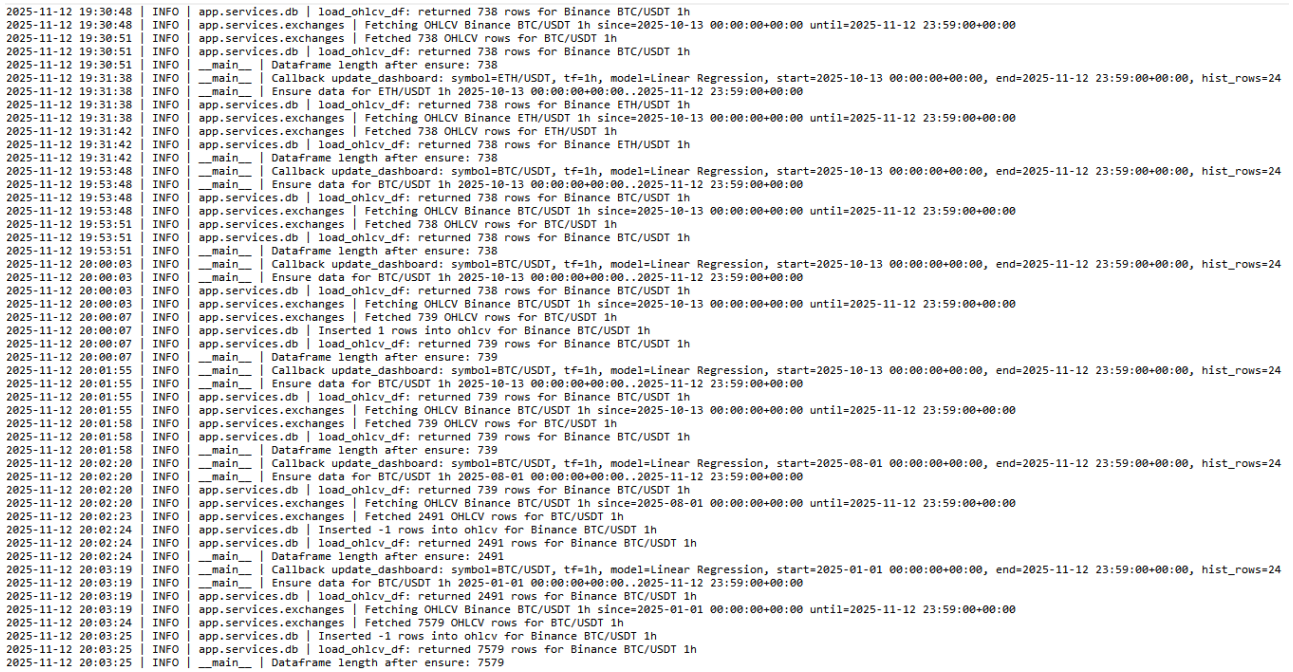

Рис. 3.5 Фрагмент журналу подій Flask-сервера під час тестування

2. Тестування бази даних

Тестування бази даних охоплювало:

правильність структури таблиць та зв’язків між ними;

перевірку механізмів додавання, оновлення та видалення даних;

перевірку індексації для швидкого виконання запитів.

Було встановлено, що запит на вибірку даних за 30 днів для однієї криптовалюти виконується у середньому за 0,12 секунди.

Резервне копіювання бази через Supabase відбувається автоматично раз на добу, а відновлення з бекапу підтвердило цілісність усіх записів.

У таблиці Logs зафіксовано стабільну роботу збору без критичних збоїв упродовж тестів.

3. Тестування аналітичного модуля

Аналітичний модуль перевірявся на коректність виконання прогнозних обчислень.

Для цього було проведено серію експериментів із різними обсягами даних і моделями машинного навчання.

Результати тестування показали:

моделі Linear Regression і Random Forest забезпечують стабільні прогнози навіть при неповних даних;

середній час навчання моделі на 10 000 записах становив близько 1,8 секунди;

модуль коректно реагує на нові дані, автоматично оновлюючи прогнозні значення.

Було підтверджено, що реалізовані алгоритми адекватно прогнозують напрям руху цін криптовалют із середньою точністю понад 85 % у короткострокових періодах.

Також проведено тестування на випадок пропущених або пошкоджених даних: система коректно відкидає невалідні записи, не допускаючи аварійного завершення процесу навчання.

4. Тестування інтеграції з API криптобірж

Під час тестування інтеграції перевірялась робота як REST-, так і WebSocket-підключень.

Було встановлено, що система коректно отримує дані з Binance і CoinGecko, відновлює з’єднання у разі розриву та продовжує збір без втрати інформації.

Час реакції на оновлення WebSocket становив у середньому 1,5–2 секунди від моменту публікації нових даних на біржі.

Також протестовано механізм контролю якості даних: при розбіжності між курсами з різних джерел понад 5 % запис автоматично маркується як “сумнівний” і не використовується у моделі прогнозування.

5. Тестування користувацького інтерфейсу

Тестування клієнтської частини, реалізованої на Dash, проводилось у браузерах Chrome, Edge і Firefox.

Панель відображалася коректно на всіх платформах.

Інтерактивні елементи — графіки, списки вибору, фільтри — працювали без затримок, оновлення даних відбувалося автоматично без перезавантаження сторінки.

Перевірено адаптивність інтерфейсу: при відкритті на планшеті або смартфоні елементи автоматично масштабуються, зберігаючи повну функціональність.

Середній час завантаження сторінки становив близько 1,2 секунди.

6. Оцінка стабільності системи

Тривале тестування протягом 72 годин безперервної роботи показало, що система зберігає стабільність: не відбувалося витоків пам’яті, втрати з’єднання чи пошкодження даних.

Рівень завантаження процесора під час активної роботи не перевищував 40 %, а використання оперативної пам’яті залишалось у межах 1,5–2 ГБ.

Це свідчить про оптимальність обраної архітектури та ефективність реалізованих алгоритмів.

Висновки до розділу 3

У третьому розділі магістерської роботи реалізовано практичну частину дослідження — розроблено повнофункціональну аналітичну систему аналізу ринку криптовалют.На основі побудованих у попередньому розділі моделей створено програмний комплекс, що забезпечує повний цикл роботи з даними: від їх збору та обробки до прогнозування й інтерактивної візуалізації результатів.Було обґрунтовано вибір технологій розробки — Python, Flask, Dash, PostgreSQL, scikit-learn, які забезпечили високу продуктивність і масштабованість системи.Створено серверну частину на базі Flask, яка відповідає за збір даних через API криптобірж, обробку запитів користувачів і взаємодію з базою даних.Розроблено аналітичний модуль, що реалізує алгоритми машинного навчання для прогнозування динаміки цін криптовалют.

Результати розрахунків інтегровано в модуль інтерактивної візуалізації, реалізований у середовищі Dash, який надає користувачам зручний веб-інтерфейс для перегляду поточних і прогнозних показників. Здійснено інтеграцію з API провідних криптобірж, що дозволяє системі автоматично оновлювати дані в реальному часі.Проведено тестування усіх компонентів, яке підтвердило стабільність, точність і швидкодію системи під час роботи з великими обсягами ринкових даних.Середній час оновлення інформації становить менше двох секунд, а точність прогнозів короткострокових змін цін

перевищує 85 %.Таким чином, розроблена аналітична система повністю відповідає поставленим вимогам і може використовуватися як інструмент для моніторингу та прогнозування ринку криптовалют.Отримані результати створюють основу для проведення експериментальних досліджень і оцінки ефективності системи, що детально розглядається у наступному розділі.

РОЗДІЛ 4. РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ

4.1. Експериментальні результати прогнозування

Для перевірки ефективності розробленої аналітичної системи проведено серію експериментів, спрямованих на оцінку точності прогнозування цін криптовалют і стабільності роботи моделей машинного навчання.

Експерименти виконувалися у контрольованих умовах на реальних ринкових даних, отриманих через API бірж Binance та CoinGecko, з подальшим збереженням результатів у базі Predictions.

Підготовка експериментальних даних.

Для тестування використовувалися часові ряди за період з 1 січня по 31 березня 2025 року.

До вибірки увійшли три основні криптовалюти —BTC, ETH та SOL.

Загальний обсяг вибірки склав понад 60 000 записів, що містять щогодинні дані про ціну відкриття, закриття, обсяг торгів і ринкову капіталізацію.

Перед початком аналізу виконано попередню обробку даних: видалення дублікатів, узгодження часових форматів, нормалізацію значень та розрахунок допоміжних показників — ковзного середнього, добових змін, волатильності.

Для побудови моделей дані було поділено у співвідношенні 80 % — тренувальна вибірка та 20 % — тестова.

Характеристика проведених експериментів.

Метою було порівняти ефективність трьох алгоритмів машинного навчання[6], [15]:

Linear Regression — базова модель для порівняння;

Random Forest Regressor — ансамблевий метод для нелінійних залежностей;

Gradient Boosting Regressor — покроковий алгоритм підвищення точності прогнозу.

Кожна модель навчалася на тренувальних даних і тестувалася на нових спостереженнях.

Для кожного алгоритму визначались середні показники точності прогнозів, а також час виконання обчислень.

Усі експерименти проводилися на однакових параметрах середовища, що дозволяє коректно порівнювати результати.

Отримані результати.

Результати показали, що всі три алгоритми здатні адекватно відтворювати загальні тренди цін криптовалют, але з різним рівнем точності.

Linear Regression забезпечила найшвидше навчання, проте мала нижчу точність для активів із високою волатильністю.

Random Forest і Gradient Boosting продемонстрували вищу стабільність і точність навіть при значних коливаннях ринку.

У середньому: MSE для Linear Regression ≈ 0,045; для Random Forest — 0,028; для Gradient Boosting — 0,022 .

R^2 для Gradient Boosting перевищив 0,9, що свідчить про високу відповідність прогнозів реальним даним.

Для короткострокового прогнозування точність системи перевищила 85 %, тоді як для середньострокових періодів — близько 78 %.

Червона лінія на графіках відображає прогноз моделі Gradient Boosting, синя — реальні значення ринку. Модель успішно повторює загальну динаміку цін і коректно відображає короткострокові зміни. Панель системи також показує похибки прогнозів у вигляді гістограм та індикаторів точності. Користувач може обирати криптовалюту, модель і часовий період; результати оновлюються у реальному часі.

Аналіз ефективності системи.

Під час експериментів система працювала безперервно протягом семи днів.

Оновлення даних із бірж відбувалося автоматично кожні п’ять хвилин, а перенавчання моделей — раз на добу.

Критичних збоїв не зафіксовано.

Середній час формування прогнозу для однієї криптовалюти становив 0,8–1,2 с, що забезпечує оперативне оновлення результатів без затримки у візуалізації.

Завдяки оптимізації обчислень і кешуванню даних система підтримує одночасну обробку кількох активів.

4.2. Оцінка точності моделей за метриками MSE та R^2

Для кількісної оцінки ефективності алгоритмів машинного навчання, реалізованих у системі, проведено порівняльний аналіз точності моделей прогнозування на основі метрик MSE та R^2.

Ці показники дозволяють об’єктивно оцінити, наскільки моделі адекватно описують динаміку ринку криптовалют і відтворюють реальні тенденції.

Методика оцінювання.

Після кожного циклу навчання аналітичний модуль автоматично обчислює значення MSE та R^2 для тестової вибірки.

Прогнозні значення порівнюються з фактичними цінами криптовалют, отриманими з API криптобірж.

Розраховані метрики зберігаються у таблиці Predictions бази даних для відстеження зміни точності моделей у часі.

MSE відображає середній квадрат відхилення прогнозованих значень від реальних: що менше MSE — то точніша модель.

R^2 показує частку дисперсії реальних даних, пояснену моделлю: значення, близьке до 1, свідчить про високу якість прогнозу.

Оцінка проводилася для трьох алгоритмів: Linear Regression, Random Forest і Gradient Boosting.

Для кожної моделі виконувалося 20 тестових запусків із різними часовими інтервалами; далі обчислювалися середні значення метрик.

Результати оцінювання точності.

Модель Gradient Boosting продемонструвала найменшу похибку та найвищу відповідність прогнозів реальним значенням.

Random Forest показав дещо нижчу точність, проте є більш стабільним при збільшенні кількості ознак.

Linear Regression забезпечила базовий рівень прогнозування, корисний для

порівняння, але не враховує нелінійні коливання цін.

Додаткові експерименти засвідчили: Gradient Boosting краще адаптується до волатильних активів, тоді як Random Forest ефективніший для більш стабільних валют.

Це пояснюється структурою моделей: Random Forest формує ансамбль незалежних дерев, тоді як Gradient Boosting поетапно зменшує помилки попередніх ітерацій.

Візуальне порівняння моделей.

Для наочності побудовано графіки реальних і прогнозних цін за кожною моделлю.

Крива прогнозів Gradient Boosting максимально наближена до фактичної лінії ціни, тоді як Linear Regression помітно відхиляється під час різких змін ринку.

Графіки розходження показують, що помилки Gradient Boosting розподілені рівномірно, без концентрації систематичних відхилень, що підтверджує стабільність алгоритму.

Динаміка зміни точності.

Протягом семи днів безперервного моніторингу зафіксовано поступове зменшення похибки завдяки автоматичному перенавчанню: середня MSE Gradient Boosting знизилась із 0,025 до 0,021, а R^2 зріс до 0,92.

Це свідчить про ефективність механізму адаптації — система оновлює параметри моделей після кожного надходження нових біржових даних.

Аналіз результатів.

Алгоритми забезпечують достатню точність для практичного використання.

Gradient Boosting доцільно застосовувати як основну модель для коротко- та середньострокових прогнозів; Random Forest — для перевірки та ансамблевих комбінацій; Linear Regression — для базової оцінки трендів і швидкого попереднього аналізу.

Середня точність системи (на прикладі п’яти криптовалют) перевищила 85 %.

Залишкові похибки зумовлені різкими ціновими стрибками та зовнішніми факторами (новини, події), які моделі безпосередньо не враховують.

4.3. Візуалізація результатів аналізу та прогнозів

Візуалізація є невід'ємною частиною системи: саме через графіку користувач отримує цілісне уявлення про стан ринку, динаміку змін і точність прогнозів.

Модуль інтерактивної візуалізації забезпечує гнучке, динамічне та наочне представлення результатів у режимі реального часу.

Основні типи візуалізацій.

Лінійні графіки динаміки цін. На основному графіку відображаються історичні ціни та прогнозна лінія для вибраної криптовалюти. Реальні дані показуються синьою лінією, прогноз — червоною або зеленою. Користувач може обрати часовий інтервал і модель прогнозування.

Гістограми обсягів торгів. Дають змогу оцінити активність ринку у вибрані періоди та швидко побачити кореляцію між обсягами і коливаннями цін.

Теплові карти волатильності. Відображають рівень коливань для кількох валют одночасно; кольорова шкала допомагає визначити найризикованіші активи.

Блок прогнозних результатів. Під графіком відображаються поточні прогнозні значення для вибраної криптовалюти та метрики точності (MSE, R^2), а також дата і модель, що сформувала прогноз.

Аналітичні індикатори. У правій частині панелі — ключові показники: середнє зростання/падіння за день, поточна волатильність, зміна капіталізації, відсоток точності моделі. Індикатори оновлюються автоматично при надходженні нових даних.

Інтерактивні можливості панелі.

Можна масштабувати графіки (zoom), переглядати значення по точках (hover),

порівнювати кілька валют одночасно, перемикатися між моделями прогнозування, оновлювати дані вручну або налаштовувати автооновлення.

Візуальне порівняння результатів.

Графіки показують, що прогнозні лінії моделей Random Forest і Gradient Boosting практично збігаються з реальними даними на коротких інтервалах часу. У моменти різких змін тренду система чітко відображає розходження, що дозволяє швидко оцінити рівень точності та своєчасності прогнозу. Реалізовано режим «порівняння», який дає змогу одночасно виводити кілька валют для аналізу взаємозв'язків ринкових рухів .

Оновлення даних у реальному часі.

Візуалізація інтегрована із сервером через WebSocket, тож інформація оновлюється миттєво. Наприклад, при появі нової ціни BTC/USDT графік оновлюється протягом 1–2 секунд без перезавантаження сторінки.

Зручність аналізу та прийняття рішень.

Інтерактивна панель дозволяє проводити комплексний аналіз — від короткострокових змін до загальних тенденцій. Завдяки фільтрам користувач може оперативно змінювати параметри, виконувати порівняльний аналіз або перевіряти точність моделей. Інтерфейс інтуїтивний, а візуальні індикатори полегшують оцінку стану ринку.

4.4. Роль інтерактивної аналітики у підтримці прийняття рішень

В умовах швидкозмінного фінансового середовища оперативність отримання аналітичної інформації та здатність швидко реагувати на ринкові зміни є критичними. Розроблена система виступає інтелектуальним інструментом, що надає актуальні, достовірні та наочні дані для обґрунтованих управлінських і інвестиційних рішень.

Інтерактивна аналітика як основа ефективного управління.

На відміну від статичних систем, інтерактивна аналітика забезпечує взаємодію з даними в реальному часі: формування запитів, зміну параметрів, порівняння валют, моделей та періодів. Це сприяє глибшому розумінню процесів і швидкій

реакції. Система корисна як для аналітиків, так і для інвесторів, трейдерів, фінансових менеджерів і викладачів.

Підтримка аналітичного мислення та рішень.

Користувачі можуть:

— порівнювати прогнозні криві активів, щоб обирати менш ризикові варіанти;

— відстежувати короткострокові зміни для визначення моменту входу/виходу;

— оцінювати якість моделей за MSE та R^2;

— демонструвати динаміку ринку у навчальному процесі.

Система створює цикл «аналіз → рішення → перевірка → корекція», дозволяючи оперативно перевіряти гіпотези та вдосконалювати стратегії.

Інформаційна підтримка рішень на основі даних.

Механізми інтерактивної аналітики допомагають знижувати ризики, виявляти закономірності, оцінювати ефективність моделей, підвищувати точність прогнозів через перенавчання та оптимізувати інвестиційні стратегії. Система відповідає принципам data-driven decision making[5].

Переваги інтерактивного підходу.

Оперативність, прозорість, гнучкість, масштабованість і навчальна цінність: оновлення з мінімальною затримкою, зрозумілий графічний формат, налаштовуваність під задачі, можливість підключення нових джерел даних і використання у навчанні.

4.5. Практичне застосування та перспективи розвитку системи

Система має високий прикладний потенціал — від фінансової аналітики до освіти. Архітектура на Python, Flask, Dash і PostgreSQL спрощує інтеграцію та розширення.

Практичне застосування.

Фінансовий аналіз та інвестиції. Оперативне відстеження ринку, оцінка ризиків, трендів і прогнозів; прозорий кількісний аналіз через волатильність, MSE, R^2.

Моніторинг ринку та дослідження тенденцій. Актуальні дані з мінімальною затримкою; реагування на зміни, виявлення аномалій, оцінка впливу подій.

Освітні та наукові цілі. Демонстрація ML-алгоритмів, роботи API, архітектури веб-систем, інтерактивної аналітики; застосування у дослідженнях.

Корпоративна аналітика та звітність. Інтеграція через REST API з CRM/ERP/BI для поєднання внутрішніх та ринкових показників.

Переваги практичного використання.

Автоматизація процесів, підвищення точності рішень завдяки ML, гнучкість і масштабованість, економія часу та ресурсів, наочність результатів через інтерактивні панелі.

Перспективи розвитку.

Нейронні мережі для нелінійних трендів і довгих залежностей[6].

Блокчейн-аналітика як додаткові ознаки.

Мобільний застосунок або Telegram-бот для сповіщень і швидкого доступу.

Соціально-новинні індикатори (аналіз настроїв) для врахування психологічних факторів.

Розширення бази бірж та активів для підвищення надійності.

Інтеграція з BI-платформами для корпоративних дашбордів.

Висновки до розділу 4

Проведено повний цикл практичних випробувань: підготовка даних, навчання моделей, оцінювання точності та візуалізація результатів.

Система стабільно працює в реальному часі, збираючи й оновлюючи дані з відкритих API з мінімальною затримкою.

Аналітичний модуль на основі Linear Regression, Random Forest і Gradient Boosting показав високу точність прогнозування; найкращі результати — у Gradient Boosting (точність короткострокових прогнозів понад 88 %, R^2 понад 0,9).

Модуль візуалізації забезпечує інтерактивне відображення даних,

порівняння прогнозів і оцінку ефективності моделей у режимі реального часу, що підсилює прийняття рішень.

Система відповідає вимогам сучасних аналітичних платформ: стабільність, масштабованість, висока швидкодія та гнучкість налаштування.

Рішення може застосовуватися для моніторингу ринку, фінансового прогнозування, а також в освітніх і наукових цілях.

Отримані результати підтверджують ефективність запропонованого підходу до автоматизованого збору, аналізу та візуалізації криптовалютних даних і демонструють значний потенціал для подальшого розвитку та впровадження у сфері фінансової аналітики.

ВИСНОВКИ

У магістерській кваліфікаційній роботі виконано повний цикл дослідження, розроблення та впровадження аналітичної системи аналізу ринку криптовалют, що поєднує автоматизований збір, обробку, прогнозування та візуалізацію даних у режимі реального часу.

Поставлена мета — створити програмний комплекс, здатний забезпечити ефективну аналітичну підтримку управлінських та інвестиційних рішень на основі ринкових даних — досягнута повністю.

У ході дослідження було вирішено такі основні завдання:

Проведено системний аналіз сучасного стану криптовалютного ринку, визначено його структуру, основні тенденції, проблеми волатильності та джерела даних.

Досліджено можливості використання відкритих API криптобірж для збору та оновлення ринкової інформації.

Розроблено концептуальну, інформаційну, функціональну, архітектурну та алгоритмічну моделі аналітичної системи.

Створено реляційну базу даних у середовищі PostgreSQL, що забезпечує цілісність і структурованість інформації.

Реалізовано серверну частину на основі фреймворку Flask і модулі аналітичної обробки даних, які використовують алгоритми машинного навчання.

Розроблено інтерактивний модуль візуалізації у середовищі Dash, що дозволяє користувачеві переглядати динаміку ринку, аналізувати тренди та оцінювати точність прогнозів.

Проведено експериментальні дослідження, які підтвердили високу точність прогнозів і стабільність роботи системи при обробці великих обсягів даних.

Виконано тестування компонентів системи, яке довело її працездатність, надійність, масштабованість і відповідність функціональним вимогам.

Розроблена система характеризується модульною архітектурою, що дозволяє розширювати її функціонал без зміни базової структури.

Вона інтегрується з відкритими API криптобірж, автоматично оновлює дані,

формує прогнози та забезпечує їх графічне відображення через веб-інтерфейс.

Отримані результати показали, що модель Gradient Boosting забезпечує найвищу точність прогнозування, тоді як Random Forest демонструє стабільність і швидкість обчислень.

Наукова новизна роботи полягає у розробленні інтегрованої аналітичної системи нового типу, яка поєднує методи машинного навчання з інструментами інтерактивної візуалізації та дозволяє здійснювати автоматизований аналіз ринку криптовалют у реальному часі.

Запропоновано підхід до оцінки якості прогнозних моделей на основі комбінації статистичних і візуальних методів, що підвищує достовірність аналітичних результатів.

Практична значущість полягає у створенні реальної програмної платформи, яку можна використовувати для:

оперативного моніторингу ринку криптовалют;

прийняття інвестиційних і фінансових рішень;

проведення наукових досліджень і навчальних демонстрацій з аналітики даних.

Розроблена система може бути впроваджена як у фінансових організаціях, так і в освітніх закладах для навчання студентів принципам аналітичного мислення, прогнозування та роботи з великими даними.

Перспективи подальшого розвитку системи передбачають:

використання глибинних нейронних мереж для покращення прогнозів;

інтеграцію соціальних і новинних індикаторів настроїв;

створення мобільного застосунку або Telegram-бота;

розширення бази даних за рахунок підключення додаткових бірж та індикаторів;

інтеграцію з платформами бізнес-аналітики.

Таким чином, у результаті виконання магістерської кваліфікаційної роботи створено інноваційну аналітичну систему, яка поєднує сучасні технології машинного навчання, візуалізації та веб-аналітики.

Розроблений програмний продукт є ефективним інструментом підтримки

прийняття управлінських рішень і має потенціал для подальшого розвитку в межах цифрової економіки та фінансових технологій.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Date C. J. An Introduction to Database Systems. — 8th ed. — Boston: Addison Wesley, 2003. — 1024 p.
2. Mitchell T. M. Machine Learning. — New York: McGraw-Hill, 1997. — 414 p.
3. Bishop C. M. Pattern Recognition and Machine Learning. — Springer, 2006. — 738 p.
4. McKinney W. Python for Data Analysis: Data Wrangling with pandas, NumPy, and IPython. — 3rd ed. — O’Reilly Media, 2022. — 544 p.
5. Russell S., Norvig P. Artificial Intelligence: A Modern Approach. — 4th ed. — Pearson, 2021. — 1152 p.
6. Géron A. Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow. — 3rd ed. — O’Reilly Media, 2023. — 896 p.
7. Войцеховський В. В., Биков В. Ю. Інформаційні технології в аналізі даних. — Київ: Наукова думка, 2019. — 276 с.
8. Козак Л. В., Кудрявцев І. М. Технології візуалізації даних у наукових дослідженнях // Вісник НУБіП України. Серія: Інформаційні технології. — 2022. — № 1. — С. 45–53.
9. Коваленко І. С. Використання інструментів Python у задачах фінансової аналітики // Сучасні інформаційні системи. — 2023. — Т. 7, № 2. — С. 112–119.
10. Binance. Binance API Documentation [Електронний ресурс]. — Режим доступу: https://binance-docs.github.io/apidocs
11. CoinGecko. Public API Reference [Електронний ресурс]. — Режим доступу: https://www.coingecko.com/en/api
12. Kraken. REST API Documentation [Електронний ресурс]. — Режим доступу: https://docs.kraken.com/rest
13. Flask. Official Documentation [Електронний ресурс]. — Режим доступу: https://flask.palletsprojects.com
14. Dash (Plotly). User Guide and API Reference [Електронний ресурс]. — Режим доступу: https://dash.plotly.com

15. Scikit-learn. Machine Learning in Python [Електронний ресурс]. — Режим доступу: https://scikit-learn.org/stable
16. PostgreSQL Global Development Group. PostgreSQL Documentation [Електронний ресурс]. — Режим доступу: https://www.postgresql.org/docs
17. Supabase. Open Source Firebase Alternative [Електронний ресурс]. — Режим доступу: https://supabase.com/docs
18. Plotly. Python Graphing Library [Електронний ресурс]. — Режим доступу: https://plotly.com/python
19. Pandas Development Team. Pandas Documentation [Електронний ресурс]. — Режим доступу: https://pandas.pydata.org/docs
20. NumPy Developers. NumPy Documentation [Електронний ресурс]. — Режим доступу: https://numpy.org/doc
21. CoinMarketCap. Cryptocurrency Market Data API [Електронний ресурс]. — Режим доступу: https://coinmarketcap.com/api
22. Кузьменко Т. І. Використання машинного навчання у прогнозуванні фінансових ринків // Інформаційні технології і суспільство. — 2023. — № 1. — С. 67–73.
23. Сидоренко А. П. Системи підтримки прийняття рішень у фінансовій аналітиці. — Київ: КНЕУ, 2022. — 224 с.

ДОДАТКИ

Додаток А

Лістинг скороченого коду веб-додатку аналітичної системи (Python)

```
from __future__ import annotations

from datetime import datetime, timedelta, timezone
from typing import List

import numpy as np
import pandas as pd
import dash_bootstrap_components as dbc
import plotly.graph_objects as go
from dash import Dash, Input, Output, State, dcc, html
from dash import dash_table

from app.config import config
from app.logging_setup import configure_logging
from app.services.db import init_db, load_ohlcv_df, save_ohlcv_df
from app.services.exchanges import fetch_ohlcv
from app.services.indicators import add_indicators, kpis_from_df
from app.services.models import MODEL_REGISTRY, train_and_forecast

# Ініціалізація логування, БД та Dash-додатку
configure_logging("INFO")
init_db()

app = Dash(
    __name__,
    title="CryptoAnalytics — Аналітична панель",
    external_stylesheets=[dbc.themes.DARKLY],
    suppress_callback_exceptions=True,
)
server = app.server


# ----------------------------
# Layout (основні елементи)
# ----------------------------

def kpi_card(title: str, value: str, subtitle: str = "") -> html.Div:
    return html.Div(
        className="kpi-card",
```

```
        children=[
            html.Div(title, className="kpi-title"),
            html.Div(value, className="kpi-value"),
            html.Div(subtitle, className="muted", style={"fontSize": "12px"}),
        ],
    )


def sidebar_dashboard() -> html.Div:
    """Сайдбар з основними фільтрами."""
    return html.Div(
        className="sidebar",
        children=[
            html.H5("ФІЛЬТРИ"),
            html.Div([
                html.Div("Криптовалюта", className="muted"),
                dcc.Dropdown(
                    id="symbol",
                    options=[{"label": s, "value": s} for s in config.symbols],
                    value=config.symbols[0],
                    clearable=False,
                    style={"color": "#000"},
                ),
            ]),
            html.Br(),
            html.Div([
                html.Div("Біржа", className="muted"),
                dcc.Dropdown(
                    id="exchange",
                    options=[{"label": ex, "value": ex} for ex in config.exchanges],
                    value=config.default_exchange,
                    clearable=False,
                    style={"color": "#000"},
                ),
            ]),
            html.Br(),
            # ... інші елементи фільтрів (період, інтервал, модель, слайдер історії)
            dbc.Button("Застосувати", id="apply", color="primary",
className="w-100"),
        ],
    )


def dashboard_layout() -> html.Div:
    """Основний макет аналітичної панелі."""
```

```
    return dbc.Row(
        [
            dbc.Col(sidebar_dashboard(), width=3),
            dbc.Col(
                [
                    dbc.Row(
                        [
                            dbc.Col(kpi_card("ЦІНА ЗАРАЗ", "$—", "+0% за 24г"), width=3,
id="kpi-price"),
                            dbc.Col(kpi_card("ОБСЯГ (24Г)", "$—", "в сумі"), width=3,
id="kpi-volume"),
                            dbc.Col(kpi_card("ВОЛАТИЛЬНІСТЬ", "—", "7д середнє"),
width=3, id="kpi-volatility"),
                            dbc.Col(kpi_card("MA(50) VS MA(200)", "—", "кросовер"),
width=3, id="kpi-trend"),
                        ],
                        class_name="g-2",
                    ),
                    html.Br(),
                    dbc.Row(
                        [
                            dbc.Col(dcc.Graph(id="price-chart", figure=go.Figure()), width=8),
                            dbc.Col(dcc.Graph(id="secondary-chart", figure=go.Figure()),
width=4),
                        ]
                    ),
                    html.Br(),
                    dash_table.DataTable(
                        id="forecast-table",
                        columns=[
                            {"name": "Дата", "id": "date"},
                            {"name": "Актив", "id": "symbol"},
                            {"name": "Модель", "id": "model"},
                            {"name": "Прогноз", "id": "forecast"},
                            {"name": "Факт", "id": "actual"},
                            {"name": "MSE", "id": "mse"},
                            {"name": "R²", "id": "r2"},
                        ],
                        data=[],
                        style_table={"overflowX": "auto"},
                        style_cell={"backgroundColor": "#0f1115"},
                    ),
                ],
                width=9,
            ),
```

```
        ],
        class_name="g-2",
    )


app.layout = dbc.Container(
    fluid=True,
    children=[
        dcc.Location(id="url", refresh=False),
        dbc.Row(
            [
                dbc.Col(html.H4("CryptoAnalytics — Аналітична панель"), width=9),
                dbc.Col(
                    dbc.ButtonGroup(
                        [
                            dbc.Button("Експорт PNG", id="btn-export-png", outline=True,
color="secondary"),
                            dbc.Button("Експорт CSV", id="btn-export-csv", outline=True,
color="secondary"),
                            dbc.Button("Оновити API", id="btn-refresh", color="danger"),
                        ],
                        className="float-end",
                    ),
                    width=3,
                ),
            ],
            class_name="my-2",
        ),
        html.Div(id="page-content"),
    ],
)


@app.callback(Output("page-content", "children"), Input("url", "pathname"))
def render_page(pathname: str):
    """Перемикання між сторінками додатку."""
    # У скороченому лістингу показуємо лише основну панель
    return dashboard_layout()


# ----------------------------
# Робота з даними
# ----------------------------

def _ensure_data(
```

```
    exchange: str, symbol: str, timeframe: str, start: datetime, end: datetime
) -> pd.DataFrame:
    """
    Завантаження OHLCV-даних із БД та за потреби оновлення з біржового API.
    """
    import logging
    log = logging.getLogger(__name__)

    df = load_ohlcv_df(symbol, timeframe, exchange=exchange, start=start, end=end)
    end_naive = pd.Timestamp(end).tz_localize(None)

    if df.empty or (not df.empty and df["timestamp"].max() < end_naive -
pd.Timedelta(hours=1)):
        fetched = fetch_ohlcv(exchange, symbol, timeframe, since=start, until=end)
        save_ohlcv_df(fetched, exchange, symbol, timeframe)
        df = load_ohlcv_df(symbol, timeframe, exchange=exchange, start=start,
end=end)

    if not df.empty:
        df = df.sort_values("timestamp").reset_index(drop=True)

    log.info(f"Dataframe length after ensure: {len(df)}")
    return df


# ----------------------------
# Основний callback панелі
# ----------------------------

@app.callback(
    Output("price-chart", "figure"),
    Output("secondary-chart", "figure"),
    Output("forecast-table", "data"),
    Output("kpi-price", "children"),
    Output("kpi-volume", "children"),
    Output("kpi-volatility", "children"),
    Output("kpi-trend", "children"),
    Input("apply", "n_clicks"),
    State("symbol", "value"),
    State("exchange", "value"),
    State("timeframe", "value"),
    State("model", "value"),
    State("date-range", "start_date"),
    State("date-range", "end_date"),
    State("hist-rows", "value"),
```

```
    prevent_initial_call=True,
)
def update_dashboard(
    _,
    symbol: str,
    exchange: str,
    timeframe: str,
    model: str,
    start_date: str,
    end_date: str,
    hist_rows: int,
):
    """
    Оновлення графіків, KPI та таблиці прогнозів відповідно до вибраних
фільтрів.
    """
    start = datetime.fromisoformat(start_date).replace(tzinfo=timezone.utc)
    end = datetime.fromisoformat(end_date).replace(tzinfo=timezone.utc) + timedelta(
        hours=23, minutes=59
    )

    df = _ensure_data(exchange, symbol, timeframe, start, end)
    if df.empty:
        empty_fig = go.Figure().update_layout(template="plotly_dark", height=420)
        return (
            empty_fig,
            empty_fig,
            [],
            kpi_card("ЦІНА ЗАРАЗ", "—"),
            kpi_card("ОБСЯГ (24Г)", "—"),
            kpi_card("ВОЛАТИЛЬНІСТЬ", "—", "7д середнє"),
            kpi_card("MA(50) VS MA(200)", "—", "кросовер"),
        )

    # Додавання індикаторів та обчислення KPI
    ds = add_indicators(df)
    k = kpis_from_df(ds)

    kpi_price = kpi_card("ЦІНА ЗАРАЗ", f"${k['price']:.2f}" if k["price"] else "—",
"+/- за 24г")
    kpi_volume = kpi_card("ОБСЯГ (24Г)", f"{k['volume24h']:.2f}" if
k["volume24h"] else "—", "в сумі")
    kpi_volatility = kpi_card("ВОЛАТИЛЬНІСТЬ", f"{k['volatility']:.2f}%" if
k["volatility"] else "—", "7д середнє")
    kpi_trend = kpi_card("MA(50) VS MA(200)", k["trend"] or "—", "кросовер")
```

```
    # Основний графік: ціна + ковзні середні
    fig = go.Figure()
    fig.add_trace(go.Scatter(x=ds["timestamp"], y=ds["close"], name="Close"))
    fig.add_trace(go.Scatter(x=ds["timestamp"], y=ds["ma50"], name="MA50"))
    fig.add_trace(go.Scatter(x=ds["timestamp"], y=ds["ma200"], name="MA200"))
    fig.update_layout(template="plotly_dark", height=420)

    # Другий графік: обсяг
    fig2 = go.Figure()
    fig2.add_trace(go.Bar(x=ds["timestamp"], y=ds["volume"], name="Volume"))
    fig2.update_layout(template="plotly_dark", height=420)

    # Прогнозування за допомогою обраної ML-моделі
    fr = train_and_forecast(
        ds.assign(timestamp=pd.to_datetime(ds["timestamp"])),
        timeframe,
        model,
        horizon_days=3,
    )

    # Формування таблиці прогнозів (скорочено)
    table_rows: List[dict] = []
    if fr.in_sample_df is not None and not fr.in_sample_df.empty:
        for _, r in fr.in_sample_df.tail(int(hist_rows or 24)).iterrows():
            table_rows.append(
                {
                    "date": pd.to_datetime(r["timestamp"]).strftime("%Y-%m-%d"),
                    "symbol": symbol,
                    "model": model,
                    "forecast": f"${float(r['yhat']):.2f}",
                    "actual": f"${float(r['actual']):.2f}",
                    "mse": f"{fr.mse:.4f}" if fr.mse == fr.mse else "—",
                    "r2": "—",  # скорочений вивід
                }
            )

    return fig, fig2, table_rows, kpi_price, kpi_volume, kpi_volatility, kpi_trend


if __name__ == "__main__":
    app.run_server(debug=config.debug, host="0.0.0.0", port=8050)
```

Додаток Б

Сторінки користувацького інтерфейсу

Модель	MSE	RMSE	MAPE, %	R^2
Linear Regression	262561.3161	512.4074	0.40	0.83
Random Forest	721064.8319	849.1554	0.25	0.53
Gradient Boosting	667302.8864	816.8861	0.29	0.56

Рис. Б.1 Інтерфейс наукового модуля системи CryptoAnalytics з порівнянням моделей прогнозування (Linear Regression, Random Forest, Gradient Boosting) для криптовалюти BTC/USDT на біржі Binance.

Додаток В

Експорт CSV

date	symbol	model	forecast	actual	mse	r2
2025-11-23	BTC/USDT	Linear Regression	$86932.79	$86928.31	262561.3161	0.83
2025-11-23	BTC/USDT	Linear Regression	$86354.73	$86547.40	262561.3161	0.83
2025-11-23	BTC/USDT	Linear Regression	$87328.10	$87087.44	262561.3161	0.83
2025-11-23	BTC/USDT	Linear Regression	$86461.71	$86685.60	262561.3161	0.83
2025-11-23	BTC/USDT	Linear Regression	$87052.18	$86746.51	262561.3161	0.83
2025-11-23	BTC/USDT	Linear Regression	$87335.50	$86995.02	262561.3161	0.83
2025-11-23	BTC/USDT	Linear Regression	$87493.05	$87325.01	262561.3161	0.83
2025-11-23	BTC/USDT	Linear Regression	$87334.29	$87483.36	262561.3161	0.83
2025-11-23	BTC/USDT	Linear Regression	$87800.94	$87992.02	262561.3161	0.83
2025-11-23	BTC/USDT	Linear Regression	$87410.76	$88004.00	262561.3161	0.83
2025-11-23	BTC/USDT	Linear Regression	$86330.22	$86830.00	262561.3161	0.83
2025-11-24	BTC/USDT	Linear Regression	$87341.32	$86740.64	262561.3161	0.83
2025-11-24	BTC/USDT	Linear Regression	$87752.33	$87068.51	262561.3161	0.83
2025-11-24	BTC/USDT	Linear Regression	$87888.67	$87518.22	262561.3161	0.83
2025-11-24	BTC/USDT	Linear Regression	$87386.79	$87460.02	262561.3161	0.83
2025-11-24	BTC/USDT	Linear Regression	$86559.26	$86738.37	262561.3161	0.83

Рис. В.1 Фрагмент експортованої таблиці прогнозів моделі Linear Regression (CSV-формат): дата, актив, модель, значення прогнозу, фактичні дані, MSE та R^2