МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ВЕТЕРИНАРНОЇ МЕДИЦИНИ ТА БІОТЕХНОЛОГІЙ ІМЕНІ С.З. ҐЖИЦЬКОГО

ФАКУЛЬТЕТ МЕХАНІКИ, ЕНЕРГЕТИКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

першого (бакалаврського) рівня вищої освіти

на тему: **“ Проектування моніторингової системи для синтезу даних з різних API крипто-валютних платформ ”**

Виконав: ст. гр. Кн-41_____

Спеціальності 122 – «Комп’ютерні науки»_____
(шифр і назва)

_____Явдик Максим Романович_____
(Прізвище та ініціали)

Керівник: __к.т.н., доц. Луб П.М._____
(Прізвище та ініціали)

Рецензент: _____
(Прізвище та ініціали)

ДУБЛЯНИ-2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ПРИРОДОКОРИСТУВАННЯ

ФАКУЛЬТЕТ МЕХАНІКИ, ЕНЕРГЕТИКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Перший (бакалаврський) рівень вищої освіти

122 – «Комп'ютерні науки»

"ЗАТВЕРДЖУЮ"
Завідувач кафедри_____
д.т.н., проф. А.М. Тригуба
"____"_____2025 р.

ЗАВДАННЯ

на кваліфікаційну роботу студенту

Явдику Максиму Романовичу

1. Тема роботи: «Проектування моніторингової системи для синтезу даних з різних API крипто-валютних платформ»

Керівник роботи Луб Павло Миронович, к.т.н., доцент
Затверджені наказом по університету від 123/к-с від 25.02.2025.

2. Строк подання студентом роботи – 16.06.2025.

3. Початкові дані до роботи: 1. Довідкова література. 2. Інформаційні джерела розробників мов програмування. 3. Платформи та технологія блокчейну, криптовалюти, майнінгу, транзакцій. 4. API криптобірж. 5. Методика проектування інформаційних систем IDEF.

4. Зміст розрахунково-пояснювальної записки:

1. Аналіз принципів функціонування та реалізації криптовалюти
2. Інструменти прогнозування курсу та робота криптобіржі
3. Розробка моніторингової системи крипто-валютної платформи
4. Реалізація та тестування програмного засобу
5. Охорона праці та безпека в надзвичайних ситуаціях
Висновки;
Список літератури;
Додатки.

5. Перелік презентаційного матеріалу : 1 та 2 – Тема, мета, завдання роботи; 3 – Головні поняття віртуальних грошей; 4 – Аналіз сучасних криптовалют та бірж; 5 – Способи проведення торгів з криптовалютою; 6 – Веб-системи оцінення курсу; 7 – API криптобіржі та його використання; 8 – Концептуальна модель та декомпозиція ПЗ; 9 – Процеси і потоки даних; 10 – Розробка діаграми класів ПЗ; 11 – Реалізація програмного засобу.

6. Консультанти з розділів:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1, 2, 3, 4	*Луб П.М., доцент кафедри інформаційних технологій*		
5	*Городецький І.М., доцент кафедри інженерної механіки*		

7. Дата видачі завдання 25 лютого 2025 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Етапи виконання кваліфікаційної роботи	Строк виконання етапів роботи	Приміт-ка
1.	*Написання першого розділу та означення головних завдань роботи*	25.02.2025 – 01.03.2025	
2.	*Виконання другого розділу та формування головних показників для розрахунків*	01.03.2025 – 01.04.2025	
3.	*Виконання третього розділу, побудова коду та розробка презентації*	01.03.2025 – 01.04.2025	
4.	*Виконання четвертого розділу, реалізація розробки та тестування*	01.04.2025 – 01.05.2025	
5.	*Написання розділу: «Охорона праці та безпека в надзвичайних ситуаціях»*	01.04.2025 – 01.05.2025	
6.	*Завершення оформлення розрахунково-пояснювальної записки та презентації*	01.05.2025 – 01.06.2025	
7.	*Завершення роботи в цілому*	01.06.2025 – 16.06.2025	

Студент _____ Явдик М.Р.
(підпис)

Керівник роботи_____ Луб П.М.
(підпис)

УДК 004.42:004.85:336.74

Проектування моніторингової системи для синтезу даних з різних API крипто-валютних платформ. Явдик М.Р. Кафедра ІТ. – Дубляни, Львівський НУВМБ, 2025.

Кваліфікаційна робота: 62 с. текст. част., 23 рис., 1 табл., 11 слайдів, 23 джерело.

Розкрито теоретичні та практичні аспекти створення програмного засобу для прогнозування курсу криптовалют.

У першому розділі розглянуто основні поняття віртуальної валюти, проведено огляд популярних онлайн криптобірж, а також проаналізовано сучасні веб-системи прогнозування динаміки криптовалютного ринку.

У другому розділі висвітлено принципи здійснення торгових операцій користувачем, описано процес перевірки та поширення транзакцій у блокчейні, а також наведено архітектурні особливості API та його обмеження.

Третій розділ присвячений моделюванню програмного засобу: побудовано структурну модель у техніці IDEF, здійснено декомпозицію процесу прогнозування курсу, розроблено логічну модель та діаграму класів, а також представлено фізичну модель програмного забезпечення.

У четвертому розділі описано реалізацію програмного продукту на базі фреймворку ASP.NET MVC та подано результати його тестування.

П’ятий розділ включає структурно-функціональний аналіз умов експлуатації програмного засобу, зокрема розрахунок освітлення комп’ютерного кабінету та заходи безпеки в надзвичайних ситуаціях.

Робота поєднує технічні, програмні та організаційні аспекти створення сучасного інструменту для роботи з криптовалютами.

Ключові слова: транзакції, блокчейн, криптовалюта, API криптобіржі, веб-додаток, прогнозування.

ЗМІСТ

ВСТУП

Зростаюча потреба у прозорих системах обліку особливо загострилася з розвитком цифрових платіжних технологій, які пред’являють високі вимоги до швидкості обробки даних та рівня безпеки [4, 6]. Одним із негативних наслідків підвищення ефективності інформаційних платформ стала їхня глибока централізація та непрозорість, що суттєво вплинуло на цифрове середовище — зокрема на електронну комерцію, інтернет-спільноти та навіть цілі держави. Можливість від'єднання від платіжних систем, таких як SWIFT, перетворилась на інструмент політичного впливу. Закритість таких систем послаблює довіру, обмежує відкриту конкуренцію та надає привілейований доступ до історії транзакцій лише окремим організаціям. Усе це обмежує повноцінне використання платіжних ресурсів.

До появи децентралізованих цифрових валют, таких як Bitcoin, фінансові інфраструктури були закритими та захищалися здебільшого традиційними механізмами — через спеціалізовані програмно-апаратні рішення: фаєрволи, системи контролю доступу тощо.

Виникнення та функціонування децентралізованих платіжних систем продемонструвало, що фінансова інфраструктура може успішно працювати без централізованого керування, залишаючись водночас відкритою для учасників, з можливістю перевірки та контролю даних. При цьому забезпечується конфіденційність користувачів та захищеність транзакцій за рахунок криптографії, структури даних і їх взаємозв’язків.

Архітектура системи Bitcoin, яка базується на технології блокчейн, має потенціал до застосування у багатьох сферах — від електронного голосування та розрахунків між користувачами до оптимізації ланцюгів постачання й автоматизації виробничих процесів. Блокчейн як модель колективного зберігання та обробки даних є перспективною основою для створення безпечних і прозорих інформаційних рішень.

Мета роботи — розробка програмного додатку для прогнозування курсу криптовалют на основі аналізу даних з криптобіржі з використанням фреймворку ASP.NET MVC.

Завдання кваліфікаційної роботи:

- проаналізувати основні поняття віртуальної валюти, принципи функціонування криптовалютних бірж та інструменти прогнозування курсу криптовалют.

- розробити структурну, логічну та фізичну моделі програмного засобу з використанням методів IDEF-моделювання та UML-діаграм.

- реалізувати програмний продукт для прогнозування курсу криптовалют за допомогою фреймворку ASP.NET MVC.

- провести тестування створеного додатку та проаналізувати його використання.

РОЗДІЛ 1. АНАЛІЗ ПРИНЦИПІВ ФУНКЦІОНУВАННЯ ТА РЕАЛІЗАЦІЇ КРИПТОВАЛЮТИ

1.1. Головні поняття віртуальної валюти

Криптовалюта, або ж цифрова валюта, являє собою електронний засіб обміну, створений та підтримуваний приватними особами або спільнотами. Через те, що більшість таких валют не підпадають під державне регулювання, вони належать до категорії альтернативних валют, тобто функціонують незалежно від офіційної грошової політики країн. Найбільш відомим і першим прикладом такої валюти є Bitcoin. Проте на ринку постійно з'являються нові криптовалюти — так звані «альткоїни», що не мають прямого зв'язку з біткоїном [1, 4].

Основою функціонування криптовалют є використання криптографічних алгоритмів — складних математичних методів шифрування, які забезпечують захист даних під час їх передачі. Завдяки застосуванню сучасних досягнень в галузі криптографії та обчислювальної техніки, ці протоколи забезпечують високий рівень безпеки, роблячи криптовалюти практично захищеними від підробок чи несанкціонованого копіювання. Крім того, вони підтримують анонімність користувачів, ускладнюючи відстеження руху коштів і встановлення особи учасника транзакції.

Блокчейн — це базова структура даних криптовалют, яка забезпечує реєстрацію та збереження всієї історії транзакцій. У будь-який момент вона може підтвердити право власності на будь-яку одиницю валюти. Цей ланцюг блоків постійно розширюється у міру зростання кількості транзакцій, тому має змінну довжину, яка збільшується з часом [16].

Блок транзакцій — це спеціальний формат зберігання кількох транзакцій у системах типу Bitcoin. Після проходження перевірки правильності структури та цифрових підписів, транзакція об’єднується з іншими і вноситься в блок. Таким чином, транзакція визнається завершеною та підтвердженою.

Кожна транзакція — це цифровий запис, за допомогою якого змінюється інформація в базі даних: відбувається переказ монет між адресами. Такий запис містить суму переказу, адресу одержувача та умови, які необхідно виконати для витрачання цих коштів. У структурі транзакції також наявна інформація про джерело отриманих монет (посилання на попередні транзакції), підтвердження прав власності, нові адреси отримувачів та суму операції.

Оскільки кожен новий блок містить хеш попереднього, усі блоки формують єдиний, неперервний ланцюг, який зберігає всі транзакції з моменту створення мережі. Перший блок у цьому ланцюзі — так званий блок-генезис (англ. *genesis block*) — є винятковим, адже він не має попередника (рис. 1.1).

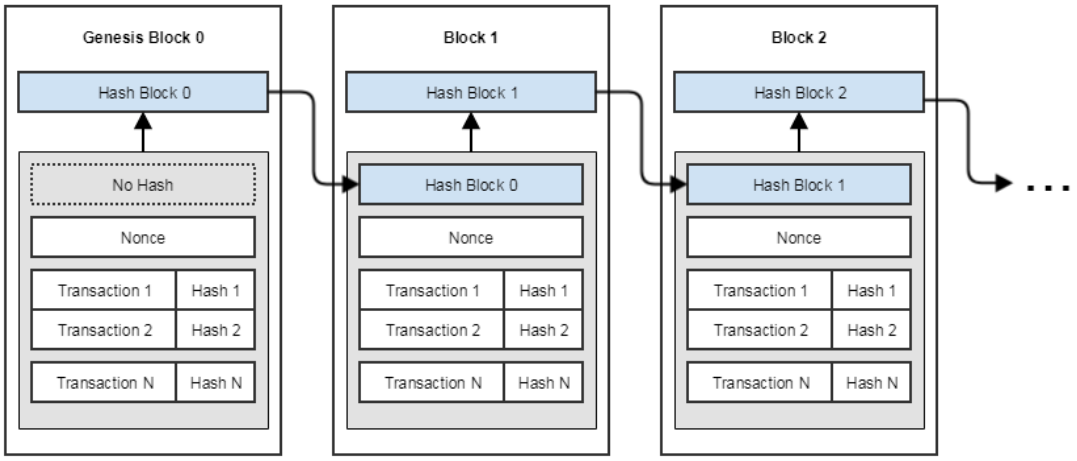


Рисунок 1.1. – Схема отримання хешу транзакцій

Блокчейн представляє собою послідовний ланцюг блоків, який постійно доповнюється новими записами про транзакції. Його копії або окремі частини одночасно зберігаються на великій кількості комп’ютерів у мережі, де синхронізація даних відбувається за чітко визначеними правилами формування блоків. Хоча інформація в блоках не є зашифрованою й доступна публічно, її

незмінність гарантується криптографічними механізмами через використання хеш-зв’язків між блоками.

Життєвий цикл транзакції повністю представлений на рисунку 1.2.

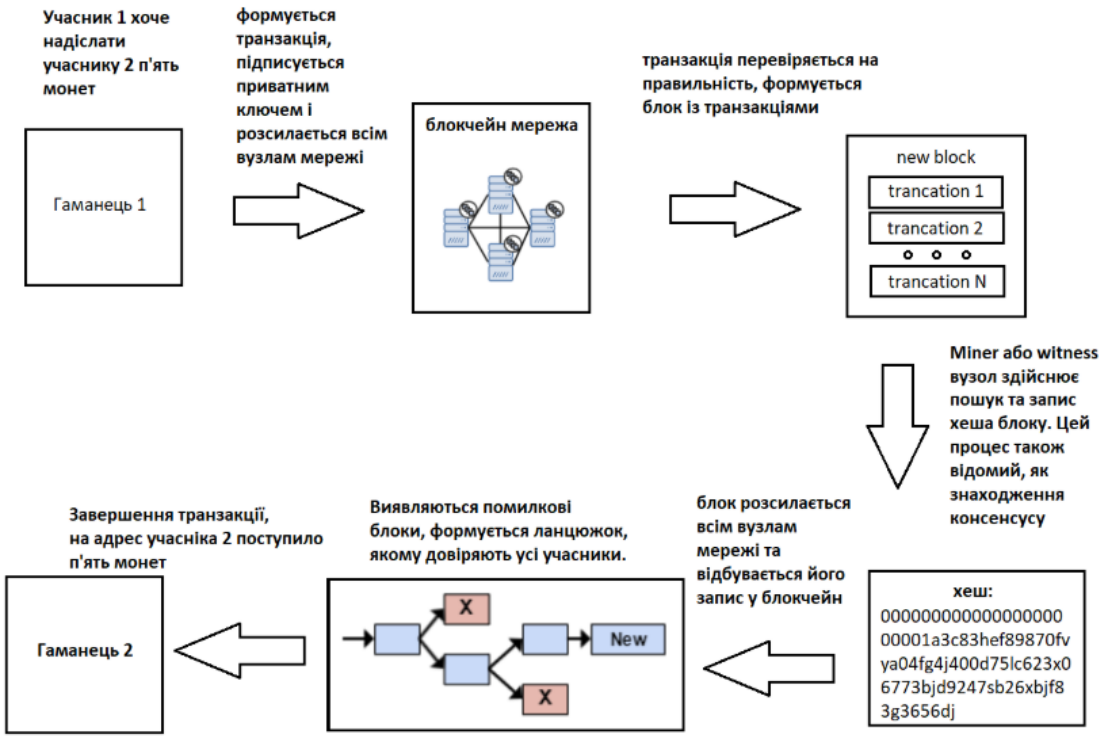


Рисунок 1.2. – Життєвий цикл транзакції в блокчейн

Усі вузли децентралізованої мережі криптовалюти містять ідентичні копії блокчейну. Ці вузли — це сервери, які управляються технічно підготовленими користувачами або об'єднаннями, що відомі під назвою «майнери». Вони займаються верифікацією операцій та внесенням їх до блокчейну. Транзакція вважається завершеною лише після того, як вона потрапляє до ланцюга блоків, що зазвичай займає декілька хвилин.

Щоб забезпечити ефективну обробку великих обсягів даних у стислий проміжок часу, блокчейн-системи застосовують структуру збереження даних, яка називається деревом Меркла. Цю концепцію розробив Ральф Меркл ще у 1979 році [23]. Одним із перших прикладів її практичного використання був протокол BitTorrent. Дерева Меркла дозволяють об’єднати окремі частини інформації в єдине кореневе значення, що дає змогу підтвердити належність

певного фрагмента даних до відповідної структури без потреби перевірки всієї бази.

Дерево Меркла містить такі компоненти (рис. 1.3):

– листя дерева (Merkle leaves);

– вузли дерева (Merkle nodes);

– корінь дерева (Merkle root).

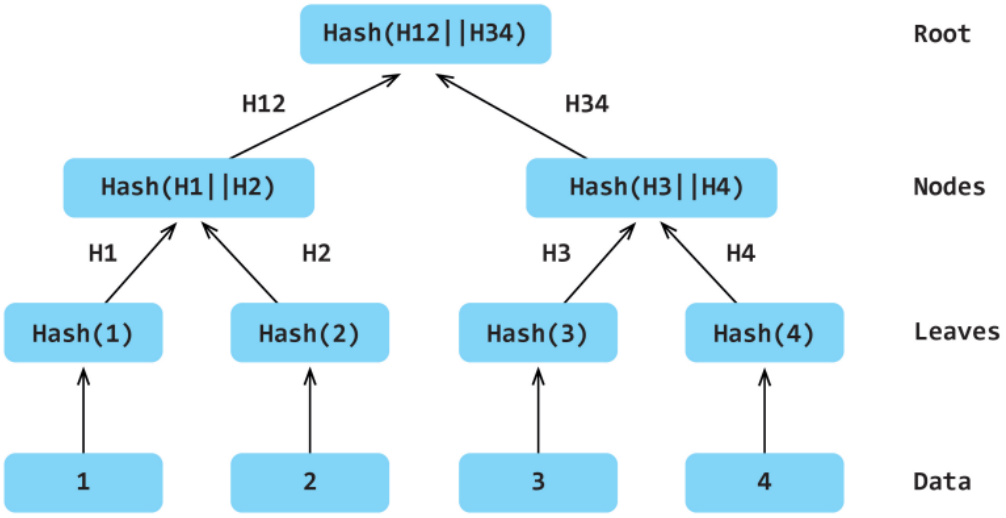


Рисунок 1.3. – Дерево Меркла із компонентами

У структурі дерева Меркла листовими елементами виступають хеші окремих блоків даних, які формують основу цієї структури. Внутрішні вузли дерева утворюються шляхом поєднання (конкатенації) пар дочірніх елементів із подальшим обчисленням хешу від отриманого значення. Кореневий вузол (або Merkle root) — це найвищий рівень у дереві, який слугує підсумковим хеш-ідентифікатором усієї сукупності даних.

Ця схема зберігання має низку переваг, що забезпечують високу швидкість і надійність перевірки достовірності великих обсягів інформації. По-перше, навіть мінімальні зміни в будь-якому з блоків спричиняють зміну всього кореневого хешу. По-друге, у випадку порушення цілісності можна оперативно визначити, який саме блок було змінено. І нарешті, за допомогою дерева Меркла можливо швидко перевірити, чи входить конкретний блок до загальної структури, не аналізуючи всі елементи повністю.

1.2. Огляд популярних онлайн криптобірж

Для об’єктивної оцінки переваг тієї чи іншої криптовалютної біржі необхідно звертати увагу на низку ключових аспектів [6, 12]:

– *Надійність і репутація.* Важливим індикатором стабільності платформи є її репутація на спеціалізованих форумах, таких як Reddit чи bitcointalk. Чим більше позитивних відгуків, тим вищий рівень довіри до ресурсу.

– *Комісійні збори.* Як і будь-які фінансові сервіси, біржі стягують комісію за обмінні операції. Її розмір може варіюватися, але зазвичай становить близько 0,2%.

– *Асортимент валютних пар.* Чим ширший вибір криптовалют для торгівлі, тим більше можливостей для користувача. Це можуть бути як ліквідні основні активи, так і менш популярні альткоїни. Водночас кількість валют не завжди гарантує ефективність торгівлі.

– *Платіжна інфраструктура.* Різноманіття способів поповнення та виведення коштів розширює функціональні можливості біржі. Одні платформи працюють виключно з криптовалютами, тоді як інші також підтримують фіатні валюти, що полегшує фінансові операції для широкого кола користувачів.

– *Політика верифікації.* На деяких біржах існують суворі вимоги до підтвердження особи, які можуть обмежувати обсяги торгів. Інші платформи дають можливість працювати навіть без повної верифікації, але з певними обмеженнями на щоденні транзакції.

– *Зручність інтерфейсу.* Високий рівень зручності та наявність аналітичних інструментів сприяє кращій навігації й дозволяє трейдерам швидше приймати обґрунтовані рішення.

– *Юрисдикція біржі.* Географічне розташування платформи може впливати на доступ до неї. Деякі біржі, наприклад японські чи південнокорейські, можуть працювати лише для громадян цих країн, незважаючи на привабливі умови торгівлі.

– *Технічна підтримка.* Оперативна і якісна служба підтримки дозволяє своєчасно вирішувати технічні труднощі та сприяє безперервності торгових операцій.

Однією з популярних платформ на ринку є ***Poloniex***. Вона надає доступ до понад 140 валютних пар і щодня приваблює понад 5 тисяч користувачів. Інтерфейс біржі зручний у користуванні: на одному екрані можна бачити чат, новини та поточні ринки. Також платформа підтримує детальні графіки з таймфреймами та рівнями Фібоначчі, що робить аналітику більш ефективною [6, 12].

Рисунок 1.4. – Робоче вікно криптобіржі ***Poloniex***

Exmo – це багатофункціональна криптовалютна біржа, яку активно використовує понад 240 трейдерів. Її керівництво включає спеціалістів з різних країн, зокрема і з Росії. Платформа надає можливість купівлі та обміну таких цифрових активів, як Bitcoin, Ethereum, Dogecoin, Litecoin і Dash. Розрахунки можливі як у криптовалюті, так і у фіатних грошах – доларах США, євро, гривнях та рублях.

Для досвідчених трейдерів сервіс пропонує розширений функціонал, зокрема:

1. *stop loss* – обмеження можливих втрат;
2. *trailing stop* – функція, що дозволяє змінювати параметри ціни продажу;
3. можливість отримання позик;
4. створення спеціалізованих ордерів на купівлю чи продаж.

LiveCoin підтримує широкий спектр торгових пар, серед яких *BTC/EUR, BTC/USD, BTC/RUR, EMC/USD, EMC/BTC, LTC/BTC, LTC/EUR, LTC/USD* та інші.

Основні сильні сторони платформи:

– орієнтація виключно на найпопулярніші криптовалюти;

– необмежена кількість виведень коштів на електронні гаманці за короткий період;

– миттєве зарахування фіатних коштів на рахунок;

– простий інтерфейс для обміну, купівлі/продажу валют та здійснення фінансових операцій;

– інтеграція з найбільш популярними платіжними сервісами;

– наявність бонусної програми з адаптивними умовами;

– доступність англомовної та російськомовної версій сайту.

Bitfinex пропонує маржинальну торгівлю, де ліквідність надають самі учасники платформи. Користувачі можуть видавати кредити іншим трейдерам, заробляючи на цьому комісію, яка виступає джерелом прибутку [6, 12].

CEX.IO – це багатофункціональний торговий майданчик, що підтримує як лімітні, так і ринкові ордери. Біржа працює з криптовалютами та фіатними активами, дозволяючи автоматичне поповнення та виведення коштів із мінімальними комісіями. Серед доступних способів – банківські перекази, а також сервіси Skrill, SEPA, AstroPay.

Cryptonit надає змогу здійснювати операції з криптовалютами за євро або долари. Платформа підтримує наступні торгові пари:

– *USD: BTC, NMC, LTC, PPC;*

– *EUR: BTC, NMC, LTC, PPC;*

– *BTC: LTC, PPC, NMC, TRC, FTC.*

Поповнення та виведення коштів можливе через широкий вибір електронних платіжних систем.

BTC-Trade підтримує такі цифрові активи, як *LTC, BTC, PPC, DOGE, DRK, CLR, NVC, RMS*. Зароблені кошти можна виводити на банківські картки Visa та MasterCard. Розмір комісії залежить від напрямку переказу:

– 1,3% при виведенні на картки українських банків;

– $1,95 + 1% — для переказів на закордонні картки.

Bittrex вважається одним із лідерів світового ринку за обсягом криптовалютного обміну.

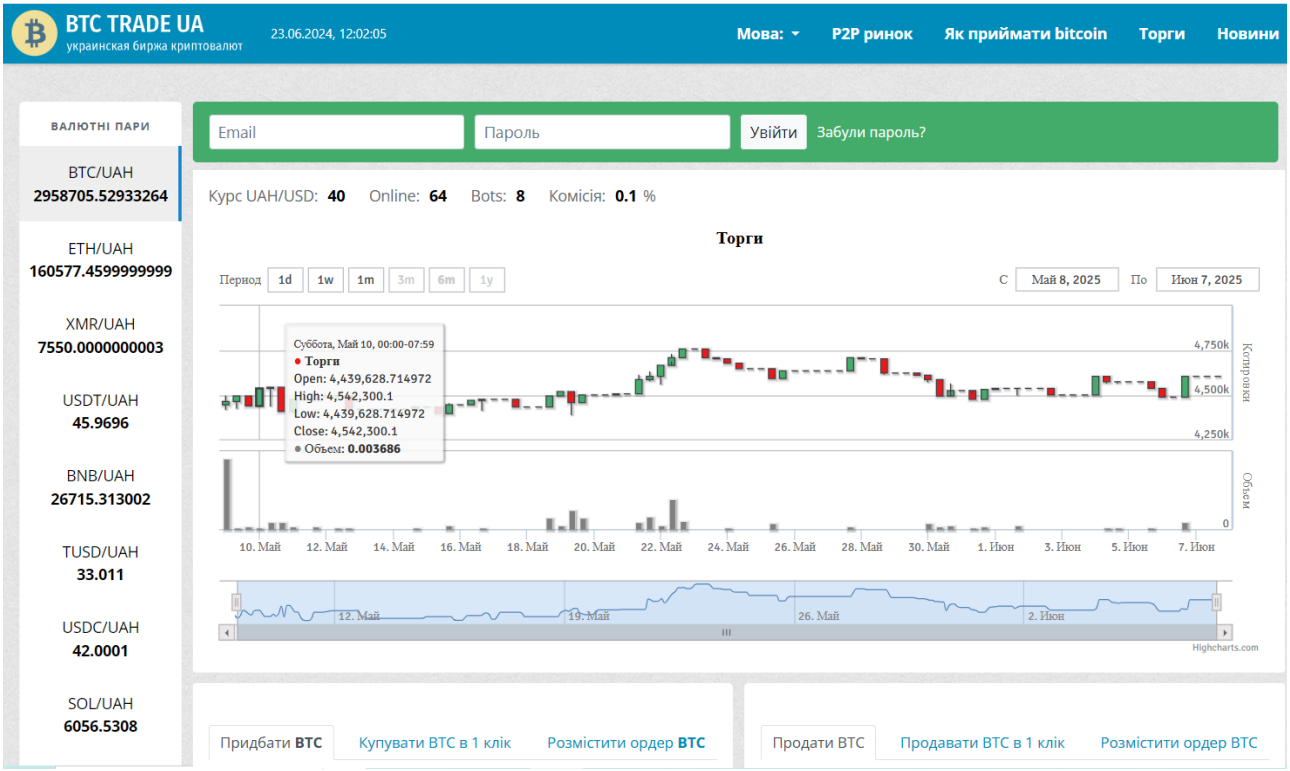


Рисунок 1.5. – Робоче вікно криптобіржі ***BTC-Trade*** [6, 12]

Хоча платформа підтримує широкий вибір торгових пар, виведення у фіатні валюти можливе лише для Bitcoin. Середній розмір комісії за конвертацію становить приблизно 0,5%.

1.3. Аналіз веб-систем для прогнозу руху курсу криптовалют

Trader – система дозволяє прогнозувати рух курсів валют на валютній біржі [6, 12]. Як вихідна використовується інформація за попередніми результатами торгів (часовий ряд): максимальна, мінімальна ціна, ціна закриття і обсяг угод за день. В системі застосовуються наступні алгоритми для аналізу даних:

- ковзне середнє трьох видів – лінійне, експоненціальне, за рангами що задаються;
- MACD-гістограми;
- індикатори – RSI, OBV, Williams R%, CandleSticks, Point & Figure.

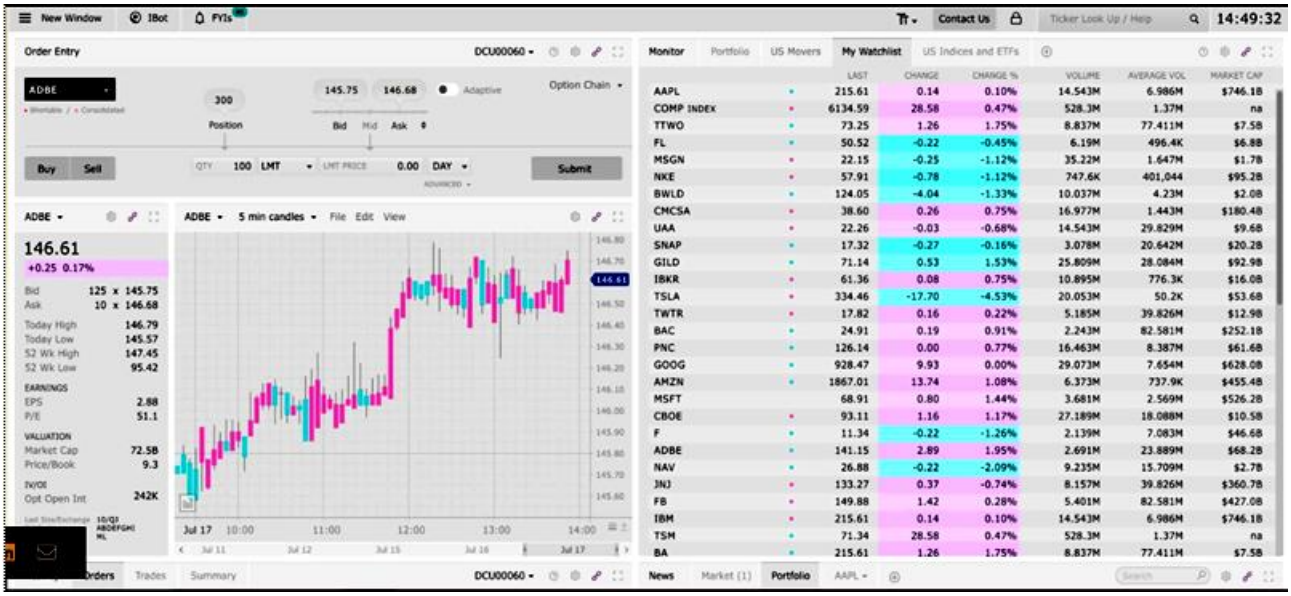


Рисунок 1.6. – Торгова система глобальної платформи *Trader*

Користувач може створювати власні формули для аналізу даних. До переваг також можна віднести можливість застосування індикатора до вже побудованого індикатору, що наприклад потрібно при побудові MACD-гістограми, де ковзне середнє обчислюється для різниці двох ковзних середніх.

Walletinvestor – має вигляд веб-додатку з можливісті вибору однієї конкретної криптовалюти, або ж спостерігати дані про всі наявні з них у вигляді таблиці. Має прогнози на періоди – два тижні, три місяці, пів року, один

рік, п’ять років. Програма пропонує користувачу окрім прогнозів поточний курс найпопулярніших криптовалют і усю попередню інформацію (рис. 1.2).

Binance – провідна в світі біржа криптовалют, що обслуговує 150 мільйонів зареєстрованих користувачів у більш ніж 180 країнах. Завдяки низьким комісіям і понад 350 представленим криптовалютам Binance вважається найкращою біржею для торгівлі Bitcoin, альткоїнами й іншими віртуальними активами.

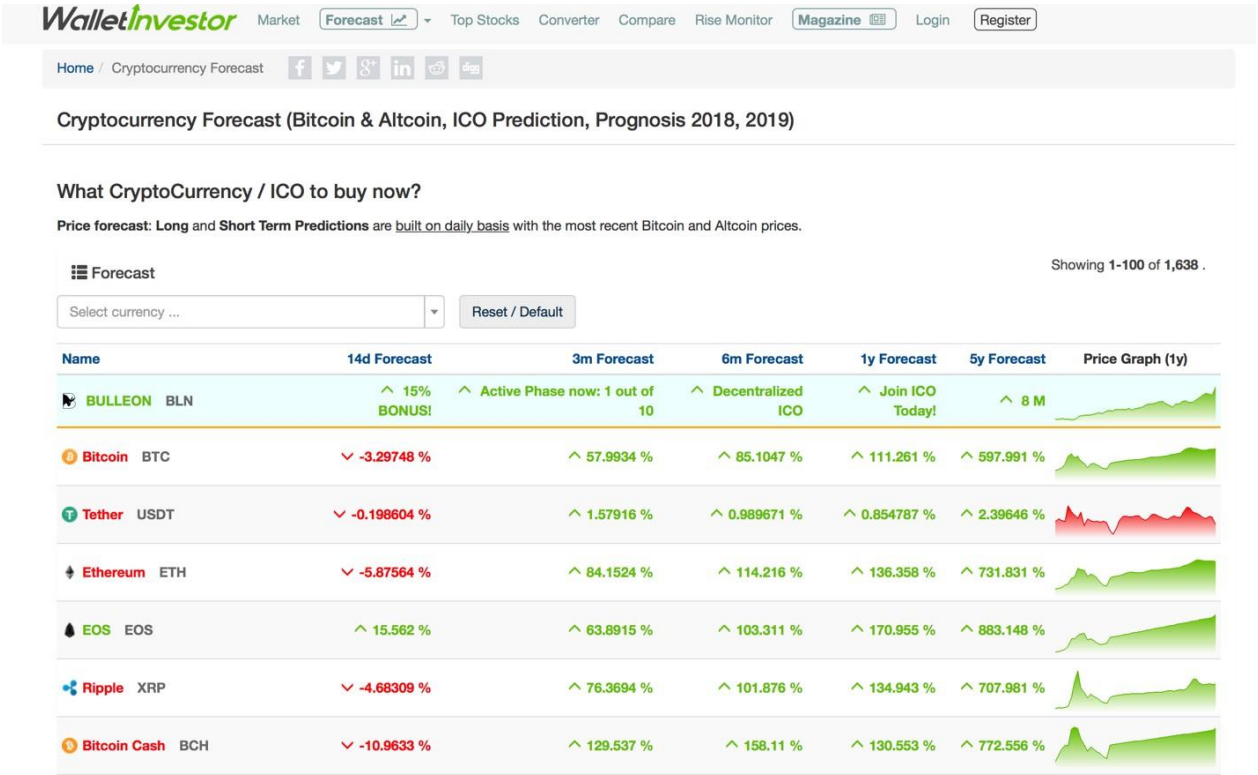


Рисунок 1.7. – Веб-додаток для прогнозів Walletinvestor

З Binance користувачі можуть: 1) торгувати сотнями криптовалют на спотовому, маржинальному і ф’ючерсному ринках; 2) купувати та продавати криптовалюту за допомогою Binance P2P; 3) заробляти відсотки на криптоактивах завдяки Binance Earn; 4) купувати або заробляти нові токени у Binance Launchpad; 5) торгувати, розміщувати в стейкінгу та позичати NFT на маркетплейсі Binance NFT.

Bitcoin. *NeuroShell* – являється цілим набором нейронних мереж, які створені та навчені з єдиною метою – передбачення курсу котирувань валют на

фінансових ринках. Оскільки сам принцип застосування полягає в процесі схованим у «чорному ящику» – користувач не повинен алгоритму обробки, чи певних його частин. А за допомогою мінімалістичного інтерфейсу все приходить лише до задання параметрів та отриманню результату. Тобто з цією процедурою може впоратись людина з будь-яким рівнем досвіду у фінансових операціях. *NeuroShell Day Trader* має ще одну сильну сторону – він вміщує в своїх методах обробки оптимізації, засновані на принципах генетичних алгоритмів. *NeuroShell Day Trader* концентрується на побудові торгової системи.

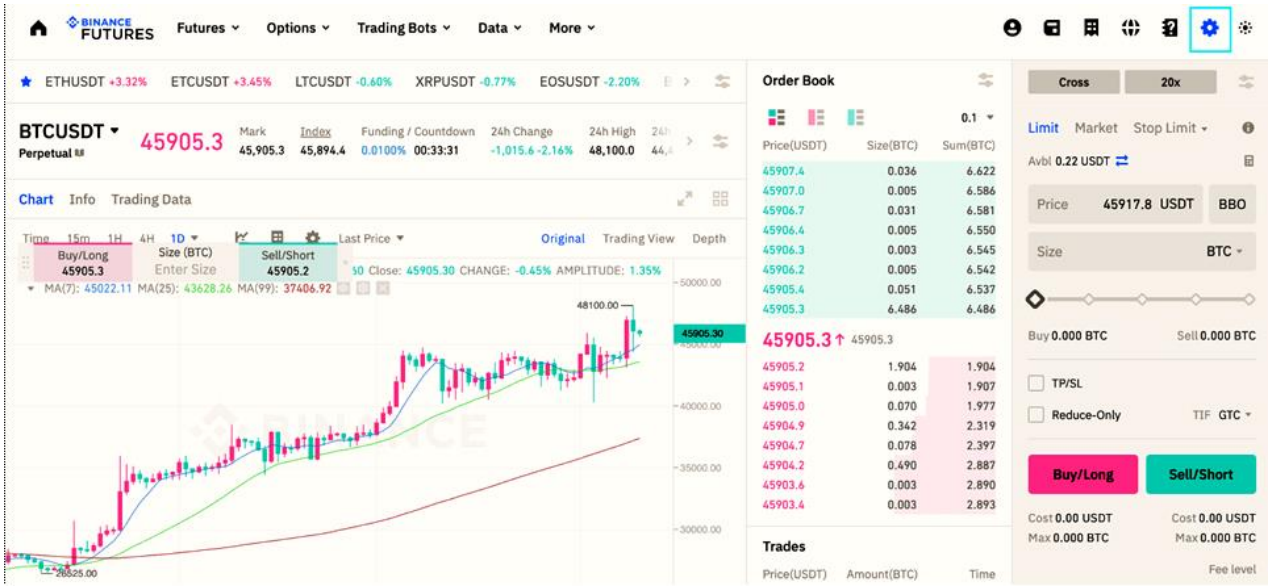


Рисунок 1.8. – Аналітично-прогнозна таблиця торгів в криптобіржі *Binance*

Сама ж торгова система може використовувати в своїх правилах як індикатори, так і спрогнозовані значення, отримані за допомогою нейронних мереж [18].

Elliott wave analyser professional 6.0 – програмний продукт призначений для аналізу валютного ринку з використанням принципів теорії хвиль Елліотта (ТХЕ), а також за допомогою стандартних алгоритмів технічного аналізу. У 1930 році Ральф Елліотт виявив, що емоційний стан натовпу впливає на курси валют, і цей вплив описується декількома зразками, які тепер відомі як хвилі Елліотта.

РОЗДІЛ 2. ІНСТРУМЕНТИ ПРОГНОЗУВАННЯ КУРСУ ТА РОБОТА КРИПТОБІРЖІ

2.1. Принцип виконання торгів із валютою з боку користувача

Біткоїн *(англ. Bitcoin)* — це криптовалюта, створена у 2008 році невідомою групою осіб під псевдонімом «Сатоші Накамото» і запущена в обіг у 2009 році, коли було опубліковано її реалізацію у вигляді програмного забезпечення з відкритим кодом [12].

Ця технологія представляє собою децентралізовану електронну валюту, яка функціонує без центрального банку або єдиного органу управління. Bitcoin пересилається через однорангову мережу, що складається з учасників рівноправних між собою, без посередницьких вузлів. Підтвердження транзакцій здійснюється кінцевими користувачами мережі — так званими «нодами» — за допомогою криптографічних алгоритмів, а самі транзакції фіксуються у публічній базі даних, відомій як блокчейн. Нові біткоїни утворюються у вигляді винагороди за процес майнінгу. Вони можуть використовуватися для обміну на інші валюти, товари або послуги.

Блокчейн являє собою публічний реєстр, що містить записи про всі транзакції Bitcoin у вигляді послідовного ланцюга блоків. Кожен блок включає хеш попереднього блоку, формуючи таким чином єдину неперервну структуру аж до первинного, або генезис-блоку. Підтримка мережі Bitcoin здійснюється через ноди, на яких запущено відповідне програмне забезпечення.

Транзакції в мережі Bitcoin описуються за допомогою мови сценаріїв четвертого покоління. Вони складаються з одного або кількох вхідних і вихідних елементів. При відправленні біткоїнів користувач задає кожну адресу отримувача та суму криптовалюти у вихідних даних. Для запобігання подвійним витратам кожен вхід повинен посилатися на попередні невикористані вихідні дані, зафіксовані у блокчейні. У разі потреби використовується допоміжний вихід, який повертає залишок відправнику.

Вхідні дані, що не враховані у вихідних, формують комісію за проведення транзакції. Приклад структури транзакції наведено на рис. 2.1.

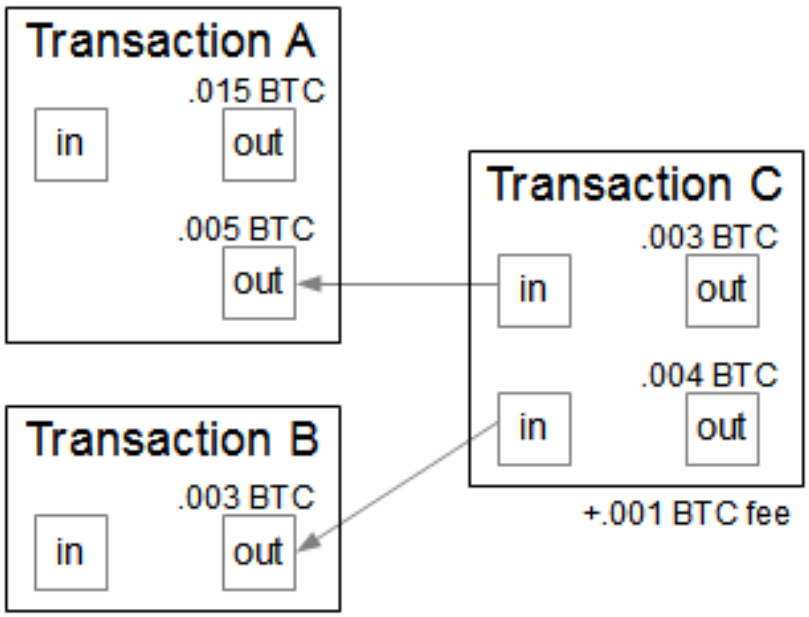


Рисунок 2.1 – Приклад транзакції з біткоїном [13]

Комісії за транзакції. Хоча комісія за проведення транзакції в мережі Bitcoin не є обов'язковою, майнери самостійно визначають, які транзакції вони оброблятимуть, віддаючи перевагу тим, що приносять їм більший дохід. Крім того, при виборі транзакцій майнери враховують співвідношення між розміром комісії та обсягом сховища, необхідним для зберігання транзакції. Комісія за транзакцію вимірюється в одиницях "sat/b" (сатоші на байт), що відображає кількість сатоші, яку сплачують за кожен байт транзакції. Розмір самої транзакції залежить від кількості вхідних та вихідних даних, які вона містить.

Право власності. У блокчейн-системі біткоїни прив'язані до конкретних адрес. Процес створення Bitcoin-адреси полягає у випадковому виборі приватного ключа та генерації відповідної публічної адреси. Ця операція виконується за частки секунди, однак зворотній процес — отримання приватного ключа на основі публічної адреси — вважається практично неможливим. Завдяки цій властивості користувачі можуть безпечно надавати іншим свою публічну адресу, не ризикуючи втратити приватний ключ.

Майнінг (видобуток). Майнінг — це процес підтримки збереження даних у мережі за допомогою обчислювальної потужності. Майнери забезпечують цілісність, послідовність та незмінність системи Bitcoin шляхом об’єднання недавно здійснених транзакцій у блок, який поширюється мережею та підтверджується іншими вузлами. Кожен блок містить хеш SHA-256 попереднього блоку, формуючи таким чином блокчейн. Для прийняття блоку мережею Bitcoin він повинен містити доказ виконаної роботи (proof-of-work, PoW). Система PoW вимагає від майнерів знайти певне число — «nonce», завдяки якому хеш блока матиме значення, що менше за встановлений мережевий поріг складності. Результат легко перевірити, але знайти його дуже складно [21].

Кожні 2016 блоків (приблизно раз на 14 днів, з інтервалом у 10 хвилин на блок) складність мережі автоматично коригується на основі статистики роботи за попередній період. Це дозволяє підтримувати середній час створення блоків близько 10 хвилин, адаптуючись до загальної обчислювальної потужності мережі. Крім того, система PoW, з’єднуючи блоки у ланцюг, робить зміну інформації у блокчейні надзвичайно складною. Щоб успішно змінити один блок, зловмиснику необхідно буде перерахувати усі попередні блоки. Зі збільшенням кількості нових блоків модифікувати історію стає дедалі важче і затратніше за часом.

2.2. Перевірка, підтвердження та поширення блоку транзакцій

У моделі транзакцій Bitcoin передбачено, що за їх обробку сплачується комісійний збір у монетах мережі. Розмір цієї комісії встановлюється відправником у момент створення транзакції і, як правило, повинен перевищувати певний мінімальний поріг. Ця комісія виступає додатковою винагородою для того учасника мережі, який підтверджує транзакцію, додаючи її до свого блоку.

Зі збільшенням популярності Bitcoin суттєво зріс обсяг нових транзакцій у мережі. Водночас протокол встановлює жорстке обмеження на розмір блоку — він не може перевищувати 1 мегабайт. У результаті виникають ситуації, коли кількість нових транзакцій перевищує пропускну здатність мережі. У таких випадках кожен вузол формує чергу непідтверджених транзакцій, в якій першочергово обробляються ті, що мають вищу комісію за одиницю розміру транзакції. Вартість запису даних у блокчейн визначається як співвідношення комісії до розміру транзакції у байтах.

Для створення нового блоку комп'ютер користувача, на якому запущене програмне забезпечення Bitcoin, зберігає весь ланцюг блоків, які вже були підтверджені та визнані дійсними. Проте, оскільки в мережі постійно відбуваються перекази і з'являються нові непідтверджені транзакції, ці дані поширюються між усіма вузлами. Комп'ютер користувача перевіряє ці транзакції та об'єднує ті з них, які вважає коректними, у новий блок (рис. 2.2).

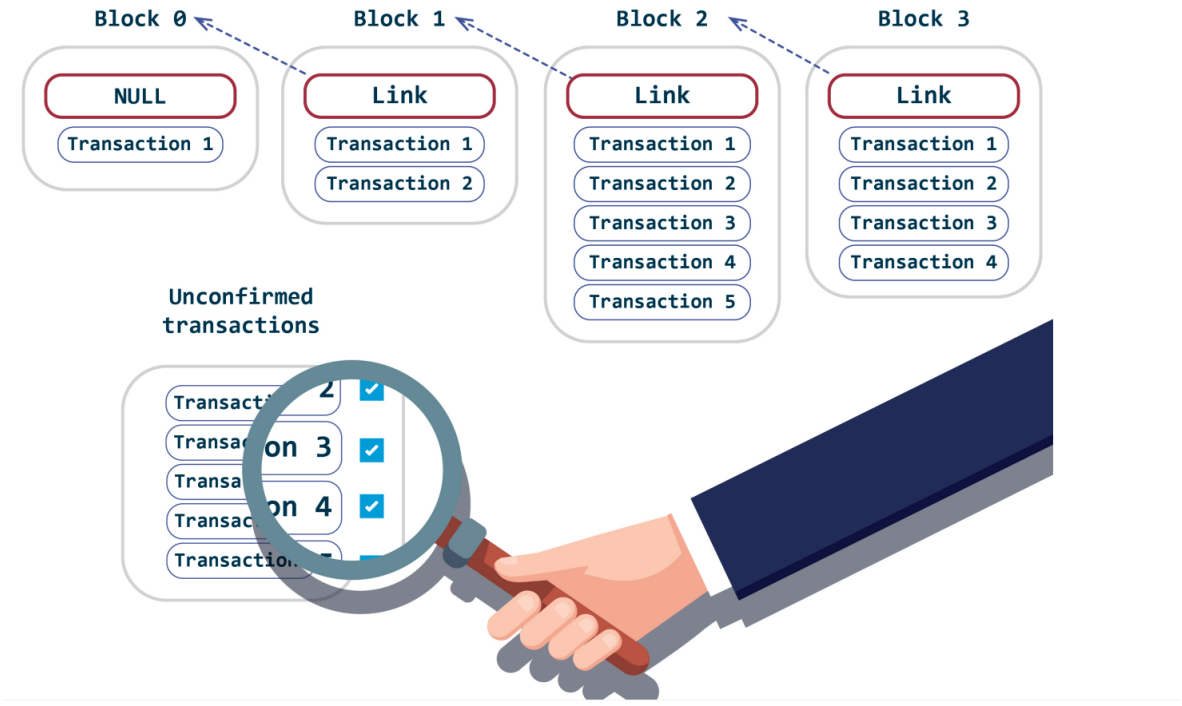


Рисунок 2.2. – Схема створення нового блоку користувачем

Обов'язковою умовою є включення посилання на попередній блок, що забезпечує відстеження історії та цілісність ланцюга. Після формування блоку він пропонується іншим вузлам як наступний елемент загального блокчейну.

Якщо в мережі з’явиться користувач, який почне діяти шахрайським шляхом, пропонуючи іншим вузлам підроблені блоки, або якщо така діяльність буде організована групою зловмисників, що контролюють мережу з модифікованих вузлів (ботнет), які ігнорують протокольні правила, це створить загрозу цілісності блокчейну.

Для захисту від таких атак введено механізм підтвердження блоків за допомогою витрат обчислювальних ресурсів. Блок визнається дійсним лише в тому разі, якщо його створення супроводжувалося розв’язанням складної обчислювальної задачі, що вимагає значних ресурсів. Таким чином, учасник мережі повинен надати доказ виконання «proof-of-work», який інші користувачі можуть легко перевірити і підтвердити правильність блоку. Це унеможливлює масове створення підроблених блоків зловмисниками.

Крім того, даний механізм забезпечує регулювання частоти появи нових блоків і визначення черговості валідаторів. Хоча всі користувачі можуть починати формування нового блоку, право запропонувати свій блок іншим отримує лише той, хто першим розв’язав ресурсомістку задачу.

В основі транзакцій у блокчейні лежать геш-значення, які застосовуються як контрольні суми для перевірки коректності передачі даних. Щоб упевнитися, що інформація не була пошкоджена під час передачі, приймаюча сторона обчислює геш від отриманих даних і порівнює його із заданим значенням. Подібні геш-функції використовують для пошуку дублікатів при зберіганні інформації або для порівняння великих обсягів даних. Замість безпосереднього порівняння великих масивів, порівнюють лише їх геші, що значно прискорює процес.

Якщо геші двох наборів даних співпадають, це свідчить про високу ймовірність ідентичності самих даних. Також під час цифрового підпису зазвичай підписується не саме повідомлення, а його геш-значення. Це значення разом із підписом передається одержувачу, який таким чином може перевірити і цілісність повідомлення, і правильність підпису.

Для перевірки цілісності даних у блоках використовується структура двійкового дерева Меркла (див. рис. 1.3). Верифікація і-го блоку у різних рівнях ієрархії виконується за допомогою кортежу (2.1):

$$\left\langle \left(h_n^i, h_n^{i+1}\right), \left(h_{n-1}^i, h_{n-1}^{i+1}\right), ..., \left(h_1^i, h_1^{i+1}\right), h_0 \right\rangle, \qquad (2.1)$$

де h – геш-код блоку даних; i – номер блоку; n – загальна кількість блоків у ланцюгу.

Для підтвердження цілісності блоку потрібно виконати не більше O(log2 n) операцій та порівнянь геш-кодів. Загальна схема побудови «ланцюжка блоків» із застосуванням дерева Меркла показана на рисунку 2.5.

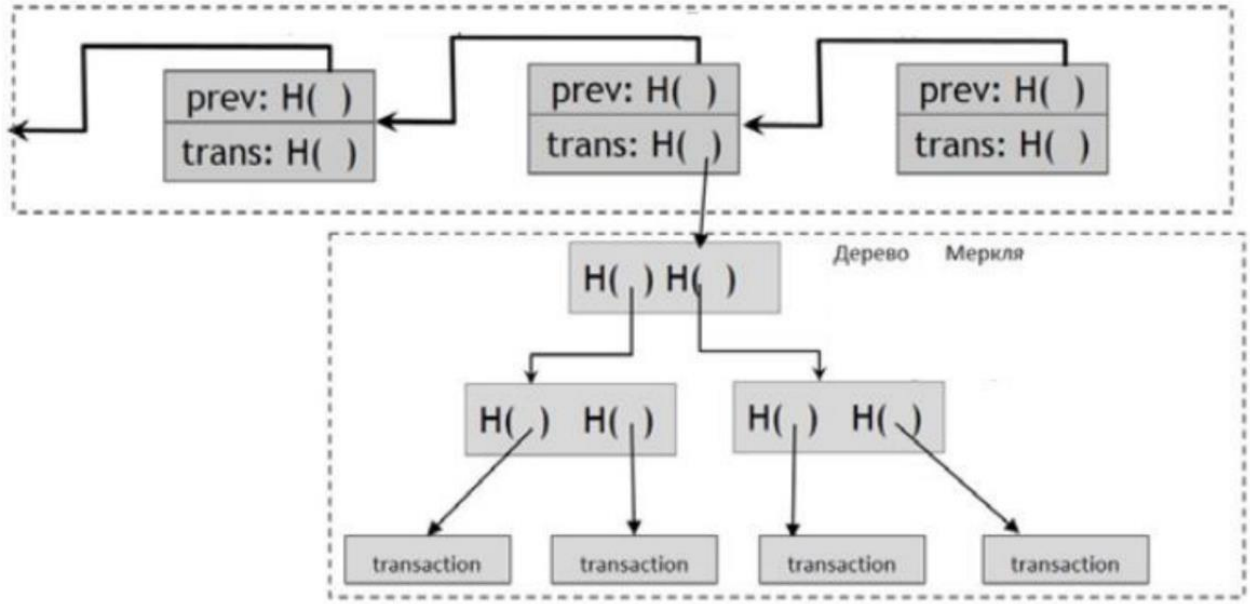


Рисунок 2.3. – Загальний принцип побудови конструкції «Ланцюжка блоків транзакцій»

У структурі кожного блоку в мережі Bitcoin можна виокремити дві основні частини: 1) заголовок, що включає ключову інформацію – мітку часу створення, посилання на попередній блок у ланцюгу та хеш кореня дерева Меркла, яке охоплює всі транзакції в межах блоку [23]; 2) список транзакцій, що безпосередньо входять до складу блоку.

Ідентифікація окремого блоку здійснюється шляхом подвійного хешування його заголовка за допомогою алгоритму SHA-256 [20]. Результат обчислення належить до діапазону $[0,2^{256}-1]$.

Загальна хеш-функція в контексті різних реалізацій блоку позначається як hash(дані). Її вхідними параметрами можуть бути кілька бінарних послідовностей, які конкатенуються в один рядок перед подачею на вхід SHA-256. Значення функції буде належати до проміжку [0; M], де M — максимально можливе значення у відповідній реалізації.

Для визнання блоку дійсним у системі, що використовує доказ виконаної роботи (Proof-of-Work), результат хешування повинен бути меншим або дорівнювати певному пороговому значенню (2.2):

$$hash(X) \leq MD, \qquad (2.2)$$

де D – параметр складності в діапазоні [1, M]; X – вхідний набір даних (зокрема заголовок блоку); M – максимальне значення області значень.

На сьогодні не існує ефективного способу знаходження такого значення X, окрім як перебирати можливі варіанти заголовків доти, доки умова (2.2) не буде виконана. Чим вищий рівень складності D, тим більше обчислювальних кроків потрібно здійснити. Середнє очікуване число ітерацій прямо пропорційне D.

Таким чином, можна стверджувати, що блокчейн характеризується високим рівнем захисту, оскільки жоден окремий учасник мережі не здатен впливати на результати валідації для прискорення чи компрометації процесу. Усі операції з формування і перевірки блоків ґрунтуються на використанні математичних алгоритмів і криптографічних методів, що забезпечує прозорість і цілісність системи.

2.3. Архітектура API та її обмеження

API (*Application Programming Interface*) – це інтерфейс прикладного програмування, який забезпечує взаємодію між окремими програмними компонентами через заздалегідь визначені протоколи та структури [6].

У даному випадку термін «додаток» охоплює будь-яке програмне забезпечення з конкретною функціональністю. Інтерфейс виконує роль умовної угоди між програмами, що описує правила взаємодії: як саме клієнтська програма надсилає запити, а серверна – надає відповіді. У технічній документації до API зазвичай детально прописано, яким чином мають бути оформлені ці запити і як очікується отримання відповідей.

Типова структура API базується на взаємодії між двома сторонами: клієнтом, який ініціює запити, та сервером, який надає відповідь. Залежно від призначення та періоду створення, API реалізуються в різних формах. Однією з ранніх моделей є SOAP (*Simple Object Access Protocol*) – це протокол для обміну даними між програмами, що використовує формат XML для передачі повідомлень. Такий підхід є менш гнучким порівняно з новішими API, хоча раніше він був досить поширеним.

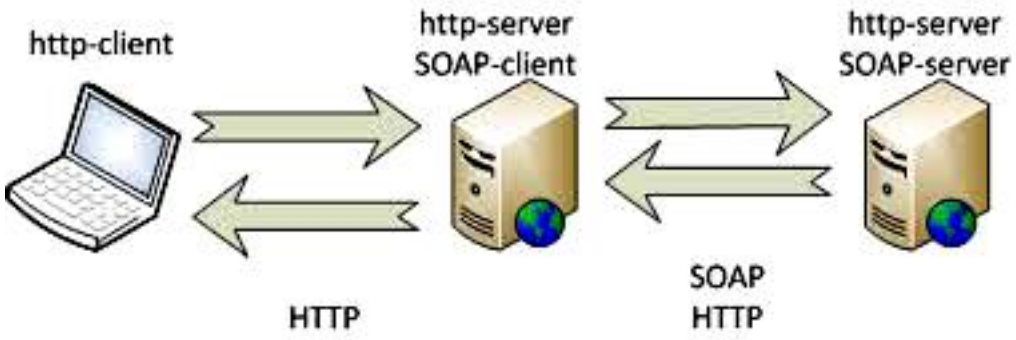


Рисунок 2.4. – Схема роботи SOAP

RPC API – такі API називаються системою віддаленого виклику процедур. Клієнт виконує функцію (або процедуру) на сервері, і сервер надсилає результат назад клієнту.

Websocket API – це ще одна сучасна розробка web API, яка використовує об'єкти JSON для передачі даних. WebSocket API підтримує двосторонній зв'язок між клієнтськими програмами та сервером. Сервер може надсилати повідомлення зворотного дзвінка підключеним клієнтам, що робить його ефективнішим, ніж REST API.

REST API – на сьогоднішній день це найпопулярніші та гнучкіші API–інтерфейси в Інтернеті. Клієнт надсилає запити на сервер у вигляді даних. Сервер використовує це введення клієнта для запуску внутрішніх функцій і повертає вихідні дані назад клієнту. REST – *Representational State Transfer*, тобто передача репрезентативного стану. REST визначає набір функцій, таких як GET, PUT, DELETE тощо, які клієнти можуть використовуватиме доступу до даних сервера. Клієнти та сервери обмінюються даними за протоколом HTTP.

Рисунок 2.5. – Схема роботи REST API

Головною особливістю REST API і те, що така передача виконується без збереження стану. Без збереження стану означає, що сервери не зберігають дані клієнта між запитами. Клієнтські запити до сервера подібні до URL-адрес, які ви вводите в браузері для відвідування веб-сайту. Відповідь від сервера є простими даними без типового графічного відображення веб–сторінки [18].

Усі API повинні бути захищені за допомогою належної автентифікації та моніторингу. Опишемо два основні способи захисту REST API.

Токени автентифікації використовуються для перевірки особистості користувачів під час здійснення API-запитів. Вони забезпечують підтвердження, що користувач справді є тим, за кого себе видає, і має відповідні права доступу для виконання певної дії через API.

На відміну від токенів, API-ключі перевіряють не користувача, а саме застосунок або програму, що надсилає запит. Вони слугують для розпізнавання джерела звернення та контролю доступу до окремих функцій API. Хоча ключі API вважаються менш безпечними, ніж токени, їхня перевага полягає в тому, що

за допомогою них можна здійснювати моніторинг активності та аналізувати, як саме використовується API.

Ключовим принципом архітектури RESTful API є уніфікований інтерфейс, що забезпечує стандартний підхід до передачі інформації між клієнтом і сервером. Інформація подається у форматі, що називається представленням ресурсу. Такий формат може відрізнятись від того, в якому ресурс зберігається всередині сервера. Наприклад, хоч сервер зберігає дані у вигляді тексту, він може передавати їх у вигляді HTML-сторінки.

Цей уніфікований інтерфейс накладає чотири основні архітектурні вимоги:

- Кожен запит повинен мати унікальний ідентифікатор ресурсу, який він запитує;
- Представлення ресурсу має містити всю інформацію, необхідну клієнту для редагування або видалення ресурсу, включно з метаданими;
- Сервер надсилає самодокументовані повідомлення, які включають інформацію про те, як клієнт має інтерпретувати отримані дані;
- У відповідь включаються гіперпосилання на пов’язані ресурси, що дозволяє клієнту досліджувати додаткові дані в динамічному режимі.

РОЗДІЛ 3. РОЗРОБКА МОНІТОРИНГОВОЇ СИСТЕМИ КРИПТО-ВАЛЮТНОЇ ПЛАТФОРМИ

3.1. Структурна модель та декомпозиція програмного засобу в IDEF Modeling Techniques

З метою задоволення вимог щодо розробки програмного забезпечення для аналізу й прогнозування курсу криптовалют, було описано основні принципи його функціонування (рис. 3.1) [9].

У концептуальній моделі закладено такі ключові компоненти – Вхідні дані: це «вибрана криптовалютна пара» та «зазначений період прогнозування». На основі цих параметрів формується HTTP-запит до API криптобіржі. Отримана інформація використовується як вхідні характеристики для обробки нейромережею з архітектурою рекурентного перцептрону, а також для математичного аналізу, що здійснюється із застосуванням методу експоненційного згладжування, коефіцієнта кореляції Пірсона (PCC) та осциляторів типу R. Після цього дані оновлюються.

Ціни криптовалют зазвичай встановлюються у відношенні до долара США (USD) або біткоїна (BTC). Проте, для гнучкості програмного засобу передбачено можливість самостійного вибору валютної пари, для якої буде виконуватись аналіз курсу.

Таким чином, до API криптобіржі буде надсилатися запит щодо двох криптовалют. Історичні дані за цими активами вже зберігаються на серверах відповідних торгових платформ, однак під час роботи ПЗ слід враховувати обмеження на обсяг і частоту API-запитів:

- для короткострокового прогнозування використовуються дані з високою деталізацією — інтервали в межах 1-30 хвилин;
- для довгострокових оцінок запитуються агреговані дані за тривалі періоди — від одного до семи днів.

Додаткові обмеження накладаються умовами криптобірж — це зокрема,

ліміти на частоту запитів та обмежений доступ до історичних даних. Різні біржі надають неоднаковий обсяг інформації, керуючись власною політикою відкритості. Потенційно підтримувані біржі: Binance, KUNA, YoBit [6].

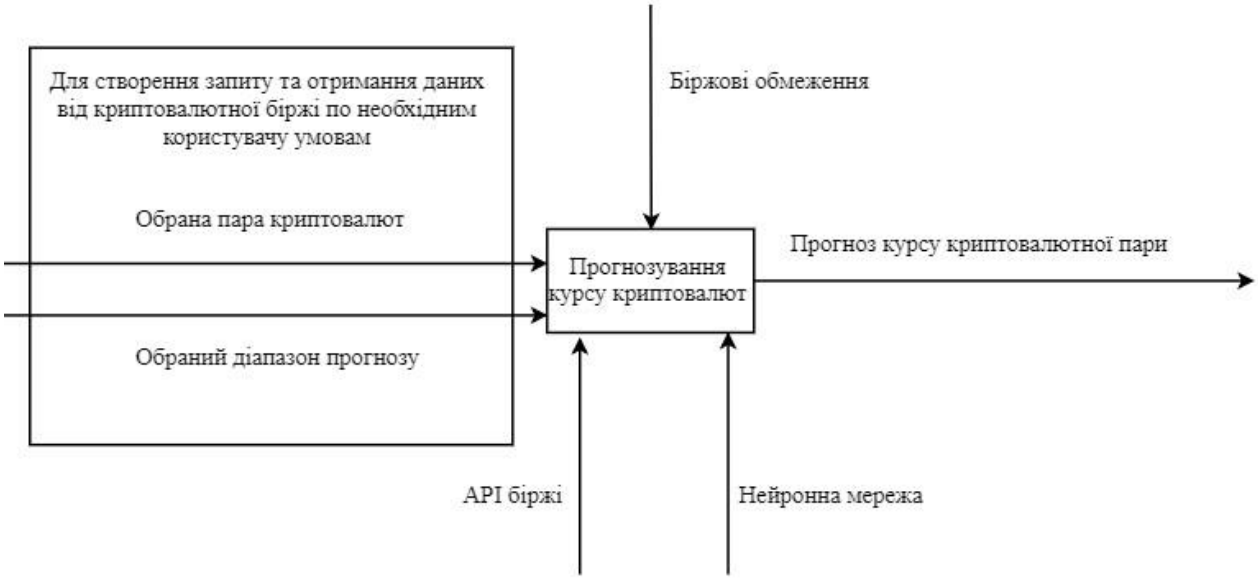


Рисунок 3.1. – Концептуальна модель ПЗ

Функціональні механізми роботи програмного забезпечення умовно поділяються на дві ключові складові:

- API криптобіржі — забезпечує обробку запитів щодо вибраної криптовалютної пари у визначеному часовому інтервалі. У відповідь API надає такі параметри, як поточний курс, кількість транзакцій, їхні максимальні та мінімальні значення, обсяг торгів, часові мітки активності тощо.

- Модуль нейронної мережі (або штучного інтелекту) — являє собою набір нейромереж, які функціонують автономно. Перед практичним використанням кожну з моделей необхідно попередньо навчити на відповідних вибірках історичних біржових даних [2, 6, 18]. Після навчання вони інтегруються в основну систему, виконують обробку отриманих API-даних, генерують прогноз курсу, а також аналізують кілька варіантів прогнозів для визначення найімовірнішого.

Вхідні параметри: це прогнозоване значення курсу для конкретної криптовалютної пари, яке формується у вигляді часового ряду відповідно до

заданого діапазону. Отримані результати виводяться у графічному інтерфейсі ПЗ у вигляді таблиці або графічної візуалізації.

Деталізована **декомпозиція концептуальної моделі програмного забезпечення виконана за стандартом IDEF0.** Основні логічні процеси, що реалізуються в ході функціонування ПЗ, відображені на рисунку 3.2.

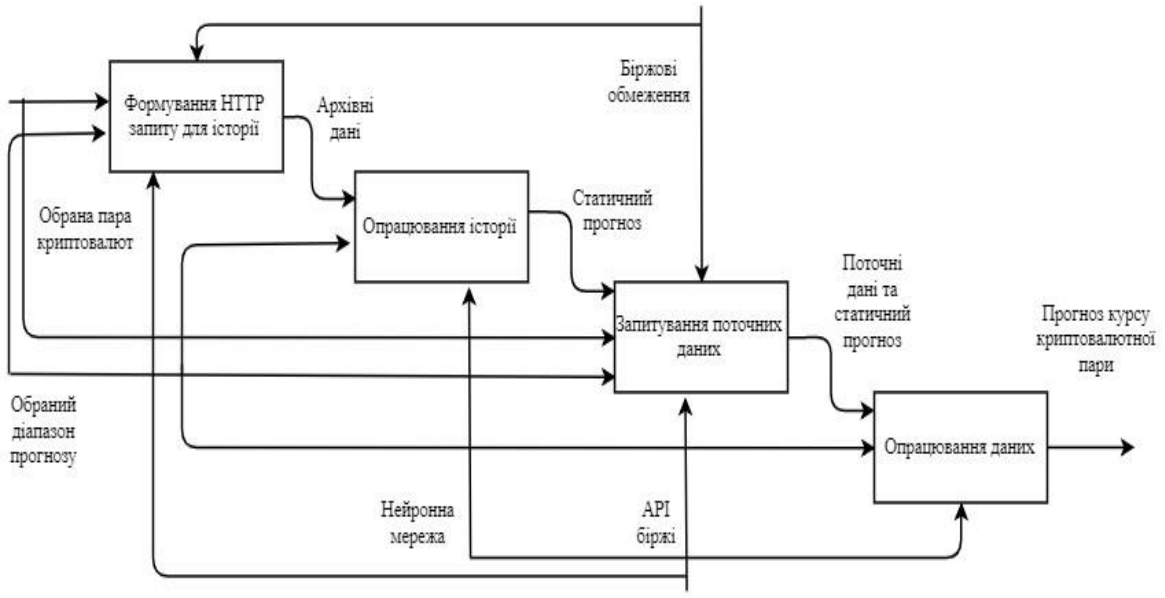


Рисунок 3.2. – Декомпозиція моделі ПЗ в IDEF0

Отримання історичних даних – на цьому етапі формується HTTP-запит, який базується на вхідних параметрах: вибраній парі криптовалют та зазначеному часовому проміжку для прогнозу. Сформований запит надсилається до API відповідної біржі.

Аналіз історичних даних – після отримання відповіді від API у форматі JSON відбувається її парсинг: із JSON-структури вибираються лише ті значення, що необхідні для подальших розрахунків [12] . На основі витягнутих даних створюються об’єкти, що будуть передані до нейронної мережі та математичних модулів (як-от алгоритм експоненціального згладжування, PCC та інші). Далі ці об’єкти проходять обробку, а результати передаються до нейромережевого модуля, що виконує зіставлення прогнозів і формує підсумковий варіант статичного прогнозу.

Отримання актуальних даних – цей процес є ключовим елементом

циклічної роботи програми в режимі реального часу. У цьому режимі ПЗ з певною періодичністю звертається до API для оновлення даних і відображає нову інформацію користувачу.

Аналіз актуальних даних – на цьому етапі відбувається додавання нових даних до наявного масиву в залежності від заданого часово́го горизонту прогнозу. Далі запускається повторна обробка оновленого набору через нейромережу та математичні методи. Якщо ж нові дані охоплюють занадто короткий проміжок часу, і суттєвих змін у порівнянні з попередніми немає, то вони фільтруються як надлишкові – не впливаючи на підсумковий прогноз, який у такому випадку залишається незмінним (статичним).

3.2. Декомпозиція моделі процесу «Прогнозування курсу крипто-валюти»

Структуру потоків інформації, основних процедур та меж системи наочно демонструє рисунок 3.3, де представлено декомпозицію концептуальної моделі у форматі DFD-нотації.

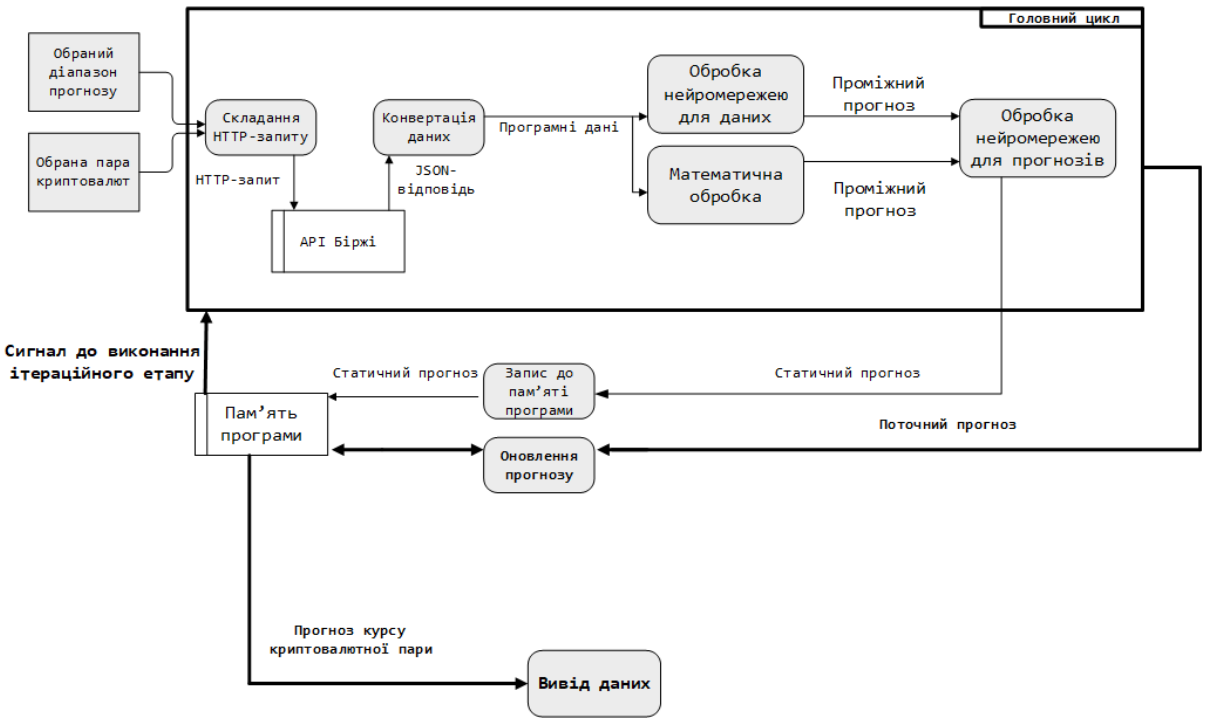


Рисунок 3.3. – Декомпозиція концептуальної моделі ПЗ в DFD

Ключовим елементом системи є процес «Прогнозування курсу криптовалют». На вхід йому передаються параметри: вибраний користувачем часовий діапазон прогнозу та конкретна пара криптовалют. Результатом обробки слугує часовий ряд прогнозних значень курсу, який надається користувачу у зручному форматі — графік, табличне подання, текстовий файл або стисле повідомлення із ключовими прогнозними точками.

У якості сховищ даних використовуються такі компоненти:

API біржі — джерело інформації про поточні та історичні операції з криптовалютами, включаючи часові мітки, курси обміну та обсяги угод;

Оперативна пам'ять програми — слугує для тимчасового зберігання оброблених даних і сформованого прогнозу. Після завершення кожного сеансу роботи або нового запиту дані оновлюються, а попередні – видаляються. Це сховище виконує роль проміжної області зберігання в межах одного сеансу взаємодії з користувачем. Для чіткого поділу між етапом ініціалізації та наступними оновленнями даних, ця частина системи виділена окремо.

На діаграмі представлено також підпроцеси, що відповідають за окремі етапи обробки даних і взаємодіють між собою шляхом обміну інформаційними потоками:

«Формування HTTP-запиту» — побудова запитного рядка згідно з введеними параметрами користувача та його передача до API відповідної криптобіржі.

«Перетворення вхідних даних» — трансформація отриманої у форматі JSON або бінарному відповіді у структури даних (об'єкти), що обробляються у динамічній пам'яті програми.

«*Обробка вхідних даних нейромережею*» — використання нейронної мережі, побудованої на основі радіально-базисних функцій, для генерації базового (атомарного) прогнозу.

«Математичний аналіз» — паралельне опрацювання тих самих вхідних об'єктів математичними моделями (експоненціальне згладжування, РСС, осцилятори R).

«*Об'єднання прогнозів нейронною мережею*» — рекурентна нейромережа здійснює об’єднання результатів, отриманих з різних джерел, у єдиний узгоджений прогноз.

«*Оновлення прогнозної моделі*» — додавання актуального прогнозу до тимчасового сховища програми у вигляді структурованих об'єктів.

«*Виведення результатів*» — подання прогнозу кінцевому користувачу у вибраному ним форматі.

3.3. Розробка логічної моделі та діаграма класів програмного засобу

Основні функціональні вимоги до серверної та клієнтської частин системи викладені в Додатку А, таблицях А.1 та А.2 відповідно.

Для побудови моделі варіантів використання необхідно визначити основних дійових осіб — акторів системи.

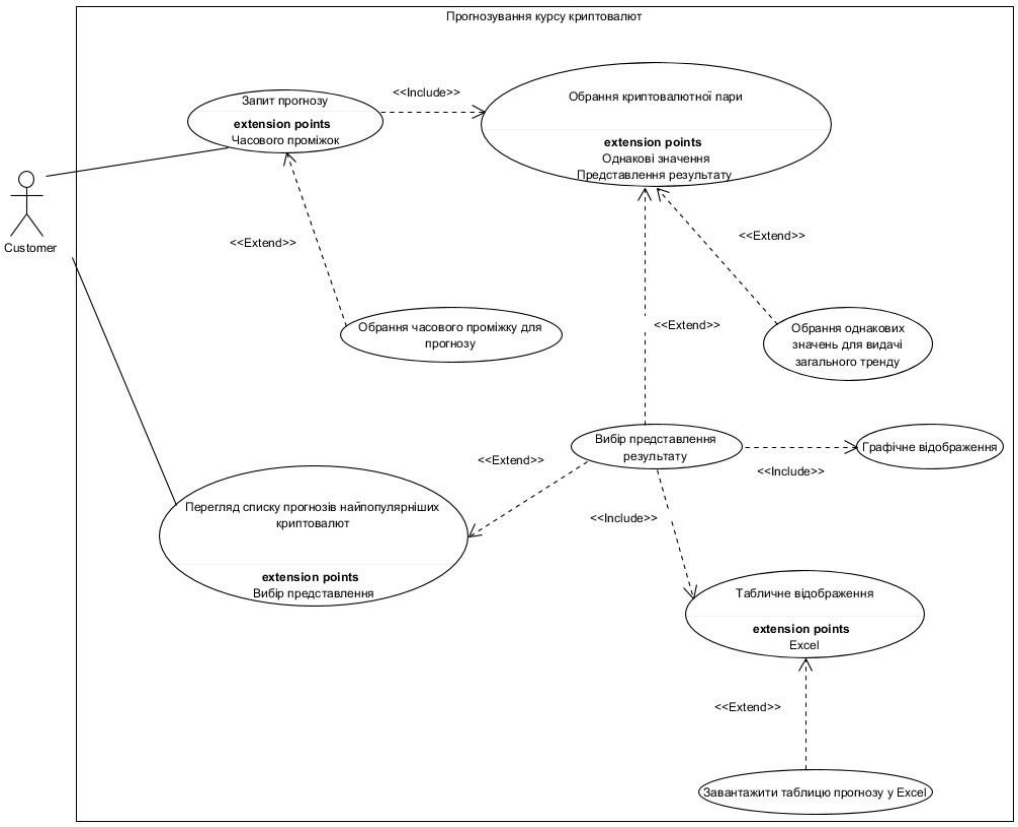


Рисунок 3.4. – Структурна схема варіантів використання програмного засобу прогнозування курсу криптовалют

Враховуючи, що призначення програмного забезпечення полягає у наданні користувачеві можливості обирати криптовалютну пару та часовий інтервал для прогнозування, а також переглядати отримані результати у зручному форматі (таблиця або графік), можна стверджувати, що єдиним актором системи є користувач.

Детальний опис усіх варіантів використання наведено в Додатку Б.

Логічна модель даних розроблена незалежно від конкретної технології реалізації програмного забезпечення і призначена для відображення основних сутностей об’єктно-орієнтованої моделі, зокрема класів, а також для точного опису роботи системи.

Для створення програмного продукту необхідно розбити інформацію, отриману через API криптовалютної біржі, на окремі самостійні одиниці, які впливають на попит певної криптовалюти та слугують основою для формування часових рядів, що використовуються для прогнозування.

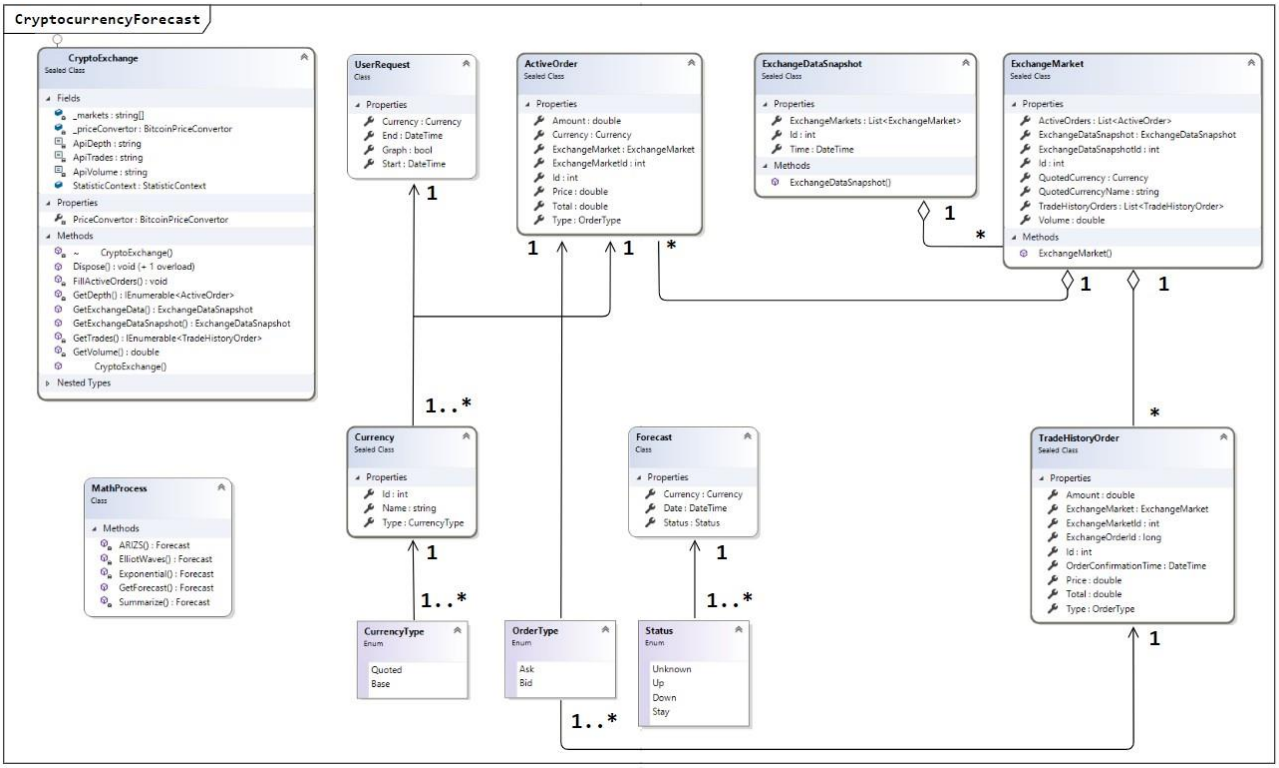


Рисунок 3.5. – Діаграма класів ПЗ

Важливим аспектом проєктування є вибір конкретної біржі, оскільки API кожної з них має свої унікальні характеристики. У подальшому для розширення

функціональності програми доцільно виділити методи взаємодії з біржею у вигляді окремого інтерфейсу. Це дозволить легко інтегрувати підтримку інших бірж, підвищить стійкість системи у разі недоступності одного з API, а також сприятиме комбінуванню даних із різних джерел задля усунення прогалин у інформації, що може виникати через обмеження окремих платформ.

Для ефективної реалізації цього механізму у програмному засобі застосовується підхід, відомий як Dependency Injection (ін’єкція залежностей). Водночас для роботи з API конкретної біржі необхідно мати спеціалізований клас, орієнтований на цю біржу. На рисунку 3.5 наведено структурну UML-схему класів програмного засобу, де показано взаємозв’язки між ними.

Клас CryptoExchange відповідає за взаємодію з криптовалютною біржею. Він містить набір шаблонів HTTP-запитів до API, які формуються на основі даних, введених користувачем, і потім надсилаються на сервер біржі. Оскільки API не підтримує запит усіх даних одразу, процедура отримання інформації поділена на кілька підзапитів, реалізованих у вигляді окремих функцій. Отримані результати об’єднуються у клас ExchangeDataSnapshot.

ExchangeDataSnapshot — це глобальна одиниця, що представляє часовий ряд біржових даних. Він має унікальний ідентифікатор і часову позначку, а також містить список об’єктів типу ExchangeMarket.

ExchangeMarket акумулює комплексні дані біржі за визначений період, а саме: 1) список відкритих угод (екземпляри класу ActiveOrder); 2) валюту (екземпляр класу Currency); 3) історію завершених угод (екземпляри класу TradeHistoryOrder); 4) обсяг торгів, що зберігається у властивості Volume.

Клас A*ctiveOrder* відображає відкриту угоду, з такими характеристиками: ціна, кількість, тип угоди, загальна сума та базова валюта у вигляді об’єкта Currency.

Currency — клас, який інкапсулює назву та ідентифікатор криптовалюти, а також її тип: базова валюта (валюта, ціна якої котирується в одиницях іншої валюти) та котирувана валюта (валюта, в одиницях якої виражається ціна базової).

TradeHistoryOrder представляє закриту угоду і містить поля: ціна, кількість, тип угоди, загальна сума та час її завершення.

UserRequest описує запит користувача і включає: базову валюту (Currency), котирувану валюту (Currency), початок та кінець часової області (тип DateTime), а також булевий параметр, що визначає формат виводу прогнозу.

MathProcess — статичний клас, який містить методи для обробки даних математичними алгоритмами або передає їх на обробку нейронній мережі; усі методи повертають об’єкт класу Forecast.

Forecast зберігає результати прогнозування курсу або тренду у вигляді часового ряду.

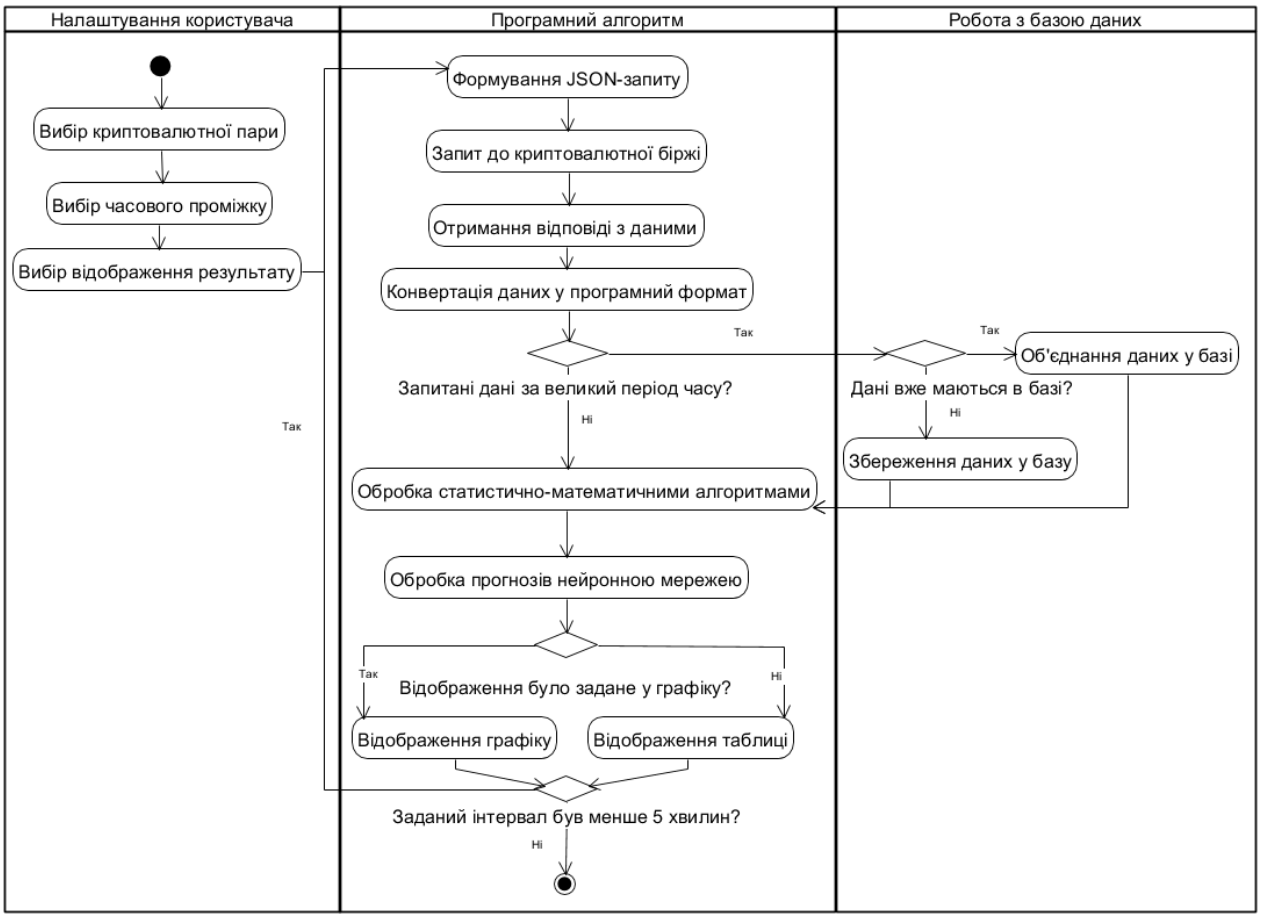


Рисунок 3.6. – Діаграма діяльності ПЗ

На діаграмі також присутні три типи перерахувань, які використовуються у сучасних об’єктно-орієнтованих мовах для групування числових значень під іменами, що відображають логіку їх застосування:

1. *CurrencyType* — визначає тип валюти: Quoted (котирувана) та Base (базова).

2. *OrderType* — визначає тип угоди: Ask (продаж) та Bid (покупка).

3. *Status* — визначає напрямок тренду у прогнозі: Unknown (початковий стан), Up (зростання), Down (спад) і Stay (без змін).

Після опису класів рекомендовано створити діаграму діяльності, що демонструє логіку робочого циклу програми. Вона наведена на рис. 3.6.

На діаграмі необхідно врахувати можливість організації циклічної роботи програми у випадку, якщо користувач замовив прогноз на період, коротший за 5 хвилин. В такій ситуації замість виводу статичного прогнозу система може періодично повторювати запити до API криптовалютної біржі, оновлюючи отримані дані та відображаючи актуальний прогноз у режимі реального часу.

3.4. Фізична модель програмного засобу

Для визначення фізичної структури програмного засобу прогнозування курсу криптовалют побудовано діаграми компонентів, які зображені на рис. 3.7.

Для визначення фізичної структури програмного засобу прогнозування курсу криптовалют було створено діаграми компонентів, представлені на рис. 3.7.

На рис. 3.7 показана структура файлів серверної частини програмного забезпечення, які забезпечують обробку даних і функціонування системи, як це було продемонстровано раніше. Зокрема, основні класи серверної частини включають:

Program.cs — клас, що виступає точкою входу для запуску програми;

Startup.cs — ключовий клас, який відповідає за початкову конфігурацію всіх необхідних сервісів, таких як MVC та інші;

HomeController.cs — клас, що забезпечує взаємодію з користувачем через головну сторінку додатку. Він надає користувачу: початковий перелік

прогнозів; форму для запиту прогнозу з фільтрами для вибору криптовалютної пари та часових меж; надсилає запит до модуля обробки та приймає результат для подальшого відображення користувачу;

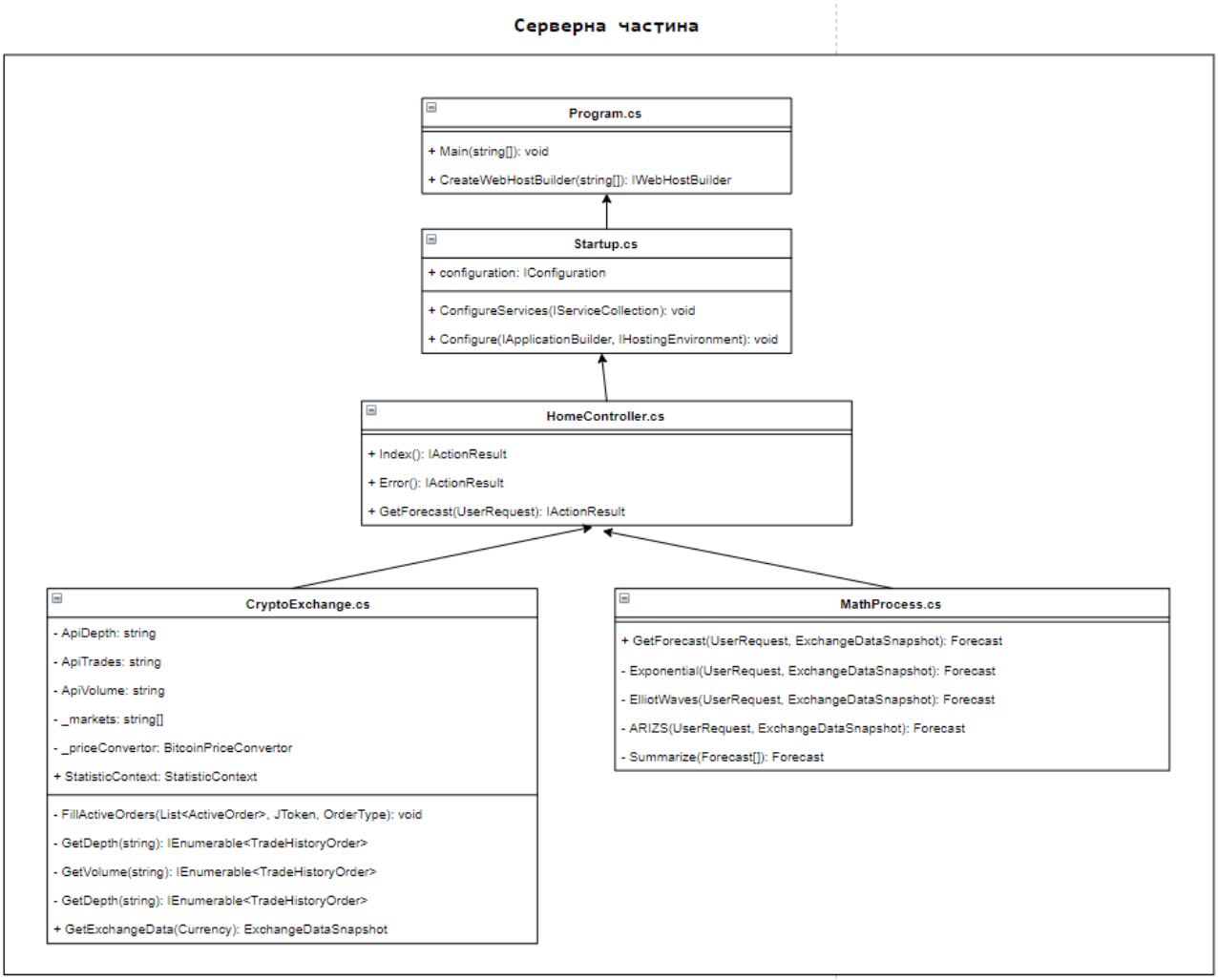


Рисунок 3.7. – Діаграма компонентів серверної частини

CryptoExchange.cs — клас, призначений для роботи з API криптовалютної біржі, детально описаний у логічній моделі та UML-діаграмі класів;

MathProcess.cs — статичний клас, що відповідає за математично-статистичну обробку даних і застосування методів штучного інтелекту для формування прогнозів.

Для роботи з клієнтом у проекті застосовуються файли розмітки:

файли з розширеннями .html або .cshtml — для розмітки веб-інтерфейсу із застосуванням HTML, JavaScript і C#;

файли .xaml — для розмітки десктопних додатків на платформі .NET.

У клієнтській частині передбачено такі файли:

Index.cshtml — головна і єдина сторінка додатку, оскільки застосунок реалізується як односторінковий (SPA), де ця сторінка завантажується цілком разом зі скриптами, а подальша взаємодія відбувається через асинхронне завантаження часткових представлень;

_MainMenu.cshtml — часткове представлення для меню, що інтегрується в Index.cshtml, через яке користувач задає параметри запиту прогнозу та форму його відображення;

_Layout.cshtml — загальний шаблон сторінки, який визначає розміщення елементів інтерфейсу;

_Table.cshtml — часткове представлення для відображення прогнозу у вигляді таблиці з датою, часом і курсом криптовалюти;

_Graph.cshtml — часткове представлення, яке містить скрипт _Graph.js для візуалізації графіку прогнозу;

_Graph.js — JavaScript-скрипт, що використовує відкриті бібліотеки для побудови графіків.

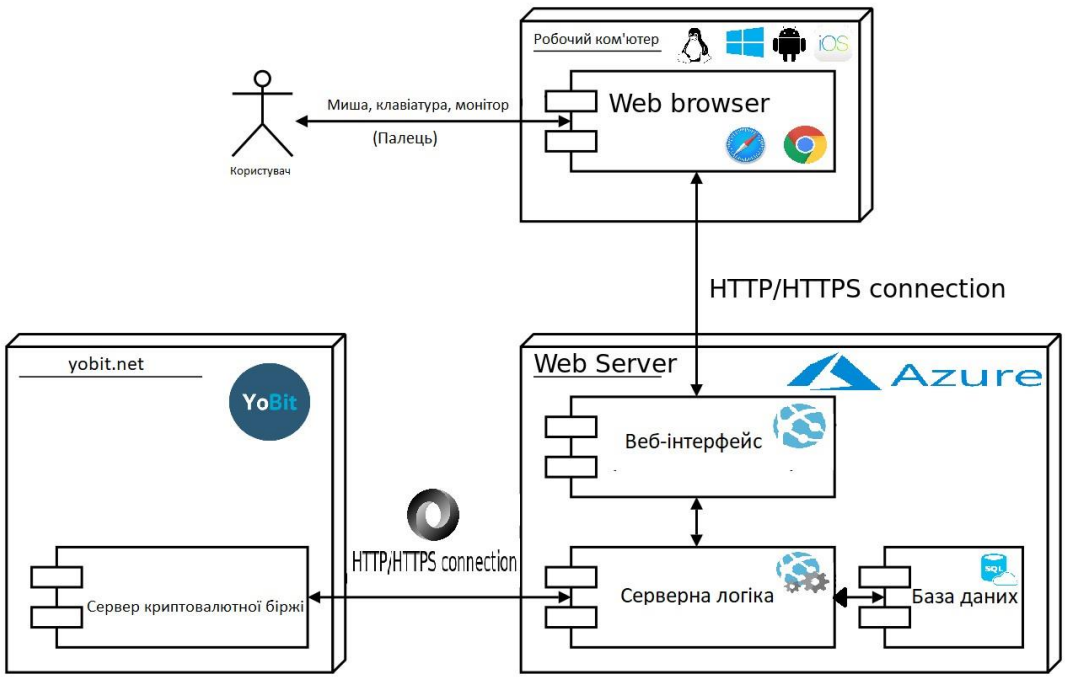


Рисунок 3.9. – Діаграма розгортки моніторингової системи

Програмний засіб передбачає одночасне використання кількома користувачами, кожен з яких працюватиме через власний інтерфейс. Для цього на рис. 3.9 наведена діаграма, що ілюструє архітектуру веб-додатку, де користувач взаємодіє із системою через браузер. Вибір веб-інтерфейсу для реалізації описано детально у наступному розділі.

Для джерела даних обрано криптовалютну біржу YoBit.Net, яка надає широкий спектр інформації через API у форматі JSON і користується значним рівнем довіри в інтернет-спільноті.

Діаграма охоплює такі ключові елементи: 1) користувацький комп’ютер із будь-якою операційною системою та веб-браузером, через який здійснюється взаємодія із сервером додатку; 2) веб-сервер, на якому розміщено веб-додаток, створений на базі ASP.NET MVC Core, а також база даних, розгорнута на платформі Azure SQL Cloud Database; 3) криптовалютна біржа, яку представляє компанія YoBit.Net [6].

РОЗДІЛ 4. РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАСОБУ

4.1. Програмна реалізація за допомогою фреймворку ASP.NET MVC

Найоптимальнішим підходом для створення програмних продуктів, які вимагають високої продуктивності на серверній стороні та зручності розробки веб-інтерфейсу з подальшим розгортанням у хмарі, є застосування платформи .NET. Вона використовує сучасну об’єктно-орієнтовану мову програмування C#, розроблену провідними фахівцями в IT-сфері та постійно вдосконалювану компанією Microsoft — лідером у галузі інформаційних технологій.

Платформа .NET оснащена менеджером пакетів NuGet, що значно спрощує підключення тисяч бібліотек для розробки, а також об’єднує всі етапи створення програмного забезпечення. Це стало можливим завдяки інтеграції з Microsoft Azure, яка забезпечує хостинг веб-додатків та надає хмарні бази даних SQL і NoSQL для повноцінного функціонування системи.

У сфері математичних обчислень традиційно широко використовується мова Python, яка має найбільший набір бібліотек, орієнтованих на наукові дослідження і обробку великих обсягів даних. Втім, сама мова має недоліки, що ускладнюють тривалу розробку. У цьому контексті корисним є Dynamic Language Runtime (DLR) платформи .NET, що дозволяє ефективно поєднувати строго типізовану мову C#, яка компілюється у байткод для виконання в середовищі Common Language Runtime (CLR), з динамічними мовами на кшталт Python.

Таким чином, можна використовувати потужні математичні бібліотеки Python без необхідності безпосередньої розробки на цій мові, що було б складним і неефективним.

Для розробки веб-інтерфейсу застосовується провідний фреймворк ASP.NET MVC Core, який є відкритим (Open Source) і підтримується спільнотою IT-розробників, що постійно вносять покращення, виправляють

помилки та оптимізують механізми. Фреймворк є кросплатформенним, однак це не має вирішального значення, оскільки хмарна платформа Azure надає послугу SaaS (Software as a Service), що дозволяє розгортати веб-додаток без необхідності налаштовувати операційну систему. ASP.NET MVC Core підтримує інтеграцію з сучасними веб-технологіями, такими як HTML5, CSS3, Angular, React.js, Vue.js, AJAX та іншими.

Щодо підвищення продуктивності, середовище розробки Visual Studio, зокрема її остання версія 2022 року, забезпечує можливість повного контролю всіх етапів створення додатка без необхідності переключатись між вікнами.

Реалізація програмного засобу виконана шляхом створення веб-додатку на базі платформи .NET у Visual Studio 2022 з використанням об'єктно-орієнтованої мови програмування C#. Додаток побудований на основі ASP.NET MVC, який передбачає наявність трьох основних компонентів: моделі (описані раніше класи у діаграмі класів), представлення (клієнтські файли розмітки HTML із вставками C#) та контролера (серверний контролер, який керує взаємодією моделей, обробкою даних та відображенням).

Першим кроком у розробці став клас YoBitCryptoExchange для роботи з API біржі, який формує запити на основі трьох URI, до яких додається криптовалютна пара:

1) ApiDepth (https://yobit.net/api/3/depth/) для отримання відкритих заявок (клас ActiveOrder);

2) ApiTrades (https://yobit.net/api/3/trades/) для отримання закритих угод за попередній період (клас TradeHistoryOrder);

3) ApiVolume (https://yobit.net/api/3/ticker/) для запиту загального обсягу торгів.

Нижче наведено приклад коду класу YoBitCryptoExchange, що реалізує доступ до API біржі:

```
using System;
using System.Collections.Generic; using System.Globalization;
using System.Linq; using System.Net;
using System.Threading.Tasks;
using CryptoExchangeService.Statistic.Helpers; using
CryptoExchangeService.Statistic.Models;
```

```
using CryptoExchangeService.Statistic.Models.Enums; using Newtonsoft.Json.Linq;
using static System.Console;
using static System.Globalization.DateTimeStyles; using static
CryptoExchangeService.Statistic.Models.Enums.CurrencyType;
using static CryptoExchangeService.Statistic.Models.Enums.OrderType;
namespace CryptoExchangeService.Statistic
{
public sealed class YoBitCryptoExchange : IDisposable
{
        private const string ApiDepth = "https://yobit.net/api/3/depth/";
        private const string ApiTrades = "https://yobit.net/api/3/trades/";
        private const string ApiVolume = "https://yobit.net/api/3/ticker/";
private readonly string[] _markets;
public readonly StatisticContext StatisticContext; private
BitcoinPriceConvertor _priceConvertor;
public YoBitCryptoExchange()
{
StatisticContext = new StatisticContext();
_markets = new[] { "btc", "eth", "doge",
"waves", "usd", "rur" };
}
private BitcoinPriceConvertor PriceConvertor =>
             _priceConvertor ?? (_priceConvertor = new
BitcoinPriceConvertor());
        public ExchangeDataSnapshot GetExchangeDataSnapshot(string currency)
{
var exchangeDataSnapshot = new ExchangeDataSnapshot
{
Time = DateTime.Now
};

Parallel.ForEach(_markets,
                                    market =>
                                        {
$"{currency}_{market}";
string pair =

ExchangeMarket
GetDepth(pair).ToList(), GetTrades(pair).ToList(),
GetVolume(pair)
var exchangeMarket = new
{
QuotedCurrencyName = market, ActiveOrders      =
TradeHistoryOrders = Volume   =
                                       };
lock (exchangeDataSnapshot)
                                       {
                                        if (exchangeMarket.ActiveOrders.Count>
0 ||
exchangeMarket.TradeHistoryOrders.Count > 0)
                                       {
exchangeDataSnapshot.ExchangeMarkets.Add(exchangeMarket);
                                       }
                                       }
                                      });
```

```
            foreach (ExchangeMarket exchangeMarket in
exchangeDataSnapshot.ExchangeMarkets)
{
exchangeMarket.QuotedCurrency = StatisticContext.Currencies.SingleOrDefault(c
=> c.Name == exchangeMarket.QuotedCurrencyName &&
c.Type == Quoted) ??
new Currency
{
exchangeMarket.QuotedCurrencyName,
     Name = Type = Quoted

 };
```

Продовження коду наведено в *додатку В*.

Оформлення веб-сторінок здійснюється за допомогою вбудованих стилів Bootstrap, які адаптивно налаштовують зовнішній вигляд простих елементів, таких як меню, кнопки, таблиці та випадаючі списки.

Відображення графіку прогнозованого курсу реалізовано за допомогою відкритої бібліотеки «Awesome Chart.js».

Кешовані біржові дані та результати прогнозів зберігаються у хмарній базі даних, яка працює паралельно з веб-додатком і підтримує автоматичне масштабування потужностей за потреби. Це дозволяє як базі даних, так і програмі ефективно розширюватися горизонтально, забезпечуючи одночасну роботу великої кількості користувачів. Єдине обмеження — це бюджет.

4.2. Тестування розробленого програмного засобу

Фрагмент тексту класу ExchangeMarket, який є основною одиницею сукупності даних отриманих з біржі:

```
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
namespace CryptoExchangeService.Statistic.Models
{
     public sealed class ExchangeMarket
     {
          public ExchangeMarket()
          {
               ActiveOrders = new List<ActiveOrder>(); TradeHistoryOrders = new
               List<TradeHistoryOrder>();
```

```
        }
        [Key]
        public int Id { get; set; }
        [Display(Name            = "Volume")]
        [Required(ErrorMessage = "{0} is required")] public
        double Volume { get; set; }
        [Display(Name = "Quoted Currency")]
        [Required(ErrorMessage = "{0} is required")] public
        Currency QuotedCurrency { get; set; }
        [Display(Name = "Active orders")]
        public List<ActiveOrder> ActiveOrders { get; set; }
set; }
 [Display(Name = "Trade history orders")]
public List<TradeHistoryOrder> TradeHistoryOrders { get;
        public int ExchangeDataSnapshotId { get; set; }
        public ExchangeDataSnapshot ExchangeDataSnapshot { get; set;
}
        [NotMapped]
        public string QuotedCurrencyName { get; set; }
    }
     }
```

Результати прогнозів для первинної обробки даних з біржі приведені на рис. 4.1.

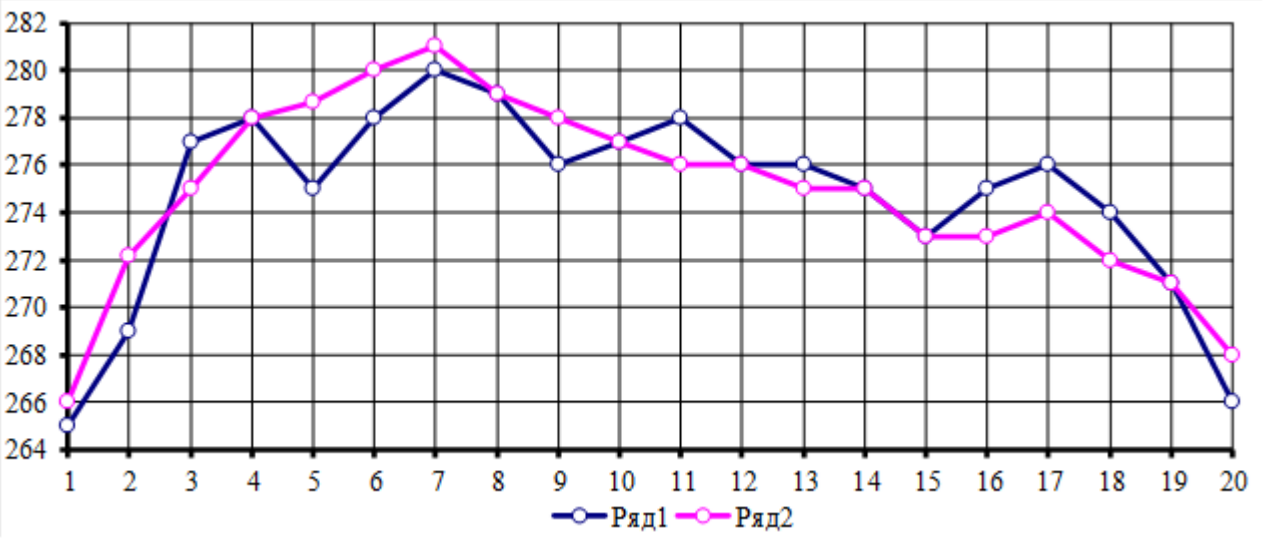
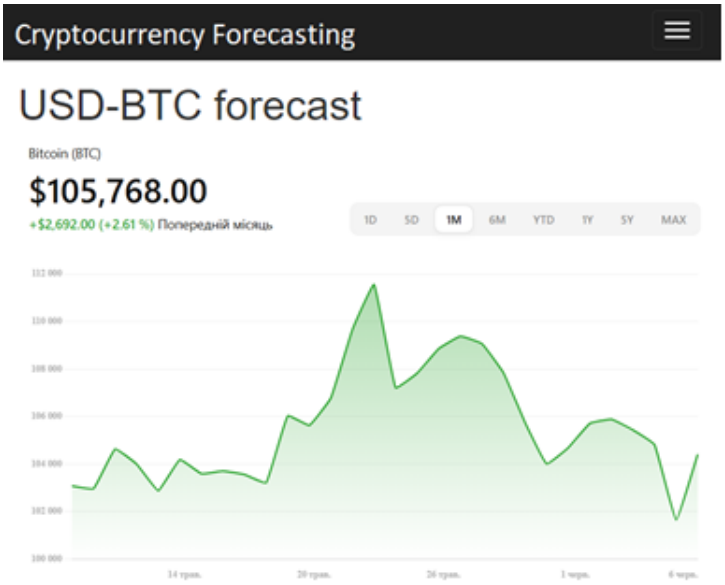


Рисунок 4.1. – Порівняння прогнозу з реальною кривою для первинної обробки даних

Після цього додаток перевірено в режимі роботи у реальному часі (рис. 4.2).



№	Date	Status
1	04.05.2025	Stay
2	05.05.2025	Stay
3	06.05.2025	Stay
4	07.05.2025	Stay
5	08.05.2025	Stay
6	09.05.2025	Stay
7	10.05.2025	Stay
8	11.05.2025	Stay
9	12.05.2025	Stay
10	13.05.2025	Stay
11	14.05.2025	Stay
12	15.05.2025	Stay
13	16.05.2025	Stay
14	17.05.2025	Stay
15	18.05.2025	Stay
16	19.05.2025	Down
17	20.05.2025	Stay
18	21.05.2025	Down
19	22.05.2025	Down
20	23.05.2025	Stay

Рисунок 4.2. – Скріншот відповіді моніторингової системи прогнозування курсу криптовалюти

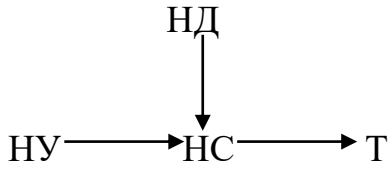
Отже, розроблено моніторингову систему у вигляді веб-додатку на базі ASP.NET MVC, реалізованого мовою C#. Додаток розгорнуто на хмарній платформі Microsoft Azure разом із супутньою базою даних. Для формування тестових і навчальних наборів даних та подальшого функціонування програми як джерело інформації обрана криптовалютна біржа YoBit.Net.

РОЗДІЛ 5. ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

5.1. Структурно-функціональний аналіз виробництва

Розробка та вживання ефективних заходів запобігання аварійним і травмонебезпечних ситуаціям можливі лише при завчасному виявлення тих небезпек, з яких починаються процеси їх формування. Оскільки небезпечні умови не завжди завчасно можна виявити, а для вивчення небезпечних дій іноді потрібно багато часу, щоб зібрати статичний матеріал, то і методи виявлення цих небезпек повинні бути відповідно диференційовані (табл. 5.1) [1].

Таблиця 5.1. Моделі формування та виникнення травмонебезпечних і аварійних ситуацій

Вид робіт, виробничий підрозділ, робоче місце	Виробнича небезпека			Можливі наслідки	Заходи запобігання небезпечним ситуаціям
	Небезпечна умова (НУ)	Небезпечна дія (НД)	Небезпечна ситуація (НС)		
Виконання робіт із електрообладнанням	Не вимкнено живлення. Відсутність заземлення.	Нехтування правилами ТБ	Ураження струмом	Травма (Т)	Проведення повторного інструктажу з ТБ. Розробка нових способів захисту. Встановлення заземлення.
					

Відповідно до аналізу небезпечних умов, які існують у виробничому процесі виокремлено такі наступні за характером дії на працівника їх групи [1]:

– характеризують стан або рівень небезпеки обладнання, які використовуються.

– сприяють виникненню технологічних помилок обслуговуючого персоналу впродовж виробничого процесу;

– створювати умови та можливість проникнення працівника в небезпечну зону;

– приводять до виникнення небезпечних дій (внаслідок низького рівня професійної підготовки працівників та організації навчання з охорони праці).

Моделі формування та виникнення травмонебезпечних і аварійних ситуацій в комп'ютерному кабінеті представлено у вигляді моделі формування та виникнення травмонебезпечних і аварійних ситуацій – табл. 5.1.

5.2. Розрахунок освітлення приміщення комп'ютерного кабінету

Освітленість виробничих приміщень може бути штучною і природною. Природне освітлення при правильному обладнанні найбільш сприятливе для людини. Основні вимоги для освітлення наступні [1]:

– освітлення повинне бути достатнім для швидкого і легкого розпізнання об'єктів роботи;

– освітлення повинно бути рівномірне без різких тіней;

– джерело світла не повинно осліплювати працівника;

– рівень освітленості не повинен обмежуватись часом.

Природне освітлення забезпечується обладнанням вікон (бокове освітлення) фонарів і світильних покрить приміщень (верхнє освітлення). Природне освітлення нормується коефіцієнтом природної освітленості. Коефіцієнт природної освітленості – це процентне відношення фактичної освітленості $Fв$ в будь-якій точці приміщення до освітленості $Fн$ розсіяної світлом небозводу точки, яка лежить на відкритій місцевості. Розрахунок природного освітлення через бокові вікна по нормам освітленості ведеться для

самої дальньої від вікон точки, тобто знаходять мінімальне значення стик коефіцієнта природної освітленості [1]:

$$e_{\min} = \frac{Fb}{F_н} \cdot 100. \qquad (5.1)$$

Значення коефіцієнта природної освітленості визначається не менше чим в п'яти точках. Значення коефіцієнта природної освітленості для сільськогосподарських виробничих приміщень в даному випадку ремонтній майстерні, беремо $e_{min} = 5\%$.

Розрахунок природного освітлення зводиться до визначення площі світлових променів.

Сумарну площу світлових променів $\sum F_o\,(м^2)$ по коефіцієнту природної освітленості для бокових променів визначаємо по формулі:

$$\sum F_o = \frac{F_н \cdot e_{\min} \cdot r_o \cdot K}{100 \cdot \tau \cdot Г_1}, \qquad (5.2)$$

де $F_н$ – площа підлоги, м^2; e_{min} – величина мінімального коефіцієнта природного освітленості; τ – загальний коефіцієнт світловикористання віконного отвору із врахуванням його забруднення, τ = 0,25; r_o – світлова характеристика вікна, r_o – 9,5; $Г_1$ – коефіцієнт, який враховує підвищення освітленості за рахунок світла, яке відбивається від стін і стелі, $Г_1$ = 1,2; K – коефіцієнт, який враховує затінення вікон сусідніми приміщеннями і загорожею, K = 1.

$$\sum F_o = \frac{36 \cdot 0,5 \cdot 9,5 \cdot 1}{100 \cdot 0,25 \cdot 1,2} = 5,7 м^2.$$

Кількість світлових променів визначимо:

$$N = \frac{\sum F_o}{F_o}, \qquad (5.3)$$

де F_o – площа вікна згідно стандарту, м^2.

$$N = \frac{5,7}{6} = 0,95.$$

Приймаємо кількість вікон – одне вікно.

При розрахунку природного освітлення найбільш поширеним і простим є метод світлового потоку. При цьому методі розраховуємо світловий потік $F_л$ (Лк), який повинна випромінювати кожна лампа (при заданій кількості ламп).

$$F_л = \frac{k \cdot S_п \cdot E}{n_л \cdot \eta \cdot r^2}, \qquad (5.4)$$

де k – коефіцієнт запасу, $k = 1,3$; $S_п$ – площа підлоги, м^2; $S_п = 36$ м^2. E – нормативна освітленість, Е = 300 Лк; $n_л$ – кількість встановлених ламп, $n_л$=6 од; η – коефіцієнт використання світлового потоку, $\eta = 0,25$; r – коефіцієнт нерівномірності освітленості, $r = 0,545$.

Коефіцієнт запасу (K) враховує можливість забруднення світильників пилом, що залежить від характеру виробництва.

Розрахунок штучного освітлення починаємо із визначення висоти розташування світильника і їх кількості. Висоту $h_н$ (м) розташування світильників над робочим місцем знаходимо за формулою:

$$h_н = H - \left(h_1 + h_2\right), \qquad (5.5)$$

де H – висота приміщення, м; h_1 – віддаль від підлоги до освітлювальної поверхні, м; h_2 – віддаль від стелі до світильника, м.

$$h_н = 4,5 - \left(2,2 + 1,5\right) = 0,8 \text{ м}.$$

При симетричному розміщенні світильників по вершинах квадратів їх кількість визначається за формулою:

$$n_c = \frac{S_п}{l^2}, \qquad (5.6)$$

де l – віддаль між світильниками, м.

Підставивши значення отримаємо:

$$n_c = \frac{36}{9} = 4 \text{ од}.$$

Тоді світловий потік буде становити

$$F_л = \frac{1,3 \cdot 36 \cdot 300}{4 \cdot 0,25 \cdot 0,545} = 2576,2 \text{ Лк}.$$

При світловому потоці 2576,2 Лк для заданої лампи вибираємо тип і потужність.

Вибираємо тип лампи – люмінесцентну, потужністю 40 Вт.

5.3. Безпека в надзвичайних ситуаціях

Забезпечення захисту населення і території у разі загрози і виникнення надзвичайних ситуацій є одним з най важливіших завдань держави.

Захист населення є системою загально державних заходів, які реалізуються центральними і місцевими органами виконавчої влади, виконавчими органами влад, органами управління з питань надзвичайних ситуацій та цивільного захисту населення, підпорядкованими їм системами, та підприємств, що забезпечують виконання організаційних, інженерно – технічних, санітарно – гігієнічних, проти епідемічних та інших заходів у сфері запобігання та ліквідації наслідків надзвичайних ситуацій.

Загрози життєво важливих інтересів громадян, держави, суспільства поділяють на зовнішні та внутрішні, виконують під час надзвичайних ситуацій техногенного та природного характеру та воєнних конфліктах.

Принципи захисту випливають з основних положень Женевської конвенції щодо захисту жертв війни та додаткових протоколів до неї, можливого характеру воєнних дій, реальних можливостей держави щодо створення матеріальної бази захисту. З метою захисту населення, зменшення втрат та шкоди економіці в разі виникнення надзвичайних ситуацій має право проводитись спеціальний комплекс заходів.

Оповіщення та інформування, яке досягається завчасним створенням і підтримкою в постійній готовності загально державної, територіальних та об’єктивних систем оповіщення населення.

ВИСНОВКИ ТА ПРОПОЗИЦІЇ

Станом на сьогодні існує понад 2500 криптовалют, з яких сім мають капіталізацію у мільярди доларів, а сотні інших оцінюються у сотні мільйонів. Популярність різних криптовалют зросла: за оцінками, у 2024 році було «активними» мільярди в ринковій капіталізації та мільйони криптогаманців. Інноваційність, значні перспективи та інтерес світової спільноти роблять криптовалюти важливим фінансовим активом.

Як існують валютні біржі для традиційних валют, так само для нових криптовалют були створені спеціалізовані криптобіржі. Криптовалюти, або віртуальні валюти, — це цифрові засоби обміну, які створюються і використовуються приватними особами чи групами. Більшість із них не контролюються урядами, тому розглядаються як альтернативні валюти — фінансові інструменти, що функціонують поза межами державної монетарної політики.

Найвідомішою та першою широко вживаною є Bitcoin. Bitcoin — криптовалюта, створена у 2008 році під псевдонімом «Сатоші Накамото» і введена в обіг у 2009 році. Це децентралізована електронна валюта без центрального банку чи керуючої структури. Передача Bitcoin здійснюється через однорангову мережу без посередників. Транзакції підтверджуються користувачами мережі — «нодами» — за допомогою криптографії та записуються в публічну базу даних — блокчейн.

Для виконання кваліфікаційної роботи встановлено зв’язки та набори даних між етапами прогнозування курсу криптовалют, побудовано концептуальну модель та її декомпозицію у нотаціях IDEF0, IDEF3 і DFD. Отримані моделі забезпечують чітке розуміння процесів і слугують базою для створення об’єктно-орієнтованої моделі з подальшою реалізацією програмного засобу.

Вибір програмних технологій обґрунтовано на основі умов, викладених у попередніх розділах, а також аналізу існуючих рішень. В результаті було

прийнято рішення реалізувати програмний засіб прогнозування криптовалют на платформі .NET. Розглянуто актуальні методи прогнозування курсів валют і криптовалют, що застосовують трейдери та спеціалізовані аналітичні програми, проведено детальний аналіз їх зручності, часових витрат, типів даних та ефективності прогнозування.

Було запропоновано об’єктно-орієнтовану модель програмного засобу прогнозування курсу криптовалют, що включає опис діаграми варіантів використання, а також логічної і фізичної моделей.

Проведено аналіз вимог до існуючих інструментів розробки програмного забезпечення для впровадження обраного методу. Обґрунтовано вибір технологічного стеку та детально описано процес створення програмного засобу за його допомогою. Розроблений засіб пройшов тестування на працездатність і оцінку ефективності.

Розроблено веб-застосунок на базі ASP.NET MVC з використанням мови C#. Додаток розгорнуто на хмарній платформі Microsoft Azure разом із супровідною базою даних. Запропоновано об’єктно-орієнтовану модель програмного засобу прогнозування криптовалют із описом діаграм варіантів використання, логічної та фізичної моделей.

Проаналізовано вимоги до сучасних інструментів розробки програмного забезпечення для реалізації обраного методу. Визначено технологічний стек та описано процес створення програмного засобу прогнозування криптовалют. Отримане програмне забезпечення пройшло перевірку на працездатність.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бен Мезрич. Біткоїнові мільярдери. Правдива історія про геніальність, зраду та реванш. Київ: КМ-Букс, 2023. 320 с.

2. Документація Binance API // Binance Docs. URL: https://binance-docs.github.io/apidocs/spot/en/#change-log (дата звернення: 05.05.2025)

3. Докуметнація git // Git-Scm. URL: https://git-scm.com/doc (дата звернення: 25.05.2025)

4. Долонов А. О. Криптовалюти: сутність, роль в економіці та особливості курсоутворення. Національний університет «Києво-Могилянська академія». 2022. 45 с. URL: http://surl.li/tugmz (дата звернення: 25.05.2025)

5. Жидецький В.Ц. Основи охорони праці / Жидецький В.Ц., Джигирей В.С., Мельников О.В. [5-е вид.]. Львів : Афіша, 2010. 350 с.

6. Когут Ю. Технології блокчейн та криптовалюта: ризики та кібербезпека. Київ: Сідкон, 2022. 316 с.

7. Криптовалютна біржа YoBit.Net. URL: https://yobit.net. (дата звернення: 05.05.2025)

8. Мова програмування C# . URL: https://uk.wikipedia.org/wiki/C_Sharp. (дата звернення: 05.05.2025)

9. Основи технічного аналізу. URL: https://pidruchniki.com/72559/finansi/osnovi_tehnichnogo_analizu (дата звернення: 06.06.2025)

10. Сайт статистики мережі Bitcoin. URL: https://blockchain.info. (дата звернення: 06.06.2025)

11. Стандарти ECMA C# та CLI. URL: http://www.webcitation.org/6HZF1RbUz. (дата звернення: 05.05.2025)

12. Стандарт ECMA-404 JSON . URL: http://www.json.org/. (дата звернення: 05.05.2025)

13. Тепскотт Д., Тепскотт А. Блокчейн-революція. Львів: Літопис, 2019. 492 с.

14. Термінологічний словник з питань запобігання та протидії легалізації (відмиванню) доходів, одержаних злочинним шляхом, фінансуванню тероризму, фінансуванню розповсюдження зброї масової знищення та корупції / Чубенко А. Г., Лошицький М. В., Павлов Д. М., Бичкова С. С., Юнін О. С. Київ : Ваіте, 2018. 826 с. URL: https://fiu.gov.ua/assets/userfiles/books/3_slovnyk.pdf (дата звернення: 06.06.2025)

15. Технічний аналіз фондового ринку. URL: https://stud.com.ua/50044/finansi/tehnichniy_analiz_fondovogo_rinku (дата звернення: 06.06.2025)

16. Хейфедін Аммус. Біткоїновий стандарт. Альтернатива центральним банкам / The Bitcoin Standard. Наш Формат. 2025. 352 с.

17. Adem Efe Gencer. Decentralization in Bitcoin and Ethereum Networks / Adem Efe Gencer, Soumya Basu, Ittay Eyal, Robbert van Renesse, Emin Gün Sirer // Financial Cryptography and Data Security (FC). 2018. 18 с.

18. Arvind Narayanan. Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction / Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller & Steven Goldfeder // Princeton University Press. 2016. 304 с.

19. Bitcoin days destroyed. URL: https://blockchain.info/charts/bitcoin-days-destroyed. (дата звернення: 05.05.2025)

20. Dr Garrick Hileman, Michel Rauch. Global cryptocurrency benchmarking study / Cambridge Centre for Alternative Finance, 2017. 114 с.

21. Fergal Reid and Martin Harrigan. An Analysis of Anonymity in the Bitcoin System. arXiv, 2011. 168.

22. Jae Kwon. Tendermint: consensus without mining. URL: http://tendermint.com/docs/tendermint.pdf. (дата звернення: 05.05.2025)

23. Ralph C. Merkle. A Digital Signature Based on a Conventional Encryption Function. In Advances in Cryptology – CRYPTO '87, Lecture Notes in Computer Science. Vol. 293. https://people.eecs.berkeley.edu/~raluca/cs261-f15/readings/ merkle.pdf. (дата звернення: 07.06.2025)

Додатки

Додаток А.
Функціональні вимоги

Таблиця А.1 Функціональні вимоги серверної частини, що ставляться до системи

№ п/п	Вимога
1	Можливість отримання форми з парою криптовалют та часовим проміжком
2	Можливість створення запиту у форматі HTTP для API принаймні однієї криптовалютної біржі
3	Можливість прийому відповіді на запит у форматі JSON
4	Можливість парсингу даних з JSON формату в об’єкти
5	Можливість запису даних у вигляді об’єктів до бази даних у разі великої кількості запитів до конкретної криптовалюти
6	Можливість перенаправлення запиту до бази даних, замість криптовалютної біржі, у разі наявності в базі потрібних даних
7	Можливість передачі даних у вигляді часового ряду об’єктів до класів з методами обробки даних у прогнози
8	Можливість видачі класами-обробниками вихідних даних прогнозів у єдиному вигляді, по екземпляру з кожного обробника
9	Можливість обробки екзмеплярів прогнозів ШНМ для генерації єдиного прогнозу
10	Можливість запису прогнозу у базу даних, у разі його довгостроковості
11	Можливість перенаправлення запиту прогнозу до бази даних, для його безпосередньої видачі, у разі збігу параметрів криптовалютної пари та часового діапазону з наявним в базі прогнозом

Таблиця А.2 Функціональні вимоги клієнтської частини, що ставляться до системи

№ п/п	Вимога
1	Можливість перегляду списком готових прогнозів найпопулярніших криптовалют у часових проміжках: 5хв, 10 хв, 30 хв, 1 год, 1 день, 1 тиждень, 1 місяць, 3 місяці, 6 місяців.
2	Можливість задання власної пари криптовалют та часового проміжку для отримання персонального прогнозу
3	Можливість задання бажаного вигляду прогнозу: таблицею чи графіком
4	Можливість вивантаження таблиці прогнозу у Excel

Додаток Б.
Таблиці варіантів використання

Таблиця Б.1 – Варіант використання «Перегляд списку прогнозів найпопулярніших криптовалют»

Назва	**Перегляд списку прогнозів найпопулярніших криптовалют**
Первинний актор	Користувач
Інші актори	Немає
Опис	Можливість одразу побачити прогнози на стандартні часові проміжки на головній сторінці додатку без запитів
Попередні умови	Відкрита головна сторінка додатку
Вихідні умови	Відкрита головна сторінка додатку

Таблиця Б.2 – Варіант використання «Задання стандартного представлення для списку прогнозів найпопулярніших криптовалют»

Найменування	**Задання стандартного представлення для списку прогнозів найпопулярніших криптовалют**
Первинний актор	Користувач
Інші актори	Немає
Опис	При обранні криптовалюти зі списку замість стандартного представлення графіком можна обрати табличний вигляд
Попередні умови	Відкрита головна сторінка додатку
Вихідні умови	Відкрита головна сторінка додатку зі зміненим представленням прогнозів

Таблиця Б.3 – Варіант використання «Завантаження прогнозу зі списку у вигляді Excel документу»

Найменування	**Завантаження прогнозу зі списку у вигляді Excel документу**
Первинний актор	Користувач
Інші актори	Немає
Опис	При обранні прогнозу з представленого списку можна натиснути «Завантажити в Excel» для завантаження прогнозу у вигляді таблиці в документі Excel
Попередні умови	Відкрита головна сторінка додатку
Вихідні умови	Завантажено Excel документ з прогнозом

Таблиця Б.4 – Варіант використання «Запит прогнозу конкретної криптовалюти»

Найменування	**Запит прогнозу конкретної криптовалюти**
Первинний актор	Користувач
Інші актори	Немає
Опис	Зверху на головній сторінці обираємо у випадаючих списках пару криптовалют та натискаємо на кнопку «Прогноз» для отримання прогнозу
Попередні умови	Відкрита головна сторінка додатку
Вихідні умови	Замість списку валют відображений графічний прогноз обраних криптовалют з кнопками перемикачами між стандартними інтервалами

Таблиця Б.5. – Варіант використання «Запит прогнозу конкретної криптовалюти на встановлений часовий проміжок»

Найменування	**Запит прогнозу конкретної криптовалюти на встановлений часовий проміжок**
Первинний актор	Користувач
Інші актори	Немає
Опис	Зверху на головній сторінці обираємо у випадаючих списках пару криптовалют, після чого обираємо діапазон прогнозу зі списку діапазонів та натискаємо на кнопку «Прогноз» для отримання прогнозу
Попередні умови	Відкрита головна сторінка додатку
Вихідні умови	Замість списку валют відображений графічний прогноз обраних криптовалют на заданий часовий проміжок

Додаток В.

Фрагмент тексту класу *YoBitCryptoExchange*, що реалізує доступ до API біржі (*продовження*)

```
exchangeMarket.ActiveOrders.ForEach(ao => ao.Price = PriceConvertor.ConvertToBTC(ao.Price,
exchangeMarket.QuotedCurrencyName));
}
return exchangeDataSnapshot;
}
           private static IEnumerable<TradeHistoryOrder> GetTrades(string pair)
{
const string timestamp = "timestamp"; const string price= "price"; const string amount= "amount";
const string type= "type"; const string ask= "ask";
using (var webClient = new WebClient())
{
pair);
string data = webClient.DownloadString(ApiTrades +
try
{
JObject jData = JObject.Parse(data); return jData[pair]
.Select(part => new TradeHistoryOrder
{
part[price].Value<double>(),
Price=
Amount=
part[amount].Value<double>(),
Type=
part[type].ToString() == ask ? Ask : Bid,
OrderConfirmationTime =
part[timestamp].UnixTimeStampToDateTime(),
ExchangeOrderId=
part[timestamp].Value<long>()
                                                 });
}
catch (Exception ex)
{
WriteLine($"{ApiDepth}{pair} {ex.Message}");

return Enumerable.Empty<TradeHistoryOrder>();
}
}
}
pair)
private static IEnumerable<ActiveOrder> GetDepth(string
{
const string asks  = "asks"; const string bids  = "bids"; const string error = "error";
using (var webClient = new WebClient())
{
pair);
string data = webClient.DownloadString(ApiDepth +try
{
JObject jObject = JObject.Parse(data);
var activeOrders = new List<ActiveOrder>();
                            if (jObject.TryGetValue(error,
StringComparison.InvariantCultureIgnoreCase, out JToken jToken))
{
WriteLine($"{ApiDepth}{pair} {jToken}");

return Enumerable.Empty<ActiveOrder>();
}
                            if (!jObject.TryGetValue(pair,
StringComparison.InvariantCultureIgnoreCase, out jToken))
{
correct answer");
WriteLine($"{ApiDepth}{pair} doesn't contain
return Enumerable.Empty<ActiveOrder>();
```

```
}
                            FillActiveOrders(activeOrders, jToken.Value<JToken>(asks), Ask);
                            FillActiveOrders(activeOrders, jToken.Value<JToken>(bids), Bid);
return activeOrders;
}
catch (Exception ex)
{
WriteLine($"{ApiDepth}{pair}{ex.Message}");
return Enumerable.Empty<ActiveOrder>();
}
}
}
private static double GetVolume(string pair)
{
const string volume = "vol";
try
{
using (var wc = new WebClient())
{
                            JObject jObject = JObject.Parse(wc.DownloadString(ApiVolume + pair));
                            return jObject.Value<JToken>(pair).Value<double>(volume);
}
}
catch (Exception ex)
{
                       WriteLine($"{DateTime.Now} {ex.Message} Thrown from GetVolume method");
return default(double);
}
}
           private static void FillActiveOrders(List<ActiveOrder> activeOrders, JToken
orderTokens, OrderType orderType)
{
if (orderTokens != null)
{
ActiveOrder
activeOrders.AddRange(orderTokens.Select(part => new

{
Price=
                                      part[0].Value<double>(),
                                      part[1].Value<double>(),
                                                  }
}
~YoBitCryptoExchange()
{
}));
Amount =
Type     = orderType
Dispose(false);
}
#region IDisposable
private void Dispose(bool disposing)
{
if (disposing)
{
StatisticContext?.Dispose();
}
}
public void Dispose()
{
Dispose(true); GC.SuppressFinalize(this);
}
#endregion
private sealed class BitcoinPriceConvertor
{
private const string ApiPrice =
"https://yobit.net/api/3/ticker/eth_btc-doge_btc-waves_btc-btc_usd- btc_rur";
private readonly Dictionary<string, double>
_currenciesCoefficients;
```

```
internal BitcoinPriceConvertor()
{
double>();
                                                                                                    }
_currenciesCoefficients = new Dictionary<string,
GetBtcPrice();
private double getAveragePrice(JToken price)
{
const double averageCoef = 2.0;
                    return (price["low"].Value<double>() + price["high"].Value<double>()) /
averageCoef;
}
private static double GetRevertPrice(double price)
{
return 1 / price;
}
private void GetBtcPrice()
{
try
{
using (var wc = new WebClient())
{
string data = wc.DownloadString(ApiPrice);
JObject jdata = JObject.Parse(data);
                                _currenciesCoefficients.Add("eth",
getAveragePrice(jdata["eth_btc"]));
                                _currenciesCoefficients.Add("doge",
getAveragePrice(jdata["doge_btc"]));
                                _currenciesCoefficients.Add("waves",
getAveragePrice(jdata["waves_btc"]));
                                _currenciesCoefficients.Add("rur",
GetRevertPrice(getAveragePrice(jdata["btc_rur"])));
_currenciesCoefficients.Add("usd",
GetRevertPrice(getAveragePrice(jdata["btc_usd"])));
}
}
catch (Exception ex)
{
                          WriteLine($"{DateTime.Now} {ex.Message} Thrown from
{nameof(PriceConvertor)}");
}
}
                public double ConvertToBTC(double price, string quotedCurrencyName)
{
                     if (_currenciesCoefficients.TryGetValue(quotedCurrencyName, out double
coef))
{
return price * coef;
}

return price;
}
}
}
}
```