ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ БІЗНЕС-КОЛЕДЖ

Кафедра комп'ютерної інженерії та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА

на тему

СИСТЕМА АНАЛІЗУ КРИПТОВАЛЮТ З АГРЕГАЦІЄЮ BLOCKCHAIN МЕТРИК

Студента Групи КІ-22

_____Деркача Т. С._____

(прізвище та ініціали)

Керівник роботи:

_____Розломій І. О.

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Кількість балів: _____

Оцінка:ECTS _____

Члени комісії:

_доцент КІІТ, к.т.н.__

_____Бурмістров С. В.,

_доцент КІІТ, доцент, к.т.н.___

_____Захарова М. В.

_доцент КІІТ, доцент, к.т.н.___

_____Михайлюта С. Л.

Черкаси, 2024 рік

ЗМІСТ

ВСТУП

Криптовалюти (також їх називають монетами або токенами) це різновид цифрової або віртуальної валюти, яка захищена криптографічним шифруванням, що унеможливлює її підроблення або подвійне витрачання [2]. Вони вважаються децентралізованою альтернативою фіатних грошей. Однак, слід зауважити, що номінальна вартість криптовалют не гарантується жодною державою [3]. Натомість їх вартість часто забезпечується приватними особами або компаніями, що підвищує ризик втрати їх цінності [4].

Отже, виникає потреба у програмному аналізі криптоактивів з агрегуванням та обчисленням середніх значень ряду показників, таких як ціна в доларах, розподілення токенів між унікальними адресами, транзакційна активність в blockchain мережі, джерела забезпечення вартості монети. й розраховувати ризики продажу або купівлі активу в фінансовий портфель [5].

Інформація про транзакції є загальнодоступною для всіх і зберігається в мережах blockchain. Щоб переглянути останні дії в мережі або активність конкретного гаманця, можна скористатись дослідниками мереж (blockchain explorers). Blockchain можна описати як нерозривний впорядкований ланцюг записів (блоків), які зберігаються в розподілену базу даних (БД). При появі нових транзакцій в мережі, кількість блоків збільшується. Кожен запис пов’язаний з попереднім за допомогою криптографії та містить інформацію, яка допомагає відстежувати активність в мережі та об'єми, що в ній циркулюють [6].

Однією з проблем аналізу криптовалютного ринку є волатильність, тобто значні коливання ціни за короткий період. Інвестори з великими об'ємами активів роблять ставки на зростання чи спадання ціни, щоб спекулятивно отримати прибуток. Ці дії викликають раптовий приплив фінансових об’ємів або їх раптовий відтік, що і призводить до високої волатильності [7]. Навіть такі дорогі криптовалюти, як Bitcoin і Ethereum, схильні до раптових коливань ціни. У зв'язку з цим виникає проблема

знаходження та відстежування тенденцій руху ціни або потенційно маніпуляційних дій для захисту власних збережень [5].

Темою кваліфікаційної роботи є розробка аналізатора криптовалют з агрегацією blockchain метрик.

Поширеною практикою при аналізі токенів є використання методів технічного та фундаментального аналізу, які вже понад 50 років на практиці застосовуються аналітиками фінансових ринків.

Технічний аналіз (ТА) являє собою сукупність інструментів прогнозування ймовірної зміни цін на основі закономірностей змін цін в минулому в аналогічних обставинах. За основу в технічному аналізі розглядають зміни ціни на графіках та таблиці лімітних замовлень біржі на придбання або продаж певних активів [8].

Фундаментальний аналіз (ФА) – це підхід, який використовуються аналітиками для встановлення справжньої вартості активу або бізнесу. Розглядаючи низку внутрішніх і зовнішніх факторів, головною метою аналітика є визначення того, чи є ринкова вартість активу завищеною, справедливою або недооціненою [9].

Актуальність теми. Поступова адаптація віртуальних валют в сучасні фінансові системи призводить до популяризації цього виду активів серед населення. Аналізатори криптовалют є корисним інструментом для користувачів, які використовують фінансові рішення на основі технології blockchain. Їх користь полягає у швидкій агрегації інформації з багатьох джерел у вигляді зручного інтерфейсу з важливими для вивчення показниками, такими як історія ціни, капіталізація і т.д. [5].

При наявності великої кількості монет виникає проблема з відстежуванням вартості кожної з них. Отримуючи інформацію про попит та добові об’єми торгів монетами з декількох незалежних бірж, аналізатор може показувати середні значення максимально наближені по точності. Ринки криптовалют торгуються цілодобово, а це означає, що вартість монет може змінюватися у будь-який проміжок часу.

Агрегація інформації мінімізує часові витрати на ручний пошук, перевірку, порівняння та обчислення усереднених показників активу, що займає значну кількість часу і може зіграти ключову роль при прийнятті фінансових рішень. Часто трапляються випадки запізнілого аналізу ситуації щодо сумнівного активу, коли кожна секунда призводить до безповоротного зменшення його вартості.

Отже, якісним можна вважати аналізатор, який швидко та систематично надає достовірні показники, що в свою чергу допомагає користувачу зменшити кількість прийнятих збиткових фінансових рішень. Наявні програмні продукти першочергово застосовують методи ТА і ФА або нейронні мережі. Використовуючи комбінацію кількісних та якісних показників, можливо розробити програмний продукт, який буде допомагати користувачам без досвіду взаємодії з криптовалютою, швидко отримати уявлення про актив у зрозумілій формі чисел або повідомлень природною мовою.

Мета і задачі розробки. Метою даної роботи є проєктування та розробка додатка для аналізу криптовалют на основі даних агрегованих з декількох відкритих джерел, мереж blockchain та інтегрованих сервісів.

Виходячи з мети роботи можна сформувати основні завдання на розробку, які включають:

1. Аналіз предметної області blockchain мереж та криптовалют.
2. Пошук та дослідження наявних рішень та технологій доцільних для використання при розробці аналізатора.
3. Вибір оптимальних технічних складових та алгоритмів для аналізу blockchain мереж та криптовалют.
4. Визначення функціональних вимог до аналізатора криптовалют.
5. Проєктування майбутньої системи аналізатора криптовалют.
6. Реалізація системи з використанням обраних технологій.
7. Тестування програмного продукту.

Об’єкт і предмет розробки. Об’єктом розробки є технічний та фундаментальний методи аналізу криптовалют на основі даних отриманих з blockchain мереж та інших сторонніх джерел.

Предметом розробки є дослідження відкритих показників отриманих з декількох джерел, blockchain мереж для аналізу цих метрик за допомогою інструментів технічного та фундаментального аналізу.

Практичне значення отриманих результатів. Практична значущість роботи полягає у агрегації аналітичної інформації про понад 13 тисяч унікальних криптовалют зі сторонніх джерел. Додаток надає зручний та інтуїтивно зрозумілий користувацький інтерфейс з підтримкою пристроїв різних розмірів, зміною мови та кольору теми. Користувач подібний до аналогів інтерфейс та отримує різноманітні необхідні для технічного та фундаментального аналізу показники та інструменти їх дослідження.

Апробація результатів: На основі створеного аналізатора криптовалют було опубліковано тези на XIII-ій міжнародній науково-практичній онлайн конференції «FREE AND OPEN SOURCE SOFTWARE» [5] із виступом 18 листопада 2021 р. на онлайн-конференції (Харків, 2021).

Структура та обсяг роботи. Кваліфікаційна робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел (31 найменування). Загальний обсяг роботи становить 64 сторінки основного тексту, 24 рисунки та 1 таблиця.

РОЗДІЛ 1
ОГЛЯД ТА ПОРІВНЯННЯ КРИПТОАНАЛІЗАТОРІВ

1.1 Аналіз предметної області

Криптовалюти не контролюються жодним центральним органом влади, що теоретично робить їх не сприйнятливими до маніпуляцій або прямого втручання, що неодноразово спростовувалося зафіксованими випадками фінансових маніпуляцій. Не зважаючи на концепцію децентралізації, яка лежить в основі blockchain мереж, майже всі торгові біржі є централізованими та можуть блокувати відкриті у них рахунки. Крім того, вони піддаються регуляції центральних органів влади [10].

Правовий статус криптовалют істотно відрізняється від однієї юрисдикції до іншої, і досі не визначений або змінюється в багатьох з них. У більшості країн їх використання чітко не регулюються та не прописано в законодавстві. Тобто можливість використання криптоактивів, як легального платіжного засобу варіюється за нормативними наслідками [9].

Як було зазначено вище, найважливішим фінансовим показником активу, який першочергово потрібно аналізувати є волатильність. Цей показник характеризує мінливу зміну ціни за період та застосовується в управлінні фінансовими ризиками для визначення міри ризику використання фінансового інструменту за заданий промiжок часу. Для розрахунку волатильності застосовується статистичний показник вибіркового стандартного відхилення, що дозволяє інвесторам визначити ризик придбання фінансового інструменту [7].

Не менш значною проблемою є велика кількості шахрайських проєктів. Через популяризацію віртуальних активів серед населення, їх створення стало доступне для будь-якого просунутого користувача інтернету. Тому інвестування в нові токени завжди пов'язане зі значним ризиком.

Нині кількість сумнівних проєктів з власними токенами постійно збільшується і значна кількість користувачів не розуміє, у що вкладає свої гроші. Тому виникає необхідність в інструменті для виявлення небезпечних проєктів, які можуть бути залучені у проведення фінансових злочинів [5].

Потенційно шахрайські проєкти можуть за короткий проміжок часу різко змінювати ціну власних токенів, щоб згодом раптово повністю продавати. Крім того, навіть перевірений blockchain протокол може бути зламаний через вразливості в програмному коді. Такі випадки як правило призводять до знецінення криптовалюти, втрати довіри до проєкту та всіх залучених до його розвитку осіб. Розглянемо детальніше інструменти й практики з технічного та фундаментального аналізу фінансових ринків, які можна застосувати при вирішенні вище вказаних проблем.

1.2 Технічний аналіз криптовалют

При ТА ринку головна мета – це визначити настрої ринку та спробувати передбачити тенденцію, тобто напрям руху цін. У його основі лежить аналіз чартів, тобто часових рядів ціни. Окрім цінових рядів, в ТА використовується інформація про об'єми торгів та інші статистичні дані. Найчастіше методи ТА використовуються для аналізу цін, що змінюються вільно, це як правило відбувається на біржах [8].

Технічний індикатор (індикатор технічного аналізу) – функція, побудована на значеннях статистичних показників торгів (ціни, обсяг торгів тощо). Аналізом поведінки індикаторів займаються професійні трейдери (учасники біржової торгівлі з фінансовою освітою). На основі аналізу технічних індикаторів, приймають рішення про відкриття (розширення) або закриття (скорочення) позицій. Технічні індикатори зазвичай застосовуються у вигляді графіків, накладених або суміщених з графіками ціни чи об'ємів активів, що торгуються. Крім того, технічні індикатори тією чи іншою мірою використовуються механічними торговими системами при алгоритмічній торгівлі [11].

Аналіз технічних індикаторів дозволяє знаходити поточну тенденцію руху ціни певного активу або фінансового ринку в цілому, та прогнозувати коли тренд може змінитись. За критерієм прогнозної значущості сигналів технічні індикатори поділяються на індикатори тренду, осцилятори та допоміжні індикатори [11].

Індикатори тренду (trend-following indicators) дозволяють виділити поточний ціновий тренд, який є основним об'єктом вивчення у ТА. Більшість з індикаторів тренду засновані на усередненні та згладжуванні цінового ряду за допомогою рухомих середніх. Індикатори цієї групи подають сигнали про розворот тренду, а також підтверджують або спростовують стійкість поточного тренду, проте не можуть використовуватися для прогнозування коливань цін у найближчому майбутньому.

Осцилятори – це найпоширеніший вид випереджальних індикаторів, які дозволяють прогнозувати поведінку цін у найближчій перспективі, що відрізняє їх від індикаторів тренду, які подають лише сигнали з запізненням.

Історично фінансові ринки перебувають у стані тренду (різке зростання або спадання) приблизно третину часу або менше, а в стані бокового руху (значення ціни обмежене і не виходить за межі діапазоном) понад половину часу всього існування. Індикатори тренду можуть бути дуже шкідливими при відсутності тренду на ринку, осцилятори ж навпаки придатні для бокового ринку й безкорисні при трендах. ТА умовно можна поділити на два напрями, в залежності від методів, що застосовуються. Перший напрям – це класичний ТА, заснований на вивченні та аналізі саме цінових графіків. Більш сучасним напрямком є комп'ютерний технічний аналіз, заснований на використанні методів математичної статистики (математичний аналіз) і спеціальних алгоритмів обробки даних (значень ціни).

Існує велика кількість методів та побудованих на їх принципах індикаторів. До найвідоміших показників відносяться [11]:

1. Первинна інформація:
   - ціна, по якій здійснюються операції;

- об’єм торгів і ліквідність цінних паперів або валюти;
- попит і пропозиція на ринку – ці значення дає біржовий стакан.

2. Інформація, що запізнюється (або інерційна):
   - «Японські свічки (Candles)», показують волатильність цін»;
   - ковзані середні (Moving Average, MA) – лінія, що запізнюється, показує напрямок довгострокового тренду;
   - метод сходження і розбіжності – пошук точок перетину «швидкою» і «повільною» середніх ліній;
   - алігатор (фондовий ринок) – пошук точок розбіжності на основі трьох середніх, що запізнюються.
3. Швидкість і прискорення ринку (перша і друга похідні від ціни):
   - «Моментум» (Momentum) – аналіз швидкості й напрямку зміни ціни (перша похідна, «зміна в часі»);
   - стохастичний індикатор (Stochastic Oscillator) – аналіз прискорення зміни ціни (друга похідна або «швидкість росту швидкості»);
   - індекс відносної сили RSI – порівняння швидкості росту та падіння ціни за обраний проміжку часу.
4. Інші індикатори:
   - лінія тренду;
   - лінії (смуги) Боліндже́ра (Bollinger Bands, BB);
   - лінії та рівні підтримки (support) і опору (resistance);
   - відкрита цікавість;
   - зважений об'єм;
   - спрямованість ціни й об'єму;
   - індекс накопичення/розподілу;
   - сходження/розбіжність ковзних середніх (Moving Average Convergence/Divergence, MACD);
   - «Хмара Ішимоку»;
   - метод хвильової симетрії;
   - індикатори настрою ринку.

Недоліком технічних індикаторів є те, що завжди існує можливість побачити хибний напрямок руху вартості. Тому вони не є надійним джерелом при аналізі. Помилки можуть виникати через різні складові, починаючи від алгоритмів та формул розрахунку показників, закінчуючи провалами в часі й неправдивими джерелами інформації.

1.3 Фундаментальний аналіз криптовалют

Основна мета ФА криптовалют полягає у зменшенні ризиків втрати фінансів та оцінці потенційних прибутків. ФА базується на трьох видах метрик отриманих з мережі blockchain, фінансових та проєктних [9].

Blockchain (або on-chain) метрики – це показники, які отримують безпосередньо з blockchain мережі. До таких метрик відносять число та суму всіх транзакцій в мережі, інформацію про активні адреси, комісії в мережі, обчислювальні потужності та суму стейкінгу.

Число транзакцій – хороший показник активності у мережі. Зобразивши на графіку число транзакцій за певні періоди (або використовуючи індикатор MACD і т.п.), можна побачити, як змінювалася активність у різний час. Проте, з цим показником слід бути обережним, тому що при переказі коштів між різними гаманцями, у них може бути один і той самий господар. Такі маніпуляції можуть виконуватись для штучного підняття активності мережі.

Сума транзакцій складається зі всіх транзакцій проведених в мережі за певний період. Більшість аналізаторів вимірює суму транзакцій одночасно у валюті протоколу мережі й фіатній валюті, такій як долар.

Під терміном активна адреса в мережі blockchain розуміють ту, яка використовувалася в певний період. Існують різні підходи до їх підрахунку. Поширений метод полягає в тому, щоб вважати адреси відправників та одержувачів транзакцій активними за певний період. Деякі аналітичні сервіси також відстежують загальну кількість унікальних адрес за весь час.

Аналіз комісій може розповісти про попит на кожне місце в новому блоці мережі. Їх можна розглядати як ставки на аукціоні, де кожен користувач

змагається один з одним, щоб їх транзакції були швидше інших включені в новий блок. Транзакції тих, хто робить вищі ставки, тобто заплатить вищу комісію, будуть швидше підтверджені в процесі майнингу, а тим, чиї ставки менші, доведеться чекати довше.

У blockchain мережах майнінгом називають процес перевірки транзакцій [6]. Перевірка необхідна для створення нових блоків у blockchain задля забезпечення функціонування мережі. У більшості мереж майнінг зводиться до серії обчислень із перебором параметрів для знаходження хешу із заданими властивостями. Криптовалюти використовують різні моделі обчислень, але вони завжди досить тривалі за часом для знаходження прийнятного варіанту та швидкі для перевірки знайденого рішення. Такі обчислення використовуються алгоритмами токенів для забезпечення захисту від повторного витрачання одних і тих самих одиниць [12].

За створення нових структурних одиниць в мережі, зазвичай передбачено винагороду, яка стимулює людей надавати свої обчислювальні потужності. Нагородою можуть бути комісійні збори або нові емітовані одиниці криптовалюти. Blockchain мережі використовують різні консенсусні алгоритми й кожен з них може мати власний механізм. Оскільки вони відіграють важливу роль у безпеці мережі, вивчення пов'язаних із ними даних може бути цінним для фундаментального аналізу.

Механізм консенсусу – це стійкий до відмови механізм, який використовується в комп'ютерних системах для досягнення необхідної згоди по одному значенню даних або одному стану мережі серед розподілених процесів або багатокористувацьких систем. У blockchain мережах консенсусом вважають угоду про те, які транзакції та в якому порядку фіксувати у БД. Наприклад blockchain Bitcoin використовує механізм консенсусу доказу роботи [12].

Доказ роботи (Proof of Work, PoW) – це загальний алгоритм консенсусу, що використовується в найпопулярніших blockchain мережах, таких як Bitcoin та Litecoin. Він вимагає від вузла-учасника довести, що виконана та

представлена ним робота дає йому право на отримання права додавати нові транзакції до blockchain. Недоліками цього алгоритму є тривалий час обробки та потреба у великих витратах електроенергії, що послужило поштовхом до створення нових та ефективніших механізмів.

Доказ частки (Proof of State, PoS) – ще один поширений алгоритм консенсусу, який розвивався як недорога та енерговитратна альтернатива алгоритму PoW. Він передбачає розподіл відповідальності за підтримку blockchain між вузлами-учасниками пропорційно до кількості токенів, що належать їм. Недоліком цього алгоритму є накопичення криптовалюти замість її витрачання.

Хоча PoW і PoS є найбільш поширеними серед blockchain мереж, існують інші алгоритми консенсусу. Доказ простору (Proof of Space, PoSpace) або доказ дієздатності (Proof of Capacity, PoC) дозволяє розділити простір пам'яті вузлів, що беруть участь в blockchain. Чим більше пам'яті чи місця на жорсткому диску вузол надасть для мережі, тим більшу частку роботи він отримає від алгоритму.

Доказ діяльності (Proof of Activitiy, PoA), використовується в blockchain «Decred» і є гібридом, який використовує аспекти як PoW, так і PoS. Доказ згоряння (Proof of Burn, PoB) є ще одним методом консенсусу, який вимагає відправляти невеликі суми монет на недоступні адреси гаманців, фактично спалюючи їх без можливості повернення. Доказ історії (Proof of History, PoH), проєкт Solana має механізм подібний до доказу минулого часу (Proof of Elapsed Time, PoET), тобто сам процес криптографічно шифрується для досягнення консенсусу без значних витрат ресурсів.

Хешрейтом (hashrate) мережі називають її обчислювальну потужність. Для криптовалют з доказом виконання роботи, хешрейт можна вважати показником здоров'я мережі. Чим більше обчислювальна потужність, тим складніше успішно провести атаку на мережу. Збільшення хешрейту може вказувати на зростання зацікавлених у майнінгу, через низькі витрати й високу прибутковість. І навпаки, зменшення обчислювальної потужності свідчить про

те, що майнери відключаються від мережі, тому що їм не вигідно забезпечувати її безпеку.

Щоб отримати on-chain метрики, можна встановити вузол бажаної мережі й експортувати дані безпосередньо з мережі, але це може вимагати значних часових та фінансових витрат. Значно швидшим та простим, але водночас менш надійним рішенням є отримання інформації зі сторонніх ресурсів та API.

На відміну від on-chain метрик, які базуються на доступних кількісних даних про мережу, проєктні метрики використовують якісний підхід до оцінки ефективності криптовалюти. Вони фокусуються на внутрішніх та зовнішніх факторах, таких як мета створення, сценарії використання, план розвитку, показники активності команди розробників та її досвід.

При аналізі проєктних метрик інформацію беруть переважно з офіційних сайтів blockchain проєктів. Їх вміст досліджують на наявність технічних документів, схем, переліку інвесторів та членів команди, яка працює над проєктом. Крім з'ясування проблем, які вирішує нове blockchain рішення, потрібно порівняти його результативність з вже наявними конкурентами та їх загальною екосистемою [13].

Whitepaper – це найважливіший з технічних документів, який описує мету та принцип роботи проєкту. Технічні документи аналізуються людиною вручну. Тому потрібно робити це уважно та скептично. Документація проєкту повинна мати, як мінімум, такий перелік інформації: перелік використаних технологій, сценарії використання, заплановані функції та оновлення, інформацію про економіку та розпродажі токенів, інформацію про команду.

Більшість blockchain проєктів мають «дорожню карту» на майбутнє, що показує термін проведення релізів з запланованими функціями та тестувань. В той час як дорожня карта дає уявлення про вже реалізовану частину роботи та майбутній план розвитку, вивчення професійних досягнень членів команди допоможе визначити вірогідність успішного виконання всього план з розробки

проєкту. Перелік інвесторів та консультантів можна розглядати як умовний рівень довіри до проєкту.

Терміном «токеномiка» (tokenomics) описують збалансовану економічну модель зацікавленості учасників ринку в певній криптовалюті та її токенах. Попит та пропозиція показують економічну обґрунтованість ціни токенів та причину їх існування. Чим вище попит відносно пропозиції, тим вище ціна активу.

Фінансові метрики базуються на інформації про ринкову капіталізацію, ліквідність та обсяги торгів монетою. Ринкова капіталізація розраховується шляхом множення обсягу пропозиції, що циркулює, на поточну ціну. По суті, вона є гіпотетичною вартістю купівлі кожної одиниці криптовалюти. Значення ринкової капіталізації може ввести користувача в оману. Теоретично можливо випустити безкорисний токен із запасом у десять мільйонів одиниць. Якщо один такий токен продаватиметься за 1 долар, то ринкова капіталізація становитиме 10 мільйонів доларів. Проте ця оцінка спотворена через відсутність реального попиту на пропозицію. Насправді такий токен ніяк не зацікавить широкий ринок.

З іншого боку, неможливо точно визначити, скільки одиниць даної криптовалюти знаходиться в обігу. Монети можуть згорати, потрапляючи на не активні адреси, ключі від гаманців можуть бути втрачені й більше ніколи не повернуті. Натомість аналізатори показують приблизні дані з урахуванням відфільтрованих монети, які вийшли з обігу.

Ліквідність – це показник того, наскільки легко актив можна купити чи продати. Ліквідним вважається той актив, який ми одразу можемо придбати та продати за його ринковою ціною. Важливо завчасно виявляти малоліквідні та неліквідні активи, тому що виникне проблема продажу особистих активів за справедливою ціною. При відсутності пропозицій від покупців на торговій платформі, у власника неліквідного активу залишається вибір: знизити ціну своєї пропозиції або чекати підвищення ліквідності.

Обсяг торгів – це показник, який може допомогти визначити ліквідність. Він може вимірюватися декількома способами та показує, об'єми які було продано за певний період. Зазвичай на графіках зображують денний обсяг торгів в одиницях криптовалюти чи доларах.

1.4 Аналіз існуючих аналогів

CoinMarketCap – найпопулярніший у світі сайт для відстеження цін на криптоактиви, який надає користувачам неупереджену та точну інформацію про криптовалюти (рис. 1.1). Сервіс заснований у травні 2013 року, і на 2021 рік налічував понад 6 тисяч монет. Уряд США використовує дані з цього джерела у своїх дослідженнях і звітах, що свідчить про високий рівень довіри до джерела та надійність інформації [14]. Окрім інформації про зміну ціни, CoinMarketCap пропонує власні програмні продукти та сервіси з можливістю підписки за різними тарифами.

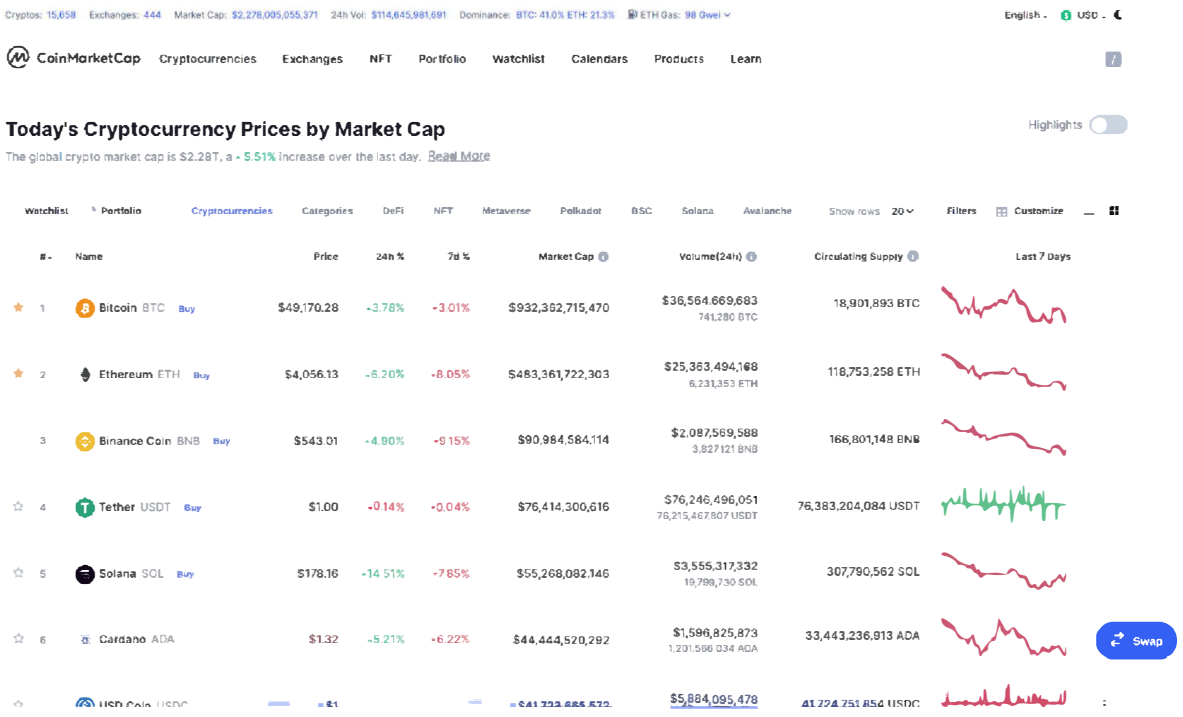

Рисунок 1.1 – Головна сторінка сайту CoinMarketCap

Після натиснення на монету з переліку відкривається сторінку з деталями (рис. 1.2). На цій сторінці доступні декілька розділів, а саме: огляд,

ринки, історичні дані, тримачі активу, проєктна інформація, гаманці, новини, оновлення з соціальних мереж, рейтинги та аналітика.

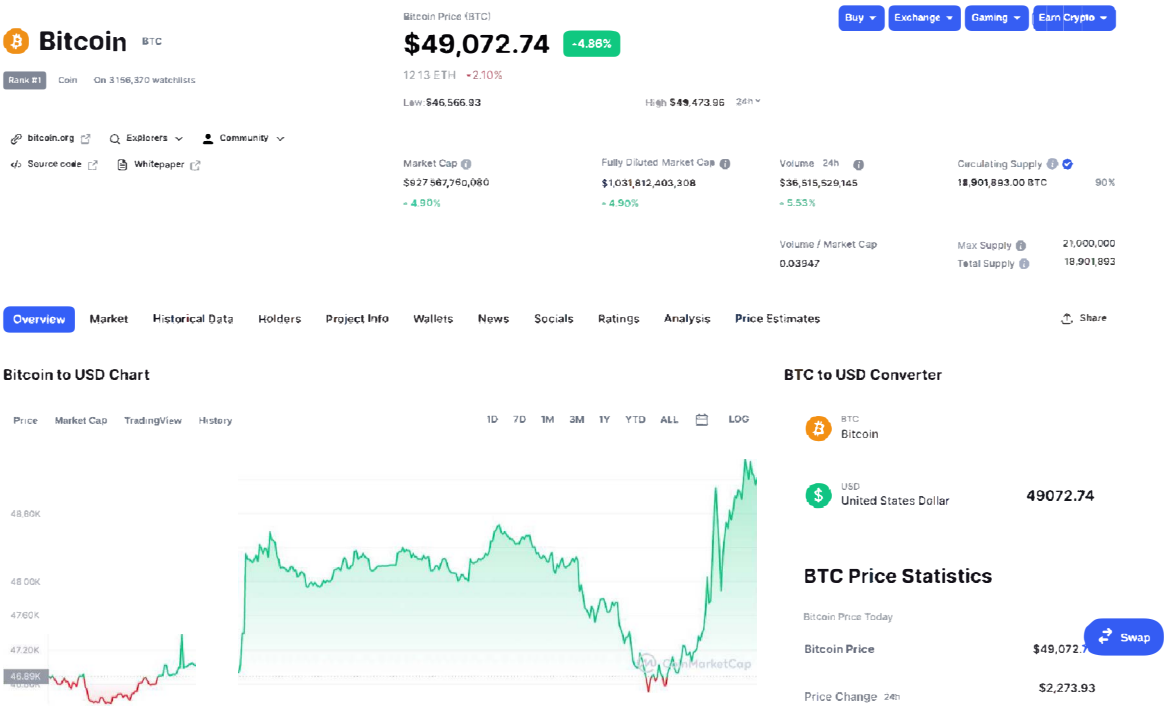

Рисунок 1.2 – Сторінка криптовалюти Bitcoin на CoinMarketCap

У верхній частині вказана ринкова вартість, ціна, обсяг торгів, пропозиція, зміна ціни за останні періоди (24 години, 1 день, 7 днів, 30 днів і т.д.), графік зі зміною ціни за аналогічні періоди які можна обрати та багато інших метрик. Крім того, зверху можуть знаходитись корисні посилання на ресурси проєкту, такі як офіційний сайт, дослідник блоків об'єднання в соціальних мережах та репозиторії. Ціна криптовалюти обраховується як середнє значення на основі даних зі всіх торгових бірж. На вибір можна обрати чотири види даних для побудови графіку. Після графіку, на сторінці знаходяться всі ринкові біржі та обмінники які містять торгові пари з обраним активом.

Перевагою CoinMarketCap є зручний інтерфейс, який об'єднує в собі велику кількість джерел інформації й дозволяє отримати надійні, високоточні показники та широкий спектр посилань на зовнішні джерела. Крім того, нові

проєкти прагнуть потрапити в перелік монет, які відображені на сайті тому, що це може стати джерелом нових інвесторів та користувачів активом.

Ще одним поширеним ресурсом для аналізу криптоактивів є сервіси компанії «Blockchain.com» (рис. 1.3). Ця криптовалютна компанія надає платформу, де можна одночасно зберігати, використовувати, керувати активами та надає інструменти для відстежування транзакцій.

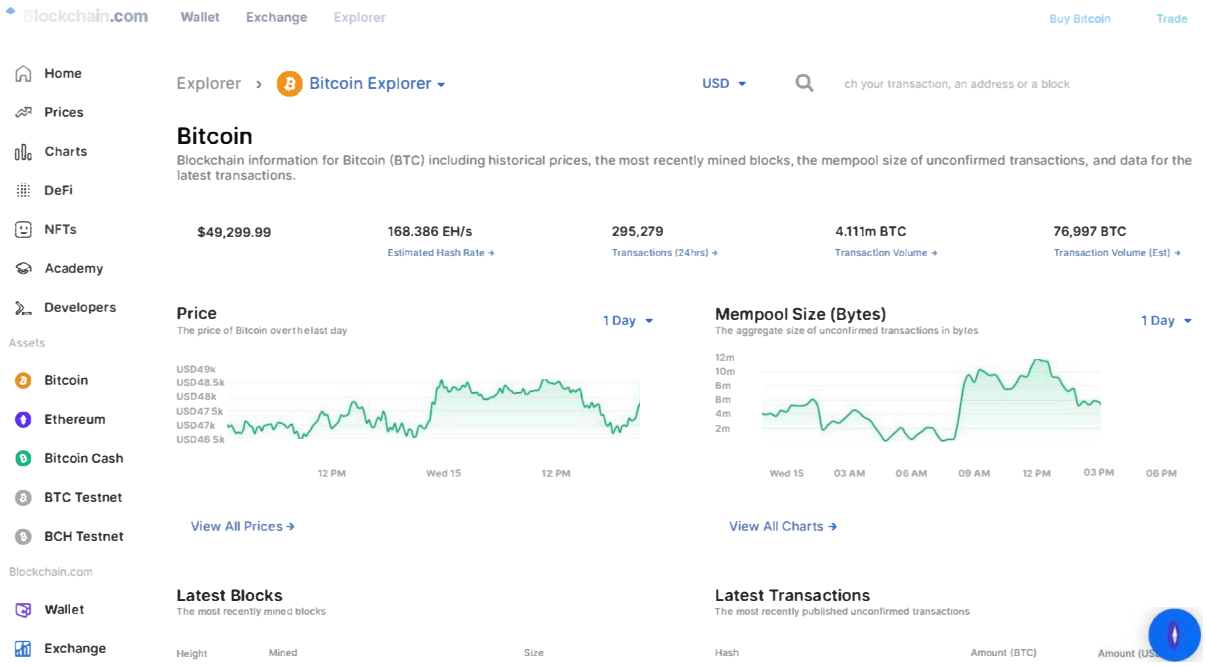

Рисунок 1.3 – Сторінка дослідника блокчейну Bitcoin на Blockchain.com

Компанія має власні дослідники blockchain мереж, які дозволяють користувачеві відслідковувати транзакції та пов'язану з ними інформацію [15]. Компанія також має власну біржу криптовалют, з якої можна отримати фінансові показники.

Перевагою blockchain.com є наявність власних дослідників blockchain мереж Bitcoin, Ethereum та Bitcoin Cash. Це дозволяє будь-кому, хто має хеш-код транзакції, побачити адреси гаманців відправника і отримувача, кількість токенів та комісію. Інструмент можна використовувати для аналізу активності мережі, та рівномірності розподілу монет серед унікальними гаманцями.

1.5 Постановка задачі на розробку аналізатора криптовалют

При проєктуванні та розробці нової системи для аналізу криптоактивів варто поєднувати агрегацію загальних і фінансових метрик разом з інтеграціями сторонніх сервісів в єдиному користувацькому інтерфейсі. Інтеграція сторонніх попередньо перевірених сервісів є доцільною, якщо це спрощує та пришвидшує процес розробки не накладаючи значних обмежень на систему. Для імплементації власного інструменту технічного аналізу фінансових чартів, потрібна невизначена кількість часу на додаткові дослідження суміжних предметних областей та специфічних алгоритмів. Тому варіантом рішенням є інтегрування існуючих сервісів для аналізу. Критеріями вибору є високий рівень надійності, відповідність наявного функціонала до необхідного та частота використання в інших програмних продуктах.

Розроблена система повинна надавати простий та зручний у використанні користувацький інтерфейс, інтегрувати сторонні сервіси для аналізу даних з перевірених джерел. Крім того, короткі аналітичні висновки у вигляді речень природною мовою повинні формуватися для користувачів. Для більшої точності, застосунок повинен агрегувати та інтегрувати декілька джерел інформації й підтримувати такі функціональні можливості:

1. Агрегація загальних, фінансових, blockchain та соціальних метрик для знайдених криптовалют.
2. Побудова інтерактивних таблиць криптовалют з можливостями сортування, зміни кількості вмісту та пагінації.
3. Пошук криптовалют за назвою з можливістю збереження в обране.
4. Побудова інтерактивних чартів та графіків з інструментами технічного аналізу фінансових показників, таких як історія зміни ціни, ринкова капіталізація і подібне.
5. Відображення підказок, аналітичних оцінок та висновків у форматі чисел або повідомлень природною мовою.
6. Інтегрування в додаток сторонніх джерел інформації та сервісів аналізу.

Для стабільної роботи застосунку слід забезпечити такі технічні характеристики:

1. Операційна системи: Windows 7+, MacOS 10.10+, Android 4.4+, IOS 7+, дистрибутиви Linux або інші ОС з підтримкою сучасних веб-браузерів.
2. Оперативна пам’ять: від 1 Гб.
3. Частота процесора: від 1500 МГц.
4. Швидкість інтернету: від 1 Мб/с.
5. Веб-браузер: Chrome, Firefox, Opera, Edge, Safari або інший з підтримкою сучасних web-стандартів.

Для локального запуску розробленої системи та внесення змін у програмний код додатка, необхідно встановити пакетний менеджер (npm або yarn) і встановити всі необхідні залежності (пакети).

Для редагування коду можна використати довільний текстовий редактор, редактор коду (VS Code, Sublime і т.п.) або спеціалізоване середовище розробки (WebStorm, Intellij Idea і т.п.).

Визначивши функціональні та технічні вимоги до розроблюваного аналізатора криптовалют, зробимо висновки на основі інформації вивченої в ході дослідження предметної області.

Висновки до першого розділу

Отже, головними проблемами в предметній області аналізу криптовалют можна вважати високий рівень волатильності всього ринку в цілому та велику кількість проєктів з високим ризиком шахрайства. Як наслідок, обидві проблеми зводяться до прийняття збиткових фінансових рішень та втрати вартості інвестицій користувача. Для розв'язання цих проблем вже існує ряд безплатних та платних програмних продуктів. Для пришвидшення процесу розробки, доцільним підходом є інтегрування наявних сервісів та інструменти, оскільки розробка власних аналогів вимагає значну кількість часу на дослідження суміжних предметних областей.

Для забезпечення високої точності та надійності інформації, поширеною практикою є агрегація показників з декількох джерел, з подальшим порівнянням результатів між собою. Варто зауважити, що одне джерело не може одночасно надавати точні дані про загальні, технічні, фінансові, соціальні та подібні аналітичні метрики. Тому що дозволяє вивести середні значення і перевірити відхилення в точності інформації з кожного місця.

Обидва вище вказані підходи до аналізу активів передбачають, що кінцеві висновки формуються людиною. Важливими складовими успішного аналізу є точність та правдивість інформації наданої системою. Варто пам'ятати, що першочергово аналізатори використовуються для прийняття фінансових рішень спрямованих на мінімізацію ризиків втрати цінності власних активів. На основі даних від аналізатора, користувач має самостійно формувати остаточні висновок та приймати фінансові рішення.

РОЗДІЛ 2
ОПИС І ПРОЄКТУВАННЯ АГРЕГАТОРА ІНФОРМАЦІЇ

2.1 Опис функціонування системи

2.1.1 Вибір технології для опису та передачі даних

Вибір правильного формату API безпосередньо впливає на архітектуру програми, ефективність роботи та кількість зусиль необхідну для впровадження й підтримки системи протягом терміну її існування.

При проєктуванні API, стандартною практикою є дотримання архітектури передачі стану репрезентації (REST). Цей стандарт був розроблений у 2000 році для отримання даних (ресурсів) від сервера, за URL-адресами [16]. Ця архітектура поширила нові концепції дизайну API – сервери без збереження стану та структурований доступ до ресурсів. Проте, система призначена для агрегації даних з різних джерел, кожне з яких має власний формат, вимагає більш гнучкої обробки складних та об’ємних структур даних. Розглянемо три недоліки REST архітектури.

Зі збільшенням кількості користувачів мобільних пристроїв, виросли вимоги до ефективного завантаження даних. Другий недолік полягає у різноманітності REST клієнтів, що ускладнює створення та синхронізацію між API, оскільки потрібна фіксована структура даних. З цього випливає повільна швидкість розробки. Для внесення змін на стороні клієнта в REST запити, часто доводиться змінювати серверну сторону для підтримки запитів, що уповільнює ітерації при розробці продукту. Архітектурно більшість відповідей REST сервера містить або надлишкові дані або недостатню їх кількість, що створює потребу в відправці нових запитів залежних від кількості інформації отриманої з попередніх.

В ході порівняння переваг та недоліків двох поширених форматів API для обміну складно структурованими даними було вирішено використовувати

специфікацію GraphQL. Ця технологія розроблена компанією Facebook як ефективна альтернатива формату REST з усуненням ряду його недоліків. Порівняння технологій наведено у таблиці 2.1.

Таблиця 2.1 – Порівняння REST та GraphQL

Характеристика	GraphQL	REST
Архітектура	Клієнто-орієнтована	Серверно-орієнтована
Організація даних	Схема даних, система типів	Окремий шлях для кожного ресурса
Операції	Запит, Мутація, Підписка	GET, POST, PUT, DELETE, PATCH
Отримання даних	Лише потрібні дані одним запитом	Фіксовані дані декількома запитами
Продуктивність	Один швидкий запит	Декілька повільних запитів
Швидкість розробки	Висока	Низька
Складність вивчення	Висока	Помірна
Само документованість	+	-
Завантаження файлів	Ускладнений процес	+
Кешування	З використанням сторонніх бібліотек	+
Стабільність	Автоматична перевірка вмісту запитів та типів	Вбудована перевірка вмісту запитів відсутня

GraphQL – це специфікація для створення API, яка описує окрему мову для опису і взаємодії з API та середовище виконання на стороні сервера для виконання запитів за допомогою власної системи типів. Ця технологія є «language agnostic», тобто відсутня прив'язка до конкретної мови програмування, БД або механізму зберігання, що дозволяє обирати між

технологіями [17]. Специфікація базується на декількох ключових понять, коротко розглянемо їх:

схема (schema) це ядро реалізації сервера, яке описує функціональні можливості, доступні для клієнтських програм;

- запит (query) на читання або отримання значень;

мутація (mutation) це запит, який змінює дані в сховищі даних;

- тип (type) визначає структуру даних, які використовуються в системі;

обробник (resolver) являє собою одну або колекцію функцій, які генерують відповідь на запит клієнта.

Після запуску служби GraphQL на сервері, можна надсилати запити за однаковим URL, який складається з протоколу (HTTP/HTTPS), домену та шляху «/graphql» для перевірки та виконання. Спочатку запит перевіряється, чи відноситься він до визначених типів і полів, а потім запускає виконання описаних в запиті функцій для отримання результату.

Схема (або система) типів GraphQL формується з описаних розробником типів, які можна отримати від сервера, і які поля він має. Крім цього, GraphQL має вбудований набір стандартних типів даних (скалярів):

1. Int (32-бітове ціле число зі знаком).
2. Float (число з плаваючою комою та подвійною точністю).
3. String (рядок з послідовністю символів UTF-8).
4. Boolean (логічний тип, що може мати лише значення true або false).
5. ID (скалярний тип для унікальних ідентифікаторів).

У більшості реалізацій GraphQL також є можливість створювати власні скалярні типи. Таким чином ми отримуємо гнучку систему типів, в якій можна описати як власні скалярні примітиви, так і високорівневі класи з ієрархіями. Більш того, кожен запит в системі має чітко описаний інтерфейс, тобто перелік вхідних параметрів та вихідних типів [18].

Серверна частина GraphQL дозволяє клієнту робити точні запити даних і отримувати виключно те, що потрібно. Розробнику легше орієнтуватись маючи перед собою єдину строго типізовану схему (модель) даних та запитів,

які може обробити сервер. Тобто, схема служить уніфікованим інтерфейсом між клієнтом та сервером і визначає способи доступу до даних.

Записані мовою визначення схем (SDL), основні компоненти схеми GraphQL слід описати відповідно до типу кожного поля та об'єкта. Сервер буде обробляти лише описані в схемі типи. Схема визначає, які запити можна робити, які типи даних допустимі в системі, а також визначає відносини між цими типами. Навколо схеми можна будувати серверні додатки будь-якою мовою програмування, яка підтримує специфікацію GraphQL.

З GraphQL можна адаптувати запити до технічних потреб. Як зазначалося, при створенні документації REST API потрібно описувати окремі кінцеві точки, їх призначення та параметри, які розробник може їм передати. Описуючи типи даних, поля та точки взаємодії між ними, GraphQL дозволяє розробникам об'єднувати нові сервери з вже існуючими у вигляді єдиної кінцевої точки (див. рис. 2.1).

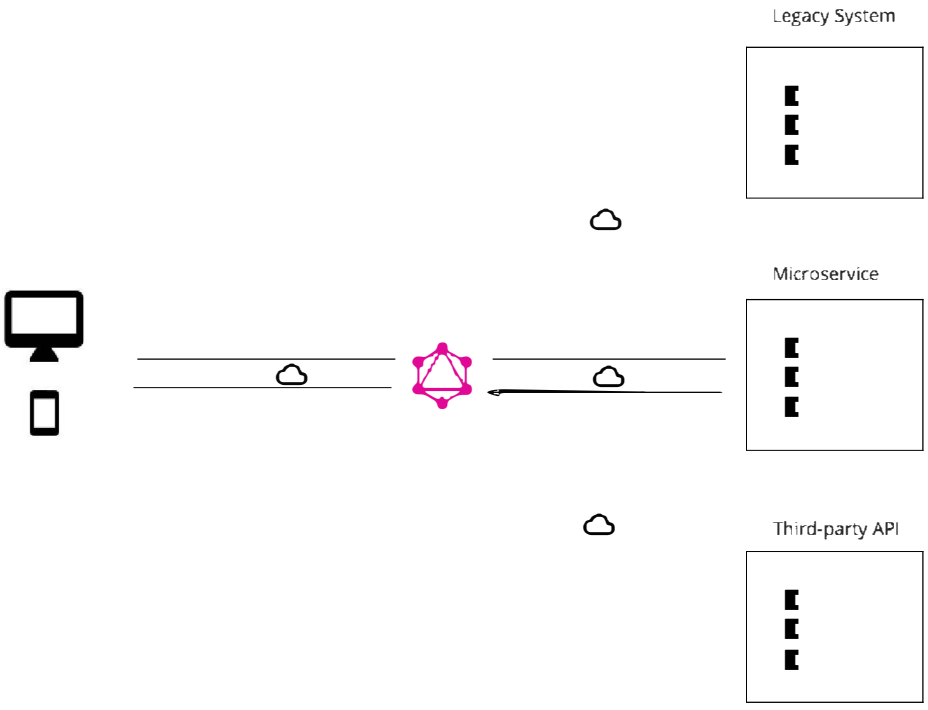

Рисунок 2.1 – Загальна діаграма архітектурного застосування GraphQL

Функція інтроспекції GraphQL додає до типів посилання, що дозволяє виявити їх в схемі та гарантувати повернення лише строго описаних типів

даних у відповідному форматі. Завдяки простому процесу створення документації, розробник одразу може побачити типи даних та запити описані в схемі. Виходячи з цього, можна додавати нові поля до існуючих запитів зі зворотною підтримкою старого інтерфейсу. Крім того, зникає потреба перевіряти формат даних, оскільки GraphQL робить це автоматично. Розробникам потрібно лише написати функцію обробник, котра описує структуру та допустимі типи даних.

GraphQL технічно підходить для розробки складних систем і мікросервісів. Інтегруючи кілька систем за своїм API, він об'єднує їх за єдиним URL і зменшує складність написання запитів. Сервер GraphQL несе відповідальність за отримання даних з існуючих систем та їх форматування у відповіді. Це особливо актуально для застарілих проєктів або сторонніх API, які з роками оновлюються та розширювалися і вимагають довготривалого обслуговування.

Під час міграції з монолітної серверної програми на мікросервісну архітектуру, GraphQL спрощує налагодження зв'язку між кількома сервісами, об'єднуючи їх в одну схему. Хоча кожна сервіс визначає свою власну схему і має власну кінцеву точку, один шлюз, який об'єднує всі схеми в одну глобальну схему.

Основна відмінність між GraphQL і REST полягає в тому, що останній зосереджений навколо окремих кінцевих точок, тому для збору та агрегації всіх необхідних даних розробник повинен виконати декілька запитів та об'єднати всі результати. У той час як GraphQL дозволяє розробнику отримати потрібне за один запит.

Формування документації за замовчанням дозволяє за мінімальну кількість дій доповнити існуючий опис. Крім того, документація автоматично синхронізується зі змінами назв, типів або параметрів запиту внесеними розробників у файл зі схемою типів. Для коректної роботи GraphQL сервера обов'язково потрібно описати схему представлення (або типів) в одному або декількох окремих файлах з розширенням «.gql».

Зміни в REST API створюють необхідність оновлювати документацію та підтримувати застарілі версію доки клієнти не мігрують на нову версію. Таким чином, REST зазвичай пропонує кілька версій API. В свою чергу GraphQL усуває потребу використання версій, припиняючи підтримку запиту на рівні поля. Застарілі поля можна пізніше видалити зі схеми, не впливаючи на наявні запити. Відсутність різних версій API спрощується підтримку та впровадження нових функцій в існуючу систему і допомагає створювати більш гнучкі та передбачувані запити.

При використанні REST архітектури необхідно самостійно перевірити заголовки HTTP запитів на наявність одного з десятків можливих статусів відповіді. Лише на основі статусу або не уніфікованого повідомлення доданого розробником можна визначити, що пішло не так. В той час як, при обробці запитів GraphQL сервер автоматично перевіряє запит на синтаксичні помилки та відповідність типів до існуючої схеми. При виявленні помилки сервер повертає детальне повідомлення про можливу причину з посиланням на рядок з запиту, де вона була виявлена. Повідомлення про помилки можна змінювати на власні виходячи з потреб, будь то трасування стека, спеціальний код помилки програми чи просто текст.

Крім того, GraphQL підтримує складну і часто використовувану операцію підписки (SUBSCRIPTION), яка дозволяє клієнтам отримувати повідомлення в режимі реального часу від сервера. Підписки на оновлення сервера можна використовувати для автоматичного надсилання сповщень клієнту, коли додається новий коментар або дані, або отримано повідомлення.

Отже, враховуючи всі наведені вище переваги GraphQL можна підсумувати, що ця технологія дозволяє проводити швидке створення прототипів програмних продуктів, які використовують CRUD операції. GraphQL прискорює процес розробки, надаючи єдину кінцеву точку API, яка служить проксі-сервером даних між інтерфейсом користувача та сховищем даних. Крім того, швидкість розробки тісно пов’язана з покращеним досвідом

використання при розробці. Технологія пропонує швидкий процес строгого опису схеми даних, що дозволяє автоматично виявляти та обробляти помилки.

До недоліків GraphQL можна віднести складність вивчення його концепцій. Хоча поняття складності вивчення є відносним, не кожен розробник знайомий з технологією і REST архітектура часто вивчається першою, що часто є причиною вибору саме її. Тому великі часові затрати на належне ознайомлення з концепціями є недоліком. Ще одним значним недоліком є відсутність зручного способу для роботи з файловими типами даних, тому що функція завантаження файлів не включена в специфікацію.

2.1.2 Вибір архітектури та платформи для серверного додатка

Екосистема побудована довкола мови JavaScript (JS) постійно збільшується, це помітно по динаміці збільшення нових бібліотек та пакетів за останні місяці. Оскільки предметна область аналізу криптовалют пов’язана з web технологіями, для серверного додатка доцільним буде використання типізованої надбудови над JS, тобто мову TypeScript (TS).

Серед високорівневих backend рішень, найбільш гнучким та детально задокументованим є фреймворк NestJS розроблений на мові Typescript. В його основі використовується більш низькорівневі серверні HTTP бібліотеки, такі як Express або Fastify [19]. Це технічне рішення використовує перевірені часом архітектурні шаблони проєктування з високим рівнем покриття тестами. Крім того, перевагами можна вважати детальну документацію та наявність правил структурування коду.

Перевагою NestJS в порівнянні з Express є наявність чіткої модульної архітектури, що дозволяє дотримуватись концепцій ООП та створювати системи з мінімальною зв'язністю компонентів. За необхідності таку монолітну структуру можна розділити на окремі мікросервіси. Завдяки використанню типізованої мови (TS), при розробці можна налаштувати

статичний аналіз коду, виведення попереджень та помилок компіляції з точним посиланням на рядки коду.

NestJS містить вбудований контейнер впровадження залежностей (Dependency Injection, DI). Цей шаблон проєктування використовується, щоб зробити програми більш ефективними та модульними [19]. Він використовується для підтримки коду, та спрощує процес розробки вирішуючи проблему порядку створення компонентів системи, використовуючи граф залежностей.

Модульна організація проєкту дозволяє розділити систему на окремі функціональні частини предметної області. В результаті, виходять окремі слабко зв'язані між собою підсистеми, що підвищує надійність та спрощує використання зовнішніх бібліотек. NestJS спрощує тестування, підтримуючи такі функції, як контейнеризація та DI модулів.

Nest підтримує всі поширені інструменти розробки серверних додатків, такі як проміжні функції (middlewares), фільтри, канали, перехоплювачі, GraphQL, сокети (websockets) та подібне. Варто зауважити, що фреймворк надає два різні способи створення GraphQL застосунків: спочатку схема (schema first) та спочатку код (code first). У першому підході джерелом істини є файли, що містять мову визначення схеми (schema definition language або SDL), а всі описані в схемі типи будуть автоматично згенеровані у вигляді TS типів. У другому випадку для генерації GraphQL типів використовуватимуться TS класи з декораторами, тобто з метаданими.

Чіткі рекомендації по структуруванню проєктів спрямовані на отримання організованої структури логічно та шаблонно розділених папок. Nest має окремий (Command Line Interface, CLI), який спрощує генерацію та наповнення проєкту шаблонним кодом.

Отже, для імплементації серверної частини було обране відносно нове рішення в області серверної розробки, з якісною документацією та великим переліком готових рішень для швидкого створення та розгортання сервісів, з

дотриманням кращих архітектурних принципів та підтримкою вимог сучасних клієнтських додатків.

2.1.3 Вибір архітектури та платформи для клієнтського додатка

Для створення найбільш універсального інтерфейсу, першочерговою платформою для клієнтського додатка було обрано web-браузер. Для пришвидшення процесу розробки доцільно використати детально задокументований високорівневий інструмент побудови користувацьких інтерфейсів такий як Angular, React або Vue. Серед трьох перелічених варіантів, кожен може бути взаємозамінним в предметній області аналізу. Однак, на відміну від конкурентів в React вся логіка компонентів написана на JavaScript та його розширенні (JSX), що спрощує процес вивчення можливостей бібліотеки. Бібліотека дозволяє швидко компонувати власні та сторонні компоненти в складні користувацькі інтерфейси [20].

У ході дослідження ряду пакетів для взаємодії з GraphQL сервером, систем готових компонентів побудови інтерфейсу та чартів, було встановлено, що лише React має інтеграцію з найбільш функціональним GraphQL клієнтом – Apollo Client, який повністю відповідає технічним вимогам кешування. Першочергово було обрано пакети які передбачають інтеграцію в React додаток, такі як Apollo Client, Chakra UI, React Table та ApexCharts. Коротко розглянемо переваги кожної обраної бібліотеки.

Apollo Client – це повноцінна бібліотека керування станом застосунків для мови JavaScript, яка дозволяє керувати локальними та віддаленими даними за допомогою GraphQL [21]. Крім того, унікальною перевагою є наявність вбудованих механізмів кешування та автоматичної синхронізації даних програми при оновленні інтерфейсу користувача. Apollo Client допомагає структурувати код декларативним і передбачуваним способом, що відповідає сучасним методам розробки. Варто зауважити що цей клієнт має ряд інструментів, які значно спрощують процес написання та відлагодження клієнтських запитів.

Однією з найважливіших переваг Chakra UI є налаштування бібліотеки під власні вимоги дизайну, щоб визначити такі речі, як кольори, розміри шрифту, відступи, точки зупинки для адаптивного дизайну, тіні та подібне [22]. Достатньо виконати невелику конфігурацію.

Chakra UI підтримує сучасну тенденцію побудови користувацьких інтерфейсів – перемикання світлої та темної теми оформлення. Темний режим зручний при поганому освітленні, тобто зменшується яскравість кольорів інтерфейсу на темніші. Більшість компонентів Chakra сумісні з темним режимом за замовчуванням, що зменшує обсяг роботи. Крім того, бібліотека дозволяє виявляти колір обраний в операційній системі клієнта та автоматично застосувати її у себе.

Важливим критерієм для одночасної підтримки користувацьких інтерфейсів для комп'ютерів та мобільних пристроїв є використання і підтримка адаптивних стилів. Chakra UI дозволяє автоматично створює необхідні медіа-запити, вказавши потрібні значення в JSX об’єктах і масивах адаптивних стилів. Крім того, можна встановлювати власні точки зупинки, при досягненні яких, стилі компонентів будуть змінюватися

Бібліотека React Table призначена для побудови інтерактивних комплексних таблиць. Вона легко інтегрується в наявні системи без порушень та обмежень попередньої функціональності. Варто підкреслити що при побудові таблиць, їх стилі та будь-яка логіка повністю контролюється розробником. Бібліотека має вбудовану фільтрацію, сортування, групування, агрегації, розбиття на сторінки та відображення наборів даних з мінімальною кількістю налаштувань [23].

ApexCharts – це сучасна бібліотека діаграм написана для JavaScript, яка дає змогу розробникам створювати інтерактивні візуалізації даних для комерційних і некомерційних проєктів. Перевагами бібліотеки є детальна документація API, понад 100 готових до використання зразків інтерактивних діаграм та гнучка система налаштування їх вигляду [24].

2.1.4 Проєктування основних функцій криптоаналізатора

Навігація між сторінками буде відбуватись через меню з посиланнями на всі доступні сторінки. Сторінки повинні бути логічно розділеними за їх наповненням. До переліку сторінок мають входити: «Монети», «Діаграми», «Тренди», «Дослідження», «DeFi» та «Обране».

Головною сторінкою вважатиметься сторінка з назвою «Монети», на якій можна буде побачити коротку інформацію про активи відсортовані за рейтингом. Для швидкого завантаження сторінки, список всіх монет буде розбито на лімітовані за кількістю одиниць сторінки, кожну з яких можна отримати виконавши окремий запит. Колонки таблиці сортуватимуться за вмістом від меншого до більшого і навпаки. Кожен рядок в таблиці має містити посиланням на сторінку з детальнішою інформацією про актив.

На сторінці з детальною інформацією про монету відображатимуться загальні, фінансові, соціальні, розробницькі та аналітичні показники. Крім того, на сторінку буде інтегрований сервіс побудови фінансових чартів «TradingView», аналітична платформа «IntoTheBlock» та таблиця з посиланнями на сторонні ресурси для купівлі активу.

При достатній кількості лімітованих запитів на хвилину, наприклад 50 запитів і більше, агрегація інформації зі сторонніх джерел може відбуватись без збереження даних у власну БД, натомість буде допустимим кешування запитів на невеликі проміжки часу.

До загальної інформації буде відноситись назва активу, його символ (або біржевий тикер), посилання на пов'язані з активом web-сайти та оцінки від сторонніх аналітичних джерел. До фінансових показників відноситься ціна активу відносно долару, ринкова капіталізація, торгові об’єми, циркулюючий об’єм, відсоток домінації на ринку, історія ціни.

Соціальні метрики являють собою дії користувачів, такі як перегляд, підписка на оновлення, створення потів або коментарів зі згадуванням активу та подібні дії зафіксовані в соціальних мереж та тематичних сайтах. Також

можна відображати кількість підписників, однак ці цифри можуть штучно підвищуватися і не відповідати дійсності.

Метрики розробки отримуються з проєктів розміщених у відкритих репозиторіїв таких як Github та Gitlab. Серед них береться: кількість зірок (вподобань), копіювань репозиторію (forks), загальна кількість обговорень, число вирішених питань, кількість дій розробників (commit) та запитів на злиття змін (merge request) і число змінених рядків за останні 4 тижні.

Для наочного зображення динаміки змін показників, вони подаватимуться у вигляді інтерактивних графіках з можливістю зміни часового періоду, графічного способу побудови та масштабу графіка. Дані з графіків можна буде зберегти у вигляді файлів в розповсюджених форматах json та cvs. На сторінку з трендами будуть таблиці з інформацією про активи або назви пов'язані з ними, що мають найбільшу кількість переглядів або згадувань користувачами за останню добу.

На сторінці для досліджень будуть розміщені діаграми з відсотковою зміною вартості активів з найбільшою капіталізацією, співвідношенням капіталізації криптовалют. У розділі децентралізованих фінансів (DeFi) буде інтегровано сервіс «DefiLama», який надає інформацію про децентралізовані проєкти та кількість активів заблоковану в blockchain мережах цих проєктів.

У користувача має бути можливість зберігати та видаляти монети до переліку обраних. Крім того, повинна бути можливість швидкого пошуку за назвою з кнопкою збереження до списку.

2.2 Розробка структурної схеми системи

Розроблюваний програмний модуль матиме клієнт-серверну архітектуру з двома обов'язковими та одним додатковим рівнем (див. рис. 2.2). Перевагою такої архітектури є гнучкість розроблюваної системи. Даний архітектурний підхід передбачає наявність одного чи декількох серверів, клієнтського додатка та джерела отримання даних або БД для збереження персистентних даних, тобто тих, які мають бути доступні після

перезавантаження програми. Така архітектура найкраще підходить для систем в яких можуть часто змінюватися й варіюватися формати даних, методи їх обробки або платформи для відображення. При збільшенні кількості відвідувачі, можна масштабуватись й адаптуватись до вищих вимог. Предметна область передбачає обробку декількох джерел даних та інтегрування з сервісами, тому важлива можливість розподіляти навантаження та додавати нові функції мінімально впливаючи на систему.

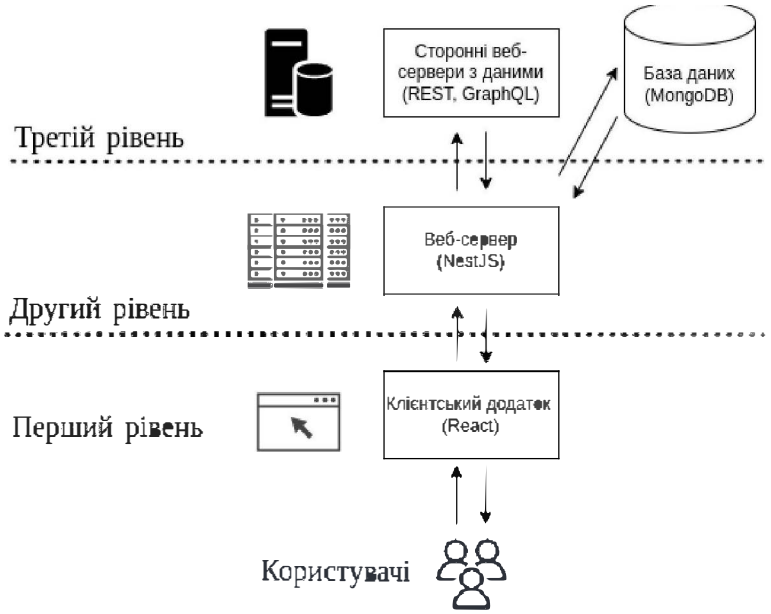

Рисунок 2.2 – Структурна схема системи

Взаємодію між клієнтським застосунком та сервером буде здійснюватися за допомогою технології GraphQL. Клієнт можна розробляти під окремі специфічні платформи. Доступ до серверів буде відбуватися по єдиному URL, таким чином створення та обробка запитів буде простіше в розробці. Крім того, для тестування розроблених запитів можна буде використати вбудовану в Apollo Server опцію «playground» [25]. Вона надає готовий інтерфейс для написання та відправки запитів на сервер без необхідності розробляти інтерфейс для тестування запитів.

Сервери Node.js працюють незалежно один від одного. Клієнтські застосунки також функціонують паралельно і незалежно. Перевагою є відсутність жорсткої прив'язки одного клієнта до сервера. Більш ніж типовою

є ситуація, коли один сервер одночасно обробляє запити від різних клієнтів, тому додатковим архітектурним рівнем системи може стати «API Gateway», який відповідає за розподіл навантаження (запитів) між доступними серверами. Клієнти мають знати про кількість доступних серверів, але можуть не мати жодного уявлення про існування інших клієнтів.

2.3 Розробка функціональної схеми системи

Основа архітектури розроблюваної системи включає клієнтську та серверну частини, тому розділимо їх на схемі та зобразимо ключові елементи кожної підсистеми. Клієнтська частина й web-сервер представлені як окремі додатки, що взаємодіють між собою через узгоджений програмний інтерфейс (API). Перевагою цього підходу є гнучкість системи, її частини можна технічно змінювати дотримуючись єдиного інтерфейсу взаємодії (рис. 2.3).

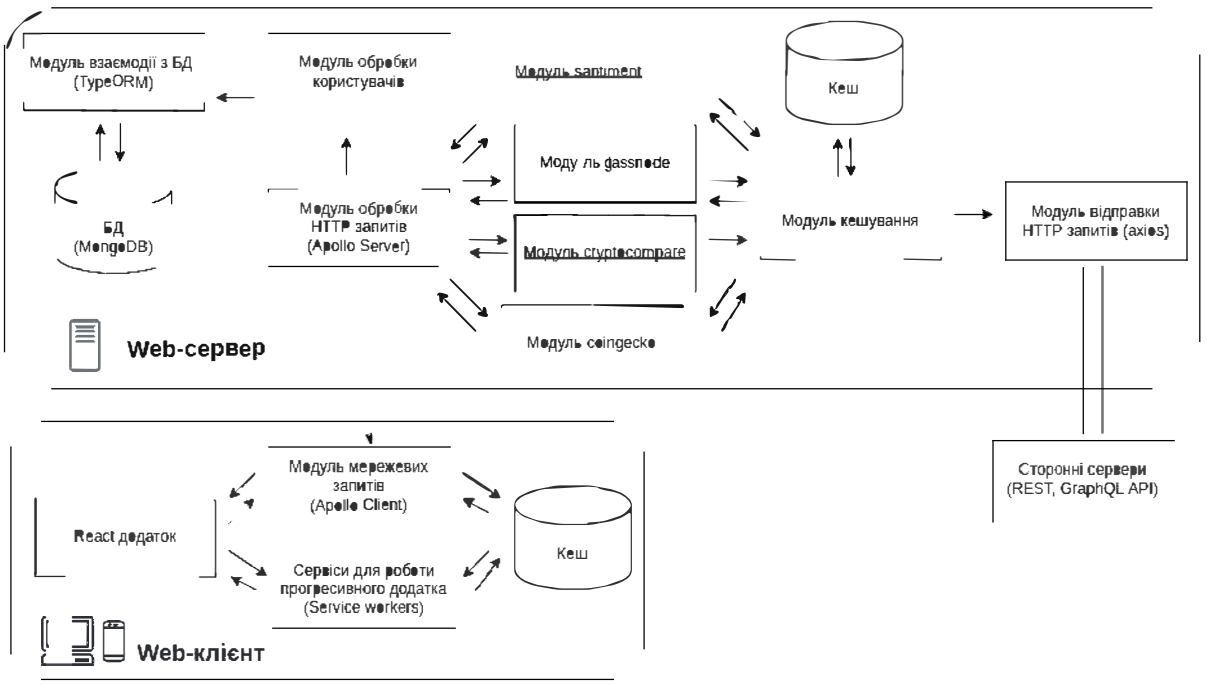

Рисунок 2.3 – Функціональна схема системи

Оскільки інформація присутня в аналізаторі є загальнодоступною, відсутня необхідність в реєстрації користувачів. Налаштування мови або

кольору теми не є критичною інформацією, тому допустиме локальне збереження у вигляді кешу на пристроях користувачів.

Клієнтський додаток відповідатиме за взаємодію з користувачем і обробку його дій. Він надає користувачу інформацію у візуальному інтерактивному вигляді. При взаємодії користувача з інтерфейсом, за певних умов клієнтська частина буде надсилати захищені HTTPS запити і обмінюватись даними з web-сервером. У якості клієнту для створення та обробки GraphQL запитів обрано Apollo Client. Зменшення надлишковості даних покращує швидкість виконання та розмір запитів.

Сервер відповідатиме за обробку запитів та даних за описаною логікою, взаємодію з кешем, сторонніми джерелами інформації та БД. До бізнес-логіки входить валідація вхідних та вихідних даних та їх більш специфічна обробка й агрегація, відповідно до строго визначеної системи типів. Крім того, на сервері можна використати механізми кешування.

Документо-орієнтована СКБД MongoDB є опціональною частиною архітектури, яку в базовій імплементації системи можна замінити на кешування. Використання БД може збільшити пропускну здатність системи шляхом локального збереження, оновлення складно структурованих даних. Перевагою цієї СКБД є вбудовані механізми масштабованості. При необхідності збільшити кількість оброблюваних запитів можна створити кластер з вузлами, тобто декількома БД, які розподіляють інформацію між собою та синхронізують інформацію за певними алгоритмами консенсусу. Однак з цього випливає, що розгортання власної БД та її масштабування вимагає значних обчислювальних потужностей та об’ємів дискової пам’яті.

Першочергово система може використовувати БД для збереження унікальних ідентифікаторів кожної монети з різних джерел інформації, таких як «CoinGecko», «IntoTheBlock», «Glassnode» і побідних. Це потрібно, щоб перевіряти чи підтримується криптовалюта обраним джерелом і відображати інформацію лише за умови наявності в БД.

2.4 Розробка діаграми процесів, які відбуваються в системі

Враховуючи вище описані функціональні можливості та вимоги до програмного продукту, спроєктуємо на їх основі діаграму процесів, які будуть відбуватися в аналізаторі криптовалют (рис. 2.4).

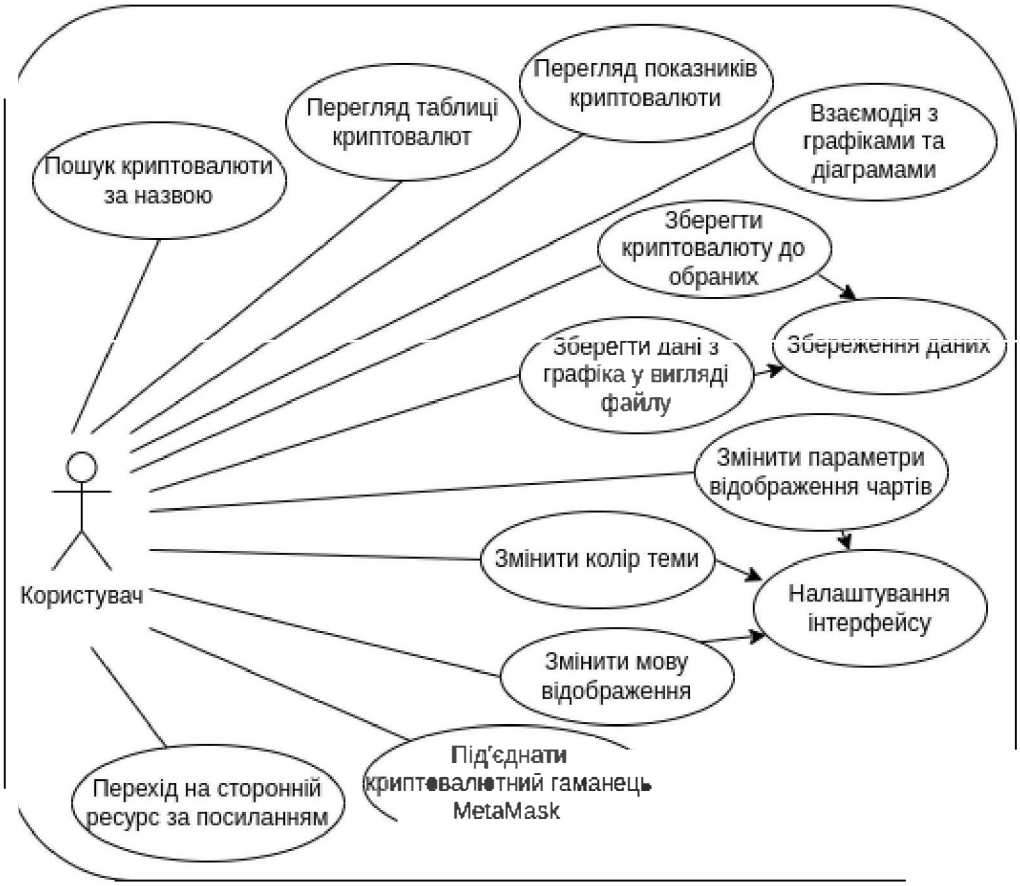

Рисунок 2.4 – Діаграма прецедентів аналізатора криптовалют

Висновки до другого розділу

Отже, для пришвидшення розробки, в програмний продукт доцільно інтегрувати сторонні програмні модулі, сервіси та готові візуальні компоненти. Таким чином фінальний програмний продукт матиме весь перелічений у вимогах набір функцій відповідно до поставлених завдань. Маючи достатній ліміт запитів до сторонніх API, достатньо буде налаштовано дворівневий механізм кешування для сервера та клієнта.

РОЗДІЛ 3
РОЗРОБКА АНАЛІЗАТОРА КРИПТОВАЛЮТ

3.1 Розробка блок-схем для аналізатора криптовалют

Отже, протокол передачі гіпертексту (HTTP) призначений для забезпечення зв’язку між клієнтами та серверами. Протокол передбачає обмін інформацією шляхом відправки запитів від клієнта та поверненням відповіді від сервера. Наприклад: клієнт (браузер) надсилає HTTP-запит на сервер, потім сервер повертає відповідь клієнту. Відповідь містить інформацію про статус запиту, а також може містити запитуваний вміст. Розглянемо та порівняємо життєві цикли HTTP-запитів на REST та GraphQL сервер.

REST сервер отримує від клієнта HTTP запит, з певним методом (GET, POST, PUT, DELETE) та URL-шлях. Після чого назва методу та шлях зіставляє з функцією, зареєстрованою для обробки цього шляху. Функція виконується один раз та повертає результат, який серіалізує, додає статус код відповіді та заголовки та надсилає його назад клієнту.

Коли GraphQL сервер отримує запит від клієнта (Apollo Client), він його перевіряє і якщо запит синтаксично правильний для кожного розпізнаного поля викликається відповідний обробник типу query (аналог GET) або mutation (аналог POST, PUT і т.п.). Далі викликається функція прив'язана до обробника і повертає результат. Після чого бібліотека Apollo Server приєднує цей результат до відповіді сервера.

Отже, клієнт-серверна архітектура передбачає, що для коректної роботи системи, сервер має реагувати на запити клієнтів. Для того, щоб визначити, узагальнену схему взаємодії користувача з системою та як вона реагуватиме на його дії, спроєктуємо діаграму послідовності аналізатора криптовалют з підтримкою специфікації GraphQL (див. рис. 3.1). Для наочності, на діаграмі зображено приклад з вмістом запитів, які обробляються GraphQL сервером.

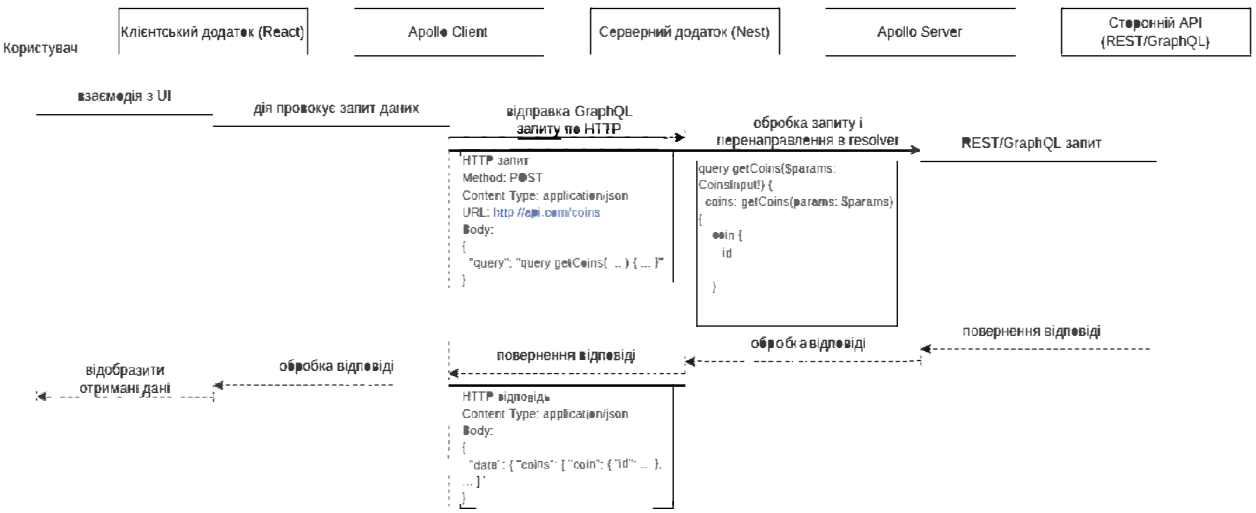

Рисунок 3.1 – Загальна діаграма послідовності системи

Отже, коли потрібно зробити POST запит на сервер, він відправляється клієнтом у вигляді рядка. Сервер обробляє рядок і намагається перетворити його в JSON об'єкт з даними про запит. Відповідно до синтаксису GraphQL та описаної розробниками схеми даних, сервер обробляє та перевіряє запит. Далі сервер знаходить функції обробники (resolvers) за назвами з запиту і викликає їх виконання. Після чого, сервер формує та повертає клієнту відповідь.

Оскільки при агрегації інформації зі сторонніх джерел єдиним лімітом є кількість запитів за певний проміжок часу, зберігання всієї інформації у власну БД не є обов'язково. Для забезпечення мінімальних умов функціонування системи, достатньо буде налаштувати кешування запитів. Це допоможе пришвидшити повернення відповідей на запити клієнтів та дозволить уникнути проблем з лімітами кількості запитів на сторонні API. Більша частина інформації про криптовалюти є незмінною і може виключно доповнюватись новими показниками через певний час.

Слід зауважити, що для забезпечення часткової роботи web-додатка без інтернет з’єднання та якісного кешування вмісту, компанія Google рекомендує розробляти прогресивні застосунки. Нижче наведено діаграму послідовності виконання GraphQL запиту з кешуванням (див. рис. 3.2).

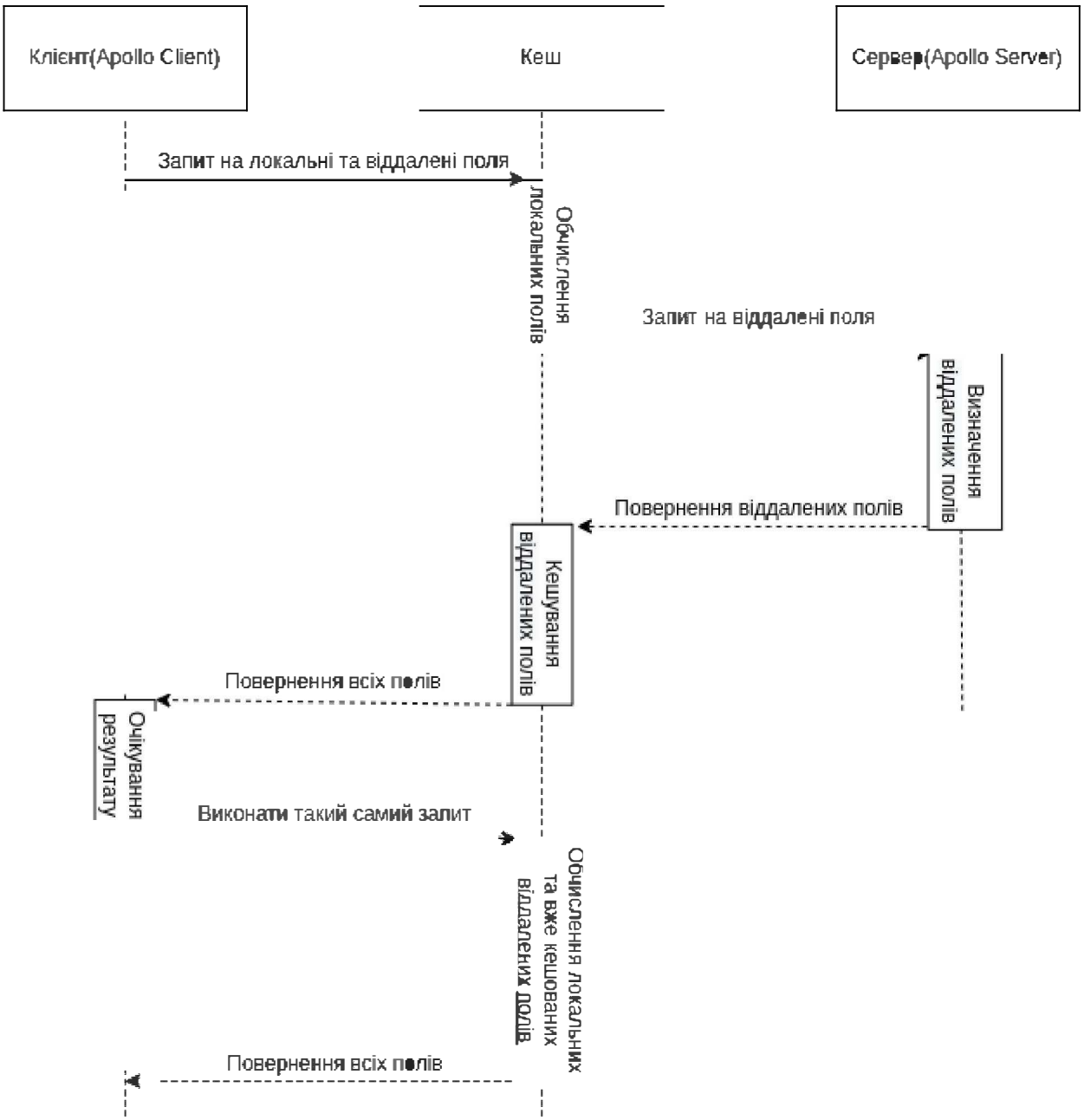

Рисунок 3.2 – Діаграма послідовності GraphQL запиту з кешуванням

Враховуючи що перелік криптовалют та детальна інформація про них береться з «CoinGecko» та «CryptoCompare» API, в першу чергу варто описати використовувані типи даних та за потреби створити на їх основі ієрархію класів. Завдяки зручній інтеграції NestJS з Apollo Server, можна за допомогою використання метаданих (декораторів) описати яким GraphQL типам та скалярам відповідають атрибути класів. Після опису всіх використовуваних типів, отримаємо таку діаграму класів (див. рис. 3.3).

Рисунок 3.3 – Діаграма класів з модуля coingecko

Перевагою «CoinGecko» є широкий набір даних про понад 13 тисяч криптоактивів та понад 600 бірж, що є значною базою знань. Всі дані, які отримуються з різних джерел, проходять через алгоритми валідації, які перевіряють цілісність даних. Для прикладу розглянемо як визначаються деякі основні метрики криптовалют.

Ціна монети розраховується на основі торгових пар, зібраних «CoinGecko» з різних обмінників [26]. Значення ціни розраховується на основі даних з живої торгової сесії. Для цього береться звичайна та кумулятивна вартість активу, помножена на об’єм і ділиться на кумулятивний обсяг торгів. Отже, середньозважена ціна – це загальна середня ціна щодо обсягів активу.

Середньозважена ціна для кожної криптовалюти обраховується за формулою Volume-Weighted Average Price (VWAP), яку наведено нижче (формула 3.1):

$$VWAP = \frac{C_P \times (P \times V)}{C_V}, \quad (3.1)$$

де P – ціна активу;

V – обсяг торгів;

C_P – кумулятивна ціна активу;

C_V – кумулятивний обсяг торгів.

Отримане значення допомагає інвесторам вирішувати, коли увійти або вийти з ринку, тобто угод, протягом дня. Для прикладу розглянемо формулу визначення глобальної середньозваженої ціни за обсягами торгів двох обмінників A та B, валютами долар (USD), єна (JPY) і Bitcoin (BTC). Припустиму, що обмінник A в парі BTC до USD ≈ 1000 доларів і має обсяг торгів 15 000 BTC за добу. Біржа B в парі BTC до JPY ≈ 109 000 єн з обсягом торгів 10 000 BTC за добу. «CoinGecko» спочатку конвертує JPY в USD за курсами валют [26]. При умові що 1 USD = 110 JPY, виходить що 109 000 JPY конвертується в ≈ 990 USD. Отже для цього прикладу формулу середньої вартості BTC матиме наступний вигляд (формула 3.2):

$$VWAP_{BTC} = \left(\frac{V_A}{V_B + V_A}\right) \times P + \left(\frac{V_B}{V_A + V_B}\right) \times CP, \quad (3.2)$$

де V_A, V_B – об’єми торгів біржі A та B;

P – ціна активу в USD ;

CP – ціна конвертована з JPY в USD.

Обсяг торгів криптоактиву обраховується як сума обсягів торгів усіх торгових пар цього активу. Торговий обсяг біржі це сума обсягів всіх торгових пар доступних на біржі.

Слід зауважити, що відслідковування активності розробників допомагає визначити технічно слабкі та сильні проекти. Монети, які більше не

підтримуються розробниками, як правило поступово з часом або раптово втрачають актуальність та свою цінність. Дивлячись на те, скільки зусиль докладає спільнота розробників, є шанс, що монета продовжуватиме розвиватися відповідно до того, які тенденції з'являються на ринку криптовалют. Активність та темпи розробки інфраструктури довкола токенів вимірюється за показниками з публічних репозиторіїв вихідного коду Github.

Для визначення актуальності криптовалют можна систематично вимірювати активність спільноти користувачів, з статистичних показників найбільших соціальних мереж, таких як Facebook, Twitter, Reddit та Telegram. Якщо криптоактив не має спільноти або не обговорюється, це може свідчити про штучність його ціноутворення та відсутність реального попиту. В протилежному випадку постійні обговорення унікальними користувачами є підставою вважати актив релевантним. Як правило активна спільнота в поєднанні з командою розробників ентузіастів привносить в індустрію нові ідеї та функціонал до криптоактивів, що підвищує цінність технології та активу.

3.2 Захист розробленого аналізатора криптовалют

Оскільки вся вразлива інформація зберігається та отримується від сервера, переважна частина алгоритмів та практик захисту сконцентрована на серверній частині. Тому, для організації базового захисту web-додатків існує ряд загальних практик.

Важливо використовувати виключно перевірені та стабільні пакети і модулі. Варто регулярно проводити аудити встановлених залежностей та оновлювати їх до останньої стабільної версії. Крім того, не можна додавати чутливу інформацію, таку як приватні ключі доступу до сторонніх API в репозиторій з програмою. Критично важливим питанням є забезпечення надійної обробки та збереження конфіденційної інформації користувачів у захищене місце.

Більшість користувачів не розуміють яким чином клієнтський додаток надсилає запити на сервер. Відповідно, HTTP-заголовки які відправляються в запитах є вразливим місцем. Зловмисники можуть отримати з заголовків інформацію про застосунок, тому важливо налаштувати їх захист і використовувати їх безпечним чином.

Для захисту поширених слабких місць серверних Node.js додатків існує пакет під назвою Helmet. Він об'єднує набір з 12 спеціалізованих модулів, які взаємодіють з Express. Кожен модуль надає опції конфігурації для захисту вразливих HTTP заголовків [27]. Коротко розглянемо перелік цих заголовків та їх призначення:

- Cache-Control – містить директиви (інструкції) в запитах та відповідях, які контролюють час та інші опції кешування;

  Content-Security-Policy (CSP) – директиви з дозволеними web джерелами та захист від скрипт атак між сайтами або Cross-Site Scripting (XSS);
- Expect-CT – дозволяє сайтам увімкнути звітність та забезпечити виконання вимог щодо прозорості SSL і подібних сертифікатів;
- Public-Key-Pins – використовується для пов’язування криптографічного відкритого ключа з певним web-сервером, щоб зменшити ризик Man in the middle (MITM) атак за допомогою підроблених сертифікатів;
- Referrer-Policy – контролює, скільки інформації з заголовка Referer має бути включено в запити;

  Strict-Transport-Security (HSTS) – забезпечує доступ до сайту лише за допомогою HTTPS, а спроби доступу по HTTP автоматично конвертуються в HTTPS;
- X-Content-Type-Options – дозволяє уникнути перебору типів сторінки (Multipurpose Internet Mail Extensions, MIME);

- X-DNS-Prefetch-Control – керує попереднім завантаженням функції Domain Name System (DNS), за допомогою якої браузери виконують розділення доменних імен на декілька посилань;
- X-Download-Options – директиви завантаження файлів;
- X-Frame-Options – вказує чи слід дозволити браузеру вбудовувати сторінку. Це використовується для захисту від click-jacking атак;
- X-Powered-By – вказує назву сервера;
- X-XSS-Protection – використовується для захисту від XSS атак.

Популярним способом витоку інформації в Express додатках є заголовок X-Powered-By. За замовчуванням Express сервер присвоює цьому заголовку значення «Express», що дає зловмисникам підказку. Хакери зазвичай зіставляють цю інформацію зі списком публічно розкритих відомих вразливостей, що робить вашу програму мішенню, особливо якщо на сервері встановлена версія Express без останніх патчів (оновлень) захисту.

З цього витікає, що потрібно використовувати Long-term support (LTS) версії програмного забезпечення, які мають довгий період підтримки. Крім цього, потрібно постійно проводити аудити всіх сторонніх пакетів та їх залежності на наявність вразливостей.

Слід зауважити, що частину HTTP заголовків наведених вище Helmet.js по замовчанню розпізнає як безпечні. Однак, такий заголовок як CSP, вимагає від розробника чіткої конфігурації. Налаштування потрібне тому, що дотримання всіх практик безпеки може порушити функціональність або погіршити досвід користувача, тому необхідні конфігурації можна вмикати лише за потреби.

Для коректної роботи аналізатора в заголовок CSP було додано такі директиви: frame-src (вбудовані вікна), script-src (скрипти), img-src (зображення), style-src (стилі) та connect-src (підключення). В ці директиви було додано URL посилання на всі сторонні сервіси інтегровані на сторінки аналізатора.

Для захисту від ботів, котрі навмисно створюють трафік та призводять до перенавантаження мережі, слід встановити обмеження швидкості (rate limit) або кількості спроб авторизації. Ця стратегія захисту полягає у встановленні обмеження на кількість повторних запитів з однієї IP адреси протягом певного проміжку часу. Наприклад, при спробі Brute-force атаки, яка передбачає прямий перебір варіантів пароля, зловмисник масово надсилає запити на автентифікацію щоб увійти в обліковий запис.

Висновки до третього розділу

У цьому розділі досліджено механізм обміну запитами між GraphQL клієнтом та сервером і побудовано на його основі діаграму послідовності. Крім цього було розглянуто алгоритм обробки запиту з увімкненим кешуванням. При проєктуванні системи типів було сформовано діаграму класів одного з модулів серверного додатка.

В ході розробки було використано мову програмування TypeScript в середовищі виконання Node.js та специфікацію GraphQL для реалізації взаємодії між серверною та клієнтською частиною. Для обробки GraphQL запитів використано Apollo Server та Apollo Client відповідно.

В основу імплементації серверної частини покладено модульний фреймворк NestJS. Бібліотека axios відповідає за виконання запитів до сторонніх REST API. Клієнтський застосунок розроблено з використанням бібліотек побудови користувацького інтерфейсу React, таблиць React Table, чартів Apexcharts та готових компонентів Chakra UI.

Для забезпечення високого рівня захисту розробленої системи, було досліджено поширені вразливості протоколу передачі даних HTTP, середовища виконання Node.js та використаних пакетів, таких як Express. Для першочергового захисту серверного застосунка від перевантаження, встановлено ліміт на кількість запитів. Крім того, було налаштовано HTTP заголовки за допомогою бібліотеки Helmet, що підняло рівень захищеності програмного продукту.

РОЗДІЛ 4
ВПРОВАДЖЕННЯ АНАЛІЗАТОРА В ЕКСПЛУАТАЦІЮ

4.1 Інтегрування апаратного гаманця в аналізатор криптовалют

На сьогодні зберігання криптовалют на апаратних гаманцях є найбезпечнішим способом. Апаратний гаманець – це фізичний пристрій, спеціально призначений для зберігання закритих ключів шифрування. Підтвердження транзакцій з криптоактивами виходить за межі предметної області аналізу криптовалют, однак подібний функціонал буде зручним доповненням при подальшому розвитку програмного продукту.

Крипто транзакції відбуваються в мережі blockchain з використанням відкритих і закритих ключів. Ці ключі є еквівалентом номера банківського рахунку та паролем для входу в web-банкінг відповідно. Публічний ключ призначений для того, щоб інші користувачі могли відправити кошти на ваш рахунок, тобто адресу гаманця. Користувач має доступ до активів на гаманці за умови якщо закритий ключ не було втрачено і інші користувачі не отримали до нього доступ.

На відміну від програмних гаманців, які можуть бути зламані через неуважність користувача або після вірусної атаки, апаратний гаманець використовується лише тоді, коли ви хочете здійснити транзакцію. Тобто основну частину часу гаманець перебуватиме в автономному режим, що мінімізує ризики його злому. Розглянемо ряд переваг апаратних гаманців:

- висока мобільність (через малі габарити пристрою);
- закриті ключі вводяться лише в гаманець не виходячи за його межі;
- несприйнятливість до комп'ютерних вірусів;
- верифікація дій на апаратному пристрої.

Найнадійнішими апаратними гаманцями вважаються моделі від брендів «Ledger», «KeepKey» та «TREZOR». Серед представлених виробників

найбільш сучасне технічне рішення було розроблене компанією Ledger [28]. Розглянемо ряд переваг гаманця Ledger Nano X.

Значним покращенням програмного захисту, стало використання нової операційної системи безпеки BOLOS. Гаманець оснащений енергоефективним підключенням Bluetooth Low Energy (BLE), що дозволяє використовувати його з пристроями Android або iOS без необхідності підключення кабелю. Хоча ця функція значно покращує досвід експлуатації, існують сумніви щодо безпеки бездротового з'єднання [28].

Через Bluetooth передаються лише загальнодоступні дані. Критичні дані (наприклад, закриті ключі та початкові дані) ніколи не виходять за межі пам'яті пристрою. Навіть якщо з'єднання Bluetooth буде зламано, система безпеки моделі Nano X покладається на захищений елемент (Secure Element, SE), який запитує вашу згоду на будь-які дії.

Реалізація Ledger Nano X використовує найсучасніший протокол Bluetooth. Цей протокол забезпечує автентифікацію за допомогою створення пари на основі числового порівняння, а конфіденційність забезпечується за допомогою шифрування на основі AES. Крім цього є можливість вимкнути Bluetooth і використовувати кабель USB типу C. Розглянемо спрощену схему можливої інтеграції апаратного гаманця Ledger Nano X в розроблений аналізатор криптовалют (рис. 4.1).

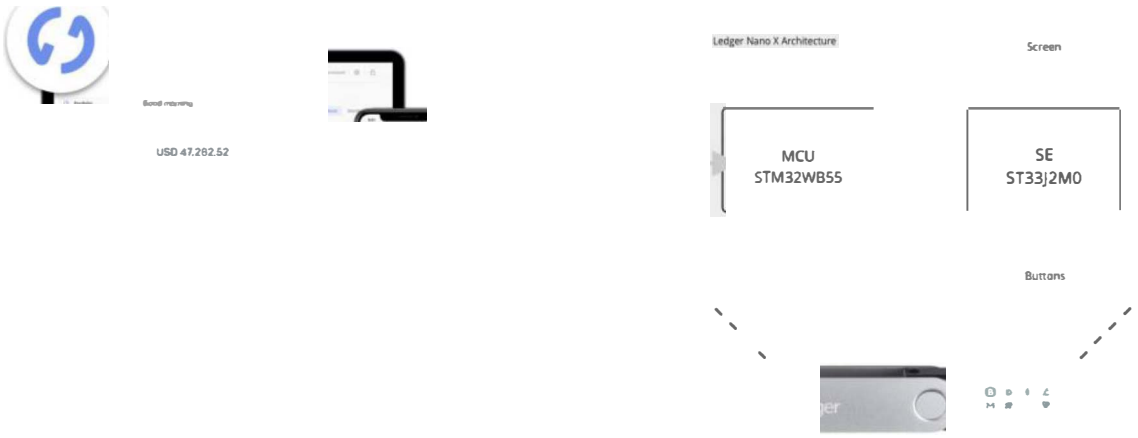

Рисунок 4.1 – Схеми взаємодії гаманця Ledger Nano X з додатком

Гаманець Nano X складається з двох мікроконтролерів (MCU) – ST33J2M0 з найсучаснішим безпековим елементом (Secure Element або SE) та двоядерний мікроконтролер з підтримкою бездротового зв’язку STM32WB55. MCU забезпечують з'єднання між смартфоном або комп'ютером та SE і відіграють роль проксі-сервера. SE відповідає за роботу невеликого екрана розміщеного в корпусі та отримання з нього вхідних даних користувача. Він зберігає початкові та приватні ключі та реалізує всі криптографічні операції для захищених дій з криптовалютами [28]. Коротко розглянемо кожен з мікроконтролерів використаних в апаратній частині цього гаманця.

ST33Jxxx – це мікроконтролер послідовного доступу, розроблений для захищених мобільних додатків. Він містить процесори ARM останнього покоління для вбудованих захищених систем. Його 32-розрядне ядро RISC SecurCore побудоване на ядрі Cortex-M3 з додатковими функціями безпеки, які допомагають захистити від передових форм атак. ST33Jxxx забезпечує високу продуктивність завдяки швидкому процесору SC300, криптографічним прискорювачам і покращеним операціям з флеш-пам'яттю [29].

Багатопротокольний бездротовий пристрій STM32WB55xx з надмалим енергоспоживанням оснащений потужним радіозв'язком, сумісним зі специфікацією Bluetooth Low Energy SIG 5.2 та IEEE 802.11-4. Цей MCU містить найбільш енергоефективний процесор розроблений на цю мить компанією Arm – Cortex-M0+, здатний виконувати операції нижнього рівня в реальному часі. Пристрій заснований на високопродуктивному 32-розрядному RISC-ядрі Arm Cortex-M4, що працює на частоті до 64 МГц. Це ядро має модуль одинарної точності з плаваючою комою (FPU), який підтримує всі інструкції та типи даних одинарної точності. Він також реалізує повний набір інструкцій DSP та блок захисту пам'яті (MPU), що підвищує безпеку [30].

Отже, щоб отримати інформацію про активи на апаратному гаманці користувача та його публічні ключі, достатньо інтегрувати в аналізатор інтерфейс з'єднання з програмним гаманцем «MetaMask» [31], який підтримує всі вище перелічені апаратні гаманці. Отримана інформація може

використовуватись для автентифікації користувачів в систему та створення криптографічних підписів.

4.2 Тестування аналізатора криптовалют

Для базового тестування програмного продукту, слід скласти контрольний список (checklist), тобто перелік вимог та функцій, які мають задовольняти певні стандарти. Пункти з цього списку будуть основою для тестування додатка. Отже, сформуємо перелік критичних функцій для ручного тестування:

можливість локально встановлення на пристрій у якості прогресивного застосунку (progressive web app, PWA) для використання його без інтернет з'єднання;

присутність сторінок «Монети», «Діаграми», «Тренди», «Досліджуй», «DeFi» та «Обране»;

робоче меню для навігації між сторінками;

коректне відображення вмісту кожної з вище наведених сторінок на різних платформах;

коректна обробка дій користувача (натискання кнопок миші, клавіатурних клавіш, наведення);

- з'єднання додатка з гаманцем «MetaMask» (отримання даних про адресу і баланс);
- зміна теми та мови інтерфейсу;

пошук криптовалют за назвою;

- збереження та видалення монет до списку обраних;
- обробка мережевих та користувацьких помилок.

Паралельно з ручним тестуванням розглянемо основні елементи користувацького інтерфейсу. Навігація між сторінками розробленого додатка відбувається за допомогою меню навігації по вертикальному списку посилань. Меню навігації фіксовано розташоване у лівій частині екрану, а при малому розмірі екрану воно ховається і відображається лише при натисканні на кнопку

меню. У лівому верхньому куті знаходиться логотип програмного продукту, його назва та меню навігації. У лівій нижній частині знаходиться перемикач мови та копірайт виконавця роботи. Верхня права частина екрану містить кнопки підключення до гаманця «MetaMask» та зміни кольору теми, а також рядок для пошуку монет за назвою.

При першому відкритті браузер пропонує встановити додаток у якості PWA. Виконаємо інсталяцію та перевіримо чи правильно відображається головна сторінка з монетами (рис. 4.2).

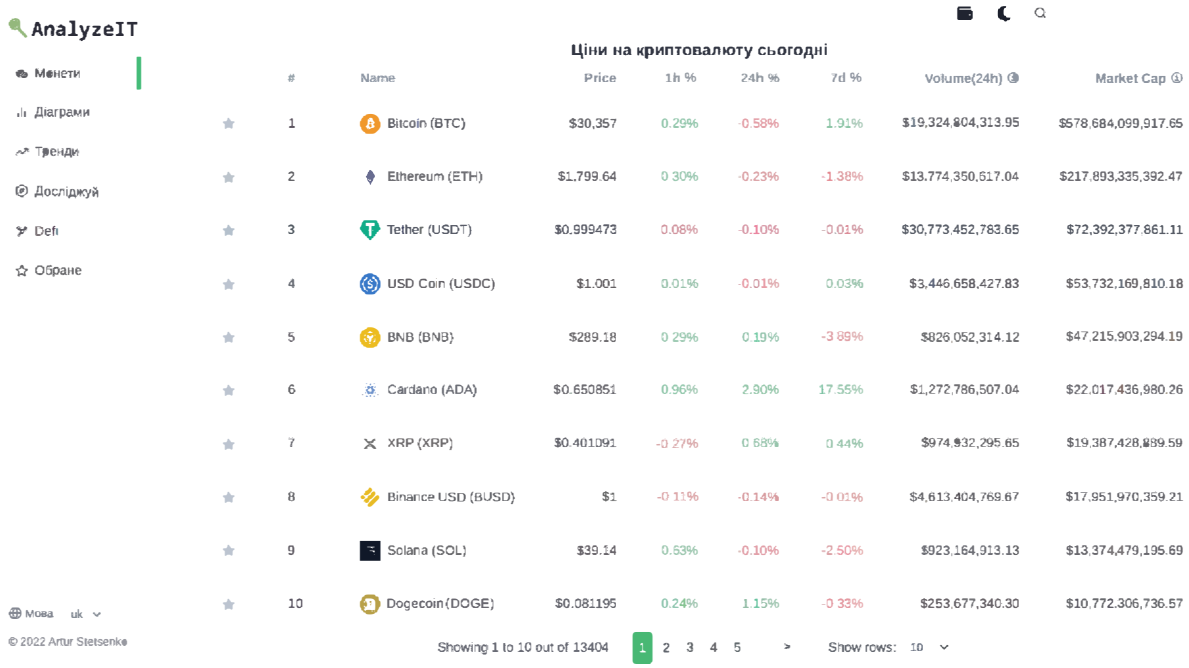

Рисунок 4.2 – Головна сторінка з монетами

Як можна бачити додаток відображається у окремому вікні та може відкриватись без підключення до інтернету. Отже функція встановлення працює і перед користувачем з'являється інтерактивна таблиця, колонки якої можна сортувати за зростанням або спаданням.

У нижній частині можна змінити кількість рядків відображених на одній сторінці, або перейти на іншу сторінку. Натиснувши на область з назвою криптовалюти ми переходимо на сторінку деталей. Разом з перевіркою вмісту, змінимо мову з української на англійську. Інформація про монету подається у

вигляді декількох блоків, які адаптуються до розміру екрана користувача і відображаються за умови наявності відповідних показників (рис. 4.3).

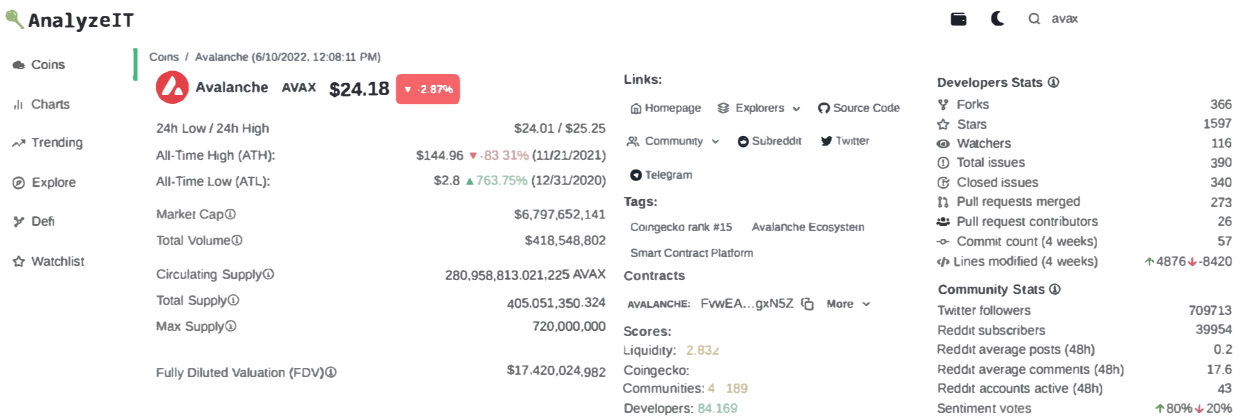

Рисунок 4.3 – Верхня частина сторінки деталей монети

В першому блоці наведені фінансові показники такі як ціна, ринкова капіталізація, торгові об'єми і подібне. Другий блок містить посилання на зовнішні джерела пов'язані з проєктом, дослідники blockchain мережі, репозиторії з відкритим кодом та соціальні мережі. Крім того, зібрані ключові слова пов'язані з активом та оцінки ліквідності, спільноти його розробників та користувачів. Третій блок містить детальну статистику активності користувачів та розробників.

Перевіримо як працює з'єднання з програмним гаманцем «MetaMask». Для цього потрібно натиснути на іконку гаманця у верхній лівій частині екрану. Якщо браузерне розширення або мобільний додаток для роботи з гаманцем не встановлено, аналізатор запропонує посилання на інсталяцію. Після встановлення, в меню з'явиться опція під'єднання або повторного під'єднання (Re/connect) гаманця «MetaMask». Далі має відкритись нове вікно браузера або мобільний додаток гаманця і запитати дозвіл на з'єднання. Надавши доступ, гаманець має виконати переадресацію назад в аналізатор і в аналізаторі повинна з'явитися інформація про публічну адресу гаманця та його баланс (див. рис. 4.4).

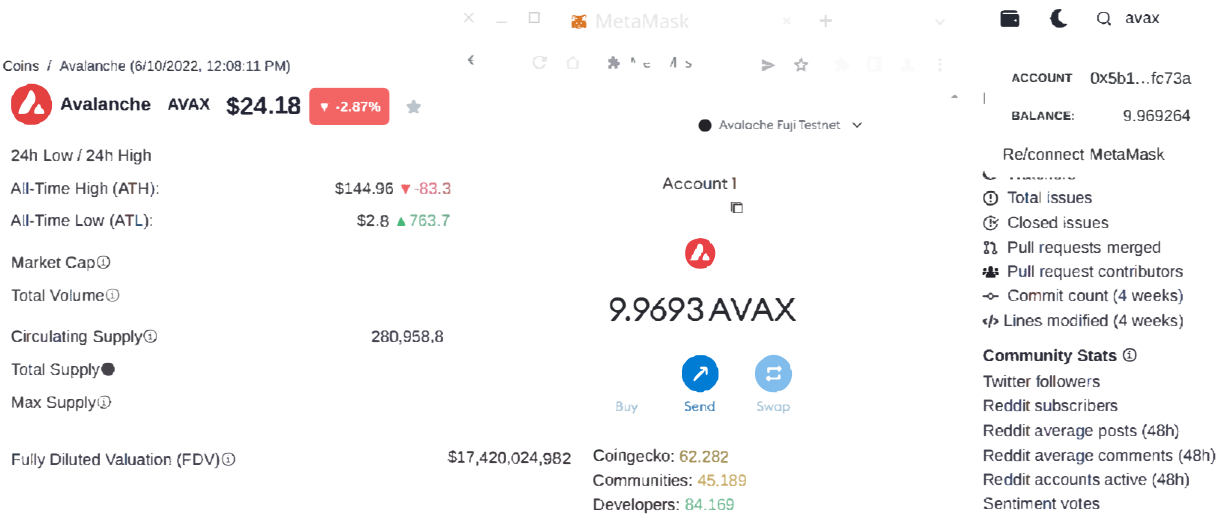

Рисунок 4.4 – З’єднання аналізатора з гаманцем «MetaMask»

Перейдемо до розгляду наступного блоку з інтерактивними графіками. На першій та другій вкладках знаходяться інтерактивні графіки з історією зміни ціни та ринкової капіталізації монети за кожен день протягом обраного періоду (див. рис. 4.5).

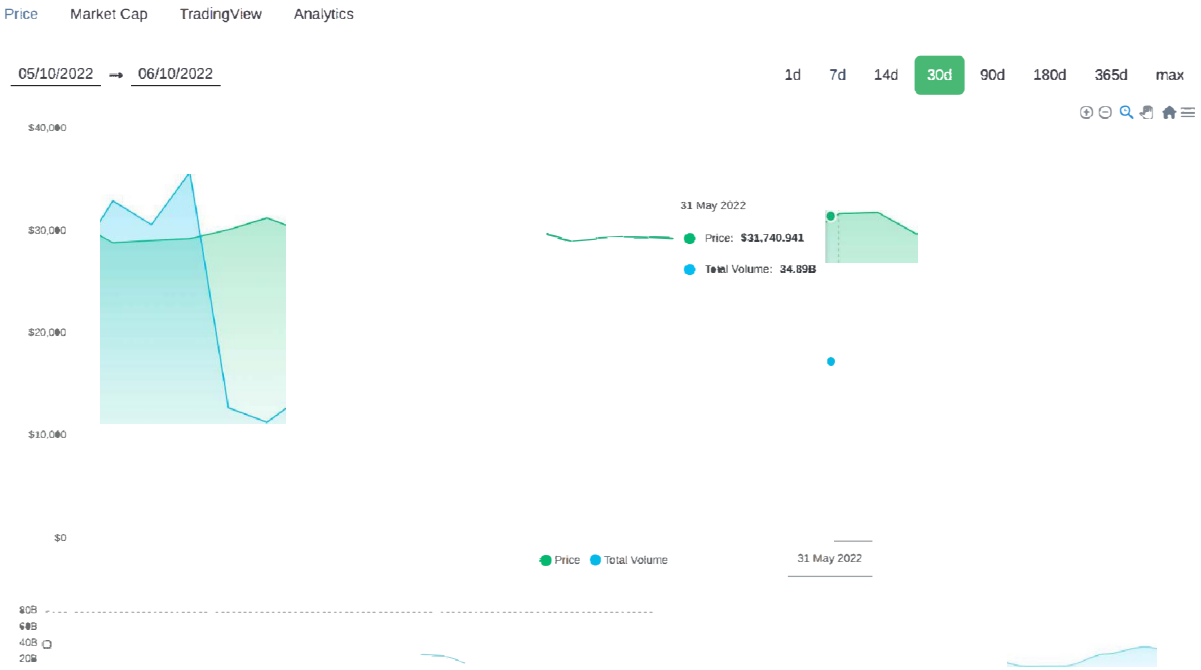

Рисунок 4.5 – Графік історії ціни на сторінці деталей монети

За замовчанням відображається проміжок останніх семи днів на інтервалі в останні 30 днів. Часовий інтервал можна обрати у правій верхній частині графіка, а нижній допоміжний графік показує історію торгових обсягів за день і дозволяє масштабувати проміжок відображення. Оскільки торговий об’єм пов'язаний зі зміною ціни та капіталізації, для зручності він накладається на їх графіки та за потреби може бути схований натиснувши на назву «Total Volume».

Перейдемо на третю вкладку з інтегрованим сервісом побудови фінансових чартів «TradingView». Цей інструмент використовується професійними трейдерами і необхідний для проведення повноцінного технічного аналізу. У лівій частині знаходиться меню з опціями для вільного малювання або побудови геометричних фігур, таких як лінії або криві на графіку. Крім того, можна додавати текст та вимірювати відсоткову зміну обраного інтервалу. У правій верхній частині вказана назва торгової пари, опції для зміни часового інтервалу, способу відображення (японські свічки, лінії або області) та меню пошуку і застосування технічних індикаторів. Спробуємо знайти та додати індикатори «Хмара Ішимоку» та індикатор «МА» для визначення довгострокового тренду за ковзаниими середніми (рис. 4.6).

Рисунок 4.6 – Інтегрований графік сервісу «TradingView»

В нижній частині сторінки також знаходиться її опис та таблиця з інформацією про доступні торгові пари зі сторонніх бірж або обмінників з посиланнями, де можна придбати актив. Оскільки компоненти для відображення тексту та таблиць вже протестовані, одразу перейдемо на сторінку з чартами. На ній інтегровано сервіси побудови діаграм з фінансовими та blockchain показниками «Glassnode» (рис. 4.7).

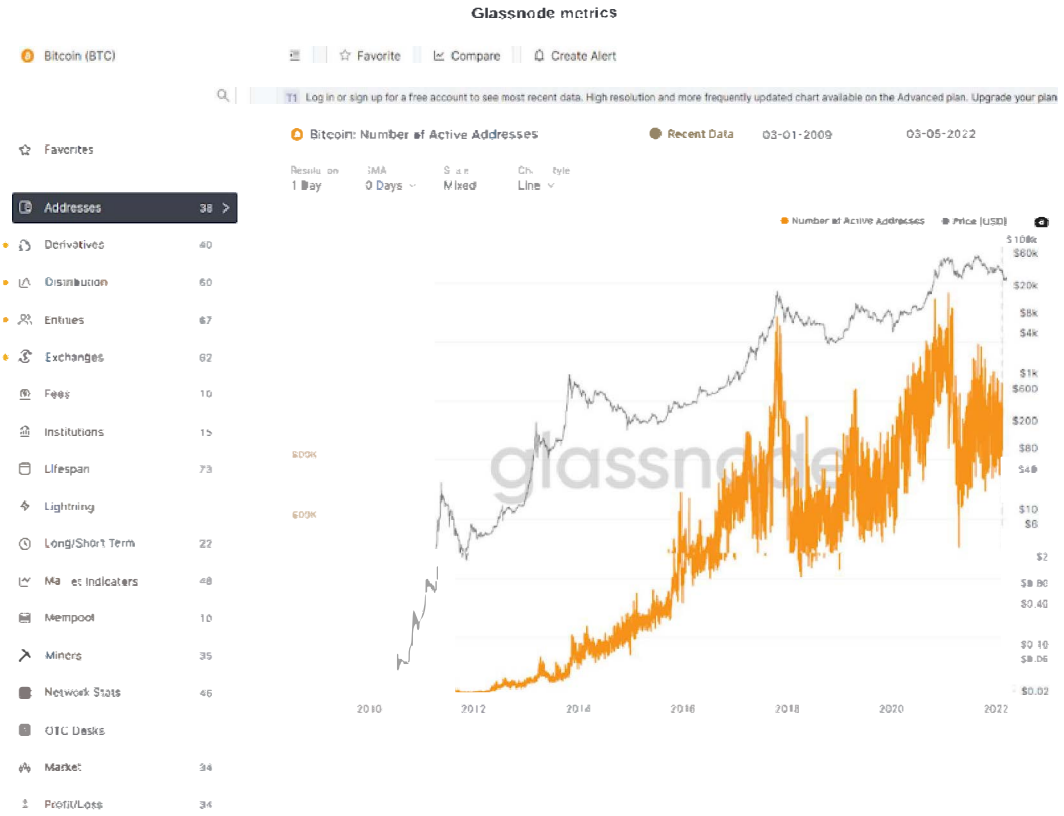

Рисунок 4.7 – Графік інтегрованого сервісу «Glassnode»

На сторінці з трендами розміщені картки з фінансовими показниками такими як: ціна, її відсоткова зміна за добу, ринкова капіталізація та торгові обсяги для семи монет, які частіше всього шукали користувачі на сервісі «CoinGecko» за останні 24 години (див. рис. 4.8). Також, на сторінку інтегровано сервіс APE Wisdom, який кожні 30 хвилин сканує найпопулярніші обговорення (subreddits) з web-сайту Reddit. Ця соціальна платформа дозволяє знаходити трендові тем, кількісно вимірюючи обговорення предметної області.

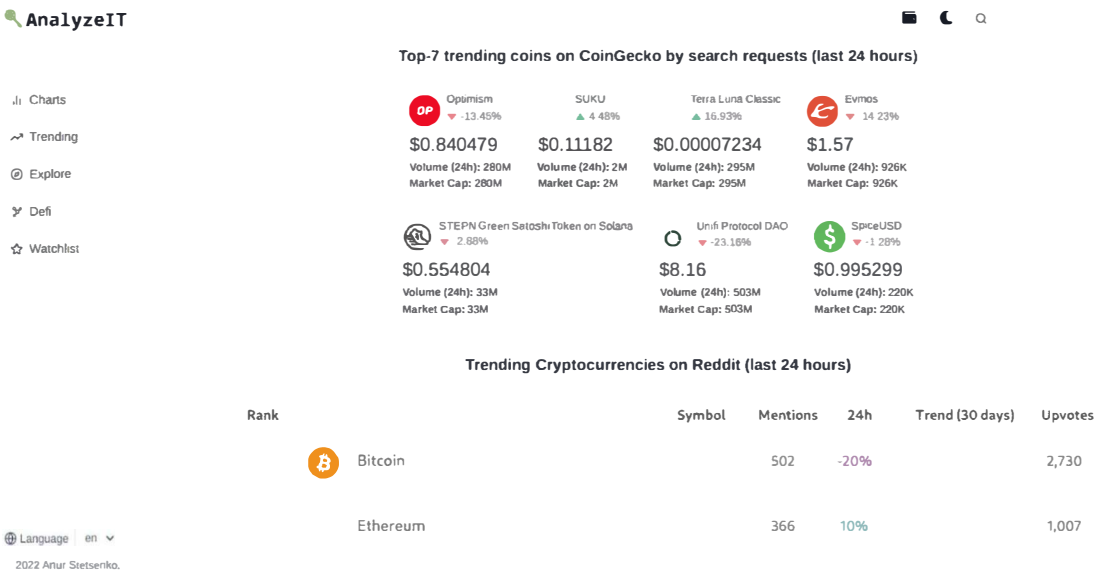

Рисунок 4.8 – Сторінка трендів

На сторінці для досліджень інтегровано три чарти розбитих як окремі вкладки. На першій розміщено діаграму домінації криптовалют з відсотковою зміною ціни за останню добу (рис. 4.9).

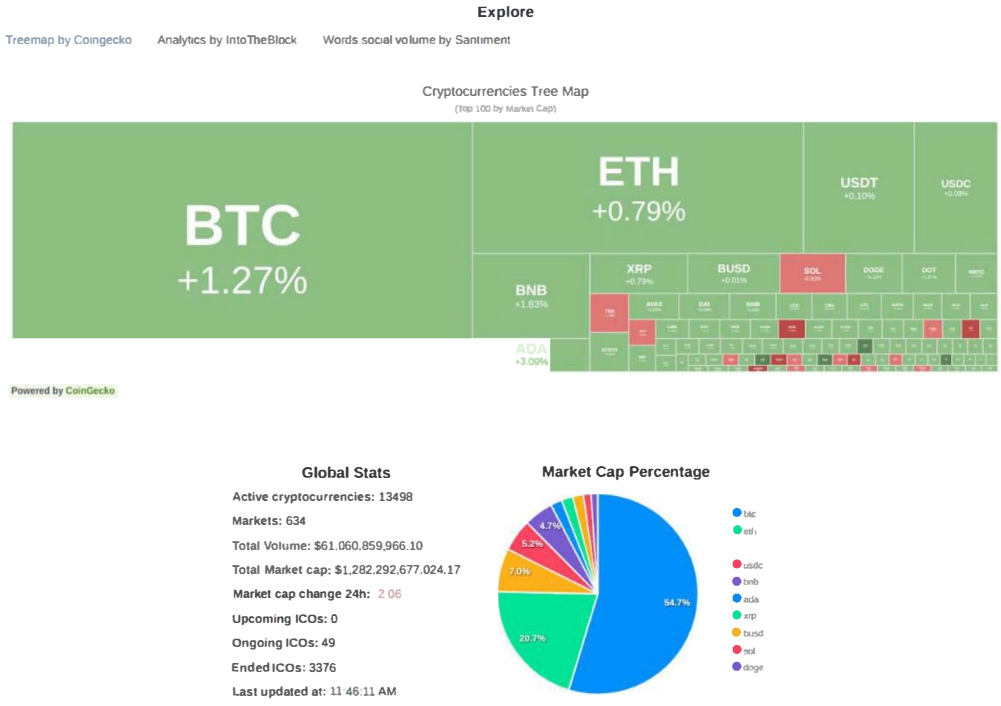

Рисунок 4.9 – Сторінка для досліджень

На другій вкладці знаходиться перелік монет з інтегрованого сервісу «IntoTheBlock» (рис. 4.10). На останній вкладці розміщено список з найбільш вживаними словами від сервісу «Santiment» (рис. 4.11).

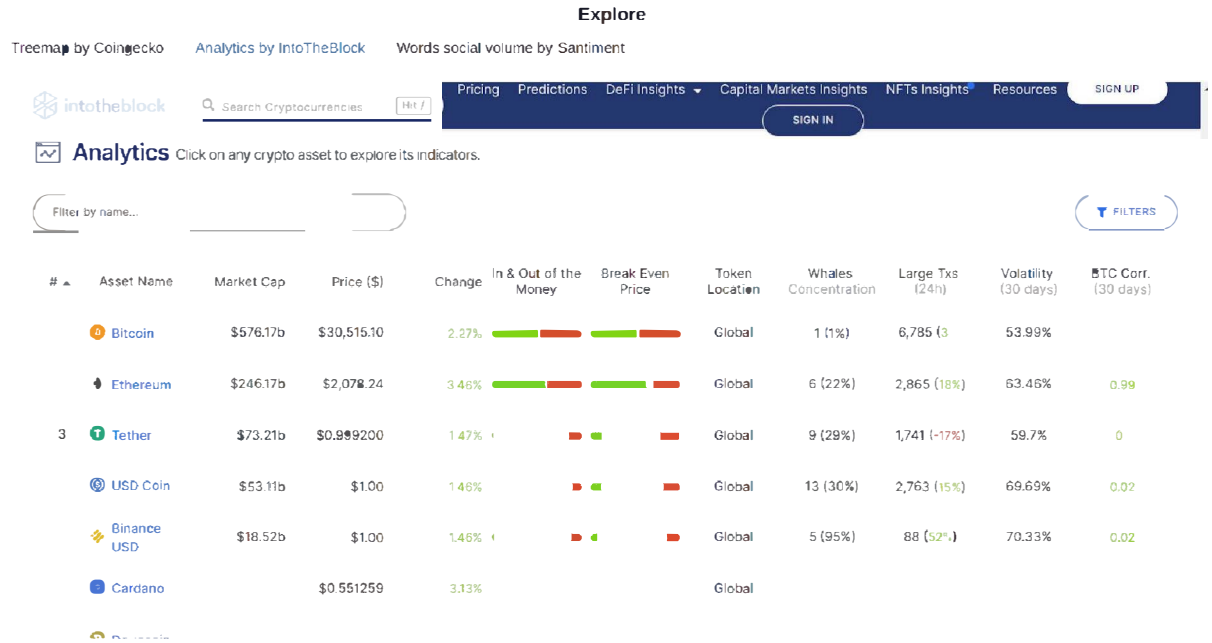

Рисунок 4.10 – Вкладка аналітики від «IntoTheBlock»

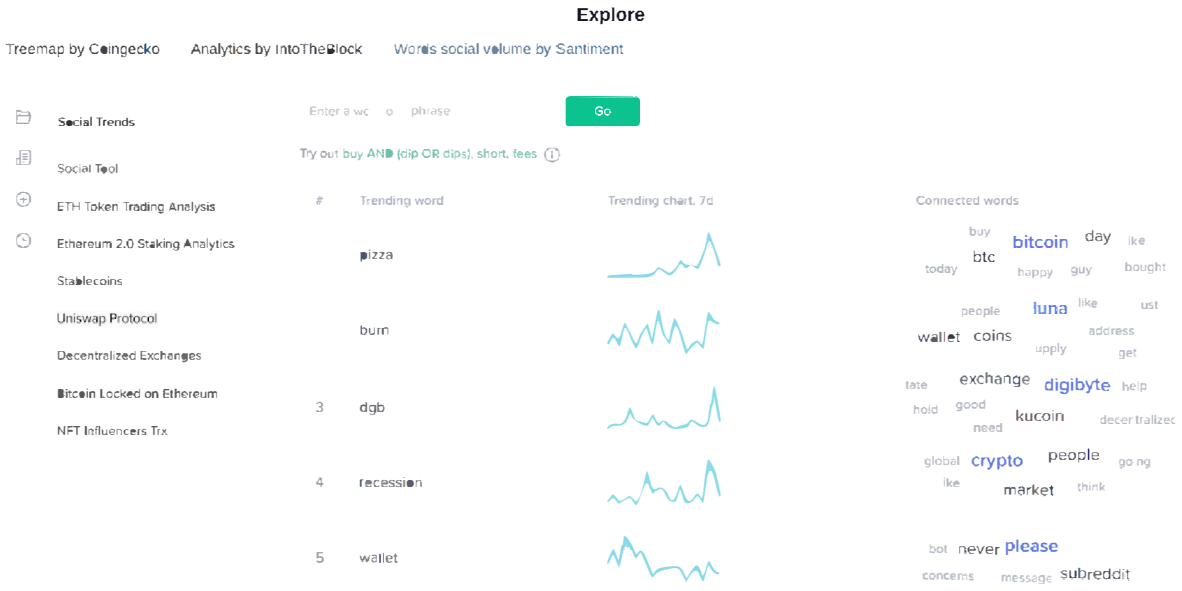

Рисунок 4.11 – Вкладка трендових слів інтегрованого сервісу «Santiment»

На сторінці децентралізованих фінансів (Defi) розміщено графік з загальною вартістю активів заблокованих в децентралізованих мережах та перелік цих мереж взятих з інтегрованого сервісу Defi Llama (рис. 4.12). Для демонстрації сторінка відображається в темному режимі.

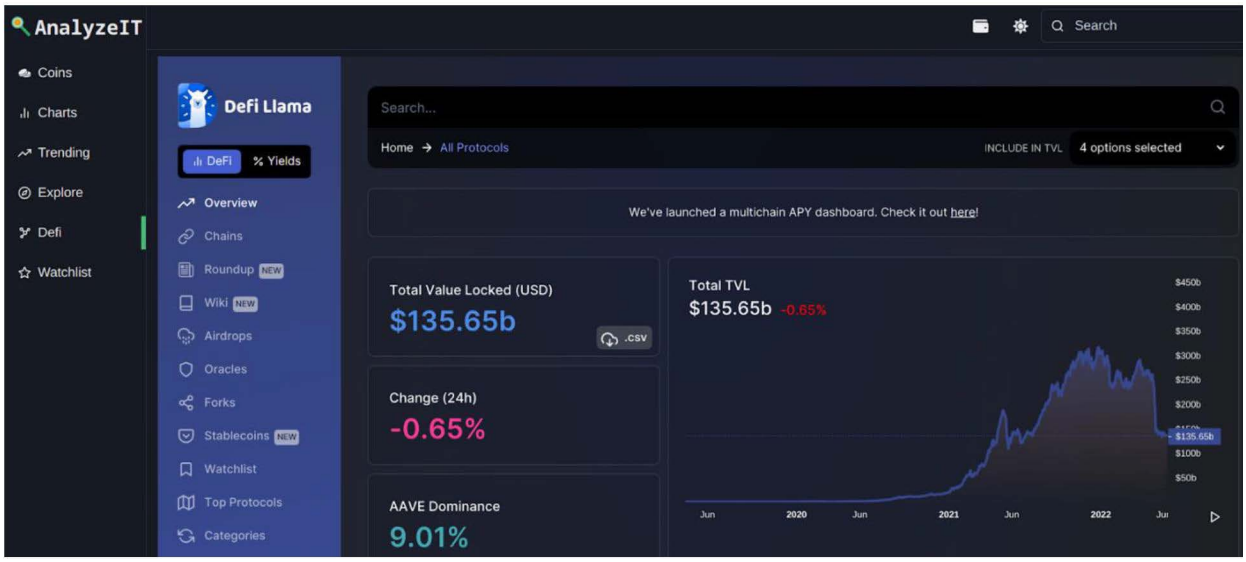

Рисунок 4.12 – Сторінка Defi з інтегрованим сервісом Defi Llama

Сторінка з обраним дозволяє додати та видаляти монети в таблицю, щоб потім швидко їх знаходити (рис. 4.13). Щоб зберегти монету в таблицю можна натиснути на іконку зірки на сторінці з монетами, деталями монети або скористатись пошуком за назвою на сторінці обраних (див. рис. 4.14).

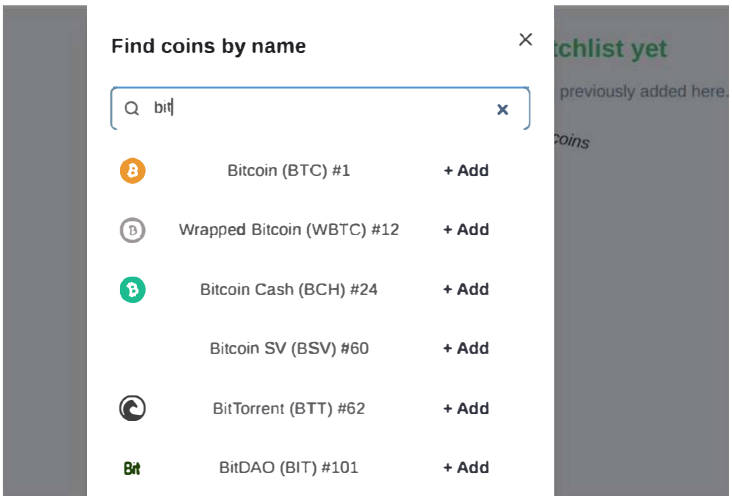

Рисунок 4.13 – Приклад пошуку криптовалюти за назвою

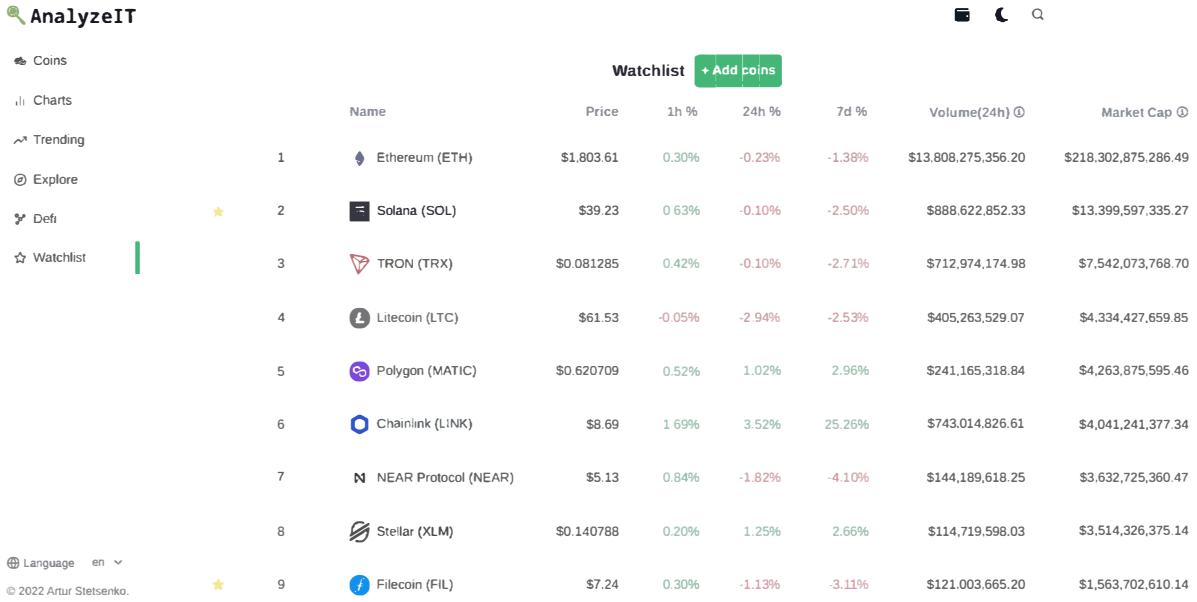

Рисунок 4.14 – Сторінка з обраним

Висновки до четвертого розділу

У цьому розділі було розглянуто схему інтеграції web-додатка з готовим апаратним гаманцем Ledger Nano X. Для взаємодії з апаратними гаманцями в аналізатор було інтегровано інтерфейс підключення до програмного гаманця «MetaMask». Надалі підключення до гаманця може слугувати механізмом автентифікації користувачів для збереження його персональних даних. Крім того, можна відправляти на гаманець повідомлення та запити на верифікацію blockchain транзакцій.

При плануванні алгоритму тестування, було сформовано контрольний список критичних функцій розробленого програмного продукту, які були згодом послідовно перевірені. Кожен з пункт було протестовано мануально, тобто шляхом взаємодії з користувацьким інтерфейсом. Всі наведені тест пройдені успішно, що підтверджує дотримання поставлених технічних та функціональних вимог. Крім того, паралельно з тестуванням було проведено короткий огляд користувацького інтерфейсу та основних можливостей додатка.

ВИСНОВКИ

У ході дослідження предметної області, було встановлено, що головною проблемою при аналізі криптовалют можна вважати великі обсяги неоднорідної інформації. Для спрощення процесу аналізу та розв'язання задач з прийняття фінансових рішень вже існують десятки програмних продуктів. Оглянувши аналоги, було сформовано перелік функціональних та технічних вимог до аналізатора.

Оскільки локальне зберігання та розгортання повноцінного вузла однієї або декількох сторонніх blockchain мереж, для отримання on-chain метрик є дорогим способом аналізу, розроблений додаток фокусується на агрегації даних доступних зі сторонніх джерел та розв'язанні проблеми надлишковості інформації. Було встановлено, що використання специфікації GraphQL дозволяє декларативно запитувати виключно потрібні значення, тим самим пришвидшується обмін неоднорідною та складно структурованою інформацією між сервером і клієнтом.

Практичне значення розробленої системи полягає в агрегації загальної та аналітичної інформації про понад 13 тисяч унікальних токенів зі сторонніх джерел. Для зручності, в розроблену систему було інтегровано спеціалізовані аналітичні сервіси такі як «Glassnode» (on-chain та фінансові метрики), «IntoTheBlock» (аналіз з прогнозування) та «Santiment» (трендові та соціальні метрики). Крім того, для побудови складних фінансових графіків та їх технічного аналізу було додано сервіс «TradingView». Кожен інтегрований інструмент спрощує процес аналізу та прогнозування поведінки криптовалют. Дослідження зібраних метрик допомагає виявити закономірності та тенденції конкретного активу або індустрії в цілому.

Створена система реалізує такий перелік функціональних можливостей:

1. Агрегація загальних, фінансових, blockchain та соціальних метрик для знайдених криптовалют.
2. Навігація між окремими сторінками з аналітичними даними.

3. Побудова інтерактивних таблиць з можливостями сортування, зміни кількості вмісту та пагінації.
4. Пошук монет за назвою з можливістю збереження в обране.
5. Побудова інтерактивних діаграм та графіків з інструментами технічного аналізу фінансових показників.
6. Формування та відображення підказок, аналітичних оцінок та висновків у форматі чисел або повідомлень природною мовою.
7. Інтегровані в додаток сторонні інструменти та сервіси для аналізу криптоактивів та взаємодії з апаратними гаманцями.
8. Адаптивний інтерфейс з підтримкою різних розмірів екрана користувача та зміни кольору теми.

Всі поставлені завдання було виконано повністю, мета роботи досягнута. Розроблений додаток використовує показники з декількох перевірених сторонніх джерел і надалі може бути концептуально та технічно удосконалений.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Методичні вказівки до написання та захисту кваліфікаційних робіт бакалавра спеціальності 123 «Комп'ютерна інженерія» (для здобувачів вищої освіти денної та заочної форм навчання) / Уклад. : Бєсєдіна С. В., Веретельник В. В., Стабецька Т. А. Черкаси : Черкаський національний університет ім. Б. Хмельницького. 2021. 53 с. URL: https://fotius.cdu.edu.ua/wp-content/uploads/2021/12/Методичні_вказівки_Кваліф_робота_123_КІ_БАКАЛАВР_2021_БєсєдінаСВ.pdf
2. Cryptocurrency. URL: https://en.wikipedia.org/wiki/Cryptocurrency (дата звернення: 23.11.2021).
3. Fiat Money. URL: https://en.wikipedia.org/wiki/Fiat_money (дата звернення: 23.11.2021).
4. Jake Frankenfield. Cryptocurrency Definition: What Is Cryptocurrency?. URL: https://www.investopedia.com/terms/c/cryptocurrency.asp (дата звернення: 23.11.2021).
5. Старкова О.В., Міхєєв І.А. Матеріали XIII-ої Міжнародної науково-практичної конференції «Free and Open Source Software», Харків, 16-18 листопада 2021. 64 с.
6. Blockchain. URL: https://en.wikipedia.org/wiki/Blockchain (дата звернення: 23.11.2021).
7. Volatility (finance). URL: https://en.wikipedia.org/wiki/Volatility_(finance) (дата звернення: 14.12.2021).
8. Technical analysis. URL: https://en.wikipedia.org/wiki/Technical_analysis (дата звернення: 14.12.2021).
9. How to Analyze a Cryptocurrency Using Fundamental Analysis. URL: https://learn.bybit.com/investing/how-to-analyze-a-cryptocurrency-using-fundamental-analysis/ (дата звернення: 14.12.2021).

10. Legality of cryptocurrency by country or territory. URL: https://en.wikipedia.org/wiki/Legality_of_cryptocurrency_by_country_or_territory (дата звернення: 15.12.2021).
11. Technical indicator. URL: https://en.wikipedia.org/wiki/Technical_indicator (дата звернення: 15.12.2021).
12. Consensus Mechanism (Cryptocurrency). URL: https://www.investopedia.com/terms/c/consensus-mechanism-cryptocurrency.asp (дата звернення: 15.12.2021).
13. A Guide to Cryptocurrency Fundamental Analysis. URL: https://academy.binance.com/en/articles/a-guide-to-cryptocurrency-fundamental-analysis (дата звернення: 15.12.2021).
14. About CoinMarketCap. URL: https://coinmarketcap.com/uk/about/ (дата звернення: 15.12.2021).
15. About | Blockchain. URL: https://www.blockchain.com/about (дата звернення: 15.12.2021).
16. GraphQL vs. REST: What You Didn't Know. URL: https://www.mobilelive.ca/blog/graphql-vs-rest-what-you-didnt-know (дата звернення: 04.04.2021).
17. Introduction to GraphQL. URL: https://graphql.org/learn/ (дата звернення: 05.04.2022).
18. Schemas and Types | GraphQL. URL: https://graphql.org/learn/schema/ (дата звернення: 05.04.2022).
19. NestJS – A progressive Node.js framework. URL: https://docs.nestjs.com/ (дата звернення: 06.04.2022).
20. React – A JavaScript library for building user interfaces. URL: https://reactjs.org/ (дата звернення: 11.04.2022).
21. Introduction to Apollo Client. URL: https://www.apollographql.com/docs/react/ (дата звернення: 12.04.2022).
22. Chakra UI – A simple, modular and accessible component library. URL: https://chakra-ui.com (дата звернення: 13.04.2022).

23. Getting Started: Overview | React Table | TanStack. URL: https://react-table.tanstack.com/docs/overview (дата звернення: 14.04.2022).
24. ApexCharts.js - Open Source JavaScript Charts for your website. URL: https://apexcharts.com/ (дата звернення: 14.04.2022).
25. GraphQL Playground – Apollo GraphQL Docs. URL: https://www.apollographql.com/docs/apollo-server/v2/testing/graphql-playground (дата звернення: 15.04.2022).
26. Methodology | CoinGecko. URL: https://www.coingecko.com/en/methodology (дата звернення: 19.04.2022).
27. Helmet. URL: https://helmetjs.github.io/ (дата звернення: 19.04.2022).
28. Ledger Nano X & Bluetooth – Security Model Of A Wireless Hardware Wallet. URL: https://www.ledger.com/ledger-nano-x-bluetooth-security-model-of-a-wireless-hardware-wallet (дата звернення: 20.04.2022).
29. ST33J2M0 – 32bit ARM® SecurCore® SC300 with secure integrity architecture, AES, DES, Nescrypt public key co-processors - STMicroelectronics. URL: https://www.st.com/en/secure-mcus/st33j2m0.html (дата звернення: 20.04.2022).
30. STM32WB55RG - Ultra-low-power dual core Arm Cortex-M4 MCU 64 MHz, Cortex-M0+ 32 MHz with 1 Mbyte of Flash memory, Bluetooth LE 5.2, 802.15.4, Zigbee, Thread, USB, LCD, AES-256 - STMicroelectronics. URL: https://www.st.com/en/microcontrollers-microprocessors/stm32 (дата звернення: 20.04.2022).
31. Create A Simple Dapp | MetaMask Docs. URL: https://docs.metamask.io/guide/create-dapp.html (дата звернення: 21.04.2022).