Київський столичний університет імені Бориса Грінченка
Факультет інформаційних технологій та математики
Кафедра комп’ютерних наук

Допущено до захисту
Завідувач кафедри комп’ютерних наук, кандидат технічних наук, доцент,
_____ Ірина МАШКІНА
(підпис)
« ___ » _____ 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня «бакалавр» Спеціальність 122 Комп’ютерні науки Освітня програма 122.00.01 Інформатика

Тема:

Інтерактивна платформа аналізу та моніторингу криптовалютного портфелю з інтеграцією API криптовалютних бірж

Виконав
студент групи ІНб-2-21-4.0д
Сидоренко Ярослав Ігорович
(ПІБ)
_____
(підпис)
Науковий керівник
к.т.н. доцент кафедри комп'ютерних наук
Рзаєва Світлана Леонідівна
_____
(підпис)

Київ – 2025

Київський столичний університет імені Бориса Грінченка

Факультет інформаційних технологій та математики
Кафедра комп'ютерних наук

Допущено до захисту
Завідувач кафедри комп'ютерних наук, кандидат технічних наук, доцент,
_____ Ірина МАШКІНА

(підпис)

« ___ » _____ 2025 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІНУ РОБОТУ БАКАЛАВРА

« Інтерактивна платформа аналізу та моніторингу криптовалютного портфелю з інтеграцією API криптовалютних бірж »

Виконавець – студент групи ІНб -2-21-4.0д
Сидоренко Ярослав Ігорович

1. Вихідні дані: *Вебзастосунок для управління криптовалютним портфелем.*
2. *Дані про криптовалюти отримуються з CoinGecko API.*
3. *Основна функціональність включає додавання, редагування та видалення активів, розрахунок загальної вартості портфеля, збереження даних на клієнтській стороні.*
4. *Реалізація ведеться з використанням React та LocalStorage*.
5. Основні завдання: *Здійснити аналіз сучасних рішень у сфері обліку криптовалютних портфелів. Обґрунтувати вибір технології React для створення вебзастосунку. Розробити структуру, функціонал і інтерфейс системи. Реалізувати основні функції: додавання, редагування та видалення криптоактивів, автоматичне оновлення цін через API CoinGecko, обчислення прибутку, побудову графіка зміни портфеля. Забезпечити збереження даних на клієнтській стороні за допомогою localStorage. Забезпечити зручний та мінімалістичний інтерфейс. Провести тестування працездатності всіх функцій у браузері. Підготувати пояснювальну записку відповідно до вимог кафедри та створити презентацію до дипломної роботи.*
6. Пояснювальна записка: *Обсяг – до 56 стор. формату А4 комп'ютерного набору з дотриманням вимог стандарту і методичних рекомендацій кафедри.*
7. Графічні матеріали: 13 рисунків
8. Додатки: *відсутні.*
9. Строк подання роботи на кафедру: *«_» червня 2025 р.*

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

№	Етапи виконання	Термін виконання	Відмітка
1	Формування теми	10.11.24	Виконано
2	Затвердження теми Бакалаврьскої роботи	29.11.24	Виконано
3	Робота над теоритичною складовою	12.02.25	Виконано
4	Написання тез	23.02.25	Виконано
5	Аналіз програмних засобів для створення проєкту	13.03.25	Виконано
6	Створення вебзастосунку по темі кваліфікаційної роботи	20.05.25	Виконано
7	Передзахист кваліфікаційної роботи	22.05.25	Виконано
8	Захист кваліфікаційної роботи	17.06.25	

Науковий керівник

к.т.н. доцент кафедри комп'ютерних наук
Рзаєва Світлана Леонідівна

_____

Здобувач

Сидоренко Ярослав

_____

РЕФЕРАТ

Кваліфікаційна робота -

с. 30, рис. 13, табл. 1, 12 посилань.

Ключові слова: криптовалюта, портфель інвестора, веб-застосунок, React, CoinGecko API, прибутковість, графік вартості, localStorage.

Метою кваліфікаційної роботи є створення простого та зручного вебзастосунку для перегляду криптовалютного портфеля з використанням даних API CoinGecko. Проведено аналіз популярних рішень (CoinMarketCap, Delta, CoinGecko Portfolio), визначено їх сильні та слабкі сторони, сформульовано вимоги до розроблюваного продукту.

Розробка здійснена з використанням HTML5, CSS3, JavaScript, React, Node.js та Express. Дані про ціни криптовалют та капіталізацію монет автоматично отримуються через API CoinGecko. Архітектура вебзастосунку включає головну сторінку, профіль користувача, таблицю активів із поточними цінами та інтерактивний графік динаміки портфеля.

Серверна логіка побудована на платформі Node.js з використанням фреймворку Express. Система забезпечує збереження базових даних користувача та конфігурації портфеля.

Функціональність вебзастосунку перевірена вручну на різних пристроях і в браузерах. Перевірено коректність відображення інтерфейсу, оновлення фінансових даних, роботу графіків та обробку можливих помилок API.

У результаті отримано прототип вебзастосунку, що дозволяє ефективно моніторити стан криптовалютного портфеля та може бути використаний як основа для створення повноцінної системи управління криптоактивами.

ЗМІСТ

ВСТУП

Актуальність. Зараз у нашому світі криптовалюти стали не просто інструментами для фінансових операцій а і обєктом інвестування. Кожний рік ринок криптовалют зростає і розвиває тим самим інфоструктуру, а саме створюються нові сервіси для аналізу обліку або управлінням своїх активів.

Такий швидкий ріст створює необхідність у комфортності цих всіх додатків щоб вони були зрозумілі для більшості людей які поступово входять у цей ринок. Тому актуальність теми зумовлена тим що зараз дуже зростає інтерес до сфери як з боку інвесторів так і зі сторони бізнесу до криптовалют як до активу. Особливо зараз популярні сервіси які в реальному часі можуть відстежувати курси криптовалют, створювати аналіз або навіть прогнозувати зміни вартості цих активів. При цьому важливо що є ресурси як CoinGecko, який дає можливість вигружати його API і отримувати повноцінний доступ до великого обсягу данних криптовалют.

Мета дослідження полягає в тому щоб розробити вебзастосунок для формування портфелю, введення даних в нього та візуалізації криптовалютного портфелю користувача який планує використовувати мій сервіс за допомогую як я вже казав данних з серверів CoinGecko. Основна ідея це створити простий і візуально зрозумілий інструмент за допомогою якого користувач зможе полегшити собі життя від легкого відсліження данних про свої активи. -

Обєкт дослідження це криптовалюти та інвестування в неї, способи відображення та використання подібних ресурсів.

Предметом дослідження є методи та інструменти створення вебзастосунку для відображення криптовалютного портфеля. У роботі розглядається використання API для отримання даних, побудова логіки

збереження та обробки інформації, а також реалізація інтерфейсу з виведенням графіків фінансових показників.

Задачі дослідження:

1. Проаналізувати основні типи криптовалют, які люди найчастіше використовують для своїх портфелів.
2. Розібратись у роботі сервісу CoinGecko і зрозуміти які саме данні мені потрібні щоб використати їх у своєму сервісі.
3. Реалізувати скрипт який отримує актуальні ціни і переносить їх у свою базу данних для відстеження цінності.
4. Створити базовий функціонал для розрахунку змін активів на прикладі декількох різних портфелів.
5. Провести тестування.

РОЗДІЛ 1. АНАЛІТИЧНА ЧАСТИНА

1.1. Аналіз відкритого ринку

На сьогоднішній день є багато різних підходів до створення криптовалютних портфелів. Одні мінімалізують ризики і беруть великі проекти з високою капіталізацією напряклад як Bitcoin або Ethereum, інші шукають менш відомі активи але з більшим потенціалом росту аніж з першого пункту через не велику популярність та кількість грошей у проектах.

Основні стратегії ж такі самі як і в інших ринках цінних активів, це може бути довгий холдинг, диверсифікації, ребалансування або поділяти свої активи на сектори різних технорлогій в криптовалюті (ШІ, мем-токени, альткоїни тощо.) Усі стратегії мають свій сенс але кожному інвестору треба навчитися підбирати під себе комфортний для нього стиль інвестування, беручи фактори часу, кількості грошей або стилю заробітку на ринку.

1.2. Принципи формування інвест-портфеля

Почнем з того що інвест портфель це набір активів, які людина обирає для певних своїх цілей. У разі криптовалют це пошук нормальних та преспективних монет в які є сенс вкласти свої гроші та розраховувати на прибуток у майбутньому. Якщо ми говоримо про розподіл між монетами, є сенс поговорити про диверсифікацію що і має на увазі не вкладатися у один проект а ділити свій бюджет на різні, щоб в якомусь сенсі занизити свої ризики і не мати можливості втратити всі свої кошти якщо у одного проекта стануться якісь проблеми.

Ринок криптовалют вважається найволатильнішим через мале розуміння людей при прикріпленність активів до чогось реального і для більшості це просто гра, через це ми маємо можливість постійних змін

настрою і стрімких падінь чи росту. Тому ці принципи дійсно важливі і про них слід ніколи не забувати. Важливо знати також і свої цілі з стратегією, тобто можна розраховувати на швидку торгівлю з більшими ризиками або довгостроково зберігати знаючи скільки саме росту потрібно інвестору. Якщо сильно розібратися є дуже багато важливих факторів які впливають на ринок криптовалют, такі як зовнішня політика, ринки акцій, стан інфляції та інші, але більшості потрібно знати лише інформацію про капіталізацію а саме обєм грошей у проекті, загальний обсяг монет та плани по майбутнім діям проектів. Ця кількість факторів зумовлює постійне спостерігання і переоцінка своїх активів для своєї ж безпеки, тому важливо мати при собі корисний інструмент який дозволяє отримувати важливу інформацію.

1.3. Порівняння API-сервісів для отримання криптоданих

При розробці застосунків, пов’язаних з криптовалютами, одним із ключових компонентів буде отримування актуальних ринкових даних, ціни, обсяг торгів, капіталізація, історія змін тощо. Найлегший спосіб організувати це, буде використання API - інтерфейс прикладного програмування, який надає доступ до потрібної інформації в зручному машинному стилі щоб використовувати її у потрібних цілях.

Найбільш популярні сервіси це CoinGecko, CoinMarketCap та різні криптобіржі для торгів.

CoinMarketCap — один із найвідоміших агрегаторів криптовалютної інформації. Його API дає доступ до великого обсягу даних, але щоб отримати повний функціонал треба отримати спецільний ключ тому цей варіант нам одразу не підходить.

Біржі як Binance, OKX, Bybit — також дуже популярні площадки але там є свої низки проблем які легше обійти просто вибором наступного ресурсу.

CoinGecko — це відкритий API, який не потребує реєстрації чи ключів для базових запитів. Він дає доступ до повного спектру інформації: ціни, графіки, зображення і тд. Дуже зручна документація по використанню, запити - прості у реалізації. Тому для реалізації проекту обираємо **CoinGecko API** як найбільш доступний і легкий у використанні застосунок.

1.4. Реалізація функціоналу інструментів

Для отримання данних планується написати скрипт на мові Python і потрібно пройти черех етапи: формування списку токенів які будуть входить у наш портфель, використання бібліотек request щоб надіслати запити до бази данних платформи та створити систему збереження усього до словника або таблиці щоб далі була можливість обробки інформації. Дуже багато функціоналу вважаю не треба добавляти щоб код працював швидко і стабільно як раз через невелику кількість запросів на ресурси, що грає нам на руку тому що це комфортність використання і додатковий фактор обрати людям саме цей інструмент.

1.5. Висновки до розділу

У цьому розділі розберуться основи того, як формується криптовалютний портфель і що взагалі варто враховувати при виборі монет. Зрозуміло, що підхід у всіх різний, але головна ідея — знайти свій комфортний стиль, де не потрібно кожен день панікувати через ринок. Також є фактор, що вибір інструменту для отримання інформації грає велику роль. CoinGecko виявився зручним і доступним без лишньої

мороки, і це реально важливо, бо не кожен студент чи новачок буде розбиратись у складних API. Не обов’язково мати складну систему — достатньо простої логіки, яка працює і дає чітке уявлення, що відбувається з твоїми грошима.

РОЗДІЛ 2. ПРОЄКТНА ЧАСТИНА ІНТЕРАКТИВНОЇ ПЛАТФОРМИ

2.1 Постанова задачі

На основі аналізу який автор провів зверху, сформовано основну мету - Створення веб-застосунку який дозволить користувачу вести свій криптопортфель, а самее доддавати активи, переглядати їх вартість та мати можливість математично розраховувати прибуток в часі. Для досягнення мети необхідно реалізувати такі задачі:

- забезпечити роботу API CoinGecko для отримання актуальних даних про криптовалюти

- реалізувати можливість вибору монет зі списку активів та додавання їх

- реалізувати функції редагування та видалення активів- зберігати дані портфеля у локальному сховищі

- автоматичне створювання розрахунку прибутку або збитку- відображати загальну інформацію портфелю- реалізувати побудову графіка

Окрім функціональних задач проект має навчальну цінність оскільки є можливість закріпити навички роботи з управління API інших ресурсів і використтання їх бібліотек для побудови та обробки данних.

2.2. Вибір технологій та інструментів

Для реалізації застосунку обрано стек технологій, який забезпечує швидку роботу, гнучкість да зручність у використанні як для розробника так і користувача.
1.React — сучасна JavaScript-бібліотека для створення користувацьких інтерфейсів. Вона дозволяє ефективно керувати станом програми, будувати компонентну структуру та створювати інтерактивні елементи. React обрав як основу клієнтської частини застосунку.

2.CoinGecko API — обрано як основне джерело даних про криптовалюти. Його великий плюс — відкритий доступ без обов'язкової реєстрації або ключа. За допомогою цього API отримується список популярних монет, їхні ціни, капіталізацію, обсяг торгів і навіть логотипи — усе, що потрібно для базового відображення інформації у портфелі. 3.Chart.js — ця бібліотека дозволяє малювати графіки прямо у браузері. Використовується вона для візуалізації змін у вартості активів з часом, що допомагає користувачу краще розуміти динаміку свого портфеля. 4. LocalStorage — це спосіб зберігати дані прямо у браузері. Він дає можливість не втрачати свій портфель навіть після оновлення сторінки або закриття браузера. Вся історія і список активів зберігається локально без потреби в базі даних.
5.HTML/CSS — класика для верстки. Дизайн зроблено у темному стилі, щоб не втомлювати очі. Загальний вигляд мінімалістичний і зручний: нічого зайвого, лише корисна інформація.
Таке поєднання технологій дозволило зробити застосунок простим у використанні, безпечним для користувача (бо всі дані локальні), і водночас достатньо функціональним, щоб відслідковувати стан криптовалютного портфеля в реальному часі.

2.3. Структура застосунку

Застосунок побудований за компонентною архітектурою, яка є стандартною практикою для бібліотеки React. Уся основна логіка в одному головному компоненті App, який відповідає за основний функціонал. При потребі цей компонент легко можна розділити на менші частини в майбутньому, якщо застосунок буде розвиватися. У межах компонента App реалізовано: завантаження топ-200 криптовалют із CoinGecko API при

запуску; збереження даних портфеля за допомогою хуку useState; обробку додавання, редагування та видалення активів; збереження інформації в localStorage, щоб вона не зникала після перезавантаження сторінки; автоматичний розрахунок прибутку/збитку для кожної монети та всього портфеля загалом; побудову графіка зміни вартості портфеля через бібліотеку Chart.js.

Кожен актив у портфелі представлений у вигляді об'єкта з такими властивостями: coin — ідентифікатор криптовалюти; amount — кількість монет; buyPrice — ціна покупки; currentPrice — поточна ринкова ціна (оновлюється з API); icon — зображення монети; portfolioHistory — масив із загальною вартістю портфеля на різні дати (для графіка). Активи зберігаються у вигляді масиву об’єктів. При кожній зміні (додаванні чи редагуванні) оновлюється як стан у React, так і історія для графіка. Це дозволяє будувати динаміку змін прямо в інтерфейсі. Запити до CoinGecko виконуються через fetch.

Для кожного активу відправляється окремий запит, що дозволяє отримувати свіжу інформацію без зайвого перевантаження API. Загалом структура застосунку вийшла лаконічною, зручною для масштабування та простою для розгортання. Усі дані зберігаються на стороні користувача, що робить проєкт автономним і безпечним у використанні.

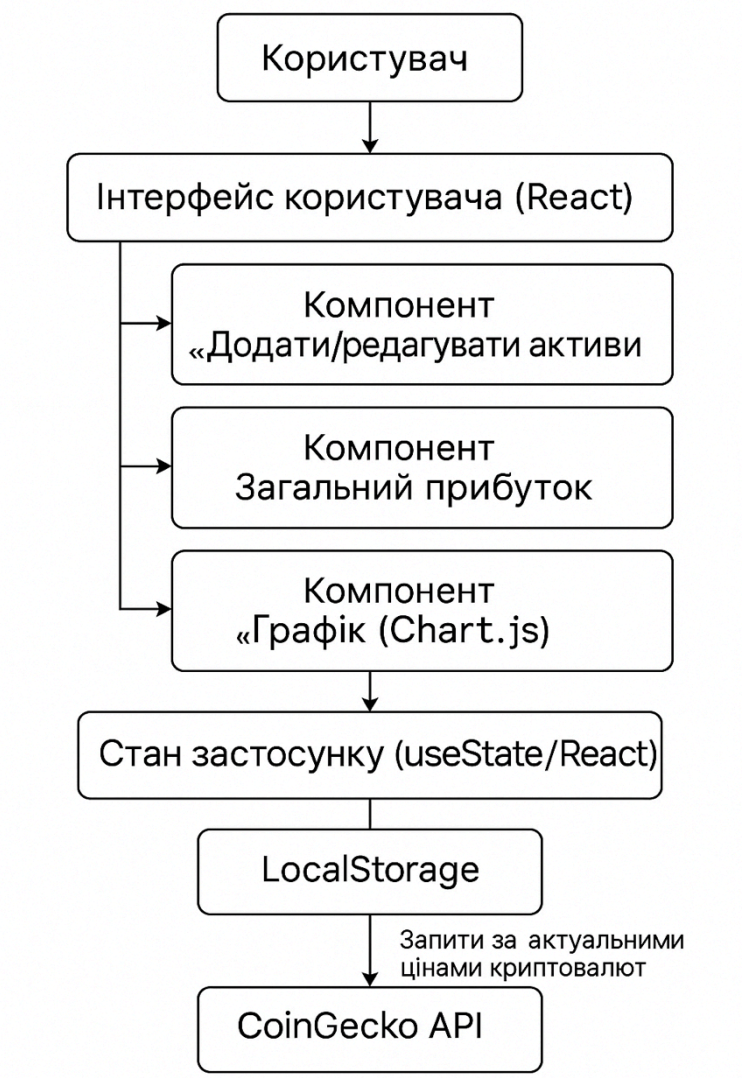

Рисунок 2.1. Структура вебзастосунку

2.4. Взаємодія з API CoinGecko

Щоб підтягувати актуальну інформацію про криптовалюти, використовується відкритий API CoinGecko. Він дуже зручний у використанні, бо не вимагає ключів чи авторизації, що ідеально підходить для невеликого навчального проєкту. Не беручи весь його функціонал — тільки те, що дійсно треба для роботи портфеля. На початку роботи застосунок автоматично робить запит до CoinGecko, щоб отримати список

найпопулярніших монет. Цей запит повертає топ-200 криптовалют за капіталізацією.

З усього обсягу даних найнеобхідніше — унікальний ідентифікатор монети (id), її повну назву (name), символ (тобто тікер типу BTC чи ETH) та посилання на зображення (icon), яке потім показується в інтерфейсі. Коли користувач додає монету до портфеля, застосунок робить ще один запит — цього разу вже за конкретною монетою, щоб дізнатися її актуальну ціну. У відповіді буде просто число, наприклад, що біткоїн зараз коштує 107 500 доларів, і це значення записує программа як поточну ціну (currentPrice). Після цього формується об’єкт активу, який містить п’ять основних полів: назву монети (coin), кількість одиниць, які придбав користувач (amount), ціну, за якою він її купив (buyPrice), поточну вартість з CoinGecko (currentPrice) та іконку (icon). Наприклад, якщо користувач купив пів біткоїна по 105 000, а зараз він коштує 107 500 — ця вся інформація зберігається разом в одному об’єкті.

Усі активи додаються в масив, який зберігається у локальному сховищі браузера (localStorage). Це означає, що навіть якщо користувач перезавантажить сторінку чи закриє вкладку — його портфель нікуди не зникне. Після кожного додавання або редагування активу також оновлюється загальна вартість усього портфеля. Вона додається в окремий масив portfolioHistory із позначкою дати, і саме на основі цього масиву потім будується графік змін. Вся ця модель даних — дуже проста, але її повністю вистачає, щоб відображати актуальну інформацію, рахувати прибуток і будувати графіки, не використовуючи ніяких серверів чи складних баз даних. Все працює прямо в браузері, і це робить застосунок автономним і зручним для користувача.

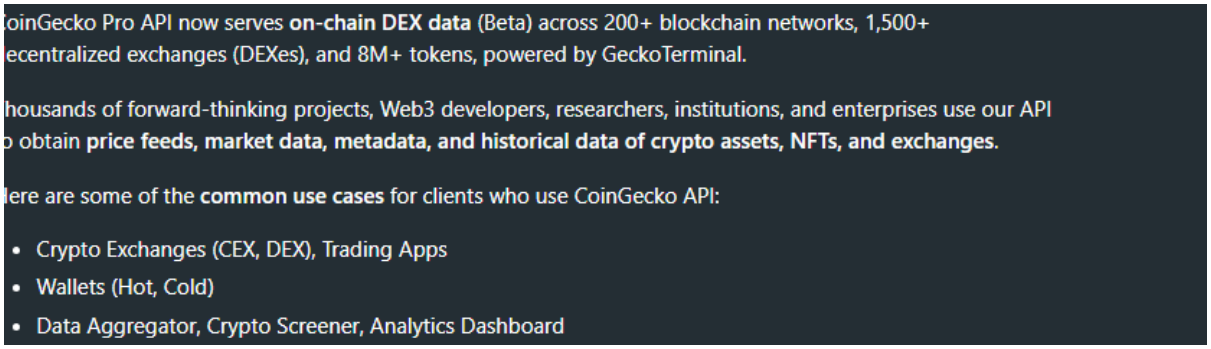

Рисунок 2.2. Можливості CoinGecko (їх документація)

2.5. Висновки до розділу

У цьому розділі вже конкретно розписано, як саме працює цей застосунок: які в нього функції, з чого він складається, як він виглядає всередині і як все це між собою з’єднується. Задачі були чіткі — зробити додаток, який дозволяє тримати свій криптопортфель під контролем. Для цього підібрано набір інструментів, які не грузять, легко реалізуються і реально роблять те, що треба: React для інтерфейсу, CoinGecko API — для даних, Chart.js — для графіків і LocalStorage — щоб усе зберігалось навіть після перезавантаження. Описано також, як виглядає структура самого коду, як працює логіка додавання монет, обробки даних, збереження і виводу результатів. Все зроблено просто, без зайвого ускладнення, але при цьому працює стабільно і вже готове до того, щоб при бажанні розширювати функціонал.

РОЗДІЛ 3. РЕАЛІЗАЦІЙНА ЧАСТИНА

3.1. Функціональні можливості

Застосунок має набір базових, але достатньо корисних функцій, щоб вести особистий криптопортфель. Основна ідея — зробити все максимально зручно, без зайвої навороченості, але щоб користувач міг повністю тримати під контролем, що відбувається з його активами. Що конкретно вміє:

1. Можна вибрати будь-яку монету зі списку топ-200, які я підтягував із CoinGecko — там є і назва, і тікер, і іконка, так що все виглядає зручно при додаванні.
2. При додаванні активу юзер просто вводить кількість і ціну покупки, а актуальну ціну система сама може підтягнути з CoinGecko — це допомагає одразу бачити прибуток.
3. Актив можна редагувати або видалити в один клік — біля кожної монети є кнопки, нічого шукати не треба.
4. Поточна ціна оновлюється автоматично при кожній зміні, і на основі цього одразу рахується прибуток або збиток.
5. Загальний прибуток по всьому портфелю теж рахується і оновлюється динамічно.
6. Всі дані (портфель, історія змін) зберігаються прямо в браузері через localStor, нічого не злітає навіть після оновлення сторінки.
7. При кожній зміні портфеля автоматично перераховується його загальна вартість і додається точка в історію щоб потім це можна було показати на графіку.
8. Сам інтерфейс зроблений акуратно, у темній темі, з простими формами і зрозумілими блоками для кожного активу.
9. Цього функціоналу вистачає, щоб тримати крипту під контролем і бачити основну інформацію без використання складних

платформ чи бірж. Усе зроблено для швидкої взаємодії і простого користування.

3.2. Реалізація інтерфейсу

Інтерфейс робився з ідеєю, щоб усе було максимально простим і зрозумілим — без зайвих заморочок, але щоб юзеру реально було зручно працювати зі своїм портфелем. Вся головна логіка винесена на одну сторінку, де відразу видно, що додаєш, що вже є в портфелі, який прибуток, і як він змінюється з часом.

Що є на сторінці: Зверху — заголовок і назва проєкту (Crypto Portfolio). Додав емодзі, щоб виглядало трошки живіше і не нудно, але все в мінімалістичному стилі. Далі — форма для додавання активу. Там можна вибрати монету зі списку, ввести кількість, ціну покупки і натиснути кнопку, щоб автоматично підтягнути поточну ціну. Кнопка внизу — або "Додати актив", або "Редагувати актив", якщо змінюєш вже існуючий. Під формою — список активів, які вже додалися. Кожен елемент — це такий окремий блок, у якому показано іконку монети, її назву (ідентифікатор), кількість, ціну покупки, поточну ціну, прибуток і кнопки "Редагувати" або "Видалити". Все в одному місці, нічого шукати не треба. Нижче — блок із загальним прибутком. Він автоматично рахує різницю між тим, за скільки ти купив усі монети, і скільки вони коштують зараз. Ще нижче — графік зміни вартості портфеля. Він оновлюється кожного разу, як ти щось додаєш або редагуєш. Побудований на Chart.js, виглядає просто, але інформативно. По дизайну — все в темній темі. Використано темно-сірий фон, світлий текст, закруглені кути, тіні і плавні ефекти при наведенні. Класичний спокійний вигляд, щоб не напрягати очі. Інтерфейс адаптований під десктоп: весь контент тримається по центру, не розтягується на всю ширину, і навіть на невеликих екранах виглядає

читабельно (зробив обмеження до 800 пікселів по ширині). Загалом, все зроблено так, щоб юзер одразу розумів, що до чого, не губився в деталях і міг спокійно керувати своїм портфелем.

```
function App() {
  const [assets, setAssets] = useState([]);
  const [coin, setCoin] = useState('');
  const [amount, setAmount] = useState('');
  const [buyPrice, setBuyPrice] = useState('');
  const [editIndex, setEditIndex] = useState(null);
  const [portfolioHistory, setPortfolioHistory] = useState([]);
  const [coinIcons, setCoinIcons] = useState({});
  const [topCoins, setTopCoins] = useState([]);
```

Рисунок 3.1. Код списку функціоналу застосунку

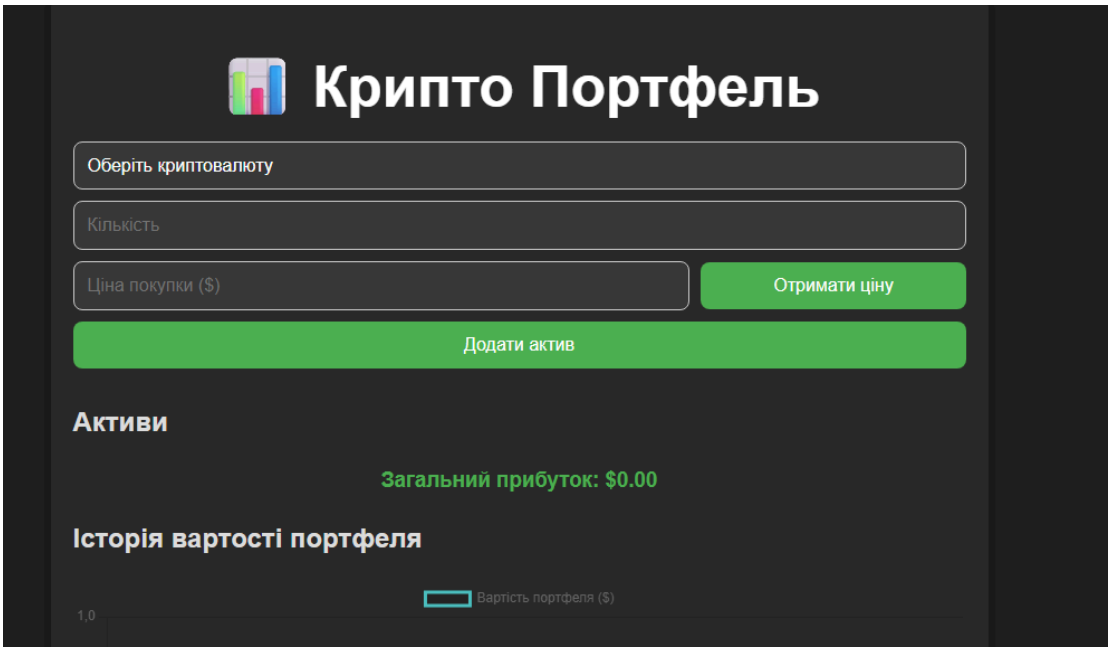

Рисунок 3.2. Початковий інтерфейс інструменту

3.3. Механізм редагування активів

Додавання і редагування активів — це по суті найголовніша річ застосунку, бо саме ці дії формують, що взагалі знаходиться в портфелі

користувача. Все працює просто: спочатку користувач вибирає монету зі списку, вводить кількість, і далі має два варіанти — або сам ввести ціну покупки, або натиснути кнопку «Цінник зараз», щоб підтягнути актуальну ціну з CoinGecko. Якщо обирає другий варіант, то в цей момент летить запит до API, і поле з ціною заповнюється автоматично.

Коли форма заповнена і юзер тисне «Додати актив» — спрацьовує функція addAsset. Вона ще раз тягне актуальну ціну з CoinGecko, формує новий об’єкт з усіма полями (назва монети, кількість, ціна покупки, поточна ціна, іконка) і або додає його в масив assets, або, якщо йде редагування — оновлює вже існуючий елемент. Після цього одразу перераховується загальна вартість портфеля, і це нове значення записується в portfolioHistory з поточною датою — щоб потім це все потрапило в графік.

Коли все зроблено — форма очищається, і можна додавати нові активи. Редагування працює так само просто: натискаєш «Редагувати» біля активу — його дані підставляються назад у форму. Далі можна змінити кількість або ціну, і після натискання кнопки оновлення актив перезаписується у списку. Усі ці зміни автоматично зберігаються в localStorage, бо через React-хук useEffect застосунок слідкує за оновленнями в assets і portfolioHistory. Загалом, механізм зроблений так, щоб усе було просто, наглядно і без всяких серверів. Все зберігається на стороні користувача, і працює швидко, без глюків.

```
  const res = await fetch(
    'https://api.coingecko.com/api/v3/coins/markets?vs_currency=usd&order=market_cap_desc&per_page=200&page=1'
  );
  const data = await res.json();

  const icons = {};
  data.forEach((coin) => {
    icons[coin.id] = coin.image;
  });

  setCoinIcons(icons);
  setTopCoins(data.map((coin) => ({ id: coin.id, name: coin.name, symbol: coin.symbol })));
} catch (error) {
  console.error('Помилка при завантаженні топ-200 монет:', error);
}
```

Рисунок 3.3. Використання API для редагування активів

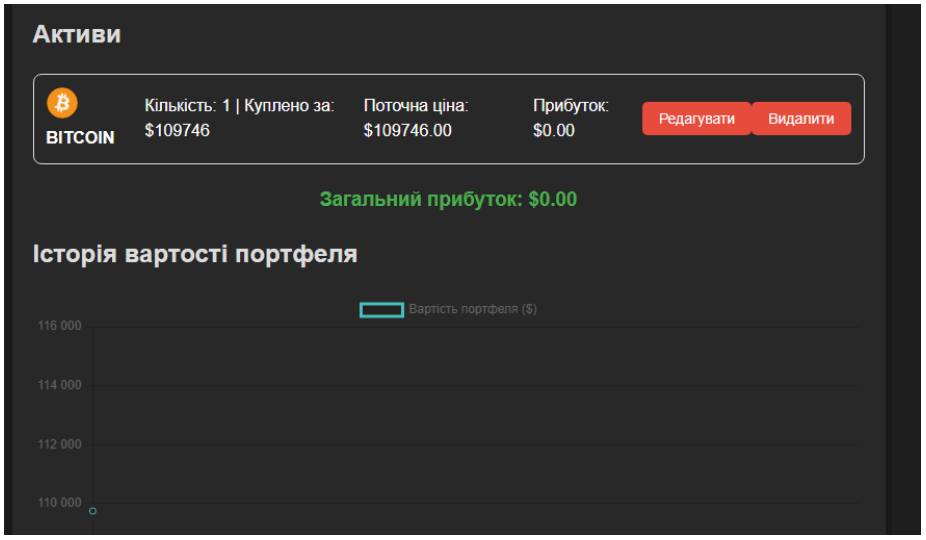

Рисунок 3.4. Інтерфейс активів які знаходяться у портфелі

3.4. Збереження портфеля

Щоб користувач не втрачав свій криптопортфель після перезавантаження сторінки або закриття браузера, у застосунку реалізоване локальне збереження всіх даних прямо в браузері. При кожному додаванні або редагуванні активу здійснюється перерахунок загальної вартості портфеля (кількість монет множиться на їх поточну ціну), формується об'єкт з новим значенням та датою змін, додається відповідний запис у масив portfolioHistory, а сам портфель — у масив assets. Обидва масиви (assets і portfolioHistory) зберігаються у localStorage, що забезпечує збереження інформації незалежно від оновлення сторінки. Під час повторного запуску застосунку збережені дані автоматично підтягуються — як поточний стан портфеля, так і історія змін його вартості. Це дає змогу не лише переглядати актуальні активи, а й відстежувати динаміку змін за допомогою графіка. Такий механізм гарантує збереження даних локально, без потреби у реєстрації, створенні акаунтів або використанні серверної частини.

.

```
useEffect(() => {
  localStorage.setItem('assets', JSON.stringify(assets));
  localStorage.setItem('portfolioHistory', JSON.stringify(portfolioHistory));
}, [assets, portfolioHistory]);

const fetchTopCoins = async () => {
```

Рисунок 3.5. Код створення та збереження портфелю

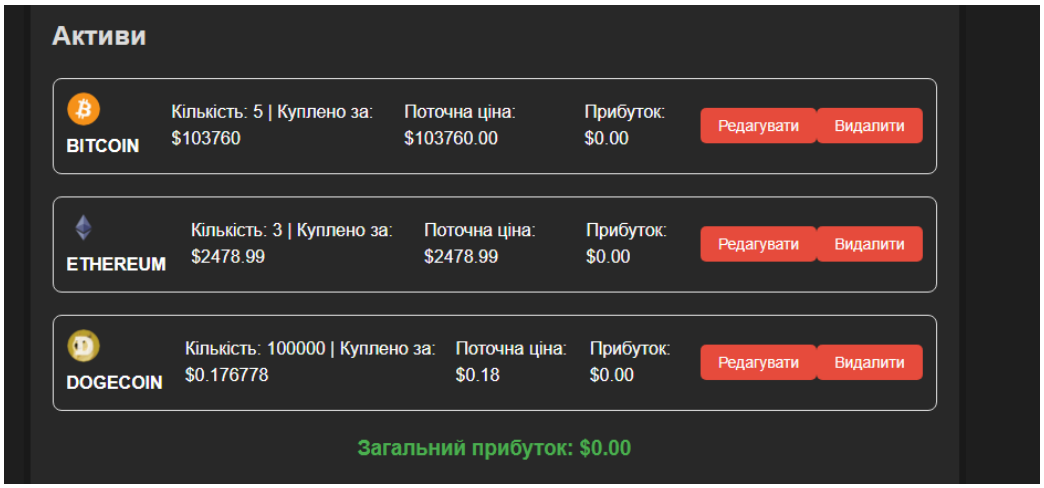

Рисунок 3.6. Інтерфейс активів в інструменті

3.5. Виведення графіків та обчислення прибутку

Застосунок не просто показує список монет — він ще й дає змогу візуально бачити, як змінюється весь портфель з часом, і одразу розраховує прибуток. Це дві важливі штуки: графік і підрахунок грошей. Графік будується за допомогою Chart.js — я використовую компонент Line через react-chartjs-2. Джерело даних — це масив portfolioHistory, де зберігаються дати і відповідна загальна вартість портфеля на той момент.

Що показує графік:

1. по горизонталі — дати (ось X),

2. по вертикалі — сума портфеля в доларах (ось Y),

3.лінія — плавна, кольорова, з підписами і легендою,

4. оновлюється автоматично після кожної зміни портфеля.

Це дозволяє юзеру буквально за пару секунд зрозуміти, коли в нього пішло вгору, а коли — вниз. Дуже зручно і наглядно.

Підрахунок прибутку -

Кожен актив показує, скільки на ньому прибутку або збитку. Формула проста:

прибуток = (поточна ціна – ціна покупки) × кількість

Це значення виводиться прямо під монетою. Якщо число червоне — значить, мінус. Якщо зелоне — все добре.

Окрім цього, під всім портфелем показується ще й загальний прибуток — тобто сума по всіх монетах. Це значення автоматично оновлюється при будь-якій дії: додав, змінив, видалив — і одразу бачиш, як це вплинуло на весь портфель.

Такий підхід дає можливість оцінити не тільки кожну окрему монету, а й ситуацію в цілому. Все чітко, в одному місці, і без калькулятора.

```
const char const portfolioHistory: never[]
  labels: portfolioHistory.map((entry) => entry.date),
  datasets: [
    {
      label: 'Вартість портфеля ($)',
      data: portfolioHistory.map((entry) => entry.value),
      fill: false,
      borderColor: 'rgb(75, 192, 192)',
      tension: 0.1,
    },
```

Рисунок 3.7. Код вирахування вартості портфеля

Зараз демонструється графік зміни в ціні коли додаєш активи у портфель. Для більш деталізованого графіка потрібно більше часу додавати активи щоб висвічувало у більш великому періоді, але є можливість показати просто зміни в ціні за період у день.

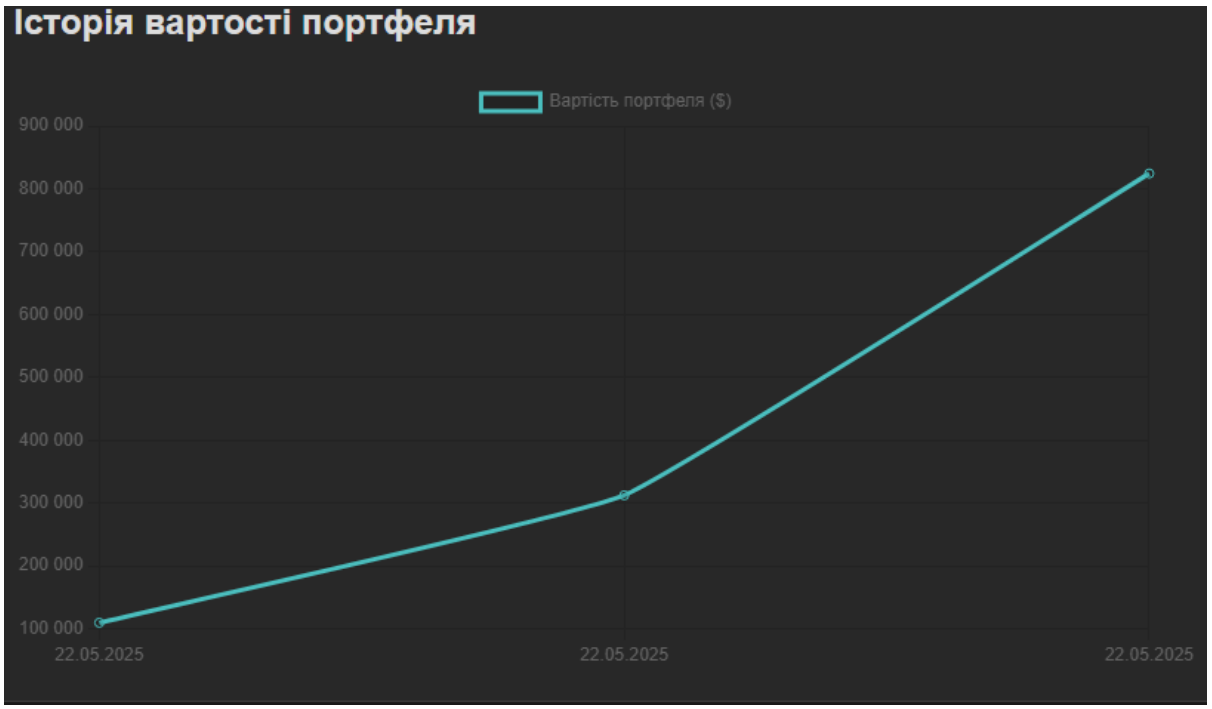

Рисунок 3.8. Інтерфейс графіку ціни портфеля

3.6. Висновки до розділу

У цьому розділі вже на практиці показано, як працює застосунок: від інтерфейсу і взаємодії з активами — до обрахунку прибутку і побудови графіка. Все зроблено так, щоб користувач міг додати або змінити актив, одразу побачити зміни у вартості, і при цьому нічого не губилось навіть після перезавантаження сторінки.

Використано сучасні інструменти — React, CoinGecko API, Chart.js і localStorage — усе це працює на клієнтській стороні, тому не треба ніякого бекенду або серверів. Просто відкрив сторінку — і все працює.

Зроблено акцент на простоті: зручна форма, плавні графіки, мінімалістичний дизайн і автоматичне збереження всього. Механізм обрахунку прибутку теж зроблений просто — але корисно. Все, що заплановано — реалізовано. Застосунок повністю відповідає задачам, які ставилися на початку.

РОЗДІЛ 4. МЕТОДИ ТЕСТУВАННЯ ТА ОЦІНКА

4.1. Перевірка працездатності функціоналу

Для перевірки працездатності вебзастосунку застосовано метод функціонального тестування. Цей метод дозволяє перевірити відповідність роботи системи початково визначеним вимогам та забезпечити коректне виконання основних функцій.

Під час функціонального тестування перевірено:

1. коректність завантаження основної сторінки;

2.правильність отримання та відображення актуальних даних з API CoinGecko;

3. стабільність роботи побудови графіків;

4. працездатність інтерактивних елементів (кнопки, фільтри, меню);

5. стійкість до некоректного або відсутнього вводу.

Тестування виконано вручну за допомогою сценаріїв використання, які дозволяють оцінити поведінку системи в умовах наближених до реального використання. За результатами функціонального тестування усі ключові функції вебзастосунку працюють стабільно та відповідають очікуванням.

4.2. Оцінка зручності інтерфейсу

Для оцінки зручності інтерфейсу вебзастосунку використано метод експертного оцінювання. Даний метод дозволяє визначити ступінь зручності використання системи з позиції кінцевого користувача. Під час проведення оцінювання враховувались такі параметри, як зрозумілість структури інтерфейсу, логіка навігації між сторінками, простота взаємодії з елементами, візуальна привабливість та адаптивність для пристроїв з різними розмірами екрану. Оцінювання проводилось за п’ятибальною

шкалою. За результатами оцінки вебзастосунок отримав середній бал 4.5, що свідчить про високий рівень зручності для користувачів. При цьому було відзначено, що інтерфейс можна доповнити додатковими можливостями кастомізації графіків, що покращить персоналізацію та користувацький досвід.

.

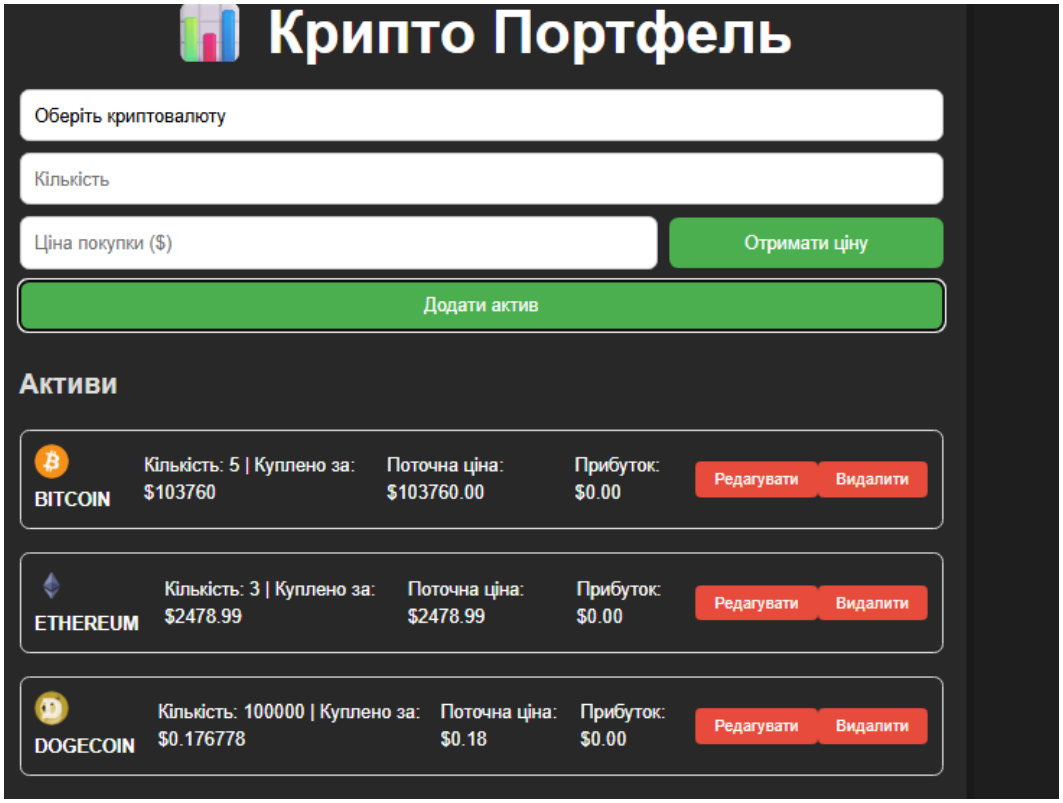

Рисунок 4.1. Інтерфейс повноцінного портфелю

4.3. Безпека роботи з API

Для оцінки зручності інтерфейсу вебзастосунку використано метод експертного оцінювання. Даний метод дозволяє визначити ступінь зручності використання системи з позиції кінцевого користувача. Під час проведення оцінювання враховувались такі параметри, як зрозумілість структури інтерфейсу, логіка навігації між сторінками, простота взаємодії з елементами, візуальна привабливість та адаптивність для пристроїв з різними розмірами екрану.

Оцінювання проводилось за п’ятибальною шкалою. За результатами оцінки вебзастосунок отримав середній бал 4.5, що свідчить про високий рівень зручності для користувачів. При цьому було відзначено, що інтерфейс можна доповнити додатковими можливостями кастомізації графіків, що покращить персоналізацію та користувацький досвід.

```
const fetchTopCoins = async () => {
  try {
    const res = await fetch(
      'https://api.coingecko.com/api/v3/coins/markets?vs_currency=usd&order=market_cap_desc&per_page=200&page=1'
    );
    const data = await res.json();

    const icons = {};
    data.forEach((coin) => {
      icons[coin.id] = coin.image;
    });
```

Рисунок 4.2. Використання апі у коді

4.4. Обмеження і можливості для покращення

Для оцінки зручності інтерфейсу вебзастосунку використано метод експертного оцінювання. Даний метод дозволяє визначити ступінь зручності використання системи з позиції кінцевого користувача. Під час проведення оцінювання враховувались такі параметри, як зрозумілість структури інтерфейсу, логіка навігації між сторінками, простота взаємодії з елементами, візуальна привабливість та адаптивність для пристроїв з різними розмірами екрану. Оцінювання проводилось за п’ятибальною

шкалою. За результатами оцінки вебзастосунок отримав середній бал 4.5, що свідчить про високий рівень зручності для користувачів. При цьому було відзначено, що інтерфейс можна доповнити додатковими можливостями кастомізації графіків, що покращить персоналізацію та користувацький досвід.

4.5. Висновки до розділу

Проведене тестування вебзастосунку з використанням різних методів дозволило отримати всебічне уявлення про якість його роботи, зручність інтерфейсу та рівень безпеки взаємодії з зовнішніми сервісами. В результаті застосування функціонального тестування було підтверджено стабільність основного функціоналу системи та коректність обробки ключових сценаріїв взаємодії користувача з вебзастосунком. Експертне оцінювання інтерфейсу показало високий рівень його зручності та зрозумілості, що є важливим фактором для забезпечення позитивного користувацького досвіду.

Аналіз безпеки взаємодії з API CoinGecko підтвердив, що застосунок відповідає базовим вимогам інформаційної безпеки, а обробка потенційних загроз реалізована на належному рівні. Разом з тим було виявлено ряд обмежень, пов’язаних як із технічними аспектами реалізації, так і із зовнішніми залежностями, зокрема з необхідністю забезпечення стабільного доступу до API та оптимізації роботи з даними. Аналітичний огляд дозволив визначити можливі напрямки подальшого розвитку вебзастосунку, серед яких особливу увагу слід приділити впровадженню механізму кешування, розширенню можливостей налаштування графічного представлення даних та удосконаленню мобільної версії інтерфейсу.

Загалом результати тестування свідчать про те, що створений вебзастосунок є стабільним та функціональним рішенням, яке відповідає визначеним вимогам та завданням. Виявлені можливості для покращення відкривають перспективи для подальшого розвитку проєкту, спрямованого на підвищення ефективності, масштабованості та якості користувацького досвіду. Подальше удосконалення системи дозволить зробити вебзастосунок ще більш зручним, швидким та надійним у використанні.

ВИСНОВКИ

У роботі реалізовано веб-застосунок для управління криптовалютним портфелем із використанням відкритого API CoinGecko. Проєкт включав не лише розробку коду, а й опрацювання теоретичних аспектів: вивчення роботи крипторинку, виявлення проблем користувачів та аналіз потреб у подібному інструменті. На етапі аналітики досліджено сучасний стан ринку, основні підходи до інвестування та типові потреби користувачів — простий облік активів, підрахунок прибутку, візуалізація змін. Проведено огляд доступних API та обрано CoinGecko як найбільш зручний і простий у використанні. У проєктній частині визначено цілі, задачі, обрано технології та продумано архітектуру. У ході реалізації створено застосунок на React, що дозволяє:

1. додавати, редагувати і видаляти криптоактиви;
2. автоматично підтягувати актуальні ціни з CoinGecko;
3. обчислювати прибуток по кожній монеті та по портфелю загалом;
4. зберігати всі дані локально в браузері без використання серверів;
5. демонструвати графік змін вартості з часом.

Після завершення основної частини проведено практичне тестування функціональності — застосунок працює стабільно, без виявлених критичних помилок, інтерфейс є зрозумілим навіть для користувачів-початківців. З точки зору безпеки використання є безпечним: персональні дані не обробляються, API є відкритим, передавання зайвих даних відсутнє. Незважаючи на наявні обмеження, такі як відсутність мобільної версії або синхронізації через акаунт, усі заплановані функції реалізовано. Застосунок становить собою не просто дипломний проєкт, а повноцінний робочий інструмент із потенціалом для подальшого розвитку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Nakamoto S. Bitcoin: A Peer-to-Peer Electronic Cash System [Електронний ресурс]. – 2008. – Режим доступу: https://bitcoin.org/bitcoin.pdf.
2. CoinGecko API Documentation [Електронний ресурс]. – Режим доступу: https://www.coingecko.com/en/api/documentation.
3. CoinMarketCap. Cryptocurrency Market Capitalizations [Електронний ресурс]. – Режим доступу: https://coinmarketcap.com/.
4. Ethereum whitepaper. A Next-Generation Smart Contract and Decentralized Application Platform [Електронний ресурс]. – Режим доступу: https://ethereum.org/en/whitepaper/.
5. React Official Documentation [Електронний ресурс]. – Режим доступу: https://reactjs.org/.
6. Chart.js Documentation [Електронний ресурс]. – Режим доступу: https://www.chartjs.org/docs/latest/.
7. MDN Web Docs. localStorage API [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage.
8. І.В. Машкіна, В.О. Абрамов, Т.І. Носенко, Є.В. Іваніченко. Навчально-методичний посібник до виконання і захисту кваліфікаційної роботи бакалавра спеціальностей 122 Комп’ютерні науки для здобувачів вищої освіти денної форми навчання, Київський столичний університет імені Бориса Грінченка. Київ, 2024.- 43 с.
9. Теоретичні та практичні аспекти використання математичних методів та інформаційних технологій в освіті й науці:моногр. / за заг. ред. О. Литвин. — К.: Київ. ун-т ім. Б. Грінченка,2021. — 332 с.

10. Binance Academy. What Is a Crypto Portfolio? [Електронний ресурс]. – Режим доступу: https://academy.binance.com/en/articles/what-is-a-crypto-portfolio.
11. Investopedia. Cryptocurrency Explained [Електронний ресурс]. – Режим доступу: https://www.investopedia.com/terms/c/cryptocurrency.asp.
12. Ковальчук С. А. Основи Web-програмування : навч. посіб. – Київ : КНЕУ, 2020. – 198 с.
13. Binance Research. Understanding Decentralized Finance (DeFi) [Електронний ресурс]. – Режим доступу: https://research.binance.com/en/analysis/defi-overview.
14. Tether Whitepaper. Tether: Fiat currencies on the Bitcoin blockchain [Електронний ресурс]. – Режим доступу: https://tether.to/en/white-papers/.
15. Chainlink Whitepaper. A Decentralized Oracle Network [Електронний ресурс]. – Режим доступу: https://chain.link/whitepaper.
16. Cointelegraph. Latest Cryptocurrency News [Електронний ресурс]. – Режим доступу: https://cointelegraph.com/.
17. Messari Crypto. Crypto Research and Data [Електронний ресурс]. – Режим доступу: https://messari.io/.
18. Kraken Learn Center. What is a Blockchain? [Електронний ресурс]. – Режим доступу: https://www.kraken.com/learn/what-is-blockchain.
19. Bitfinex Learn. Understanding Stablecoins [Електронний ресурс]. – Режим доступу: https://www.bitfinex.com/learn/crypto/stablecoins.