Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Створення інструменту для візуалізації та аналізу інформації про крипто-активи.

Виконав: студент IV курсу, групи СН-41
спеціальності 122 Комп’ютерні науки
(шифр і назва спеціальності)

_____ Курян В.А.
(підпис) (прізвище та ініціали)

Керівник _____ Козбур Г.В.
(підпис) (прізвище та ініціали)

Нормоконтроль _____ Шимчук Г.В.
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

Рецензент _____ Микитишин А.Г.
(підпис) (прізвище та ініціали)

Тернопіль
2024

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«____» _червня_ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

Студенту Курян Віталій Андрійович
(прізвище, ім'я, по батькові)

1. Тема роботи Створення інструменту для візуалізації та аналізу інформації про крипто-активи.

Керівник роботи Козбур Галина Володимирівна, к. т. н., доцент кафедри КН
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «*29* « *квітня* 2024 року № *4/7-470*

2. Термін подання студентом завершеної роботи 24 червня 2024р.

3. Вихідні дані до роботи інтернет джерела та статті відповідні до теми кваліфакаційної роботи

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. Розділ 1. Аналіз предметної області та постановка задач розробки інструменту. 1.1 Аналіз предметної області. 1.2 Формування вимог до інструменту для візуалізації та аналізу крипто-активів. 1.3 Огляд існуючих рішень та обґрунтування необхідності нового інструменту. 1.4 Вибір мови програмування та середовища розробки. 1.5 Висновок до першого розділу. Розділ 2. Проектування інструменту для візуалізації та аналізу крипто-активів. 2.1 Архітектура інструменту. 2.2 Розробка функціональної моделі. 2.3 Інтерфейс користувача та візуальні компоненти. 2.4 Інтеграція з зовнішніми API для отримання даних про крипто-активи. 2.5 Висновок до другого розділу. Розділ 3. Реалізація інструменту для візуалізації та аналізу крипто-активів. 3.1 Вибір та використання бібліотек для розробки. 3.2 Розробка back-end частини на Python. 3.3 Розробка front-end частини. 3.4 Валідація та тестування інструменту. 3.5 Висновки до третього розділу. Розділ 4. Безпека життєдіяльності, основи охорони праці. 4.1 Ризик як кількісна оцінка небезпек. 4.2 Вимоги ергономіки до організації робочого місця оператора ПК. 4.3 Висновок до четвертого розділу. Висновки. Перелік джерел. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці	Сенчишин В.С. доцент кафедри МТ		

7. Дата видачі завдання 29 січня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	30.01.2024	*Виконано*
2.	Підбір джерел про інструменти для візуалізації та	31.01.2024-03.02.2024	*Виконано*
	аналізу даних про крипто-активи		
3.	Опрацювання джерел по темі кваліфікаційної роботи	04.02.2024-06.02.2024	*Виконано*
4.	Виконання дослідження щодо інформації про	07.02.2024-11.02.2024	*Виконано*
	крипто-активи		
	Розроблення інструменту для візуалізації та аналізу		
5.	Оформлення розділу «Аналіз предметної області та	03.06.2024-05.06.2024	*Виконано*
	постановка задач розробки інструменту»		
6.	Оформлення розділу «Проектування інструменту для	06.06.2024-08.06.2024	*Виконано*
	візуалізації та аналізу крипто-активів»		
7.	Оформлення розділу «Реалізація інструменту для	09.06.2024-11.06.2024	*Виконано*
	візуалізації та аналізу крипто-активів»		
8.	Виконання завдання до підрозділу «Безпека життєдіяльності»	12.06.2024-13.06.2024	*Виконано*
9.	Виконання завдання до підрозділу «Основи охорони	14.06.2024-15.06.2024	*Виконано*
	праці»		
10.	Оформлення кваліфікаційної роботи	16.06.2024-17.06.2024	*Виконано*
11.	Нормоконтроль	18.06.2024-19.06.2024	
12.	Перевірка на плагіат	20.06.2024	
13.	Попередній захист кваліфікаційної роботи	21.06.2024	
14.	Захист кваліфікаційної роботи		

Студент _____ Курян В. А.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Козбур Г. В.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Створення інструменту для візуалізації та аналізу інформації про крипто-активи. // Кваліфікаційна робота освітнього рівня «Бакалавр» // Курян Віталій Андрійович// Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група СН-41 // Тернопіль, 2024 // С. 72, рис. – 11, табл. – 3, слайди – 15, додат. – 4, бібліогр. – 27.

Ключові слова: крипто-активи, транзакція, блокчейн, трейдер, криптовалюта, токен, інвестиції.

Кваліфікаційна робота присвячена розробці інструменту для візуалізації та аналізу крипто-активів.

У першому розділі кваліфікаційної роботи проаналізовано крипто-активи, їх використання та виклики, огляд існуючих рішень і їхні обмеження. Обґрунтовано вибір технологій для розробки інструменту, зокрема Python та PyCharm Edu.

Другий розділ кваліфікаційної роботи присвячено проектуванню інструменту: визначено його архітектуру, описано інтерфейс користувача та інтеграцію з зовнішніми API для отримання даних про крипто-активи.

У третьому розділі кваліфікаційної роботи описано реалізацію інструменту на основі Python з використанням tkinter. Реалізовано функції для збору, обробки та візуалізації даних, створено користувацький інтерфейс. Проведено тестування, яке виявило здатність інструменту виконувати основні завдання та необхідність доопрацювань.

У розділі «Безпека життєдіяльності, основи охорони праці» кваліфікаційної роботи висвітлено аспекти управління ризиками та організації робочих місць для операторів ПК, вимоги до ергономіки, параметри повітряного середовища, освітленості та рівня шуму.

ANNOTATION

Creation of a Tool for Visualization and Analysis of Information About Crypto-Assets // Qualification work of the educational level «Bachelor» // Kurian Vitalii // Ternopil Ivan Pulyu National Technical University, Computer and Information Systems and Software Engineering Faculty, Computer Sciences Department, group SN-41 // Ternopil, 2024 // P. 72, fig. - 11, tabl. - 3, slides - 15, annexes. – 4, references - 27.

Keywords: crypto assets, transaction, blockchain, trader, cryptocurrency, token, investment.

The qualification work is devoted to the development of a tool for visualization and analysis of crypto-assets.

In the first section of the qualification work, crypto-assets, their use and challenges, an overview of existing solutions and their limitations are analyzed. The choice of technologies for the development of the tool is justified, in particular Python and PyCharm Edu.

The second section of the qualification work is devoted to the design of the tool: its architecture is defined, the user interface and integration with external APIs for obtaining data on crypto-assets are described.

The third section of the qualification work describes the implementation of the Python-based tool using tkinter. Functions for data collection, processing, and visualization have been implemented, and a user interface has been created. Testing was conducted, which revealed the ability of the tool to perform basic tasks and the need for improvements.

In the section «Life safety, basics of labor protection» of the qualification paper, aspects of risk management and organization of workplaces for PC operators, requirements for ergonomics, parameters of the air environment, lighting and noise level are covered.

ПЕРЕЛІК СКОРОЧЕНЬ І ТЕРМІНІВ

API (англ. Application Programming Interface) – Інтерфейс програмування додатків, набір підпрограм, протоколів і інструментів для створення програмного забезпечення.

HTTP-запит (англ. Hypertext Transfer Protocol request) – Запит протоколу передачі гіпертексту, основний механізм для комунікації між клієнтом та сервером у мережі Інтернет.

Python – Пайтон, мова програмування високого рівня загального призначення, що підтримує різні парадигми програмування.

Блокчейн – Ланцюжок блоків, розподілена база даних, яка підтримує список записів, що постійно зростає, захищених від фальсифікації та перегляду.

Криптовалюта – Цифрова валюта, яка використовує криптографію для забезпечення безпеки транзакцій і контролю створення нових одиниць.

Трейдер – Особа, що здійснює торгівлю фінансовими активами, такими як акції, облігації, валюта або криптовалюта, з метою отримання прибутку.

ЗМІСТ

ВСТУП

Актуальність теми. У сучасну епоху цифрових технологій зростає інтерес до крипто-валют як до предмета інвестування та загального інтересу. Стрімкий технологічний прогрес і збільшення уваги до цього фінансового ринку створили потребу в інструментах, які можуть ефективно візуалізувати та аналізувати інформацію про крипто-активи. З розвитком блокчейн-технологій та децентралізованих фінансових систем, ринок крипто-валют став значно більш динамічним та складним, що вимагає від користувачів глибоких знань та аналітичних навичок для прийняття зважених інвестиційних рішень.

Мета і задачі дослідження. Основна мета цього проекту – покращити доступність та якість інформації про крипто-валютні активи для широкого кола користувачів. Це вимагає вирішення різних завдань, таких як оцінка існуючих методів візуалізації та аналізу даних про ринки крипто-валют, а також створення нових підходів та інструментів для ефективного використання цієї інформації.

Для досягнення поставленої мети потрібно:

- Проаналізувати існуючі програмні рішення для візуалізації та аналізу крипто-валютних даних.
- Визначити основні вимоги до інструментів для користувачів різних категорій, таких як інвестори, трейдери, дослідники та аналітики.
- Створити зручний та інтуїтивно зрозумілий інтерфейс користувача, який дозволяє швидко та ефективно взаємодіяти з даними.

Практичне значення одержаних результатів. Висновки цієї роботи є цінними для інвесторів, трейдерів, дослідників та аналітиків ринків крипто-валют. Розробка надійного інструменту для візуалізації та аналізу крипто-валютних даних може допомогти в прийнятті рішень, покращити управління ризиками та сприяти зростанню ринку. Використання таких інструментів підвищує прозорість та доступність ринкових даних, що сприяє збільшенню довіри до крипто-валют і їх популярності.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧ РОЗРОБКИ ІНСТРУМЕНТУ

1.1 Аналіз предметної області

Крипто-активи є невід'ємною частиною сучасної цифрової економіки, привертаючи увагу інвесторів, трейдерів та аналітиків. Ці активи базуються на технології блокчейн, що забезпечує децентралізоване зберігання даних, прозорість та безпеку транзакцій [1].

Розглянемо основні аспекти крипто-активів. Найвідоміші з них – це крипто-валюти, такі як Bitcoin та Ethereum, які використовуються як засіб обміну та зберігання вартості [2]. Bitcoin, як перша крипто-валюта, залишається найбільш відомою і найціннішою завдяки обмеженій емісії та високій ліквідності. Ethereum, крім функції крипто-валюти, пропонує платформу для створення децентралізованих додатків (dApps) та смарт-контрактів, що значно розширює його функціональність [3].

Інші крипто-активи, відомі як токени, можуть представляти різні активи або давати доступ до певних послуг. Вони створюються на існуючих блокчейн-платформах і мають широке застосування, від фінансових послуг до управління цифровими правами. Наприклад, токени можуть використовуватися для краудфандингу через первинні пропозиції монет (ICO), надаючи інвесторам можливість підтримати нові проекти та отримати в обмін частку у вигляді токенів [4].

Технологія блокчейн лежить в основі всіх крипто-активів. Вона забезпечує децентралізоване зберігання даних, що підвищує їх безпеку та незмінність. Прозорість блокчейну дозволяє будь-кому перевіряти транзакції, що робить систему більш надійною.

Крипто-активи знаходять застосування у різних сферах. Інвестори та трейдери використовують їх для отримання прибутку від коливань цін, торгуючи на численних біржах. Смарт-контракти дозволяють автоматично виконувати

угоди без посередників, знижуючи витрати та підвищуючи ефективність. Вони можуть використовуватися для створення децентралізованих автономних організацій (DAO), що працюють без втручання людини [5].

Децентралізовані фінансові послуги (DeFi) відкривають нові можливості для кредитування, страхування та інших фінансових операцій без традиційних установ. DeFi-платформи дозволяють користувачам надавати кредити або позичати кошти, заробляти відсотки на свої активи або страхувати свої інвестиції без участі банків чи інших фінансових посередників [6].

Крипто-активи, попри свої переваги, мають ряд викликів та ризиків. Висока волатильність цін може призводити до значних фінансових втрат, оскільки ціна Bitcoin може змінюватися на десятки відсотків протягом одного дня. Це створює високі ризики для інвесторів [7]. Регуляторні ризики також важливі, оскільки правовий статус крипто-активів варіюється в різних країнах: у деяких вони заборонені, в інших – регулюються як фінансові інструменти або товари.

Безпека є критичним аспектом, оскільки зберігання крипто-активів потребує надійного захисту від хакерських атак. Зберігання активів у холодних гаманцях (офлайн) та використання багатофакторної автентифікації можуть допомогти знизити ці ризики [8].

Таким чином, глибоке розуміння крипто-активів та технології блокчейн є необхідним для розробки інструменту, що дозволить ефективно візуалізувати та аналізувати інформацію про крипто-валюти. Це допоможе користувачам приймати обґрунтовані рішення та знижувати ризики, пов'язані з інвестуванням у цей швидкозростаючий ринок.

1.2 Формування вимог до інструменту для візуалізації та аналізу крипто-активів

Створення ефективного інструменту для візуалізації та аналізу крипто-активів потребує врахування різних вимог користувачів, технічних аспектів та

особливостей ринку. Основні вимоги можна розділити на функціональні, нефункціональні та користувацькі.

Функціональні вимоги включають можливість збору даних з різних джерел, таких як біржі, новинні ресурси та блокчейн-платформи, обробку великих обсягів даних у режимі реального часу, підтримку методів аналізу, таких як технічний і статистичний аналіз, а також візуалізацію даних у вигляді графіків і діаграм. Важливо також забезпечити інтерактивність, щоб користувачі могли взаємодіяти з візуалізаціями, змінювати параметри та налаштування.

Нефункціональні вимоги стосуються продуктивності, масштабованості та сумісності. Інструмент має забезпечувати високу швидкодію при обробці та візуалізації великих обсягів даних, надійний захист від хакерських атак, можливість масштабування для обробки збільшеного обсягу даних та користувачів, а також сумісність з різними операційними системами та пристроями.

Користувацькі вимоги включають зручність використання, налаштовуваність та підтримку. Інтерфейс користувача повинен бути інтуїтивно зрозумілим та зручним, з можливістю налаштовування під потреби користувачів, включаючи вибір мов та тем оформлення. Також важливо забезпечити наявність детальної документації та системи підтримки для допомоги користувачам у разі виникнення проблем або питань.

Врахування цих вимог забезпечить створення зручного та ефективного інструменту для візуалізації та аналізу крипто-активів, який допоможе користувачам приймати обґрунтовані інвестиційні рішення та знижувати ризики, пов'язані з ринком крипто-валют.

1.3 Огляд існуючих рішень та обґрунтування необхідності нового інструменту

На ринку крипто-активів вже існує ряд інструментів для візуалізації та аналізу даних. Серед них найбільш популярними є такі рішення як

CoinMarketCap, TradingView, CryptoCompare та інші. Кожне з цих рішень має свої особливості, переваги та недоліки.

CoinMarketCap – один з найвідоміших ресурсів для відстеження цін крипто-валют. Він надає інформацію про поточні ціни, обсяги торгів, ринкову капіталізацію та інші важливі метрики. Однак, цей ресурс обмежений у можливостях глибокого аналізу та візуалізації даних, що робить його менш корисним для професійних трейдерів та аналітиків.

TradingView – потужний інструмент для технічного аналізу, який підтримує широкий спектр фінансових активів, включаючи крипто-валюти. Він пропонує інструменти для побудови графіків, технічних індикаторів та аналізу ринкових даних [9]. Однак, TradingView може бути занадто складним для новачків та потребує платної підписки для доступу до розширених функцій.

CryptoCompare – ресурс, що надає дані про ринок крипто-валют, включаючи історичні дані, графіки та аналітику. Він також пропонує API для доступу до даних, що дозволяє інтегрувати їх у власні додатки. Проте, CryptoCompare може не мати достатньо можливостей для аналізу даних у режимі реального часу [10].

CoinGecko – ще один популярний інструмент для відстеження цін крипто-валют, який надає інформацію про ринкові капіталізації, обсяги торгів та інші ключові метрики. Однак, його функціонал для технічного аналізу також обмежений.

Основні недоліки існуючих рішень включають обмежені можливості для глибокого аналізу, недостатню зручність для новачків та складність у використанні. Крім того, більшість з них потребує платної підписки для доступу до розширених функцій, що може бути перешкодою для багатьох користувачів.

Ще однією важливою проблемою є надмірна кількість інформації, яка надається користувачам. Усі ці платформи надають величезний обсяг даних, і якщо користувач не знає, що саме йому потрібно, він може швидко заплутатися. Неможливість зменшити обсяг цієї інформації ускладнює процес аналізу та

прийняття рішень, особливо для новачків. Це може призвести до неправильних інтерпретацій даних та, відповідно, до невдалих інвестиційних рішень.

Обґрунтування необхідності нового інструменту базується на потребі в комплексному рішенні, яке б поєднувало в собі простоту використання та широкі можливості для аналізу даних. Новий інструмент повинен забезпечувати зручний доступ до великого обсягу даних та мати інтуїтивно зрозумілий інтерфейс, який підходить як для новачків, так і для професійних користувачів.

Таким чином, новий інструмент для візуалізації та аналізу крипто-активів має значний потенціал для покращення доступності та якості інформації про ринок крипто-валют, що сприятиме прийняттю обґрунтованих інвестиційних рішень та зниженню ризиків для користувачів.

1.4 Вибір мови програмування та середовища розробки

Для роботи над інструментом було проведено порівняння різних мов програмування та середовищ розробки, щоб обрати найкращий варіант для розробки інструменту для візуалізації та аналізу крипто-активів.

Порівняння мов програмування:

Python – високорівнева мова програмування, відома своєю простотою та читабельністю.

- Переваги: Велика кількість бібліотек для аналізу даних (Pandas, NumPy, Matplotlib), підтримка машинного навчання (Scikit-learn, TensorFlow), простий синтаксис.
- Недоліки: Менша швидкість виконання порівняно з мовами на зразок C++.

JavaScript – мова програмування, яка використовується для створення інтерактивних веб-додатків.

- Переваги: Популярна у front-end розробці, зручна для створення інтерактивних веб-додатків.
- Недоліки: Обмежені можливості для аналізу даних порівняно з Python.

R – мова програмування, спеціалізована для статистичного аналізу та візуалізації даних [11].

- Переваги: Велика кількість пакетів для статистики та аналізу даних.
- Недоліки: Менш універсальна, складніший синтаксис.

Java – високопродуктивна мова програмування, відома своєю багатоплатформеністю.

- Переваги: Висока продуктивність, багатоплатформеність.
- Недоліки: Складний синтаксис, не настільки зручний для швидкого прототипування та аналізу даних.

Порівняння середовищ розробки (IDE):

PyCharm Edu – інтегроване середовище розробки, спеціалізоване для Python.

- Переваги: Зручний інтерфейс, інтеграція з Git, автодоповнення коду.
- Недоліки: Обмежені функціональні можливості у порівнянні з професійною версією PyCharm.

Visual Studio Code – безкоштовне, багатоплатформене середовище розробки з підтримкою багатьох мов програмування [12].

- Переваги: Велика кількість плагінів, підтримка багатьох мов програмування.
- Недоліки: Може вимагати додаткового налаштування для оптимальної роботи з Python.

Jupyter Notebook – інтерактивне середовище, ідеальне для аналізу даних та машинного навчання.

- Переваги: Можливість виконання блоків коду окремо.
- Недоліки: Менш зручний для великих проектів програмного забезпечення.

Eclipse – потужне середовище розробки, широко використовуване для програмування на Java [13].

- Переваги: Велика кількість плагінів.
- Недоліки: Складний інтерфейс, менш популярний для Python-розробки.

На основі проведеного аналізу було обрано мову програмування Python та середовище розробки PyCharm Edu. Python відзначається своєю простотою, широкими можливостями для аналізу даних та підтримкою машинного навчання. PyCharm Edu забезпечує зручний інтерфейс, інтеграцію з Git та всі необхідні інструменти для розробки та тестування, що робить його ідеальним вибором для цього проекту. Ця комбінація дозволяє швидко створювати прототипи, тестувати та впроваджувати нові функції, а також забезпечує потужні інструменти для аналізу та візуалізації даних.

1.5 Висновок до першого розділу

В першому розділі кваліфікаційної роботи було проведено детальний аналіз предметної області, розглянуто існуючі рішення та обґрунтовано необхідність розробки нового інструменту для візуалізації та аналізу крипто-активів.

Розпочато з аналізу предметної області, де розглянуто основні аспекти крипто-активів. Крипто-валюти, такі як Bitcoin та Ethereum, а також інші токени, які використовуються для різних цілей, зокрема для краудфандингу та управління цифровими правами. Було відзначено важливість технології блокчейн, яка забезпечує децентралізоване зберігання даних, прозорість та безпеку транзакцій. Також було розглянуто основні сфери застосування крипто-активів, включаючи інвестиції, трейдинг та децентралізовані фінансові послуги (DeFi). Окремо наголошено на викликах та ризиках, пов'язаних з високою волатильністю, регуляторними питаннями та безпекою.

Далі було проведено огляд існуючих рішень, таких як CoinMarketCap, TradingView, CryptoCompare та CoinGecko. Виявлено, що, хоча ці інструменти надають велику кількість інформації, вони мають ряд обмежень, включаючи недостатні можливості для глибокого аналізу, складність використання для новачків та надмірну кількість даних, що ускладнює прийняття рішень. Це

підкреслило необхідність створення нового інструменту, який поєднує простоту використання з широкими можливостями для аналізу даних.

Також було проведено порівняння різних мов програмування та середовищ розробки. Було обрано мову програмування Python та середовище розробки PyCharm Edu завдяки їх простоті, широким можливостям для аналізу даних та підтримці машинного навчання. PyCharm Edu забезпечує зручний інтерфейс та всі необхідні інструменти для розробки та тестування, що робить його ідеальним вибором для цього проекту.

Таким чином, у першому розділі кваліфікаційної роботи були закладені основи для подальшої розробки ефективного інструменту для візуалізації та аналізу крипто-активів, який допоможе користувачам приймати обґрунтовані інвестиційні рішення та знижувати ризики, пов'язані з ринком крипто-валют.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ІНСТРУМЕНТУ ДЛЯ ВІЗУАЛІЗАЦІЇ ТА АНАЛІЗУ КРИПТО-АКТИВІВ

2.1 Архітектура інструменту

Архітектура інструменту для візуалізації та аналізу крипто-активів включає кілька основних компонентів і модулів, кожен з яких виконує специфічні функції. Основними модулями інструменту є:

Модуль збору даних відповідає за отримання даних з різних джерел, таких як біржі, новинні ресурси та інші релевантні платформи. Цей модуль інтегрується з зовнішніми API для отримання поточних та історичних даних про крипто-активи.

Модуль обробки даних обробляє отримані дані, включаючи фільтрацію, агрегування, нормалізацію та перетворення даних у потрібний формат. Він забезпечує підготовку даних для подальшого аналізу та візуалізації.

Модуль аналізу даних виконує різні види аналізу, такі як статистичний аналіз, технічний аналіз та прогнозування. Цей модуль може видавати результати у вигляді текстових звітів, показників тенденцій, максимальних і мінімальних значень, критичних змін цін, найбільших угод тощо.

Модуль візуалізації відповідає за створення графіків, діаграм, гістограм та інших візуальних представлень даних. Він надає користувачам інструменти для інтерактивної роботи з візуалізаціями, дозволяючи змінювати параметри та налаштування для детального аналізу.

Інтерфейс користувача забезпечує зручний доступ до всіх функцій інструменту. Він дозволяє користувачам вибирати необхідні функції, налаштовувати параметри аналізу та візуалізації, а також переглядати результати у зрозумілому вигляді.

Компоненти інструменту взаємодіють між собою через чітко визначені інтерфейси та протоколи обміну даними. Основні аспекти взаємодії включають:

Кожна функція записана в окремому файлі і може використовуватися окремо. Доступ до них здійснюється через запити як до системних функцій. Користувачі роблять запит з певними даними, а функція повертає результат своєї роботи. Наприклад, функція отримання кількості токенів, отримавши на вхід назву крипто-валюти, поверне кількість токенів цієї крипто-валюти.

Функції можуть отримувати та повертати дані різних типів. Це можуть бути числові дані, рядки тексту, масиви або складніші структури даних. Це дозволяє гнучко обробляти та передавати інформацію між різними модулями.

Для передачі даних та виклику функцій будуть використовуватися текстові файли, у яких записані назви функцій та параметри для їх роботи. Це дозволяє легко змінювати та налаштовувати виклики функцій без необхідності змінювати код.

Кожен модуль може викликати функції інших модулів за необхідності. Наприклад, модуль візуалізації може звертатися до модуля аналізу даних для отримання результатів аналізу, які потім відображаються у вигляді графіків та діаграм.

Для архітектури інструменту було обрано монолітний підхід з елементами мікросервісної архітектури.

Цей підхід було обрано з кількох причин:

Монолітна архітектура дозволяє користувачам легко робити запити щодо певної інформації про крипто-активи та швидко отримувати результати. Це забезпечує зручність використання інструменту в будь-якому місці.

Незважаючи на монолітний підхід, деякі елементи мікросервісної архітектури дозволяють використовувати різні функції незалежно одна від одної, що знижує навантаження на систему та підвищує її гнучкість.

Така архітектура дозволяє ефективно виконувати аналіз та візуалізацію даних, забезпечуючи високу продуктивність та надійність роботи інструменту.

Загалом, архітектура інструменту для візуалізації та аналізу крипто-активів є комбінацією монолітного та мікросервісного підходів, що забезпечує зручність використання, гнучкість та високу продуктивність.

2.2 Розробка функціональної моделі

Для розробки функціональної моделі інструменту для візуалізації та аналізу крипто-активів було використано UML діаграми, які дозволяють чітко візуалізувати основні функції та процеси в системі [14]. Нижче на рисунку 2.1 представлено діаграму варіантів використання (Use Case Diagram), що описує взаємодію користувачів з інструментом.

Use Case Diagram ілюструє, як користувачі взаємодіють з різними функціями інструменту. Основні варіанти використання включають збір даних, аналіз даних, візуалізацію даних та перегляд звітів.

У ролі актора виступає користувач який у свою чергу може бути інвестором, трейдером, аналітиком або тощо.

Варіанти використання (Use Cases):

1. Збір даних: Отримання поточних курсів, збір історичних даних.

2. Аналіз даних: Статистичний аналіз, технічний аналіз (включаючи ковзне середнє).

3. Візуалізація даних: Створення графіків, діаграм, гістограм.

4. Перегляд звітів: Перегляд текстових звітів, перегляд візуалізацій.

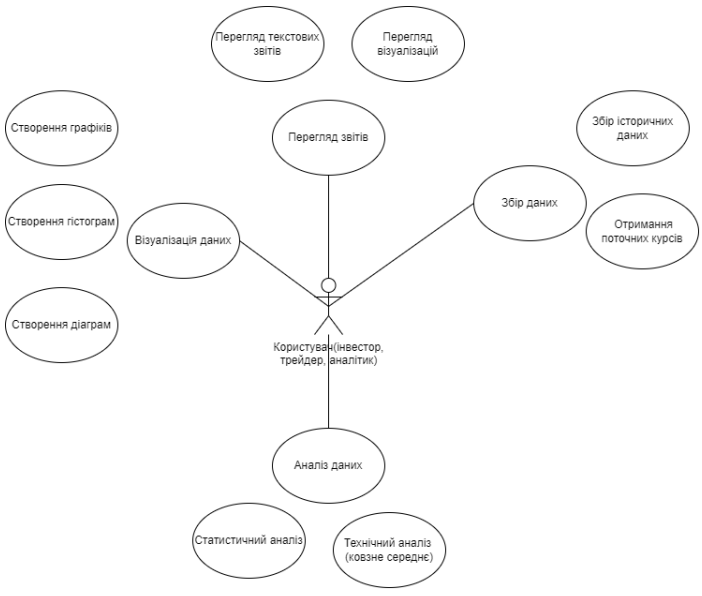

Рисунок 2.1 – Діаграма варіантів використання для розроблюваного інструменту

Нижче розглянемо і пояснено зв'язки створеної діаграми.

Користувач:

- Має доступ до всіх основних варіантів використання: Збір даних, Аналіз даних, Візуалізація даних, Перегляд звітів.
- Може вибирати, які саме варіанти використання застосувати (наприклад, переглядати лише візуалізацію, або лише графіки і гістограми).

Збір даних:

- Включає Отримання поточних курсів і Збір історичних даних. Ці варіанти можуть бути виконані незалежно один від одного.

Аналіз даних:

- Включає Статистичний аналіз і Технічний аналіз (ковзне середнє). Користувач може виконати один або обидва види аналізу залежно від потреб.

Візуалізація даних:

- Включає Створення графіків, Створення діаграм і Створення гістограм. Користувач може вибрати один, декілька або всі варіанти для створення візуалізацій.

Перегляд звітів:

- Включає Перегляд текстових звітів і Перегляд візуалізацій. Користувач може переглядати лише текстові звіти, лише візуалізації або обидва варіанти.

2.3 Інтерфейс користувача та візуальні компоненти

Інтерфейс користувача є критичним компонентом будь-якого програмного забезпечення, оскільки забезпечує взаємодію між користувачем і системою. У даному розділі описані основні сторінки інтерфейсу користувача та їхні функціональні можливості для інструменту візуалізації та аналізу крипто-активів.

Основні сторінки програми:

1. стартова сторінка;

2. основне робоче поле;

3. сторінка вибору функцій;

4. довідкова сторінка;

5. сторінка задання параметрів;

На стартовій сторінці користувач має можливість вибрати один з трьох варіантів: створити новий проект, відкрити проект з локальної папки або відкрити демонстраційний файл. Також є можливість швидко відкрити останні відкриті файли аналізу або переглянути за допомогою випадаючого списку усі створені файли. Приблизний вигляд сторінки можна переглянути на рисунку 2.2.

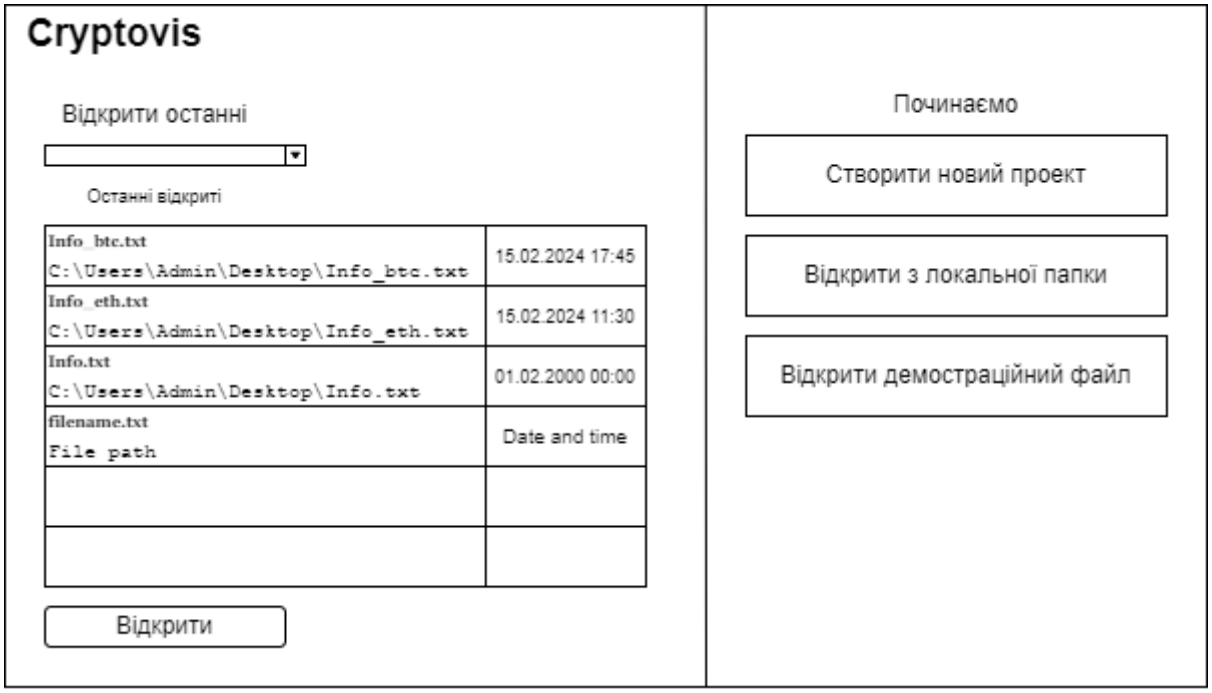

Рисунок 2.2 – Макет стартової сторінки програми

Саме вікно буде мати такі компоненти:

- Створити новий проект: відкриває сторінку вибору функцій.
- Відкрити з локальної папки: дозволяє користувачу вибрати наявний проект для відкриття.
- Відкрити демонстраційний файл: відкриває демонстраційний проект для ознайомлення.
- Відкрити останні: дозволяє швидко відкрити останні відкриті файли аналізу.

- Випадаючий список: дає змогу переглянути усі створені файли і вибрати необхідний для відкриття.

Основне робоче поле містить різні вкладки для відкритих проектів. Користувач може відкривати, змінювати функції та параметри, а також оновлювати інформацію про крипто-активи. На вкладках буде розміщуватись візуалізація даних, а в текстовому полі, відповідно до вкладок, текстова інформація. Побачити умовний вигляд основного робочого поля можна на рисунку 2.3.

Рисунок 2.3 – Макет основного робочого поля

З рисунка можна побачити що будуть розміщені такі елементи:

- Новий: створення нового проекту.
- Змінити функції: відкриває сторінку вибору функцій для редагування.
- Оновити інформацію: оновлює дані аналізу.
- Змінити дані: дозволяє змінити параметри аналізу.
- Вкладки: відображають різні проекти з візуалізацією даних.
- Текстове поле: відображає текстову інформацію відповідно до вибраної вкладки.

На сторінці вибору функцій користувач може вибрати функції, які він хоче використовувати в проекті. Доступні функції відображаються у вигляді

чекбоксів, і користувач може вибрати одну або декілька функцій. У нижній частині сторінки розташоване текстове поле, яке відображає усі вибрані функції.

Умовний вигляд цього вікна продемонстровано на рисунку 2.4.

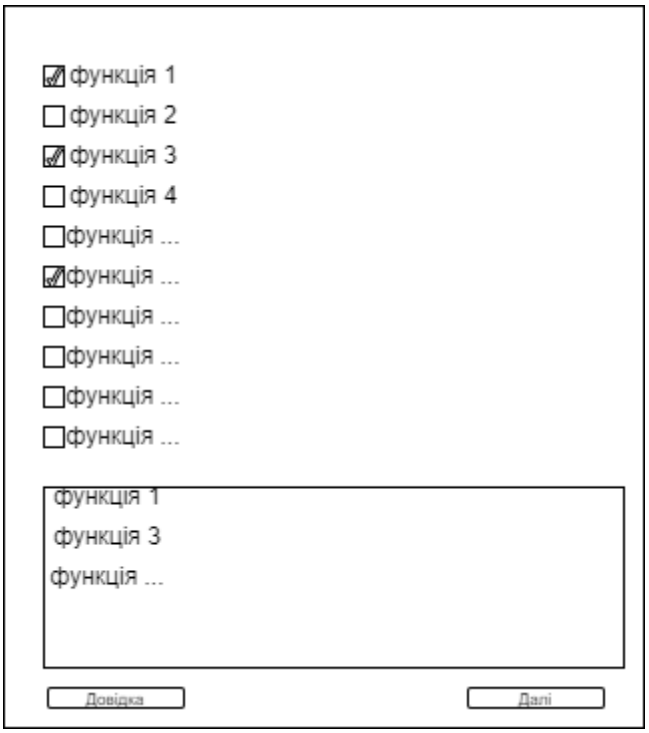

Рисунок 2.4 – Макет вікна вибору функцій

Як видно з рисунку у вікна будуть такі основні елементи:

- Список функцій: відображає всі доступні функції, які користувач може вибрати.
- Довідка: відкриває окреме вікно для перегляду опису функцій.
- Текстове поле вибраних функцій: відображає перелік функцій, обраних користувачем.
- Кнопка «Далі»: переходить на сторінку задання параметрів для вибраних функцій.

Довідкова сторінка надає користувачу інформацію про всі можливі функції та короткий опис до них. Це допомагає користувачу краще зрозуміти, які функції доступні і як вони можуть бути використані.

На сторінці задання параметрів користувач вводить необхідні параметри для вибраних функцій. Після введення параметрів користувач може запустити аналіз і переглянути результати.

Ці сторінки та їх компоненти забезпечують зручний і інтуїтивно зрозумілий інтерфейс для користувача, дозволяючи ефективно взаємодіяти з інструментом для візуалізації та аналізу крипто-активів.

2.4 Інтеграція з зовнішніми API для отримання даних про крипто-активи

Для отримання даних про крипто-активи ваш інструмент інтегрується з кількома зовнішніми API, які забезпечують доступ до різноманітної інформації про крипто-валюти. Основні API, які використовуються:

CoinGecko API використовується для отримання широкого спектру даних про крипто-валюти, включаючи поточні курси, історичні дані, обсяги торгів та інші метрики. Безкоштовний доступ забезпечує більшість необхідних даних, однак у деяких випадках можливі обмеження [15].

Binance API використовується для отримання даних про ринкові торги, поточні ціни та обсяги торгів з однієї з найбільших крипто-валютних бірж. Забезпечує доступ до детальних даних торгівлі в реальному часі.

CoinMarketCap API використовується як резервне джерело для отримання даних, коли CoinGecko API не може надати інформацію в рамках безкоштовного доступу. Платформа надає детальні ринкові дані, історичні ціни, рейтинги крипто-валют та іншу релевантну інформацію.

Etherscan API використовується для отримання специфічних даних про блокчейн Ethereum, таких як пошук найбільших тримачів крипто-валют. Забезпечує детальну інформацію про транзакції, баланси адрес та інші дані, пов'язані з блокчейном Ethereum [16].

API Alternative.me використовується для отримання індексу страху та жадібності, оскільки його обчислення є занадто важким для реалізації в поточному проєкті.

Для підключення до API використовується бібліотека requests, яка дозволяє надсилати HTTP-запити до зовнішніх API та обробляти відповіді. Для доступу до Binance API використовується також бібліотека ccxt, яка спрощує роботу з біржовими API.

Процес підключення:

- CoinGecko API: Надсилання HTTP-запитів за допомогою бібліотеки requests. Не потребує створення власного API ключа для безкоштовного доступу.
- Binance API: Використання бібліотеки ccxt для підключення та надсилання запитів. Не потребує створення власного API ключа для безкоштовного доступу.
- CoinMarketCap API: Надсилання HTTP-запитів за допомогою бібліотеки requests. Потребує створення власного API ключа для доступу.
- Etherscan API: Надсилання HTTP-запитів за допомогою бібліотеки requests. Потребує створення власного API ключа для доступу.
- Alternative.me API: Надсилання HTTP-запитів за допомогою бібліотеки requests для отримання індексу страху та жадібності.

API повертають дані у форматі JSON, що дозволяє легко обробляти і використовувати ці дані в програмі. Основні типи даних, які отримуються з API, включають поточні курси крипто-валют, історичні дані, обсяги торгів, дані про найбільших тримачів крипто-валют, а також індекс страху та жадібності.

Отримані дані обробляються для подальшого використання у візуалізації та аналізі. Основні етапи обробки включають фільтрацію, нормалізацію та зберігання даних. Фільтрація видаляє нерелевантні або зайві дані, нормалізація приводить дані до єдиного формату для зручності подальшої обробки, а зберігання забезпечує збереження оброблених даних у внутрішньому сховищі для подальшого використання.

Для перевірки коректності отриманих даних використовуються перевірка форматів і значень. Перевірка форматів включає перевірку правильності форматів даних, наприклад JSON. Перевірка значень забезпечує адекватність отриманих значень, наприклад перевірка, чи не є значення ціни негативним або занадто великим.

Для забезпечення безпеки при роботі з зовнішніми API використовуються API ключі для доступу до CoinMarketCap API та Etherscan API. Ключі зберігаються у безпечному місці і не передаються у відкритому вигляді. Кожен API має свої обмеження та квоти на використання. Наприклад, CoinGecko API і Binance API надають безкоштовний доступ з обмеженням на кількість запитів на хвилину, CoinMarketCap API потребує API ключ і має різні квоти залежно від плану підписки, а Etherscan API також потребує API ключ і має обмеження на кількість запитів на день.

Запити на оновлення даних можуть виконуватися вручну користувачем при потребі або автоматично при новому відкритті файлу з аналізом.

Ці заходи забезпечують надійний і безперервний доступ до актуальних даних про крипто-активи.

2.5 Висновок до другого розділу

У другому розділі кваліфікаційної роботи було розглянуто проектування інструменту для візуалізації та аналізу крипто-активів. Було визначено архітектуру інструменту, яка складається з кількох модулів, кожен з яких виконує певну функцію, що забезпечує модульність та розширюваність системи. Модулі включають збір даних, аналіз даних, візуалізацію даних та перегляд звітів, кожен з яких тісно взаємодіє один з одним для забезпечення комплексного аналізу крипто-активів.

Описано інтерфейс користувача та візуальні компоненти, які забезпечують зручний та інтуїтивно зрозумілий доступ до функцій інструменту. Основні сторінки інтерфейсу включають стартову сторінку, основне робоче поле,

сторінку вибору функцій, довідкову сторінку та сторінку задання параметрів. Особлива увага приділялася забезпеченню зручності використання, що дозволяє користувачу легко взаємодіяти з системою, налаштовувати параметри аналізу та отримувати необхідні дані.

Також було розглянуто інтеграцію з зовнішніми API для отримання даних про крипто-активи. Використання API CoinGecko, Binance, CoinMarketCap, Etherscan та Alternative.me забезпечує доступ до широкого спектру даних про крипто-валюти, включаючи поточні курси, історичні дані, обсяги торгів, дані про найбільших тримачів крипто-валют та індекс страху та жадібності. Було описано механізми підключення до API, обробки отриманих даних та забезпечення їхньої безпеки.

Результати, отримані у цьому розділі, є основою для подальшої розробки інструменту, забезпечуючи його функціональність, зручність використання та надійність. Інтеграція з зовнішніми API та ефективна обробка даних забезпечують своєчасний доступ до актуальної інформації, що є критично важливим для аналізу та прийняття рішень на ринку крипто-активів.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ ІНСТРУМЕНТУ ДЛЯ ВІЗУАЛІЗАЦІЇ ТА АНАЛІЗУ КРИПТО-АКТИВІВ

3.1 Вибір та використання бібліотек для розробки

Для розробки інструменту для візуалізації та аналізу крипто-активів було обрано кілька основних бібліотек, які забезпечують різноманітну функціональність та спрощують процес розробки. Використання цих бібліотек дозволило створити ефективний, зручний у використанні та функціонально багатий інструмент. Нижче наведено перелік використаних бібліотек та їх призначення:

1. Tkinter: Використовується для створення графічного інтерфейсу користувача (GUI). Забезпечує зручні інструменти для розробки вікон, кнопок, меню та інших елементів інтерфейсу. tkinter є стандартною бібліотекою для створення GUI в Python, що робить її легко доступною і добре задокументованою [17].

2. Os: Забезпечує функції для взаємодії з операційною системою, такі як робота з файловою системою, отримання системної інформації та інші операції на рівні операційної системи. Використовується для забезпечення сумісності з різними операційними системами та для управління файлами і директоріями.

3. Datetime: Використовується для роботи з датами та часом. Дозволяє виконувати операції з датами, такі як обчислення інтервалів, форматування дат та часу. Це особливо важливо для обробки історичних даних та аналізу часових рядів крипто-валют.

4. Requests: Бібліотека для надсилання HTTP-запитів. Використовується для отримання даних з зовнішніх API, таких як CoinGecko, CoinMarketCap, Etherscan та Alternative.me. requests є однією з найпопулярніших бібліотек для роботи з HTTP у Python завдяки своїй простоті та потужності [18].

5. Pandas: Потужна бібліотека для обробки та аналізу даних. Використовується для зручної роботи з табличними даними, їх фільтрації,

агрегації та аналізу. pandas забезпечує ефективні інструменти для роботи з великими наборами даних і підтримує різні формати вхідних даних, такі як CSV, Excel, SQL тощо [19].

6. Ccxt: Бібліотека для роботи з крипто-валютними біржами. Використовується для отримання даних про ринкові торги з Binance API. ccxt підтримує велику кількість крипто-валютних бірж і надає уніфікований інтерфейс для роботи з ними [20].

7. Matplotlib: Бібліотека для створення графіків та візуалізацій. Використовується для візуалізації даних, таких як графіки цін, гістограми та інші візуальні компоненти. matplotlib є однією з найбільш використовуваних бібліотек для створення графіків у Python і підтримує широкий спектр налаштувань та типів графіків [21].

Функціональність цих бібліотек є повністю задовільною і не викликає нарікань.

Створення графічного інтерфейсу користувача забеспечується бібліотекою tkinter. Її використовується для створення вікон, кнопок, меню та інших елементів GUI. Завдяки tkinter користувач має можливість взаємодіяти з інструментом через зручний і зрозумілий інтерфейс, що значно підвищує зручність використання.

Бібліотека os дозволяє виконувати операції з файлами та взаємодіяти з операційною системою, наприклад, відкривати файли проектів або зберігати результати аналізу. Ця бібліотека забезпечує сумісність з різними операційними системами та надає зручні інструменти для роботи з файловою системою.

Для обробки дат і часу використовується datetime, що необхідно для аналізу історичних даних крипто-валют. Ця бібліотека дозволяє виконувати складні операції з датами, такі як обчислення інтервалів, порівняння дат та форматування часу.

Requests дозволяє надсилати HTTP-запити до зовнішніх API та отримувати відповіді у форматі JSON. Це забезпечує доступ до необхідних даних про крипто-

валюти, таких як поточні курси, історичні дані та індекс страху і жадібності. Завдяки requests можна легко інтегрувати зовнішні джерела даних у програму.

Pandas забезпечує потужні інструменти для обробки та аналізу табличних даних, таких як історичні ціни крипто-валют. Використання pandas дозволяє ефективно маніпулювати великими наборами даних, виконувати фільтрацію, агрегацію та аналіз даних.

Бібліотека ccxt спрощує роботу з API крипто-валютних бірж, таких як Binance, дозволяючи отримувати дані про торги, обсяги та інші ринкові показники. Ця бібліотека забезпечує уніфікований інтерфейс для роботи з різними біржами, що полегшує інтеграцію та доступ до ринкових даних.

matplotlib використовується для створення графіків, гістограм та інших візуальних компонентів, що необхідні для представлення даних користувачеві. Завдяки matplotlib можна створювати різноманітні візуалізації, які допомагають краще зрозуміти та проаналізувати дані.

Бібліотека tkinter була обрана завдяки своїй простоті використання та інтеграції з Python. Вона забезпечує необхідні інструменти для створення графічного інтерфейсу користувача, що дозволяє швидко і ефективно розробляти зручні GUI-додатки.

Бібліотека os є вбудованою бібліотекою Python, яка забезпечує необхідні функції для роботи з файловою системою та операційною системою. Її використання забезпечує сумісність з різними операційними системами та спрощує виконання операцій на рівні ОС.

Бібліотека datetime була обрана завдяки своїм потужним інструментам для роботи з датами та часом. Вона дозволяє легко обробляти і маніпулювати датами, що є важливим для аналізу історичних даних крипто-валют.

Бібліотека requests є однією з найпопулярніших бібліотек для роботи з HTTP у Python завдяки своїй простоті та потужності. Вона забезпечує зручні інструменти для надсилання HTTP-запитів та отримання відповідей, що робить її ідеальним вибором для інтеграції з зовнішніми API.

Бібліотека pandas була обрана завдяки своїй потужності та гнучкості у роботі з табличними даними. Вона дозволяє ефективно маніпулювати великими наборами даних та забезпечує багатий набір функцій для аналізу даних.

Бібліотека ccxt забезпечує уніфікований інтерфейс для роботи з крипто-валютними біржами, що полегшує інтеграцію та доступ до ринкових даних. Вона підтримує велику кількість бірж та надає зручні інструменти для роботи з ними.

Бібліотека matplotlib була обрана завдяки своїй популярності та широкому спектру функцій для візуалізації даних. Вона дозволяє створювати різноманітні графіки та візуалізації, які допомагають краще зрозуміти та проаналізувати дані.

Інтеграція бібліотек у проєкт пройшла гладко, завдяки гарній документації та підтримці спільноти для кожної з обраних бібліотек. Важливою перевагою стала можливість легкої інтеграції між різними бібліотеками, що дозволило створити потужний та зручний у використанні інструмент. Наприклад, бібліотека pandas чудово інтегрується з matplotlib, що дозволяє легко створювати графіки на основі табличних даних. Бібліотека requests без проблем працює з даними у форматі JSON, що спрощує обробку відповідей від зовнішніх API.

Перед вибором використовуваних бібліотек були розглянуті інші варіанти, такі як PyQt для створення GUI, NumPy для обробки даних та Seaborn для візуалізації. Проте, обрані бібліотеки були вибрані через їхню простоту у використанні, хорошу інтеграцію з іншими компонентами проєкту та підтримку необхідної функціональності. Наприклад, PyQt є потужною бібліотекою для створення GUI, але вона має більш складний синтаксис і вимагає додаткового часу для освоєння, тоді як tkinter є простішою і більш інтегрованою у Python.

Цей вибір бібліотек забезпечує надійну основу для розробки та підтримки інструменту для візуалізації та аналізу крипто-активів, дозволяючи ефективно обробляти та візуалізувати дані, а також забезпечувати зручний інтерфейс для користувачів. Завдяки використанню обраних бібліотек можна легко розширювати функціональність інструменту та адаптувати.

3.2 Розробка back-end частини на Python

Back-end частина інструменту для візуалізації та аналізу крипто-активів виконує ключові функції, такі як збір, обробка та аналіз даних. Основними компонентами back-end частини є окремі функції, які видають різні результати і знаходяться у окремих файлах. Також є два більш глобальних файли, що здійснюють виклик функцій і вивід текстових результатів та відображення візуалізацій.

Програма складається з декількох ключових компонентів:

Окремі функції: Кожна функція виконує певну задачу, наприклад, збір даних з API, обробка отриманих даних, аналіз даних або візуалізація результатів.

Глобальні файли: Ці файли відповідають за виклик функцій, координацію їх роботи та вивід результатів. Один файл займається виводом текстових результатів, а інший - відображенням візуалізацій.

Основні функції back-end частини включають:

Збір даних: Виконується шляхом надсилання запитів до зовнішніх API для отримання необхідної інформації про крипто-активи.

Обробка даних: Отримані дані проходять через різні етапи обробки, включаючи фільтрацію, нормалізацію та підготовку для аналізу.

Аналіз даних: Виконання різних видів аналізу, таких як статистичний аналіз, технічний аналіз та інші методи для виявлення тенденцій та інсайтів.

Вивід результатів: Текстові результати та візуалізації передаються для відображення користувачеві.

Взаємодія між різними компонентами back-end частини здійснюється за допомогою запитів. Якщо якась функція потребує дані від іншої, вона здійснює прямий запит і отримує необхідну інформацію. Це забезпечує гнучкість та мобільність системи, дозволяючи функціям взаємодіяти без обмежень.

Кожна функція має свій власний запит до API і одразу обробляє отримані дані потрібним чином. Це робиться для того, щоб не передавати велику кількість даних на великі відстані всередині програми, хоча цього не вдалось уникнути для

функцій візуалізації. Наприклад, функція для отримання поточних курсів крипто-валют здійснює запит до CoinGecko API, отримує дані у форматі JSON, обробляє їх і повертає у вигляді, зручному для подальшого використання.

Дані, отримані з API, обробляються одразу після отримання, щоб зменшити навантаження на систему та уникнути зайвих передач даних між функціями. Для цього використовуються заголовки з JSON файлів, які дозволяють обробляти лише потрібні дані. Обробка даних здійснюється за допомогою класичних бібліотек pandas та matplotlib.

pandas використовується для роботи з табличними даними, їх фільтрації, нормалізації та агрегації. Це дозволяє ефективно маніпулювати великими наборами даних і підготувати їх для подальшого аналізу та візуалізації.

matplotlib використовується для створення графіків та інших візуалізацій на основі оброблених даних. Це дозволяє наочно представити результати аналізу користувачеві.

Для забезпечення мобільності дані зберігаються лише тимчасово і лише у випадках, коли це необхідно для виконання конкретних функцій. Основний акцент робиться на обробку та аналіз даних у режимі реального часу, що дозволяє користувачам отримувати актуальні результати без необхідності зберігання великих обсягів даних.

Для зберігання файлів аналізу використовуються текстові файли, в яких записані функції, які потрібно виконати, та параметри до цих функцій. Це дозволяє легко зберігати і передавати проекти аналізу, а також забезпечує простоту редагування і управління цими файлами.

Ці заходи забезпечують надійний і безперервний доступ до актуальних даних про крипто-активи, а також їх ефективну обробку та аналіз, що є критично важливим для роботи інструменту.

3.2.1 Функції отримання та обробки даних

Функції отримання даних включають запити до API для отримання різних типів інформації, таких як поточні ціни крипто-валют, історичні дані, обсяги торгів, індекси страху та жадібності тощо. Дані, отримані з API, обробляються та форматуються відповідним чином, щоб їх можна було легко використовувати для аналізу та візуалізації.

Однією з найпростіших функцій для отримання даних - функцію отримання ціни крипто-валюти за її назвою. Ця функція здійснює запит до CoinGecko API і повертає поточну ціну вказаної крипто-валюти у доларах США. Функція get_price(name) приймає назву крипто-валюти як параметр і повертає її поточну ціну у доларах США. Якщо назва крипто-валюти знайдена, функція повертає рядок з ціною у форматі «X USD». Якщо крипто-валюта не знайдена, функція повертає повідомлення «Price not found».

Практичне виконання цієї функції в коді наведено на лістингу 3.1.

Лістинг 3.1 – Отримання символу крипто-валюти за її назвою

```
import requests
def       get_price(name):                                  url       =
f»https://api.coingecko.com/api/v3/simple/price»
    params = {
        «ids»: name.lower(),
        «vs_currencies»: «usd»
    }
    response = requests.get(url, params=params)
    data = response.json()
    if name.lower() in data:
        return f»{data[name.lower()]['usd']} $»
    else:
        return «Price not found»
```

Спершу було імпортовано бібліотеку requests, яка дозволяє здійснювати HTTP-запити. Це необхідно для взаємодії з API CoinGecko. Спершу встановлюється URL для запиту до CoinGecko API, що вказує на ресурс, який надає просту цінову інформацію. Далі визначаємо параметри запиту, які

включають назву крипто-валюти у нижньому регістрі та валюту, в якій повертається ціна.

Функція здійснює GET-запит до API з вказаними параметрами, щоб отримати необхідну інформацію. Отримана відповідь від API у форматі JSON зберігається в змінну data. Далі ми перевіряємо, чи містить ця відповідь інформацію про запитану крипто-валюту. Якщо назва крипто-валюти знайдена у відповіді, функція повертає її ціну у форматі «X USD». Якщо ж крипто-валюта не знайдена, функція повертає повідомлення «Price not found».

Ця функція надає користувачеві простий спосіб отримати поточну ціну крипто-валюти за її назвою. Це може бути корисно для швидкої перевірки вартості активів у режимі реального часу, що є важливою складовою прийняття інвестиційних рішень. Наприклад, користувач може ввести назву крипто-валюти «bitcoin» і отримати поточну ціну у доларах США.

Окрім функції отримання ціни крипто-валюти за її назвою, є ще кілька інших важливих функцій, які використовуються для збору інформації про крипто-валюти. Ці функції включають отримання загальної кількості токенів, ринкової капіталізації та інформації про біржі, на яких торгується крипто-валюта. Всі ці функції працюють за одним принципом, відрізняючись лише запитами до API та обмеженнями щодо даних, які потрібно отримати.

Усі функції використовують однаковий підхід до збору даних. Вони здійснюють HTTP-запит до зовнішнього API, отримують дані у форматі JSON, обробляють їх і повертають користувачеві у зручному вигляді. Кожна функція виконує специфічний запит до API для отримання конкретного типу даних. Відмінності між функціями полягають у типі запиту до API та обмеженнях щодо даних, які потрібно отримати та обробити.

Функція для отримання загальної кількості токенів крипто-валюти здійснює запит до API і повертає інформацію про кількість токенів у зверненні. Вона обробляє відповідь API та надає користувачеві точні дані про загальну кількість випущених токенів, що може бути корисним для оцінки вартості активів і прогнозування ринкових тенденцій.

Функція для отримання ринкової капіталізації крипто-валюти здійснює запит до API і повертає поточну ринкову капіталізацію у доларах США. Вона дозволяє користувачам отримувати актуальну інформацію про загальну ринкову вартість крипто-валюти, що є важливим показником її популярності та стабільності на ринку.

Функція для отримання інформації про біржі, на яких торгується крипто-валюта, здійснює запит до API і повертає список бірж. Ця інформація допомагає користувачам дізнатися, де вони можуть придбати або продати крипто-валюту, а також отримати уявлення про обсяги торгів і ліквідність на різних платформах.

Ці функції забезпечують користувачам доступ до важливої інформації про крипто-валюти, такої як загальна кількість токенів, ринкова капіталізація та інформація про біржі. Незважаючи на те, що функції мають різні запити до API та обмеження щодо даних, принцип їхньої роботи залишається однаковим, що забезпечує консистентність та зручність використання.

Хоча ці функції досить важливі є багато різних корисних функцій, наприклад get_top_wallets(crypto_name) дозволяє отримати інформацію про топ-10 гаманців, які тримають найбільшу кількість певної крипто-валюти. Ця функція використовує API Etherscan для отримання даних про транзакції та баланси гаманців. Основна мета цієї функції - надати користувачеві інформацію про основних власників крипто-валюти, що може бути корисним для аналізу ринкової активності та оцінки розподілу активів.

Основні кроки функції:

- Отримання символу крипто-валюти: Функція використовує допоміжну функцію get_symbol, щоб отримати символ крипто-валюти за її назвою.
- Отримання контрактної адреси: Використовується функція get_contract_addresses, щоб отримати контрактну адресу крипто-валюти на платформі Ethereum.
- Збір подій передачі токенів: Функція get_transfer_events здійснює запит до Etherscan API для отримання даних про всі події передачі токенів для вказаної контрактної адреси.

- Розрахунок балансів гаманців: Функція обробляє отримані дані про події і розраховує баланси для кожного гаманця. Отримані дані про транзакції обробляються, щоб розрахувати баланси гаманців на основі подій передачі токенів.
- Сортування та вибір топ-10 гаманців: Гаманці сортуються за балансом, і обираються топ-10 гаманців з найбільшими балансами. Баланси гаманців сортуються у порядку зменшення для виявлення топ-10 гаманців з найбільшими балансами.
- Форматування результатів: Результати форматуються у вигляді рядків, що містять адреси гаманців та їх баланси, і повертаються користувачеві. Адреси гаманців і їх баланси форматуються у вигляді рядків і повертаються користувачеві.

Переглянути основну частину коду функції можна на лістингу 3.2

Лістинг 3.2 – Отримання топу гаманців для певної крипто-валюти

```
import requests
from get_symbol import get_symbol
from get_contract_addresses import get_contract_addresses

ETHERSCAN_API_KEY = 'your_etherscan_api_key'

def      get_transfer_events(contract_address,      start_block=0,
end_block=99999999):
    url                                                          =
f»https://api.etherscan.io/api?module=account&action=tokentx&contr
actaddress={contract_address}&startblock={start_block}&endblock={e
nd_block}&sort=asc&apikey={ETHERSCAN_API_KEY}»
    response = requests.get(url)
    data = response.json()
    if data['status'] == '1':
        return data['result']
    else:
        return []
```

Ця функція є важливим інструментом для аналізу ринкової активності та оцінки ризиків, пов'язаних з концентрацією активів. Вона дозволяє користувачам отримувати важливу інформацію про найбільших власників крипто-валют, що сприяє прийняттю більш обґрунтованих рішень на ринку крипто-активів [22].

Ще однією цікавою функцією є отримання найбільших угод за останній місяць вона надає важливу інформацію про значні транзакції, що відбулися за останній місяць. Це дозволяє користувачам зрозуміти ринкові тенденції та виявити великі угоди, що можуть вплинути на ринок.

Функція get_largest_trades(crypto_name) приймає назву крипто-валюти як параметр і повертає список найбільших угод за останній місяць. Вона використовує API біржі Binance для отримання даних про угоди і сортує їх за обсягом, щоб виявити найбільші.

Основні кроки пошуку найбільших угод за останній місяць:

- Отримання символу крипто-валюти: Функція використовує допоміжну функцію get_symbol, щоб отримати символ крипто-валюти за її назвою.
- Встановлення діапазону дат: Визначаються початкова і кінцева дати для періоду в 30 днів.
- Збір даних про угоди: Функція здійснює запит до API Binance для отримання даних про угоди за вказаний період.
- Фільтрація та сортування угод: Отримані дані сортуються за обсягом угод, щоб виявити найбільші.
- Форматування результатів: Результати форматуються у вигляді рядків, що містять деталі угод, такі як дата, тип угоди, обсяг, ціна та загальна вартість угоди.

На додаток до вже розглянутих функцій, існує кілька інших важливих функцій, які забезпечують різноманітну інформацію про крипто-валюти. Ці функції дозволяють користувачам отримувати ключові дані для аналізу та прийняття рішень. Нижче наведено детальний опис цих функцій.

Функція визначення відсоткового відношення long/short надає інформацію про відсоткове відношення довгих і коротких позицій для конкретної крипто-валюти. Відсоткове відношення long/short позицій дозволяє оцінити ринкові настрої серед трейдерів та інвесторів, що може бути корисним для прийняття торгових рішень.

Функція отримання посилань на соціальні мережі повертає посилання на офіційні сторінки крипто-валюти в соціальних мережах. Соціальні мережі є важливим джерелом інформації та новин про крипто-валюти, і ця функція допомагає користувачам легко знаходити офіційні канали комунікації проекту.

Функція отримання контрактних адрес повертає контрактні адреси крипто-валюти, що є важливим для користувачів, які працюють з токенами на платформі Ethereum або інших блокчейнах. Контрактні адреси дозволяють відстежувати транзакції та інші дії, пов'язані з крипто-валютою.

Функція отримання максимального та мінімального значень ціни за період надає інформацію про максимальну та мінімальну ціну крипто-валюти за вказаний період. Ці дані є важливими для технічного аналізу та визначення ключових рівнів підтримки та опору на ринку.

Функція отримання дат різких змін ціни визначає дати значних змін ціни крипто-валюти. Вона допомагає користувачам ідентифікувати періоди високої волатильності, що може бути важливим для ризик-менеджменту та стратегій торгівлі.

Функція визначення тенденцій руху ціни аналізує історичні дані про ціни крипто-валюти та визначає основні тенденції її руху. Це дозволяє користувачам зрозуміти, чи є ринок бичачим (зростаючим) або ведмежим (спадним), і приймати відповідні інвестиційні рішення.

Функція визначення критичних значень ціни визначає ключові рівні цін, які можуть бути важливими для аналізу, наприклад, рівні підтримки та опору. Визначення цих рівнів допомагає трейдерам та інвесторам приймати більш обґрунтовані рішення щодо входу та виходу з ринку.

Функція отримання обсягів торгів за період повертає дані про обсяги торгів крипто-валюти за вказаний період. Обсяги торгів є важливим показником ринкової активності та ліквідності, що допомагає оцінити попит і пропозицію на ринку.

Усі ці функції працюють за схожим принципом, що включає запити до API, обробку отриманих даних та повернення результатів у зручному форматі. Навіть

якщо конкретні функції не були детально розглянуті, їхній принцип роботи є аналогічним вже описаним.

Однак не слід забувати що користувач зможе у будь-який момент може звернутися до довідки функцій для отримання додаткової інформації про кожну з них. Довідка містить опис функцій, їхнє призначення та приклади використання, що допоможе користувачеві зрозуміти, як саме працює кожна функція і які дані вона повертає.

3.2.2 Функції візуалізації даних

Візуалізація даних є ключовим елементом аналізу крипто-активів, оскільки вона дозволяє користувачам отримувати візуальне уявлення про ринкові тенденції та динаміку цін. У цьому підрозділі будуть розглянуті основні функції візуалізації, що надає наш інструмент. Вони допомагають користувачам краще розуміти ринкові сигнали та приймати обґрунтовані інвестиційні рішення.

Для детального розгляду було обрано три основні функції візуалізації: візуалізація графіку ціни з ковзним середнім, візуалізація об'єму торгів за день та візуалізація індексу страху та жадібності. Ці функції були вибрані тому, що вони демонструють різноманітність можливостей нашого інструменту та включають ключові аспекти аналізу ринку:

- Візуалізація графіку ціни з ковзним середнім допомагає користувачам згладжувати короткострокові коливання та ідентифікувати середньострокові та довгострокові тренди.
- Візуалізація об'єму торгів за день надає інформацію про ринкову активність та ліквідність крипто-валюти.
- Візуалізація індексу страху та жадібності відображає емоційний стан ринку, що є важливим індикатором для прийняття інвестиційних рішень.

Ось принцип роботи для коду що буде відображати графік ціни з ковзним середнім:

1. Збір даних: Виконується запит до API для отримання історичних даних про ціни крипто-валюти за заданий період.

2. Обробка даних: Дані перетворюються у формат, зручний для побудови графіку. Дати конвертуються у відповідний формат, а ціни додаються до списку.

3. Обчислення ковзного середнього: Для кожного дня обчислюється середнє значення ціни за вказаний період (наприклад, 7 днів).

4. Побудова графіку: Використовується бібліотека matplotlib для побудови графіку, на якому відображається як сама ціна, так і ковзне середнє.

Сам код можна переглянути на лістингу 3.3

Лістинг 3.3 – Створення графіку з ковзним середнім

```
import requests
import matplotlib.pyplot as plt
from datetime import datetime, timedelta
def get_price_data(crypto_name, days):
    end_date = datetime.utcnow()
    start_date = end_date - timedelta(days=days)
    start_timestamp = int(start_date.timestamp())
    end_timestamp = int(end_date.timestamp())
    url                                                    =
f'https://api.coingecko.com/api/v3/coins/{crypto_name}/market_char
t/range'
    params = {
        'vs_currency': 'usd',
        'from': start_timestamp,
        'to': end_timestamp
    }
    response = requests.get(url, params=params)
    data = response.json()
    prices = [(datetime.utcfromtimestamp(item[0] / 1000), item[1])
for item in data['prices']]
    return prices
def calculate_moving_average(prices, window_size):
    return [sum(prices[i:i+window_size]) / window_size for i in
range(len(prices) - window_size + 1)]
def visualize_price_with_ma(crypto_name, days, window_size):
    price_data = get_price_data(crypto_name, days)
    dates, prices = zip(*price_data)
    moving_average = calculate_moving_average(prices, window_size)
    fig, ax = plt.subplots(figsize=(12, 6))
    ax.plot(dates,   prices,   label=f'Price   of   {crypto_name}',
color='blue')
    ax.plot(dates[window_size-1:],                  moving_average,
label=f'{window_size}-day MA', color='orange')
```

```
    ax.set_xlabel('Date')
    ax.set_ylabel('Price (USD)')
    ax.set_title(f'{crypto_name.capitalize()} Price Over Last
{days} Days with {window_size}-Day Moving Average')
    ax.legend()
    ax.grid(True)
    plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()
```

Візуалізація графіку ціни з ковзним середнім є важливою, оскільки вона допомагає згладити короткострокові коливання ціни, роблячи загальні тренди більш очевидними та легкими для аналізу. Це дозволяє краще ідентифікувати середньострокові та довгострокові тенденції на ринку, що є критичним для прийняття інвестиційних рішень.

Крім того, ковзне середнє часто використовується для визначення рівнів підтримки та опору, які є ключовими для розробки ефективних торгових стратегій. Трейдери та інвестори можуть використовувати ці дані для прийняття обґрунтованих рішень щодо купівлі або продажу активів. Таким чином, така візуалізація допомагає користувачам краще розуміти ринкову динаміку та ухвалювати більш обґрунтовані інвестиційні рішення.

Візуалізація об'єму торгів за день: ця функція будує гістограму, яка відображає обсяги торгів крипто-валюти протягом певного періоду часу. Вона важлива для розуміння ринкової активності та ліквідності крипто-валюти.

Принцип роботи у неї доволі простий і його можна розділити на 3 наступні частини:

1. Збір даних: Виконується запит до API для отримання історичних даних про обсяги торгів за певний період.

2. Обробка даних: Дані перетворюються у формат, зручний для побудови гістограми. Дати конвертуються у відповідний формат, а обсяги торгів додаються до списку.

3. Побудова гістограми: Використовується бібліотека matplotlib для побудови гістограми, на якій по осі X відкладається час, а по осі Y - обсяг торгів.

Візуалізація індексу страху та жадібності дозволяє побачити графік який відображає емоційний стан ринку крипто-валют.

Принцип роботи даної функції такий ж що і у попередньої за відмінністю того що відбувається запит інформації з іншого API. Цю інформацію можна було б отримати і власноруч на основі даних з інших функцій однак цей процес буде занадто громісткий на даний момент.

Візуалізація індексу страху та жадібності є важливою, оскільки цей індекс відображає емоційний стан ринку крипто-валют, що може бути критичним для прийняття інвестиційних рішень. Індекс страху та жадібності допомагає визначити, чи є ринок більше схильним до страху (що може свідчити про надмірну обережність та можливі низькі ціни) або жадібності (що може свідчити про надмірний оптимізм та можливі високі ціни). Візуалізація цього індексу дозволяє інвесторам і трейдерам краще розуміти поточні настрої ринку, оцінювати ризики і можливості, а також приймати більш обґрунтовані рішення щодо своїх інвестиційних стратегій.

3.2.3 Взаємодія з користувачем та управління даними

Цей підпункт зосереджений на аспектах взаємодії користувача з інструментом та управління даними, зокрема на процесах виконання функцій, запису функцій у текстовий файл, а також збереження списку останніх відкритих проектів.

Для запису обраних функцій які потрібно виконати виконується їх запис у текстовий. Ця функція зберігає всі обрані користувачем функції та їх параметри у текстовий файл для подальшого використання. Код за допомогою якого це відбувається продемонстровано на лістнигу 3.4.

Лістинг 3.4 – Процес запису обраних функцій та їх параметрів

```
def save_to_file(selected_functions, entries, crypto_name_entry, parent_window, data_entry_window, main_window):
    crypto_name = crypto_name_entry.get().strip()
```

```
    current_datetime   =   datetime.datetime.now().strftime(«%Y-%m-
%d_%H-%M-%S»)
    filename = f»{crypto_name}-{current_datetime}.txt»
    with open(filename, «w») as file:
        for function in selected_functions:
            if function == «Отримання символу за назвою»:
                file.write(f»get_symbol:{crypto_name}\n»)
            elif function == «Отримання ціни за назвою»:
                file.write(f»get_price:{crypto_name}\n»)
    messagebox.showinfo(«Success», f»Файл збережено як {filename}»)
```

Функція зберігає назву крипто-валюти та обрані функції у текстовий файл з унікальним іменем, що містить дату та час створення. Після збереження файл можна відкрити знову для подальшого використання. Повний код можна переглянути у додатку А.

Наступною є функція що відповідає за виконання всіх основних функцій інструменту на основі налаштувань, збережених у текстовому файлі. Вона зчитує файл налаштувань, виконує відповідні функції для збору та обробки даних, а також виводить результати у текстовий формат або візуалізація. Її код можна переглянути на лістингу

Лістинг 3.5 – Виклик записаних функцій

```
def execute_functions(filename, tab):
    settings = read_settings_file(filename)
    results = []
    if 'get_symbol' in settings:
        results.append(f»Символ:
{get_symbol(settings['get_symbol'][0])}»)
    if 'get_price' in settings:
        results.append(f»Ціна:
{get_price(settings['get_price'][0])}»)
    return '\n'.join(results)
```

Функція execute_functions викликає окремі функції, такі як get_symbol та get_price, і додає результати їх виконання до списку results. Після виконання всіх функцій, результати об'єднуються у рядок і повертаються або представлаються у вигляді графіків, гістограм, то що. Повний код можна переглянути у додатку Б.

3.3 Розробка front-end частини

Розробка front-end частини програми включає створення користувацького інтерфейсу, який дозволяє користувачам зручно взаємодіяти з інструментом для візуалізації та аналізу крипто-активів. Основним завданням є забезпечення інтуїтивно зрозумілого та зручного інтерфейсу, що дозволить користувачам легко виконувати необхідні дії.

Інтерфейс програми побудований на віконній системі з елементами вкладочної структури. Такий підхід забезпечує зручну та інтуїтивно зрозумілу взаємодію користувача з програмою, дозволяючи йому легко перемикатися між різними функціями та розділами інтерфейсу.

Віконна система інтерфейсу передбачає поділ програми на окремі вікна, кожне з яких виконує певну функцію. Такий підхід дозволяє користувачу концентруватися на конкретних задачах, забезпечуючи при цьому можливість швидкого доступу до інших функцій програми. Віконна система є стандартом для більшості десктопних програм, оскільки вона знайома користувачам і забезпечує високу зручність використання.

Список вікон розроблюваного інструменту:

1. Головне меню.
2. Вікно вибору функцій.
3. Основне робоче поле.
4. Вікно заповнення параметрів функцій.
5. Довідка функцій.

Вікна програми є можливість відкривати з різних її частин, так наприклад вікно вибору функцій може бути відкрите як головного вікна в процесі створення нового проекту так і з робочого простору при зміні функцій для візуалізації та аналізу вже існуючого файлу.

3.3.1 Вікно головне меню

Вікно головного меню являються першим вікном що зустрічає користувача після запуску програми. Основний функціонал виражений тим що користувачам може створювати нові проекти або відкривати існуючі. Також воно надає доступ до останніх відкритих файлів, перегляду та редагування параметрів функцій, а також оновлення даних аналізу.

Вигляд цього вікна можна побачити на рисунку 3.1

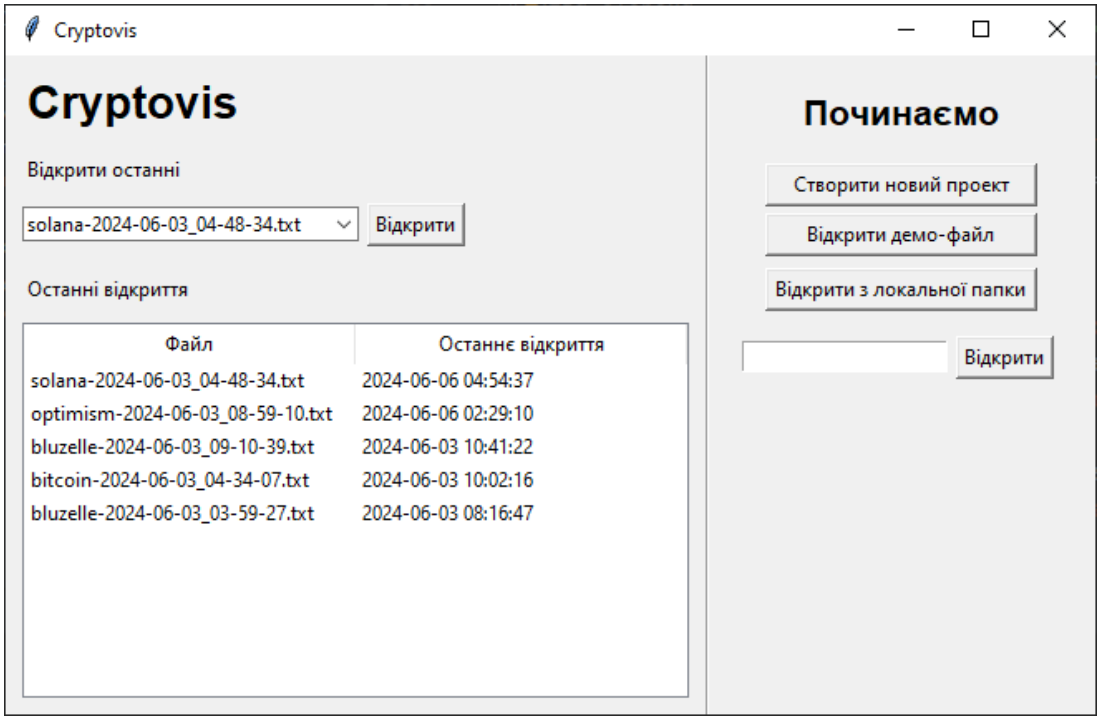

Рисунок 3.1 – Стартове вікно програми

Вікно можна розділити на дві частини і у кожного є свої елемнти:

Ліва частина:

- Заголовок «Cryptovis»: Розташований у верхній частині лівого фрейму, він відображає назву програми великим шрифтом.
- Відкриття останніх файлів: Випадаючий список, який містить назви останніх відкритих файлів, і кнопка «Відкрити», яка дозволяє швидко відкрити вибраний файл.

- Перелік останніх відкриттів: Таблиця з переліком останніх відкритих файлів, що включає дві колонки: «Файл» та «Останнє відкриття». У колонці «Файл» вказана назва файлу, а в колонці «Останнє відкриття» - дата та час останнього відкриття файлу. Таблиця дозволяє користувачам двічі клікнути на файл для його відкриття.

Права частина:

- Заголовок «Починаємо»: Розташований у верхній частині правого фрейму, він вітає користувача та інформує про доступні опції.
- Кнопки для основних дій:
- «Створити новий проект»: Відкриває вікно для створення нового проекту.
- «Відкрити демо-файл»: Дозволяє відкрити демонстраційний файл для ознайомлення з функціоналом програми.
- «Відкрити з локальної папки»: Відкриває діалогове вікно для вибору файлу з локального сховища.
- Текстове поле для введення шляху до файлу: Користувач може вручну ввести шлях до файлу, після чого натиснути кнопку «Відкрити» для його завантаження.

Фрагмент коду що дозволяє розділити робочий простір можна переглянути на лістингу 3.6.

Лістинг 3.6 – первинна розмітка стартового вікна

```
root = tk.Tk()
root.title(«Cryptovis»)
left_frame = tk.Frame(root)
left_frame.grid(row=0, column=0, sticky=«nsew»)
right_frame = tk.Frame(root)
right_frame.grid(row=0, column=2, sticky=«nsew»)
separator = ttk.Separator(root, orient='vertical')
separator.grid(row=0, column=1, sticky='ns')
root.grid_columnconfigure(0, weight=3)
root.grid_columnconfigure(2, weight=2)
root.grid_rowconfigure(0, weight=1)
```

Вікно root створюється за допомогою бібліотеки tkinter. Вікно ділиться на дві частини з відношенням 3:2. Ліва частина (left_frame) містить елементи для відкриття останніх файлів, а права частина (right_frame) містить елементи керування проектами. Також додається розділяюча лінія між лівою та правою частинами для кращого сприйняття.

Спершу розглянемо праву сторону розробленого вікна. На цій стороні розташовуються такі елементи:

- Заголовок «Починаємо» вітає користувача та інформує про доступні опції.
- Кнопки «Створити новий проект», «Відкрити демо-файл» та «Відкрити з локальної папки» дозволяють користувачу швидко розпочати роботу з програмою.
- Текстове поле для введення шляху до файлу дозволяє користувачу вручну ввести шлях до файлу, поруч з яким знаходиться кнопка «Відкрити», яка дозволяє завантажити файл за вказаним шляхом.

Код розміщення усіх цих елементів на сторінці можна переглянути на лістингу 3.7.

Лістинг 3.7 – Розміщення елементів на правій стороні вікна

```
cryptivis_label = tk.Label(left_frame, text=«Cryptovis», font=(«Helvetica», 20, «bold»))
cryptivis_label.grid(row=0, column=0, padx=10, pady=(10, 5), sticky=«w»)
open_recent_label = tk.Label(left_frame, text=«Відкрити останні»)
open_recent_label.grid(row=1, column=0, padx=10, pady=(5, 5), sticky=«w»)
recent_files_frame = tk.Frame(left_frame)
recent_files_frame.grid(row=2, column=0, padx=10, pady=(5, 10), sticky=«w»)
recent_files_combo = ttk.Combobox(recent_files_frame, width=30)
recent_files_combo.grid(row=0, column=0, sticky=«w»)
open_selected_button = tk.Button(recent_files_frame, text=«Відкрити», command=lambda: open_selected_file(recent_files_combo))
open_selected_button.grid(row=0, column=1, padx=5, sticky=«w»)
recent_label = tk.Label(left_frame, text=«Останні відкриття»)
recent_label.grid(row=3, column=0, padx=10, pady=(5, 5), sticky=«w»)
```

```
tree = ttk.Treeview(left_frame, columns=(«File», «Last Opened»), show=«headings»)
tree.grid(row=4, column=0, padx=10, pady=(5, 10), sticky=«nsew»)
tree.heading(«File», text=«Файл»)
tree.heading(«Last Opened», text=«Останнє відкриття»)
tree.bind(«<Double-1>«, on_tree_item_click)
populate_recent_files(recent_files_combo, tree)
```

У лівій ж частині вікна розміщуються такі елементи:

- Заголовок «Cryptovis» відображає назву програми великим шрифтом.
- Випадаючий список для відкриття останніх файлів, який містить назви останніх відкритих файлів. Поруч із випадаючим списком знаходиться кнопка «Відкрити», яка дозволяє швидко відкрити вибраний файл.
- Таблиця з переліком останніх відкритих файлів, що включає дві колонки: «Файл» та «Останнє відкриття». У колонці «Файл» вказана назва файлу, а в колонці «Останнє відкриття» - дата та час останнього відкриття файлу. Таблиця дозволяє користувачам двічі клікнути на файл для його відкриття.

Код розміщення перелічених елементів продемонстровано на лістингу 3.8.

Лістинг 3.8 – Розміщення елементів на правій стороній вікна

```
cryptivis_label = tk.Label(left_frame, text=«Cryptovis», font=(«Helvetica», 20, «bold»))
cryptivis_label.grid(row=0, column=0, padx=10, pady=(10, 5), sticky=«w»)
open_recent_label = tk.Label(left_frame, text=«Відкрити останні»)
open_recent_label.grid(row=1, column=0, padx=10, pady=(5, 5), sticky=«w»)
recent_files_frame = tk.Frame(left_frame)
recent_files_frame.grid(row=2, column=0, padx=10, pady=(5, 10), sticky=«w»)
recent_files_combo = ttk.Combobox(recent_files_frame, width=30)
recent_files_combo.grid(row=0, column=0, sticky=«w»)
open_selected_button = tk.Button(recent_files_frame, text=«Відкрити», command=lambda: open_selected_file(recent_files_combo))
open_selected_button.grid(row=0, column=1, padx=5, sticky=«w»)
recent_label = tk.Label(left_frame, text=«Останні відкриття»)
recent_label.grid(row=3, column=0, padx=10, pady=(5, 5), sticky=«w»)
tree = ttk.Treeview(left_frame, columns=(«File», «Last Opened»), show=«headings»)
tree.grid(row=4, column=0, padx=10, pady=(5, 10), sticky=«nsew»)
tree.heading(«File», text=«Файл»)
tree.heading(«Last Opened», text=«Останнє відкриття»)
```

```
tree.bind(«<Double-1>«, on_tree_item_click)
populate_recent_files(recent_files_combo, tree)
```

Продемонстрований підхід до організації головного меню дозволяє користувачам легко створювати нові проекти, відкривати існуючі файли, а також мати швидкий доступ до останніх відкритих проектів, забезпечуючи зручність та ефективність роботи з інструментом для візуалізації та аналізу крипто-активів. Вище було наведено лише фрагменти коду для головного вікна повний ж можна переглянути на додатку В.

3.3.2 Вікна вибору, довідки та задавання параметрів функцій

Вікно вибору функцій дозволяє користувачу обрати необхідні функції для аналізу крипто-валют. Це вікно має вкладочну структуру з вкладками «Отримання даних» та «Візуалізація даних». У вкладці «Отримання даних» є дві підвкладки: «Загальні дані про крипто-валюту» та «Дані про торгівлю і ціни».

Структура вкладок:

- Вкладка «Отримання даних»:
- Підвкладка «Загальні дані про крипто-валюту»: містить функції для отримання базової інформації про крипто-валюту, такі як символ, ціна, загальна кількість токенів, ринкова капіталізація та інші.
- Підвкладка «Дані про торгівлю і ціни»: містить функції для отримання даних про торгівлю, таких як максимальна та мінімальна ціна за період, дати різких змін ціни, обсяги торгів за період та інші.
- Вкладка «Візуалізація даних»: містить функції для візуалізації даних, такі як графік ціни, графік ціни з ковзним середнім, візуалізація обсягів торгів та інші.

Елементи вікна:

- Випадаючі списки та кнопки:
- Список функцій з чекбоксами: дозволяє користувачу обрати функції, які будуть виконуватись для аналізу.

- Поле для загального списку обраних функцій: показує список функцій, обраних користувачем.
- Кнопки «Довідка функцій» та «Далі»: кнопка «Довідка функцій» відкриває вікно з описом функцій, а кнопка «Далі» переходить до вікна введення параметрів функцій.

Спершу додаються вкладки «Отримання даних» та «Візуалізація даних». У вкладці «Отримання даних» створюються дві підвкладки для загальних даних про крипто-валюту та даних про торгівлю і ціни, код за допомогою якого це виконується можна переглянути на лістингу 3.9.

Лістинг 3.9 – Створення вкладок та підвкладок

```
tab_control = ttk.Notebook(new_project_window)
data_tab = ttk.Frame(tab_control)
tab_control.add(data_tab, text='Отримання даних')
data_notebook = ttk.Notebook(data_tab)
general_data_tab = ttk.Frame(data_notebook)
trade_data_tab = ttk.Frame(data_notebook)
data_notebook.add(general_data_tab, text='Загальні дані про крипто-
валюту')
data_notebook.add(trade_data_tab, text='Дані про торгівлю і ціни')
data_notebook.pack(expand=1, fill='both')
visualization_tab = ttk.Frame(tab_control)
tab_control.add(visualization_tab, text='Візуалізація даних')
tab_control.pack(expand=1, fill='both')
```

Наступним елементом є поле для загального списку обраних функцій. Це поле показує всі функції, які були обрані користувачем код для створення цього списку можна переглянути на лістингу 3.10.

Лістинг 3.10 – Текстове поле з обраними функціями

```
buttons_frame = ttk.Frame(new_project_window)
buttons_frame.pack(fill='x', padx=10, pady=10)
function_info_button = ttk.Button(buttons_frame, text=«Довідка
функцій», command=open_function_info_window)
function_info_button.pack(side='left')
next_button = ttk.Button(buttons_frame, text=«Далі»,
command=lambda: open_data_entry_window(new_project_window,
selected_functions_listbox, parent_window))
next_button.pack(side='right')
```

Також на вікно додаються кнопки «Довідка функцій» та «Далі». Кнопка «Довідка функцій» відкриває вікно з описом функцій, а кнопка «Далі» переходить до вікна введення параметрів функцій.

Вигляд створеного вікна вибору функцій можна переглянути на рисунку 3.2.

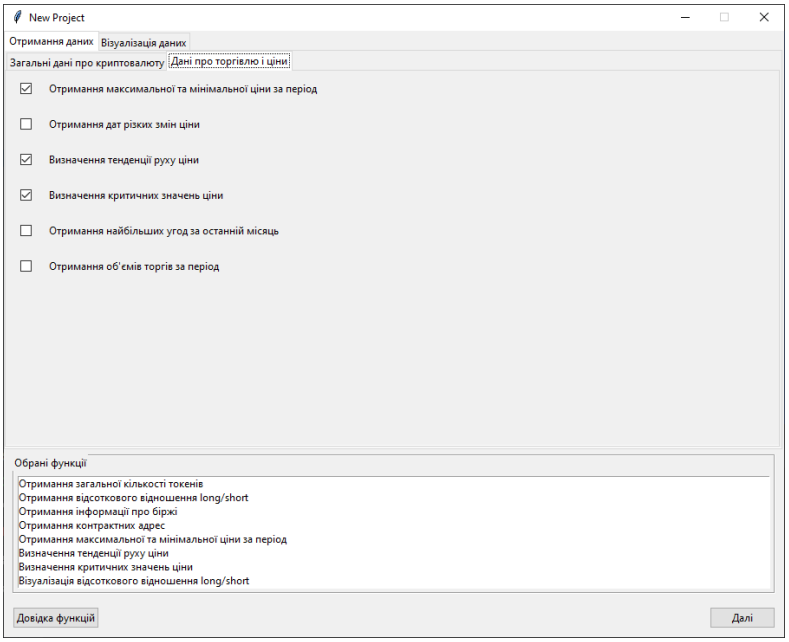

Рисунок 3.2 – Вікно вибору функцій

Наступним було розроблено вікно довідки функцій. Воно дозволяє користувачу отримати детальну інформацію про всі доступні функції. На ньому міститься список функцій з можливістю перегляду короткого опису кожної функції.

Елементами цього вікна являються список функцій, поле опису.

Список функцій розташований у лівій стороні і містить у собі список всіх доступних функцій. Користувач може обрати будь-яку функцію для перегляду її опису.

Поле опису знаходиться праворуч від списку функцій розташоване поле, яке показує короткий опис обраної функції.

Останнім вікном що дозволяє взаємодіяти напряму з функціями є введення даних.

Вікно введення даних дозволяє користувачу ввести необхідні параметри для обраних функцій. Воно забезпечує зручний інтерфейс для введення даних, таких як назва крипто-валюти, період аналізу та інші параметри.

Оскільки усі функції потребують назви крипто-валюти то воно не залежно від обраних функцій розташоване у верхній частині вікна.

Поля для введення параметрів функцій: для кожної обраної функції створюється відповідне поле для введення параметрів. Наприклад, для функції «Візуалізація графіку ціни» потрібно ввести період у днях, а для функції «Візуалізація графіку ціни з ковзним середнім» потрібно додатково вказати розмір вікна ковзного середнього.

Ну і завершальним елементом є кнопка «Створити» яка розташована внизу вікна, дозволяє зберегти введені дані та перейти до створення проекту.

Ці вікна забезпечують зручність та ефективність роботи з інструментом для візуалізації та аналізу крипто-активів, дозволяючи користувачам легко обирати та налаштовувати необхідні функції.

3.3.3 Вікно основного робочого поля

Вікно головного робочого простору забезпечує зручний інтерфейс для взаємодії з результатами аналізу та візуалізації крипто-валютних даних. Воно включає кілька функціональних елементів, які дозволяють користувачеві відкривати проекти, змінювати дані та оновлювати інформацію. Інтерфейс організований таким чином, щоб користувач міг легко і швидко отримувати необхідну інформацію, виконувати різні операції з даними та переглядати результати в зручному вигляді.

Верхня панель містить кілька основних кнопок: «Новий», «Змінити дані», «Обновити інформацію» та «Відкрити». Кожна з цих кнопок виконує специфічні функції:

- Кнопка «Новий» дозволяє створити новий проект, відкриваючи вікно вибору функцій, де користувач може вибрати необхідні йому функції для аналізу та візуалізації даних.
- Кнопка «Змінити дані» відкриває вікно для зміни параметрів функцій у вже існуючому файлі, де дані будуть перезаписані у поточний файл. Це забезпечує можливість швидкого коригування параметрів без необхідності створення нового проекту.
- Кнопка «Обновити інформацію» виконує повторний запит до API для оновлення даних, забезпечуючи актуальність інформації, що відображається. Це корисно для отримання найсвіжіших даних про крипто-валютні ринки.
- Кнопка «Відкрити» дозволяє вибрати існуючий файл для відкриття та перегляду результатів аналізу. Це дозволяє користувачеві швидко відновити роботу з попередніми проектами.

Центральна частина вікна складається з вкладок, кожна з яких містить результати аналізу для окремого файлу. Вкладки забезпечують можливість перегляду кількох файлів одночасно, що дозволяє користувачеві порівнювати різні дані та результати аналізу. Фрагмент коду що за це відповідає можна переглянути на лістингу 3.11.

Лістинг 3.11 – Створення вкладок для відкритих файлів

```
tab_control = ttk.Notebook(main_window)
tab_control.pack(side='top',  fill='both',  expand=True,  padx=5,
pady=5)
tab_frame = ttk.Frame(tab_control)
tab_control.add(tab_frame, text=filename.split(«/»)[-1])
canvas = tk.Canvas(tab_frame)
scrollbar    =    ttk.Scrollbar(tab_frame,    orient=«vertical»,
command=canvas.yview)
scrollable_frame = ttk.Frame(canvas)
scrollable_frame.bind(
    «<Configure>«,
    lambda e: canvas.configure(
        scrollregion=canvas.bbox(«all»)
    )
)
canvas.create_window((0, 0), window=scrollable_frame, anchor=«nw»)
canvas.configure(yscrollcommand=scrollbar.set)
```

```
scrollable_frame.bind_all(«<MouseWheel>«, lambda event:
on_mouse_wheel(event, canvas))
canvas.pack(side=«left», fill=«both», expand=True)
scrollbar.pack(side=«right», fill=«y»)
```

Для кожного файлу створюється окрема вкладка, яка містить прокручувану область з результатами аналізу також одразу здійснюється прив'язка події прокручування мишки що дозволяє прокручувати вміст вкладки за допомогою колеса мишки, що робить навігацію більш зручною.

Нижня частина вікна містить текстове поле для відображення текстових результатів аналізу. Це поле дозволяє користувачеві детально переглядати текстові звіти та інші текстові дані, що є результатом виконання обраних функцій. Вміст самого поля на якому видаються результати оновлюється при зміні вкладки, відображаючи результати для обраного файлу.

Вікно основного робочого поля з деякими даними можна переглянути на рисунку 3.3.

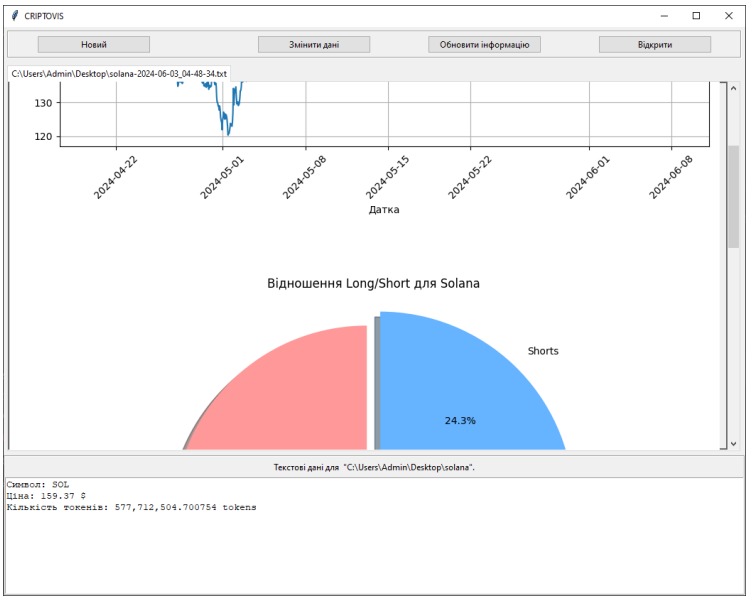

Рисунок 3.3 – Вікно основного робочого поля

Таким чином, вікно головного робочого простору забезпечує зручний інтерфейс для роботи з даними, що дозволяє користувачам легко відкривати, змінювати та оновлювати інформацію про крипто-активи. Це сприяє

ефективному аналізу та візуалізації даних, забезпечуючи користувачеві всі необхідні інструменти для роботи з крипто-валютними ринками. Повний код вікона можна переглянути у додатку Д.

3.4 Валідація та тестування інструменту

Валідація та тестування є важливими етапами розробки будь-якого програмного забезпечення, зокрема й інструментів для візуалізації та аналізу крипто-активів. Вони забезпечують коректність роботи, стабільність і надійність програми, а також гарантують, що інструмент відповідає всім вимогам користувачів.

Основною метою тестування є перевірка всіх аспектів функціональності інструменту, виявлення можливих помилок і недоліків, а також оцінка зручності використання програми. Це включає:

- Функціональне тестування: перевірка основних функцій інструменту, таких як створення нового проекту, відкриття раніше створеного проекту, перегляд інформації.
- Тестування інтерфейсу користувача: оцінка зручності використання, логічності розташування елементів інтерфейсу, а також виявлення можливих проблем у взаємодії користувача з програмою.
- Тестування продуктивності: перевірка швидкості виконання запитів до API, швидкості обробки даних і відображення результатів.
- Тестування надійності: перевірка стабільності роботи програми при різних навантаженнях і в різних умовах, виявлення можливих збоїв і помилок.

Для проведення тестування інструменту використовувалися такі методи: Користувацьке тестування: було залучено кінцевого користувача - трейдера з досвідом півтора року. Це тестування проводилося на комп'ютері, де був розроблений інструмент, оскільки інструмент ще не був перетворений у файл .exe. Трейдер виконував такі сценарії: створення нового проекту з різною кількістю функцій (мінімальна, середня, максимальна кількість функцій);

відкриття раніше створених проектів (використання демонстраційного файлу); тестування функцій оновлення та зміни даних для функцій. Тестування продуктивності: оцінка навантаження на комп'ютер під час відкриття проектів з різною кількістю інформації. Оцінювалися наступні показники: швидкість відкриття проектів; рівень завантаженості ЦП, оперативної пам'яті, жорсткого диску та мережі. Системне тестування: перевірка стабільності роботи програми при різних навантаженнях, включаючи часте оновлення даних та відкриття великої кількості проектів одночасно.

В результаті тестування було виявлено та виправлено кілька помилок у роботі окремих функцій та взаємодії між модулями. Було виявлено, що одна з функцій не видає результат через невірний формат вхідних даних. Після виправлення ця функція запрацювала коректно. Якщо задавати неіснуючу криптовалюту, виникає глобальна помилка. Це буде виправлено в подальшому за допомогою перевірки даних на коректність. Було виявлено обмеження на отримання історичних даних, пов'язане з обмеженням самого API для публічного використання. Це можна виправити придбавши платний доступ. При надмірній кількості оновлення даних виникала помилка через обмеження API на кількість запитів за одиницю часу. Відкриття великої кількості проектів одночасно призводило до зниження продуктивності ПК та зручності використання. Було виявлено, що якщо створювати проект з більш ніж 8 функціями, виникала помилка API через обмеження доступу. Виправлення можливе через придбання платного доступу. Використання функцій, що обробляють велику кількість інформації, призводило до значного часу відкриття або створення проектів. Виправлення можливе шляхом використання паралельних обчислень, збільшення фізичних ресурсів ПК або ширини доступу до мережі.

Перевірка продуктивності показала, що під час запуску проекту є значне навантаження на мережу, а також стрибки використання ЦП, жорсткого диску та оперативної пам'яті. Ці стрибки є значними лише у моменті відкриття проекту та оновлення даних. Час завантаження проектів з мінімальною кількістю функцій

займає менш ніж пів секунди, середньої кількості – від секунди до півтори, а максимальної – від 2 до 3 секунд. Однак, якщо швидкість інтернету низька, то час завантаження значно збільшується. Кількість стрибків навантаження залежить від кількості об'ємних функцій, таких як візуалізація графіку ціни або визначення найбільших тримачів крипто-валюти. Графіки стрибків крипто-валюти можна переглянути на рисунку 3.4 у виділених областях: область під номером 1 – найменше функцій, область під номером 2 – середня кількість функцій, область під номером 3 – найбільша кількість функцій.

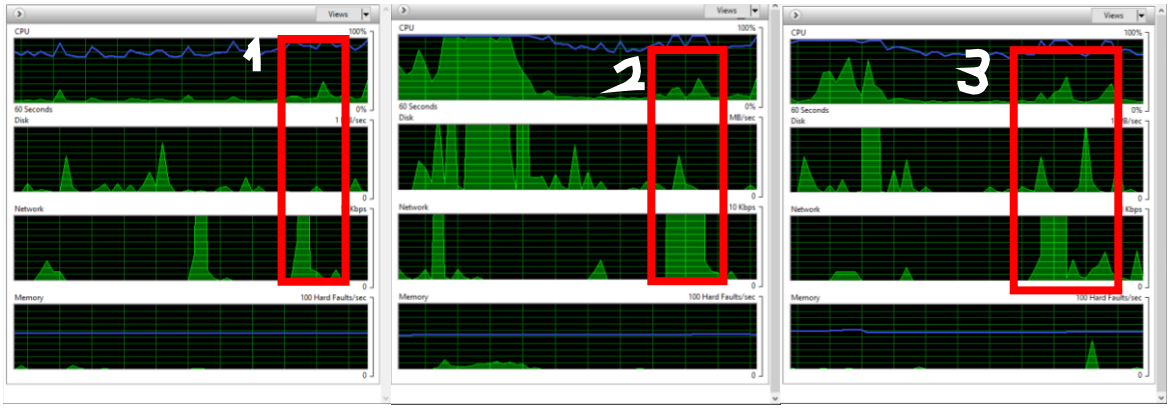

Рисунок 3.4 – графіки навантаження ресурсів ПК

Також було здійснено кілька правок в інтерфейсі користувача порівняно з початковим плануванням інтерфейсу. Це дозволило покращити зручність використання інструменту та забезпечити більш інтуїтивно зрозумілий інтерфейс.

В результаті тестування було підтверджено загальну функціональність та стабільність інструменту, хоча були виявлені деякі проблеми, які потребують подальшого доопрацювання. Основні функції інструменту працюють коректно, проте для досягнення більшої надійності та зручності використання необхідно внести певні покращення. Інтерфейс в цілому зручний для користувачів, але також потребує деяких доопрацювань для підвищення ефективності роботи з даними про крипто-активи.

3.5 Висновки до третього розділу

В третьому розділі кваліфікаційної роботи описано процес реалізації інструменту для візуалізації та аналізу крипто-активів на Python. Було реалізовано низку важливих функцій для збору, обробки та візуалізації даних. Основною технологією для розробки користувацького інтерфейсу було обрано бібліотеку tkinter, яка забезпечила створення інтуїтивно зрозумілого та зручного інтерфейсу для взаємодії користувача з інструментом. Функції back-end частини, включаючи запити до API, обробку даних за допомогою pandas та візуалізацію результатів з використанням matplotlib, були успішно інтегровані в загальну архітектуру програми.

Функції отримання та обробки даних забезпечили доступ до актуальної інформації про крипто-валюти з різних джерел, а візуалізація даних спростила їх аналіз та інтерпретацію.

Валідація та тестування інструменту показали його здатність виконувати основні завдання та надавати користувачеві актуальну інформацію про крипто-валюти. Проведене користувацьке, продуктивності та системне тестування допомогло виявити кілька проблем, які потребують подальшого доопрацювання, зокрема помилки з форматом вхідних даних, обмеження API для публічного використання та продуктивність при обробці великих обсягів даних.

Тестування також продемонструвало важливість оптимізації інтерфейсу користувача для забезпечення більшої зручності використання. Було виявлено, що деякі елементи інтерфейсу потребують доопрацювання для кращого відображення інформації та підвищення загальної ефективності роботи інструменту.

Розроблений інструмент показав свою здатність забезпечувати користувачів важливою інформацією про крипто-активи, однак для досягнення максимальної ефективності та надійності необхідно врахувати результати тестування та внести відповідні зміни.

РОЗДІЛ 4. БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Ризик як кількісна оцінка небезпек

Ризик у контексті безпеки життєдіяльності визначається як ймовірність виникнення подій з небажаними наслідками (нещасні випадки, аварії, катастрофи) протягом конкретного періоду часу. Ризик залежить від частоти прояву небезпеки та можливих наслідків цих подій [23].

Кількісна оцінка небезпек передбачає визначення шкоди, яку може завдати небезпека. Вартісна оцінка є універсальним методом, який дозволяє порівняти різні види шкоди.

Комплексною оцінкою небезпеки є ризик “R”, який визначається як добуток частоти прояву небезпеки на шкоду, яку вона завдає, як обраховувати зображено на формулі 4.1.

$$R = p \cdot E \tag{4.1}$$

де:

- R — ризик;
- p — частотна ймовірність прояву небезпеки;
- E — шкода, яку завдає небезпек;

Оцінка серйозності небезпеки допомагає визначити необхідні заходи для забезпечення безпеки. Існують чотири категорії серйозності небезпеки можна переглянути на таблиці 4.1

Таблиця 1.1 – Категорії серйозності небезпеки

Вид	Категорія	Опис нещасного випадку
Незначна	1	Менш значні травми, короткочасне захворювання
Гранична	2	Серйозна травма, стійке захворювання
Критична	3	Значний збиток, важкі травми
Катастрофічна	4	Загибель людей, зруйнування системи

Ймовірність небезпеки можна розділити на кілька рівнів, розподіл продемонстровано на таблиці 4.2.

Таблиця 4.2 – Рівні ймовірності небезпеки

Вид (по частоті виникнення)	**Рівень**	**Опис наслідків**
Часта	A	Велика ймовірність того, що подія відбудеться
Ймовірна	B	Може трапитися кілька разів за життєвий цикл
Можлива	C	Спостерігається кілька разів за період роботи
Рідка	D	Малоймовірно, але можливе виникнення
Практично неможлива	E	Настільки малоймовірно, що можна припустити, що не відбудеться

Комбінуючи категорії серйозності та рівні ймовірності, можна створити матрицю оцінки ризику, що дозволяє класифікувати ризики за ступенем їх припустимості. Сама матриця продемонстрована на таблиці 4.3.

Таблиця 4.3 – Матриця оцінки ризику

Частота, з якою відбувається подія (небезпека)	**Категорія небезпеки**	**Індекс ризику небезпеки**	**Класифікація ризику**
Часто (A)	4A, 3A, 2A, 1A	4A, 4B, 4C, 3A, 3B, 2A	Неприпустимий (надмірний)
Ймовірно (B)	4B, 3B, 2B, 1B	4D, 3C, 3D, 2B, 2C	Небажаний (гранично допустимий)
Можливо (C)	4C, 3C, 2C, 1C	4E, 3E, 2D, 2E, 1A, 1B	Припустимий з перевіркою (прийнятний)
Рідко (D)	4D, 3D, 2D, 1D	1C, 1D, 1E	Припустимий без перевірки (знехтуваний)
Практично неможливо (E)	4E, 3E, 2E, 1E		

Сучасна концепція безпеки життєдіяльності базується на досягненні прийнятного ризику, який поєднує технічні, економічні, соціальні та політичні аспекти. Прийнятний ризик визначається як компроміс між бажаним рівнем безпеки і можливостями його досягнення [24].

Управління ризиком передбачає заходи зниження ризиків до припустимих рівнів. Це може включати:

- відмову від небезпечних робіт;
- заміна небезпечних операцій менш небезпечними;
- удосконалення технічних систем;
- розробка та використання засобів захисту;
- організаційно-управлінські заходи;

Для ефективного управління ризиком використовують ризик орієнтований підхід, який включає порівняння існуючого ризику з припустимим і прийняття рішень на основі цього аналізу.

Кількісна оцінка ризику є важливим інструментом для аналізу та прийняття обґрунтованих рішень у сфері безпеки життєдіяльності. У контексті кваліфікаційної роботи, де розробляється інструмент для аналізу та візуалізації інформації про крипто-активи, ці підходи можуть бути застосовані для фінансових ризиків. Інструмент дозволить користувачам оцінювати ризики крипто-активів, що сприятиме більш інформованому прийняттю інвестиційних рішень.

4.2 Вимоги ергономіки до організації робочого місця оператора ПК

Робоче місце визначається як простір, оснащений необхідним обладнанням, де один працівник або група працівників здійснюють свою трудову діяльність. Робочою зоною розуміється простір, що знаходиться на висоті до 2 метрів над рівнем підлоги або площадки, на якій працівники перебувають постійно або тимчасово.

Ергономіка (від грецьк. ἔργον – робота) – наука про пристосування робочих місць, обладнання та комп'ютерних програм задля створення найбільш безпечних та ефективних умов праці для людини, виходячи з фізичних і психічних особливостей людського організму [25].

Основні завдання ергономіки:

1. Пристосування елементів системи «людина – трудовий процес» до можливостей працівника.

2. Забезпечення умов для максимальної ефективності праці.

3. Запобігання можливим загрозам для здоров'я працівника.

4. Оптимальна витрата біологічних ресурсів у процесі праці.

Площа кабінету для одного робочого місця з ПК повинна бути не меншою 6 м², а об'єм – не менш 24 м³. Для внутрішньої обробки приміщення повинні використовуватися дифузно-відбивні матеріали з коефіцієнтами відбиття для стелі – 0,7-0,8; для стін – 0,5-0,6; для підлоги – 0,3-0,5. Робочі місця з ПК розміщуються на відстані не менше 1 м від стін зі світловими прорізами та на відстані не менш ніж 1,2 м між бічними поверхнями відеотерміналів. Відстань між тильною поверхнею одного відеотермінала та екраном іншого не повинна бути меншою 2,5 м [26].

Організація робочого місця передбачає:

- правильне розміщення робочого місця у виробничому приміщенні;
- вибір ергономічно обґрунтованого робочого положення, виробничих меблів з урахуванням антропометричних характеристик людини;
- раціональну компоновку обладнання на робочих місцях;
- урахування характеру та особливостей трудової діяльності.

Висота робочої поверхні столу має бути 725 мм з можливістю регулювання в межах 680-800 мм. Ширина – 600-1400 мм, глибина – 800-1000 мм. Простір для ніг: висота не менше 600 мм, ширина – не менше 500 мм, глибина на рівні колін – не менше 450 мм, на рівні простягнутої ноги – не менше 650 мм. Стіл повинен бути зручним для розміщення обладнання та забезпечувати доступ до

необхідних ресурсів, оптимальне розміщення екрану, клавіатури, принтера та документів [26].

Монітор встановлюється на відстані 500-700 мм від очей користувача. Кут зору – 10-20°, але не більше 40°; кут між верхнім краєм дисплея і рівнем очей – не менше 10°. Кращим є розташування екрану перпендикулярно до лінії зору користувача. Робоче місце повинно бути обладнане поворотним майданчиком для відеотермінала, що дозволяє переміщати його в горизонтальній і вертикальній площинах в межах 130-220 мм та змінювати кут нахилу на 10-15 мм [26].

Клавіатура розміщується на робочому або окремому столі на відстані 100-300 мм від краю з боку користувача. Кут нахилу клавіатури повинен забезпечувати комфортне користування [26].

Крісло повинно підтримувати раціональну робочу позу, мати регульовану підйомно-поворотну конструкцію. Висота сидіння – 400-500 мм, ширина і глибина – не менше 400 мм. Кут нахилу сидіння – до 15° вперед і до 5° назад. Спинка висотою приблизно 300 мм, шириною – не менше 380 мм, з кутом нахилу від 1° до 30°. Стрілець повинен бути оснащений підлокітниками завдовжки не менше 250 мм і завширшки 50-70 мм, які регулюються по висоті в межах 230-260 мм і відстані між ними 350-500 мм. Поверхня крісла повинна бути напівм’якою, з покриттям, що не електризується, неслизьке та повітронепроникне, для легкого очищення [26].

Робочі місця повинні розташовуватися так, щоб природне світло падало переважно з лівого боку. Освітленість на горизонтальній площині робочого місця повинна складати 400 лк при висоті 0,8 м над підлогою. Штучне освітлення повинно забезпечувати комбіноване освітлення з рівнем 1500 лк або загальне освітлення з рівнем 400 лк [26].

Результати досліджень показують, що негативний вплив на операторів ПК пов'язаний з дискомфортними зоровими умовами через неправильно спроектоване освітлення. Згідно ДБН В.2.5-28:2018, освітленість на робочому місці оператора ПК повинна бути 400 лк при висоті 0,8 м над підлогою.

Яскравість свічення екрану має бути не менше 100 кд/м². Мінімальний розмір світної точки для кольорового дисплея – не більше 0,6 мм; контрастність зображення знаку – не менше 0,8; низькочастотне тремтіння зображення в діапазоні 0,05-1,0 Гц – в межах 0,1 мм; екран повинен мати покриття антивідблиску [26].

Температура, вологість та швидкість руху повітря на робочих місцях мають відповідати нормам. Висока температура негативно впливає на стан людини, тому важливо забезпечити ефективний відвід повітря. Рівень шуму на робочих місцях користувачів ПК повинен відповідати санітарним нормам і не перевищувати 60 дБ [26].

Для забезпечення комфортних умов роботи необхідно дотримуватися таких параметрів повітряного середовища:температура 18-22°С, вологість 40-60%, швидкість руху повітря до 0,1 м/с. Коефіцієнт відбиття світла для стелі має бути 60-70%, стін – 40-50%, підлоги – 30%, інших поверхонь – 30-40% [26].

Правильна організація робочих місць сприяє усуненню загального дискомфорту, зменшенню втомлюваності працівника, підвищенню його продуктивності. Проведені дослідження показують, що при раціональній організації робочих місць продуктивність праці зростає на 15-25% [27].

4.3 Висновок до четвертого розділу

В четвертому розділі кваліфікаційної роботи описано основні аспекти управління ризиками та організації робочих місць, зокрема для операторів ПК.

Ризик визначається як добуток частотної ймовірності виникнення небезпеки та шкоди, яку вона може завдати. Оцінка серйозності небезпек та комбінування категорій серйозності ймовірності дозволяють створити матрицю оцінки ризику для визначення необхідних заходів безпеки.

Управління ризиками передбачає зниження їх до прийнятних рівнів, включаючи відмову від небезпечних робіт, заміну небезпечних операцій, удосконалення технічних систем та організаційні заходи. Прийнятний ризик

визначається як компроміс між бажаним рівнем безпеки та можливостями його досягнення.

Організація робочого місця оператора ПК повинна враховувати ергономічні вимоги для зменшення втомлюваності та підвищення продуктивності. Це включає правильне розташування обладнання, регульованість висоти столу та стільця, оптимальне розміщення монітора та інші параметри для забезпечення безпечних і комфортних умов праці.

Дотримання оптимальних параметрів повітряного середовища, освітленості та рівня шуму є важливим для підтримання функціонального стану працівників і їхньої продуктивності. Правильна організація робочих місць сприяє усуненню загального дискомфорту, зменшенню втомлюваності та підвищенню ефективності праці.

ВИСНОВКИ

У кваліфікаційній роботі освітнього рівня «Бакалавр» було проведено детальний аналіз предметної області, досліджено існуючі рішення, обґрунтовано необхідність розробки нового інструменту для візуалізації та аналізу крипто-активів, а також здійснено реалізацію та тестування цього інструменту. Крім того, розглянуто аспекти безпеки життєдіяльності та основи охорони праці.

Перший розділ роботи присвячено детальному аналізу предметної області крипто-активів. Розглянуто основні аспекти крипто-валют, їх використання та пов'язані з ними виклики. Проведено огляд існуючих рішень для аналізу крипто-активів, виявлено їхні обмеження. На основі цього обґрунтовано вибір технологій для розробки нового інструменту, зокрема мови програмування Python та середовища розробки PyCharm Edu.

Другий розділ зосереджено на проектуванні інструменту для візуалізації та аналізу крипто-активів. Визначено архітектуру інструменту, яка забезпечує модульність та розширюваність. Детально описано інтерфейс користувача, який забезпечує зручний доступ до функцій інструменту, а також інтеграцію з зовнішніми API для отримання актуальних даних про крипто-активи.

У третьому розділі описано процес реалізації інструменту для візуалізації та аналізу крипто-активів на основі Python з використанням бібліотеки tkinter. Реалізовано функції для збору, обробки та візуалізації даних. Створено інтуїтивно зрозумілий користувацький інтерфейс, що спрощує взаємодію користувача з інструментом. Проведено тестування інструменту, яке показало його здатність виконувати основні завдання та надавати актуальну інформацію про крипто-активи, але також виявило необхідність деяких доопрацювань.

У розділі «Безпека життєдіяльності, основи охорони праці» висвітлено основні аспекти управління ризиками та організації робочих місць для операторів ПК. Описано методи оцінки та управління ризиками, а також вимоги до ергономіки робочих місць. Визначено параметри оптимального повітряного

середовища, освітленості та рівня шуму для забезпечення безпечних та комфортних умов праці.

Таким чином, у кваліфікаційній роботі закладено основи для створення ефективного інструменту для візуалізації та аналізу крипто-активів, а також враховано важливі аспекти безпеки та ергономіки робочих місць. Це сприятиме прийняттю обґрунтованих інвестиційних рішень та зниженню ризиків, пов'язаних з ринком крипто-валют.

ПЕРЕЛІК ДЖЕРЕЛ

1 Використання цифрових технологій. [Електронний ресурс] – Режим доступу до ресурсу: https://unba.org.ua/news/8874-vikoristannya-cifrovih-tehnologij-vimagae-novih-standartiv-zahistu-prav-lyudini.html

2 Тенденція розвитку криптовалюти в україні [Електронний ресурс] – Режим доступу до ресурсу: https://elartu.tntu.edu.ua/bitstream/lib/40095/2/III_MNPK_2022_Budzanivska_T-Trend_of_cryptocurrency_22-23.pdf

3 Що таке криптовалюта та що з нею можна робити? [Електронний ресурс] – Режим доступу до ресурсу: https://tyzhden.ua/shcho-take-kryptovaliuta-ta-shcho-z-neiu-mozhna-robyty/

4 Що таке альткоїни, токени та криптовалюти [Електронний ресурс] – Режим доступу до ресурсу: https://incrypted.com/ua/shcho-take-altkojiny-tokeny-ta-kryptovaljuty/

5 Що таке DAO? [Електронний ресурс] – Режим доступу до ресурсу: https://academy.binance.com/uk/articles/decentralized-autonomous-organizations-daos-explained

6 Що таке DeFi і як вони працюють? [Електронний ресурс] – Режим доступу до ресурсу: https://blog.whitebit.com/uk/what-is-defi-and-how-does-it-work/

7 Криптовалюти – загрози та перспективи функціонування [Електронний ресурс] – Режим доступу до ресурсу: https://elartu.tntu.edu.ua/bitstream/lib/34917/2/FMZKPNES_2021_Krupka_A-Cryptocurrency-threats_and_89-92.pdf

8 Як захистити свій крипто гаманець [Електронний ресурс] – Режим доступу до ресурсу: https://stormgain.com/uk/blog/how-protect-your-crypto-wallet

9 Керівництво по TradingView для новачків [Електронний ресурс] – Режим доступу до ресурсу: https://academy.binance.com/uk/articles/a-beginner-s-guide-to-tradingview

10 Огляд Cryptocompare [Електронний ресурс] – Режим доступу до ресурсу: https://happycoin.club/41066-2/

11 12 курсів з мови програмування R [Електронний ресурс] – Режим доступу до ресурсу: https://dev.ua/news/12-kursiv-z-movy-prohramuvannia-r-vid-bazovykh-do-superprosunutykh

12 Знайомство з Visual Studio Code [Електронний ресурс] – Режим доступу до ресурсу: https://romul.name/blog/znayomstvo-z-visual-studio-code/

13 Eclipse Java: середовище розробки “під себе” [Електронний ресурс] – Режим доступу до ресурсу: https://javarush.com/ua/groups/posts/uk.2359.eclipse-java-seredovijshe-rozrobki-pd-sebe

14 UML для бізнес-моделювання [Електронний ресурс] – Режим доступу до ресурсу: https://evergreens.com.ua/ua/articles/uml-diagrams.html

15 Coingecko introduction [Електронний ресурс] – Режим доступу до ресурсу: https://docs.coingecko.com/reference/introduction

16 Etherscan introduction [Електронний ресурс] – Режим доступу до ресурсу: https://docs.etherscan.io/

17 Графічний інтерфейс користувача з Tk¶ [Електронний ресурс] – Режим доступу до ресурсу: https://docs.python.org/uk/3.10/library/tk.html

18 Requests: HTTP for Humans [Електронний ресурс] – Режим доступу до ресурсу: https://requests.readthedocs.io/en/latest/

19 Pandas documentation [Електронний ресурс] – Режим доступу до ресурсу: https://pandas.pydata.org/docs/

20 ccxt – documentation [Електронний ресурс] – Режим доступу до ресурсу: https://docs.ccxt.com/#/

21 10 найкращих бібліотек Python для машинного навчання та ШІ [Електронний ресурс] – Режим доступу до ресурсу: https://www.unite.ai/uk/10-best-python-libraries-for-machine-learning-ai/

22 У пошуковику Google тепер можна відстежити криптогаманець і побачити його баланс [Електронний ресурс] – Режим доступу до ресурсу:

https://incrypted.com/ua/u-poshukovyku-google-teper-mozhna-vidstezhyty-kryptogamanets-i-pobachyty-jogo-balans/

23 Безпека життєдіяльності. Ризик як кількісна оцінка небезпек. [Електронний ресурс] – Режим доступу до ресурсу: http://opcb.kpi.ua/wp-content/uploads/2021/08/Lekc_2_BGD_CZ_2021.pdf

24 Основи небезпеки та ризику. [Електронний ресурс] – Режим доступу до ресурсу: https://naurok.com.ua/metodichni-materiali-do-temi-osnovi-nebezpeki-ta-riziku-280445.html

25 Ергономіка робочих місць [Електронний ресурс] – Режим доступу до ресурсу: http://univer.nuczu.edu.ua/tmp_metod/2557/Kurs_lekcij_po_ERM.PDF

26 ДСТУ 8604:2015 Дизайн і ергономіка [Електронний ресурс] – Режим доступу до ресурсу: https://online.budstandart.com/ua/catalog/doc-page.html?id_doc=71028

27 Ергономічні вимоги до організації робочих місць [Електронний ресурс] – Режим доступу до ресурсу: https://moodle.znu.edu.ua/pluginfile.php/554266/mod_resource/content/л13.pdf

ДОДАТКИ