Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних наук

Бакалаврська кваліфікаційна робота на тему:

ПРОГРАМНИЙ МОДУЛЬ ПРОГНОЗУВАННЯ ЦІН АКЦІЙ ФОНДОВОГО РИНКУ

Виконала: студентка 4 курсу, групи 2КН-21б
спеціальності 122 – Комп'ютерні науки
(шифр і назва напрямку підготовки, спеціальності)

_____ Краснолуцька Я. В.
(прізвище та ініціали)

Керівник: к.ф-м.н., доц. каф. КН

_____ Шевчук О.Ф.
(прізвище та ініціали)

«04» червня 2025 р.

Рецензент: к.т.н., доцент каф. САІТ

_____ Горячев Г.В.
(прізвище та ініціали)

«05» червня 2025 р.

Допущено до захисту
Зав. кафедри Яровий А. А.
«06» червня 2025 р.

Вінниця ВНТУ – 2025 р.

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних наук
Рівень вищої освіти перший (бакалаврський)
Галузь знань – 12 Інформаційні технології
Спеціальність – 122 Комп'ютерні науки
Освітньо - професійна програма – Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри КН
проф., д.т.н. Яровий А.А.
«05» травня 2025 р.

ЗАВДАННЯ
НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Краснолуцькій Яні Віталіївні

1. Тема роботи: Програмний модуль для прогнозування цін акцій фондового ринку.
Керівник роботи: Шевчук Олександр Федорович, к.ф.-м.н., доц.
Затверджені наказом вищого навчального закладу "10" 03 2025 року № 97
2. Термін подання студентом роботи 30 травня 2025 р.
3. Вихідні дані до роботи:
Мова програмування – об'єктно-орієнтована;
мінімальна кількість представлених акцій – 3;
мінімальна кількість історичних даних часового ряду – 365;
горизонт прогнозу – 30 днів;
тип інтерфейсу користувача – інтуїтивно-зрозумілий.
4. Зміст текстової частини (перелік питань, які потрібно розробити):
- проаналізувати сучасні засоби прогнозування цін акцій фондового ринку;
- розробити алгоритм роботи програмного модуля прогнозування цін акцій фондового ринку;
- спроєктувати та реалізувати модуль прогнозування цін акцій фондового ринку;
- провести тестування програмного модуля та розробити інструкцію користувача.
5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень):
- структура програмного модуля прогнозування цін акцій;
- блок-схема алгоритму роботи модулю;
- Use Case-діаграма взаємодії користувача з системою;
- діаграма архітектури програмного рішення;
- інтерфейс веб-додатку (Streamlit) з прикладом побудови прогнозу.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-3	Шевчук О.Ф., к.ф-м.н., доц. каф. КН		

7. Дата видачі завдання 06 травня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва та зміст етапу	Термін виконання		Примі
		початок	закінчення	
1	Аналіз сучасних засобів прогнозування цін акцій фондового ринку	05.05.25	07.05.25.	
2	Розробка алгоритму роботи програмного модуля прогнозування цін акцій фондового ринку	08.05.25.	11.05.25.	
3	Розробка структури програмного модуля прогнозування цін акцій фондового ринку	12.05.25.	15.05.25	
4	Реалізація програмного модуля прогнозування цін акцій фондового ринку	16.05.25.	29.05.25.	
5	Розробка інструкції користувача	30.05.25	01.06.25	
6	Оформлення матеріалів до захисту БКР	02.06.25	04.06.25	

Студент _____ (підпис) **Краснолуцька Я. В.** (прізвище та ініціали)

Керівник проекту (роботи) _____ (підпис) **Шевчук О.Ф.** (прізвище та ініціали)

АНОТАЦІЯ

Краснолуцька Я.В. Програмний модуль для прогнозування цін акцій фондового ринку. Бакалаврська дипломна робота складається з 77 сторінок формату А4, на яких представлено 26 рисунків та 5 таблиць, список використаних джерел містить 20 найменувань.

Дипломна робота присвячена розробці програмного модуля для прогнозування динаміки цін акцій на основі історичних біржових даних. У процесі дослідження було проаналізовано сучасні методи прогнозування, проведено порівняння програм-аналогів та обґрунтовано вибір моделі ARIMA як ефективного інструменту для короткострокового фінансового аналізу. Розробка здійснювалася у два етапи: спочатку створено дослідницький прототип у Jupyter Notebook, далі - повнофункціональний застосунок із графічним інтерфейсом користувача на платформі Streamlit. Реалізовано функціонал побудови прогнозу на 30 днів, оцінку точності моделі, а також додаткові модулі: стрічку фінансових новин та інвестиційний калькулятор.

Програмний модуль успішно протестовано, що підтвердило його коректну роботу, зручність використання та можливість практичного застосування в інвестиційній діяльності.

Ключові слова: прогнозування, фінансові часові ряди, ARIMA, акції, Streamlit, інтерфейс.

ABSTRACT

Krasnolutska Y.V. Software module for stock market price forecasting. The bachelor's thesis consists of 77 A4 pages, including 26 figures and 5 tables. The list of references contains 20 sources.

This thesis is dedicated to the development of a software module for forecasting stock price dynamics based on historical market data. During the research, modern forecasting methods were analyzed, existing analog software solutions were reviewed, and the ARIMA model was chosen as an effective tool for short-term financial prediction.

The development process was carried out in two stages: first, a research prototype was created in Jupyter Notebook; then, a fully functional user application was implemented using the Streamlit platform. The system supports stock price forecasting for 30 days ahead, model accuracy evaluation, and includes additional modules such as a financial news feed and an investment return calculator.

The software module was thoroughly tested, confirming its correct operation, user-friendliness, and suitability for practical use in investment analysis.

Keywords: forecasting, financial time series, ARIMA, stocks, Streamlit, user interface.

ЗМІСТ

ВСТУП

Фондовий ринок є ключовим елементом фінансової системи будь-якої країни, що забезпечує механізм перерозподілу капіталу, інвестування та оцінки економічної активності. Динаміка цін акцій безпосередньо впливає на рішення інвесторів, діяльність компаній і стабільність ринкової економіки в цілому. Прогнозування вартості цінних паперів є важливим інструментом для прийняття обґрунтованих рішень у сфері інвестицій та управління ризиками.

Традиційні підходи до аналізу фондового ринку зазвичай ґрунтуються на експертному аналізі графіків, економічних новин або фінансових звітів компаній. Однак такий підхід є суб’єктивним, залежним від людського фактору та обмеженим у можливостях обробки великих обсягів даних. В умовах стрімкого розвитку технологій та наявності відкритих джерел фінансової інформації постає необхідність створення автоматизованих інструментів прогнозування, які здатні забезпечити більш точну, швидку та об'єктивну оцінку ринкових тенденцій.

Розробка програмного модуля для прогнозування цін акцій дозволяє створити ефективний інструмент для приватних інвесторів, фінансових аналітиків, трейдерів та дослідників. Такий модуль може надавати користувачу прогнозні значення на основі математичних моделей, здійснювати візуалізацію історичних і прогнозованих даних, а також враховувати інформаційний фон фондового ринку.

Метою дослідження є розширення функціональних можливостей програмного модуля прогнозування цін акцій фондового ринку.

Об’єктом дослідження є процеси аналізу та прогнозування динаміки цін акцій фондового ринку.

Предметом дослідження є алгоритми та програмні засоби, що організовують оцінку та прогнозування цін акцій фондового ринку.

Задачі дослідження:

1. Аналіз предметної області.
2. Порівняльний аналіз сучасних програмних засобів прогнозування цін акцій фондового ринку.

3. Проєктування програмного модуля прогнозування цін акцій фондового ринку.

4. Програмна реалізація модуля прогнозування цін акцій фондового ринку.

5. Тестування функціоналу програмного модуля прогнозування цін акцій фондового ринку.

6. Розробка інструкції користувача.

Апробація роботи.

Робота апробувалась на конференції «Молодь у науці: дослідження, проблеми, перспективи, 2025р.», доповідь «Особливості розробки програмного модуля прогнозування цін акцій фондового ринку».

Публікації: електронні тези конференції «Молодь у науці: дослідження, проблеми, перспективи, 2025р.» [1]

Зареєстровано авторське право на твір № 136468 у формі комп'ютерної програми – «Програмний модуль прогнозування цін акцій фондового ринку». Оригіналом цього документа є електронний документ з ідентифікатором: CR2570220525. [2]

1 АНАЛІЗ СУЧАСНИХ ЗАСОБІВ ПРОГНОЗУВАННЯ ЦІН АКЦІЙ ФОНДОВОГО РИНКУ

1.1 Аналіз предметної області

Фондовий ринок є однією з основних складових фінансової системи, що має значний вплив на економічний розвиток держави. Це ринок, на якому обертаються різноманітні цінні папери, що дає можливість мобілізувати ресурси для розвитку компаній, урядів та інших фінансових інститутів. Завдяки фондовому ринку капітали інвесторів можуть бути направлені в економіку, що сприяє зростанню ринкової капіталізації компаній, розвитку інфраструктури і підвищенню загального рівня життя населення. Водночас фондовий ринок є індикатором економічної ситуації в країні, адже його динаміка відображає зміну очікувань учасників ринку щодо майбутнього розвитку економіки [2].

Один із основних фінансових інструментів на фондовому ринку - це акції. Вони засвідчують право власника на частку в капіталі акціонерного товариства, а також на участь у розподілі прибутку компанії. Акції поділяються на два основні типи: прості (звичайні) та привілейовані. Прості акції надають власникам право голосу на загальних зборах акціонерів, а також можливість отримувати дивіденди в залежності від фінансових результатів компанії. Привілейовані акції, як правило, забезпечують стабільний дохід за фіксованою ставкою, проте вони не надають права голосу при прийнятті рішень на рівні компанії [3].

Ціни на акції та інші цінні папери визначаються під впливом багатьох чинників. Серед них основними є фінансові результати компаній, економічні новини, зміна макроекономічних показників, політична ситуація, геополітичні ризики та дії регуляторів ринку. Тривалість та інтенсивність впливу цих факторів може суттєво відрізнятись, що ускладнює процес прогнозування цін на ринку. Важливо також відзначити, що значну роль у цьому процесі відіграють психологічні аспекти, зокрема інвесторські настрої та масові емоції, які здатні спричиняти ірраціональні рухи на ринку [4].

Інфраструктура фондового ринку включає низку важливих учасників і елементів. Основними учасниками є компанії-емітенти, які пропонують свої цінні папери для продажу на ринку, а також інвестори, як приватні, так і інституційні, які шукають можливості для розміщення своїх капіталів.

До інших важливих учасників належать фондові біржі, брокерські та дилерські компанії, депозитарії, клірингові установи, а також регуляторні органи, які контролюють дотримання законодавства та забезпечують стабільність ринку. Технічна інфраструктура фондового ринку базується на електронних торгових системах, платформах для зберігання й обліку цінних паперів, а також інформаційно-аналітичних сервісах, які дозволяють учасникам ринку отримувати актуальні дані для прийняття рішень [5].

Сучасний фондовий ринок доступний не тільки для професіоналів, а й для широкого загалу. Завдяки розвитку онлайн-брокерських сервісів, мобільних застосунків та фінансових платформ, можливість здійснення операцій з цінними паперами стала доступною для значної кількості інвесторів. Цей процес значно демократизував ринок, що дозволяє мільйонам людей брати участь у фінансових операціях, що раніше були доступні лише вузькому колу осіб. Однак з цим зростанням попиту на доступні інструменти, виникають нові виклики, пов’язані з необхідністю постійного покращення інтерфейсів і швидкості доступу до інформації, що зумовлює розробку більш зручних та надійних програмних рішень для цього сегменту ринку [6].

Фондовий ринок виконує кілька ключових функцій. По-перше, він є джерелом капіталу для компаній, даючи їм змогу залучати фінансові ресурси для розвитку. По-друге, він є важливим інструментом для громадян, які хочуть інвестувати свої заощадження і отримувати прибуток від росту вартості акцій. По-третє, фондовий ринок служить індикатором економічного стану країни, відображаючи загальні настрої ринку та майбутні прогнози щодо економічного зростання чи спаду. Також фондовий ринок визначає справедливу вартість активів на основі попиту і пропозиції, що виникають в результаті взаємодії численних учасників ринку, кожен з яких має свої інвестиційні цілі [7].

Поведінка інвесторів на фондовому ринку є дуже різноманітною, що залежить від індивідуальних стратегій, рівня досвіду та знань. Одні інвестори орієнтуються на довгострокові стратегії, спрямовані на стабільне отримання доходу, інші ж зосереджуються на короткострокових спекуляціях, намагаючись отримати швидкий прибуток від коливань на ринку.

Технічний аналіз, що включає вивчення графіків цін та використання різноманітних індикаторів, є основним інструментом для спекулянтів, тоді як фундаментальний аналіз є більш притаманним інвесторам, які шукають довгострокові перспективи для розміщення своїх капіталів. Психологічні чинники, зокрема колективний настрій, страх чи ейфорія, можуть мати суттєвий вплив на ринок, що веде до ірраціональних коливань цін [8].

В інформаційній ері фондовий ринок сильно залежить від швидкості та якості інформації, доступної для учасників ринку. Дані про ціни, обсяги торгів, фінансові звіти, новини та прогнози є основою для прийняття рішень. У сучасному світі інформація повинна бути оперативною, точними і доступною у будь-який час. Інструменти візуалізації, такі як графіки, діаграми та інтерактивні аналітичні панелі, дозволяють інвесторам швидко оцінювати ситуацію на ринку та приймати зважені рішення [9].

Одним із основних джерел фінансових даних є Yahoo Finance API, яке надає доступ до історичних котирувань, фінансових показників та інших ринкових даних. Оскільки API надає відкритий доступ до великого обсягу інформації, це створює можливість для автоматизації процесів збору, обробки та аналізу фінансових даних, що дає змогу створювати ефективні програмні рішення для інвесторів та аналітиків, надаючи їм необхідні інструменти для прийняття рішень [10].

Вплив новин є важливою складовою фондового ринку. Зміни в керівництві компаній, результати квартальних звітів, новини щодо економічної політики чи геополітичних подій можуть миттєво вплинути на ціни акцій. Тому своєчасний доступ до якісної та надійної інформації є критичним для будь-якого учасника ринку. Завдяки сучасним технологіям новини швидко інтегруються у фінансові

платформи, що дозволяє інвесторам оперативно реагувати на зміни в ситуації на ринку [11].

Отже, фондовий ринок є складною і багатогранною системою, яка включає в себе економіку, фінанси, інфраструктуру та поведінкову економіку. Він функціонує як індикатор загальних економічних тенденцій і забезпечує ефективне перерозподілення фінансових ресурсів, що дозволяє оптимізувати процеси економічного розвитку. Для цього необхідні інструменти, що дозволяють здійснювати аналіз та прогнозування руху на ринку. Розробка програмних рішень для роботи з фондовим ринком є надзвичайно важливим завданням в умовах динамічних змін на фінансових ринках, оскільки правильний аналіз і своєчасне прогнозування дозволяють значно підвищити ефективність інвестицій.

1.2 Порівняльний аналіз програм-аналогів

Сучасна фінансова аналітика активно використовує інструменти прогнозування часових рядів для оцінки змін цін на фондовому ринку. Це зумовлено тим, що фінансові ринки є дуже динамічними та чутливими до різноманітних факторів, що вимагає використання сучасних методів для точного прогнозування. Для цієї мети застосовуються як класичні статистичні моделі, так і методи машинного навчання, кожен з яких має свої особливості та переваги. У цьому розділі розглянуто найпоширеніші програмні рішення, які використовуються для прогнозування фінансових показників, зокрема Prophet, QuantConnect та Google Vertex AI.

Одним із найвідоміших інструментів є Prophet - бібліотека з відкритим кодом, розроблена компанією Meta (Facebook). Prophet базується на адитивній моделі, що поєднує тренд, сезонність та святкові дні, що дозволяє моделювати як лінійні, так і нелінійні тренди. Однією з основних переваг Prophet є його простота використання, що робить його зручним для аналітиків без глибокої математичної підготовки.

Завдяки своїй здатності працювати з нерегулярними та неповними даними, Prophet дуже популярний серед початківців та малих компаній, які не мають

великих обчислювальних потужностей чи спеціалізованих знань у галузі машинного навчання. Однак модель менш ефективна при аналізі високоволатильних ринкових даних, таких як акції чи криптовалюти, де коливання цін є значними та нестабільними. Для таких випадків існують більш спеціалізовані моделі, які можуть краще відображати особливості фінансових ринків. На рисунку 1.1 зображено приклад передбачення часових рядів з допомогою Prophet.

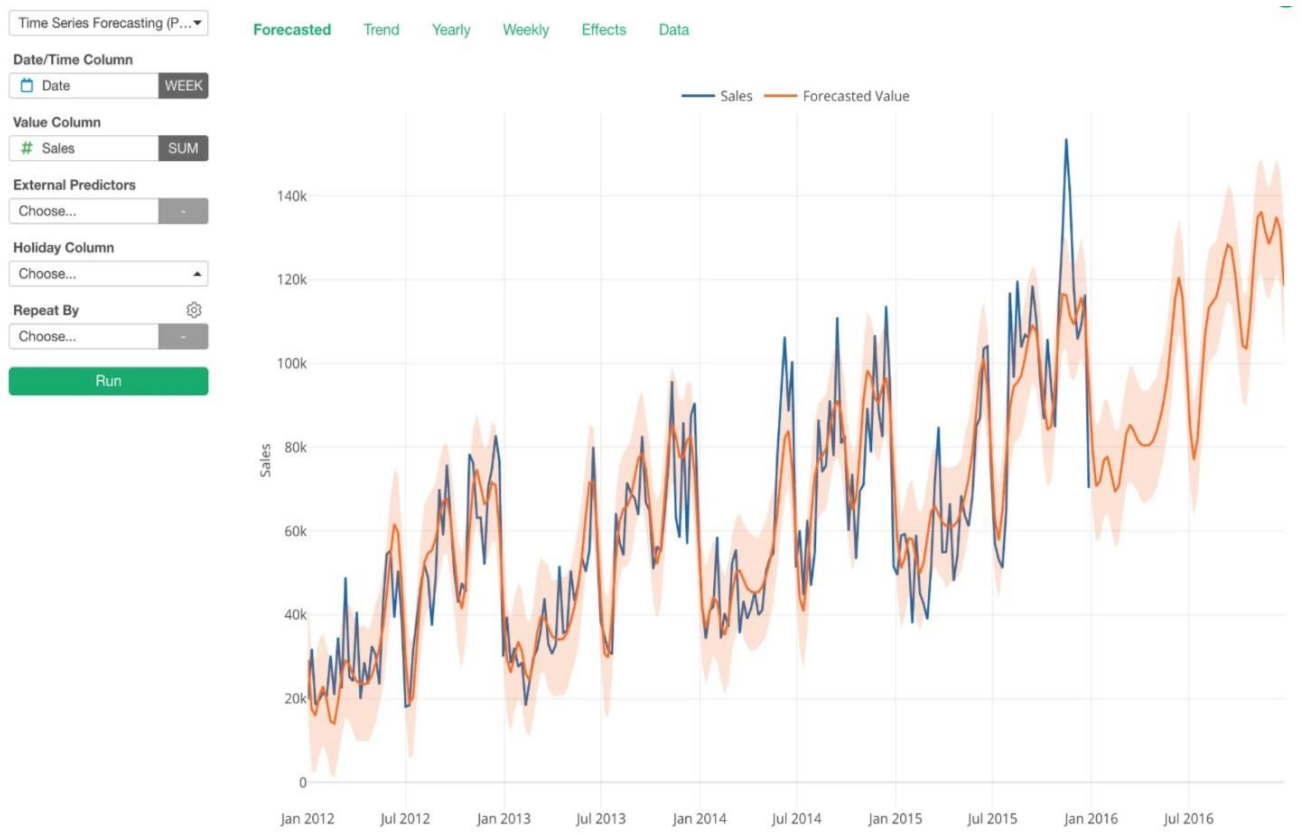

Рисунок 1.1 – Передбачення часових рядів з допомогою Prophet

QuantConnect є однією з найбільш розвинених платформ для алгоритмічної торгівлі, що надає доступ до величезного набору інструментів для фінансового аналізу. Платформа підтримує моделі глибокого навчання, зокрема Long Short-Term Memory (LSTM) та дерева рішень, що дозволяє проводити складний аналіз даних з високою точністю. QuantConnect орієнтований переважно на досвідчених

користувачів, оскільки потребує знань з програмування, структурування торгових стратегій та роботи з API.

Однією з головних переваг платформи є її висока гнучкість і підтримка великих обсягів даних, що дозволяє працювати з історичними даними фінансових інструментів, таких як акції, ф'ючерси та криптовалюти, а також з потоковими даними в режимі реального часу. Однак одним із недоліків є складність конфігурації платформи та її залежність від хмарної інфраструктури, що може створювати проблеми при великому обсязі даних або вимогах до швидкості обробки.

На рисунку 1.2 наведено приклад графіка активів для AAPL в QuantConnect.

Google Vertex AI є одним із найбільш передових рішень для автоматизованого машинного навчання. Платформа підтримує сервіс AutoML Tables, який дозволяє користувачам без великих знань у галузі програмування швидко створювати та навчати моделі для прогнозування. AutoML автоматично вибирає архітектуру моделей, проводить оптимізацію гіперпараметрів і генерує звіти з точності, що робить цей інструмент надзвичайно зручним для тих, хто не має глибоких знань у машинному навчанні, але хоче отримати точні прогнози.

Проте, незважаючи на свою зручність, Vertex AI обмежує контроль над процесом моделювання, що може бути критичним для спеціалізованих фінансових задач, де потрібна максимальна налаштуваність і точність. До того ж, обмежений доступ до метрик і можливості тонкої настройки може обмежити застосування для складних та високочутливих фінансових ситуацій.

На рисунку 1.3 зображено приклад передбачення часових рядів з допомогою Google Vertex AI.

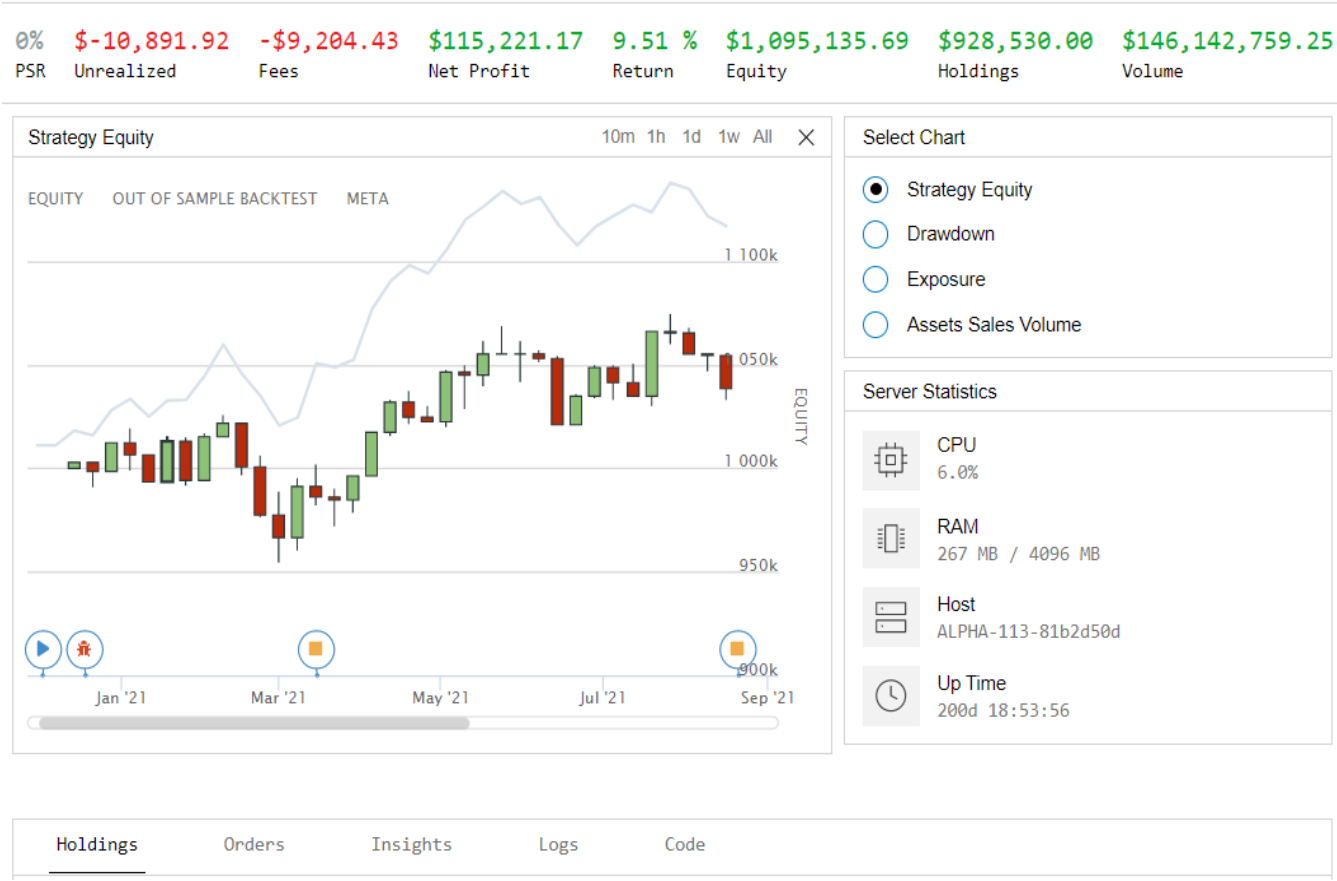

Рисунок 1.2 – Приклад графіка активів для AAPL в QuantConnect

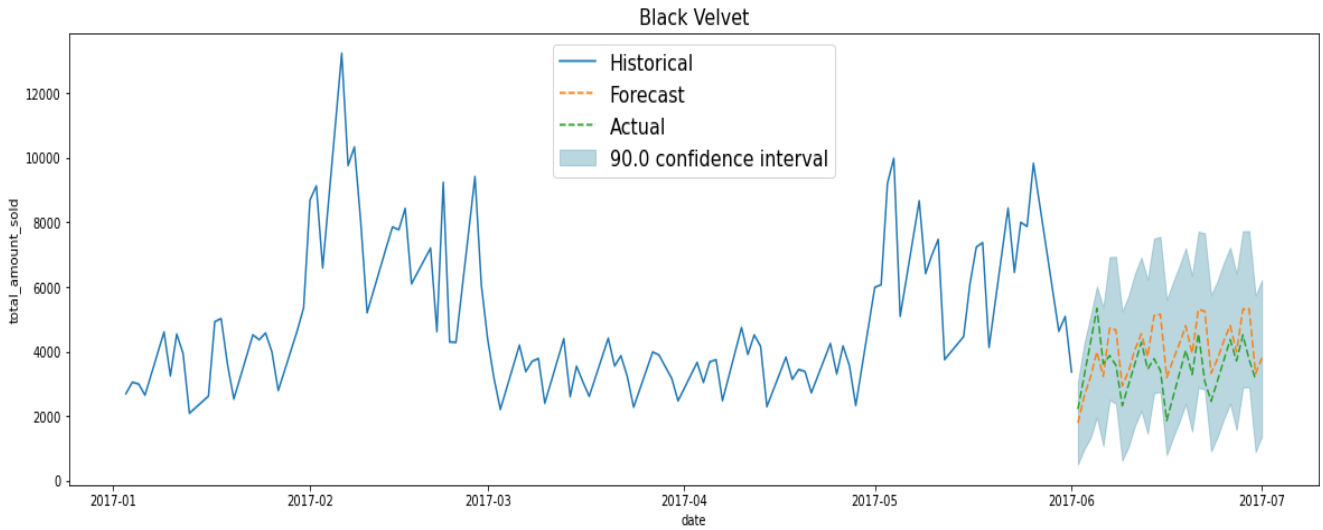

Рисунок 1.3 – Передбачення часових рядів з допомогою Google Vertex AI

Для більш детального порівняння цих трьох рішень, в таблиці 1.1 наведено їхні основні характеристики.

Таблиця 1.1 – Порівняння існуючих програм

Характеристика	**Prophet**	**QuantConnect**	**Google Vertex AI**
Тип моделі	Адитивна модель тренду	ML/AI (LSTM, Trees)	AutoML (нейронні мережі, дерева)
Гнучкість налаштування	Обмежена	Висока	Обмежена
Потреба в хмарних ресурсах	Ні	Так	Так
Інтерпретованість результатів	Середня	Низька	Низька
Джерело даних	API або CSV	API	BigQuery/API
Можливість локального використання	Так	Ні	Ні
Візуалізація	Вбудована	Частково	Автоматична

Як видно з таблиці, вибір інструменту залежить від специфіки задачі. Prophet є ідеальним для простих часових рядів і для початкового аналізу, що не вимагає значних ресурсів або складної налаштування. Він може бути корисним для швидких аналізів, однак його можливості обмежені при роботі з високоволатильними ринками або складними трендами.

QuantConnect надає великий набір інструментів для досвідчених трейдерів та дослідників, що хочуть розробити складні алгоритми торгівлі або фінансові моделі. Однак для початківців ця платформа може бути надто складною, вимагаючи знань з програмування і структурування даних. Вона підходить для роботи з великими даними та високою гнучкістю налаштування.

Google Vertex AI, в свою чергу, ідеально підходить для швидкого створення моделей на основі автоматизованого машинного навчання, але є обмеження в

точності налаштувань і повному контролі над процесом моделювання. Його можна порекомендувати користувачам, які хочуть швидко отримати прогнози без необхідності глибокого розуміння технічних аспектів моделей машинного навчання.

Незважаючи на розбіжності між цими платформами, всі вони сприяють підвищенню точності фінансового прогнозування та дають змогу приймати більш обґрунтовані рішення в умовах високої ринкової невизначеності. Залежно від потреб користувача, рівня його досвіду та специфіки задачі, кожен з цих інструментів має свої переваги та недоліки, що робить їх використання доцільним у різних контекстах.

Розвиток таких інструментів і технологій допомагає фінансовим аналітикам та трейдерам покращити прогнозування ринкових трендів і приймати стратегічно важливі інвестиційні рішення в умовах швидкозмінного фінансового середовища.

1.3 Обґрунтування вибору методу прогнозування цін акцій фондового ринку

У процесі створення програмного модуля для прогнозування цін акцій важливою складовою є обґрунтований вибір математичного підходу, який дозволяє ефективно моделювати часові ряди. Прогнозування динаміки цін на фондовому ринку є складною задачею через високу волатильність, залежність від зовнішніх чинників, наявність трендів і сезонності, а також значний рівень шуму у даних. У цьому контексті актуальним є аналіз сучасних методів прогнозування та їх придатності до використання в прикладних інструментах.

Серед найбільш поширених підходів до прогнозування часових рядів варто виокремити методи класичної статистики, машинного навчання та гібридні моделі. У контексті прогнозування цін акцій на фондовому ринку, класичні статистичні моделі є популярними завдяки їх простоті, інтерпретованості та низьким вимогам до обсягу даних. Одним з найбільш відомих методів є модель ARIMA (Autoregressive Integrated Moving Average), яка дозволяє ефективно працювати з

нестаціонарними часними рядами, де відсутні сталий тренд або сезонність. Модель ARIMA базується на припущенні, що майбутні значення часового ряду залежать від його попередніх значень, а також від похибок у попередніх прогнозах. Іншими словами, вона аналізує автокореляцію даних, що дозволяє врахувати взаємозв'язки між спостереженнями на різних етапах часу [12].

Модель SARIMA (Seasonal ARIMA) є розширенням ARIMA і додатково включає сезонні компоненти, що є важливими для фінансових рядів, в яких існує виражена сезонність. Наприклад, сезонні зміни можуть бути пов'язані з річними циклами в прибутках компаній або загальноекономічними трендами.

Популярними є також методи експоненціального згладжування, такі як Holt-Winters, які використовуються для прогнозування на основі трендів і сезонності. Ці методи також досить ефективні в умовах, коли дані мають чітко виражену сезонність, хоча вони можуть мати обмежену здатність до роботи з високоволатильними ринками [13].

В останні роки методи машинного навчання набувають дедалі більшої популярності у фінансовому прогнозуванні. Вони здатні враховувати складні нелінійні залежності та взаємодії між різними ознаками, що робить їх корисними для роботи з даними, які містять складні патерни та високу волатильність. Одним із таких методів є дерева рішень, а також їх ансамблеві варіанти, такі як Random Forest та Gradient Boosting, які здатні працювати з великими обсягами даних і коригувати прогнози на основі нової інформації.

Ще однією важливою категорією є рекурентні нейронні мережі (RNN) та їх різновид LSTM (Long Short-Term Memory), які спеціально розроблені для роботи з часовими рядами. Вони здатні «пам'ятати» попередні стани і передбачати майбутні значення на основі історичних даних. Однак для ефективного використання таких методів потрібні великі обсяги даних для навчання моделей, а також потужні обчислювальні ресурси. Крім того, такі моделі є менш інтерпретованими, що ускладнює їх застосування у фінансових системах, де важлива прозорість і контроль за результатами прогнозування.

Попри численні переваги, методи машинного навчання мають свої обмеження, зокрема необхідність в обробці великих обсягів даних та складність налаштування моделей. Більш того, для таких методів високий ризик перенавчання є важливою проблемою, оскільки вони можуть сильно залежати від вибору гіперпараметрів і характеристик навчальних даних. Перенавчання може призвести до погіршення якості прогнозів на нових, невідомих даних [14].

Для оцінки основних характеристик методів прогнозування було складено порівняльну таблицю (табл. 1.1), яка дозволяє оцінити їх переваги та обмеження.

Таблиця 1.2 – Порівняння методів прогнозування часових рядів

Критерій	**ARIMA**	**LSTM (нейронна мережа)**	**Random Forest**	**Holt-Winters**
Робота з трендом	Так	Так	Частково	Так
Робота з сезонністю	Через SARIMA	Так	Потрібна обробка	Так
Обсяг даних для навчання	Невеликий	Великий	Середній	Невеликий
Інтерпретація результатів	Зрозуміла	Складна	Середня	Зрозуміла
Вимоги до ресурсів	Помірні	Високі	Середні	Низькі
Підходить для фінансових рядів	Так	Так	Частково	Частково
Ризик перенавчання	Низький	Високий	Середній	Низький
Простота реалізації	Висока	Низька	Середня	Висока

Модель ARIMA має низку практичних переваг, що роблять її придатною для прогнозування фондових котирувань у прикладних умовах. Зокрема, вона не потребує великих обсягів даних, має зрозумілу структуру і високу інтерпретованість, що дозволяє відслідковувати зв’язок між параметрами моделі та

результатами прогнозу. Вона також стійка до перенавчання і не вимагає складних обчислень, що є перевагою в умовах обмежених ресурсів.

У той же час, моделі на основі глибокого навчання (наприклад, LSTM) мають потенціал до врахування складних патернів у даних, проте потребують значної кількості навчальних прикладів, потужних обчислювальних засобів і часу на налаштування. Такі моделі складні в інтерпретації, що ускладнює їх використання у програмних модулях, орієнтованих на прозору логіку роботи.

Методи типу Random Forest забезпечують прийнятну точність у деяких випадках, але не враховують часову послідовність даних без додаткової обробки, що знижує їх ефективність у завданнях аналізу часових рядів.

Метод Holt-Winters, хоч і добре працює при чіткій сезонності, менш стабільний при сильній волатильності, характерній для акцій, та вимагає ручного налаштування компонентів.

У рамках цієї роботи було прийнято рішення використовувати **модель** ARIMA для прогнозування цін акцій на фондовому ринку. Основною причиною цього вибору є те, що ARIMA добре підходить для аналізу нестаціонарних часових рядів, типових для фінансових ринків. Модель ARIMA дозволяє працювати з трендами та сезонними коливаннями, що є важливим при прогнозуванні цін акцій, враховуючи можливі економічні цикли та специфіку фондового ринку.

Модель ARIMA має низку переваг, серед яких:

1. Низькі вимоги до обсягу даних – ARIMA є досить ефективною навіть при обмеженому наборі історичних даних, що є важливим, коли доступ до даних обмежений або коли потрібно здійснити швидке прогнозування.

2. Інтерпретованість – одна з основних переваг ARIMA полягає в її прозорості: кожен параметр моделі має чітке економічне або статистичне значення (наприклад, параметри AR, MA, і I), що дозволяє користувачам аналізувати і коригувати моделі за потреби.

3. Низький ризик перенавчання – ARIMA, на відміну від методів машинного навчання, не схильна до перенавчання, що дозволяє знизити ймовірність отримання неадекватних прогнозів у разі зміни умов на ринку.

Таким чином, з огляду на поставлену задачу, обмеженість обсягів навчальних даних, а також орієнтацію на короткострокове прогнозування, модель ARIMA була обрана як найбільш доцільна та ефективна для реалізації програмного модуля прогнозування цін акцій.

1.4 Постановка задачі

Для розробки програмного модуля прогнозування цін акцій фондового ринку застосуємо математичну модель у вигляді:

$$Y = f(X), \tag{1.1}$$

де: Y – вектор прогнозованих значень цін акцій на певному часовому горизонті, отриманих в результаті обробки часових рядів;

$X(T, N, F)$ – вектор вхідних даних, де:

T – історичні дані про динаміку цін акцій TSLA;

N – історичні дані про динаміку цін акцій AAPL;

F – історичні дані про динаміку цін акцій NVDA.

Математична модель прогнозування базується на алгоритмі ARIMA з додатковою обробкою вхідних історичних даних, який дозволяє аналізувати й прогнозувати часові ряди з урахуванням сезонності, трендів і стохастичних компонентів.

Напрями вдосконалення:

1. Інтеграція новинного блоку. Додавання функціоналу для врахування новинного фону, що дозволить враховувати вплив зовнішніх інформаційних факторів на динаміку цін акцій та сприятиме підвищенню якості прийняття управлінських рішень.

2. Фінансовий калькулятор. Реалізація програмного засобу для автоматизованого розрахунку ключових фінансових показників і аналізу динаміки

інвестиційних змін, що дозволить користувачам оперативно оцінювати фінансову ефективність обраних рішень.

1.5 Висновки до розділу

У першому розділі проведено детальний аналіз предметної області, розглянуто особливості функціонування фондового ринку, основних його учасників та факторів, які впливають на зміну цін акцій. Окреслено важливість застосування аналітичних моделей для прогнозування фінансових показників, зокрема в умовах високої волатильності ринку, де людський фактор часто стає джерелом похибок у прийнятті рішень. Аналіз існуючих програм-аналогів продемонстрував, що на ринку вже представлені ефективні, але вузькоспеціалізовані інструменти. Prophet є простим у використанні, але менш придатним для складних ринкових умов. QuantConnect пропонує широку гнучкість і підтримку складних моделей, але вимагає глибоких технічних знань. Google Vertex AI забезпечує швидке створення моделей, проте не дає змоги повністю контролювати процес. Всі ці рішення мають свої переваги й недоліки, що вказує на відсутність універсального підходу для всіх категорій користувачів.

З огляду на це, було сформульовано задачу створення програмного продукту, що об’єднає зручність використання та функціонал для ефективного прогнозування. Передбачено впровадження модуля прогнозування на основі моделі ARIMA, інформаційної сторінки новин і інвестиційного калькулятора, що в сукупності має забезпечити додаткові інструменти як для технічного аналізу, так і для формування обґрунтованих інвестиційних рішень.

2 ПРОЄКТУВАННЯ ПРОГРАМНОГО МОДУЛЯ ПРОГНОЗУВАННЯ ЦІН АКЦІЙ ФОНДОВОГО РИНКУ

2.1 Розробка алгоритму роботи програмного модуля прогнозування цін акцій фондового ринку

Для створення програмного модуля прогнозування цін акцій було розроблено алгоритм його роботи, який описує послідовність дій від отримання вхідних даних до формування прогнозу (рис. 2.1).

На початковому етапі користувач задає параметри: біржовий тикер, період, за який потрібно завантажити дані, та частоту котирувань. Після цього модуль підключається до джерела - Yahoo Finance - і завантажує історичні дані про ціну закриття обраної акції. Для цього використовується бібліотека yfinance, яка дозволяє легко отримувати дані без додаткових ключів API.

Після завантаження даних вони проходять попередню обробку. Видаляються пропущені значення, виконується логарифмування ряду для зменшення впливу великих коливань, а також перевіряється стаціонарність. Якщо ряд нестабільний (має тренд або змінну дисперсію), застосовується диференціювання.

Далі виконується аналіз автокореляцій. На основі автокореляційної (ACF) і часткової автокореляційної (PACF) функцій визначаються параметри моделі ARIMA - тобто, скільки лагів потрібно враховувати для побудови моделі. Це допомагає підібрати оптимальні значення параметрів (p, d, q), де:

- p - порядок авторегресії;
- d - порядок диференціювання;
- q - порядок ковзної середньої.

Після цього модель ARIMA будується та навчається на підготовленому часовому ряді. Для цього використовується бібліотека statsmodels. В результаті отримується набір прогнозованих значень у логарифмічному масштабі.

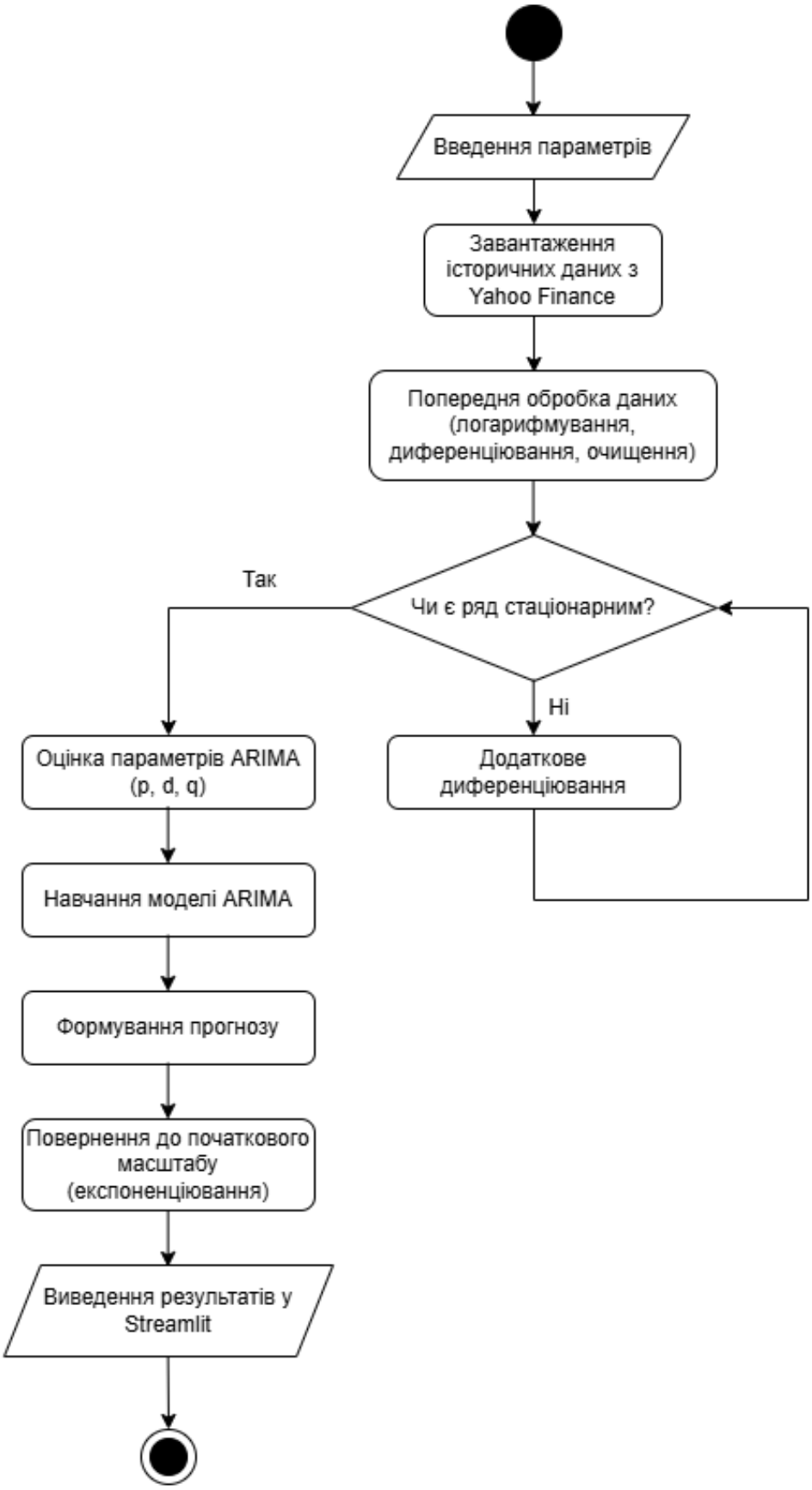

Рисунок 2.1 – Блок-схема алгоритму роботи програмного модуля

Наступний крок - повернення прогнозу до реального масштабу цін. Для цього прогнозовані логарифми спочатку інтегруються (тобто об’єднуються у вигляді суми приростів), а потім до них застосовується експоненціальна функція. Це дозволяє отримати фінальний прогноз у зрозумілому вигляді - у вигляді передбачених цін акцій.

Після формування прогнозу система оцінює його точність. Для цього використовуються такі показники як середня абсолютна відносна похибка (MAPE).

Завершальний етап - виведення результатів у вигляді графіків. Відображаються фактичні та прогнозовані значення, залишки моделі, тренди й сезонні компоненти. Це дозволяє користувачу візуально оцінити якість прогнозу.

Алгоритм побудовано таким чином, щоб він був гнучким - тобто, його легко можна адаптувати для роботи з іншими акціями або часовими проміжками. Також структура алгоритму дозволяє в майбутньому розширити його іншими моделями прогнозування, наприклад SARIMA.

2.2 Розробка структури програмного модуля прогнозування цін акцій фондового ринку

Проєктування програмного модуля прогнозування цін акцій передбачає створення структурованої системи для завантаження, обробки, аналізу та прогнозування часових рядів на основі історичних біржових даних. На початковому етапі було визначено загальну архітектуру модуля та обрано ключові інструменти й бібліотеки, які забезпечують зручну роботу з фінансовими даними. Основними критеріями при проєктуванні стали гнучкість моделі, модульність, прозорість обчислень та можливість подальшого масштабування на інші акції.

Основною метою програмного модуля є побудова прогнозу динаміки цін акцій фондового ринку на основі історичних часових рядів із використанням моделі ARIMA. Обґрунтований вибір цієї моделі зумовлений її широким застосуванням у статистичному аналізі фінансових даних, зокрема для задач короткострокового прогнозування.

Загальна структура модуля складається з таких функціональних компонентів:

1. Модуль отримання даних. Забезпечує автоматизоване завантаження історичних біржових котирувань за допомогою бібліотеки yfinance. Користувач може вказувати тикер цінного паперу (наприклад, TSLA), діапазон дат і частоту оновлення (щоденно, щотижнево тощо).

2. Модуль попередньої обробки. Виконує очищення даних від пропущених значень, логарифмування та нормалізацію часового ряду. У разі необхідності здійснюється диференціювання ряду для досягнення стаціонарності. Також формуються ковзні середні та обчислюються стандартні відхилення для аналізу волатильності.

3. Модуль аналізу автокореляцій. Використовується для побудови автокореляційної функції (ACF) та часткової автокореляції (PACF). На їх основі визначаються початкові параметри моделі ARIMA (p, d, q).

4. Модуль моделювання ARIMA. Реалізується за допомогою пакету statsmodels. Забезпечує побудову прогнозної моделі на основі оброблених даних, навчання моделі та формування відповідних залишків.

5. Модуль формування прогнозу. Забезпечує побудову прогнозованих значень цін акцій на заданий період. Передбачає відновлення масштабу ряду до початкових значень через обернення логарифмічної трансформації.

6. Модуль оцінки якості прогнозу. Здійснює обчислення стандартних метрик точності - середньої абсолютної відносної похибки (MAPE) що дозволяють оцінити ефективність побудованої моделі.

7. Модуль візуалізації. Побудова графіків часових рядів, залишків, прогнозів та елементів розкладання часових рядів виконується за допомогою бібліотеки matplotlib. Це дозволяє візуально аналізувати тренди, сезонність, залишкові коливання та відповідність моделі фактичним даним.

На рисунку 2.2 зображено логічну схему взаємодії модулів програмної системи.

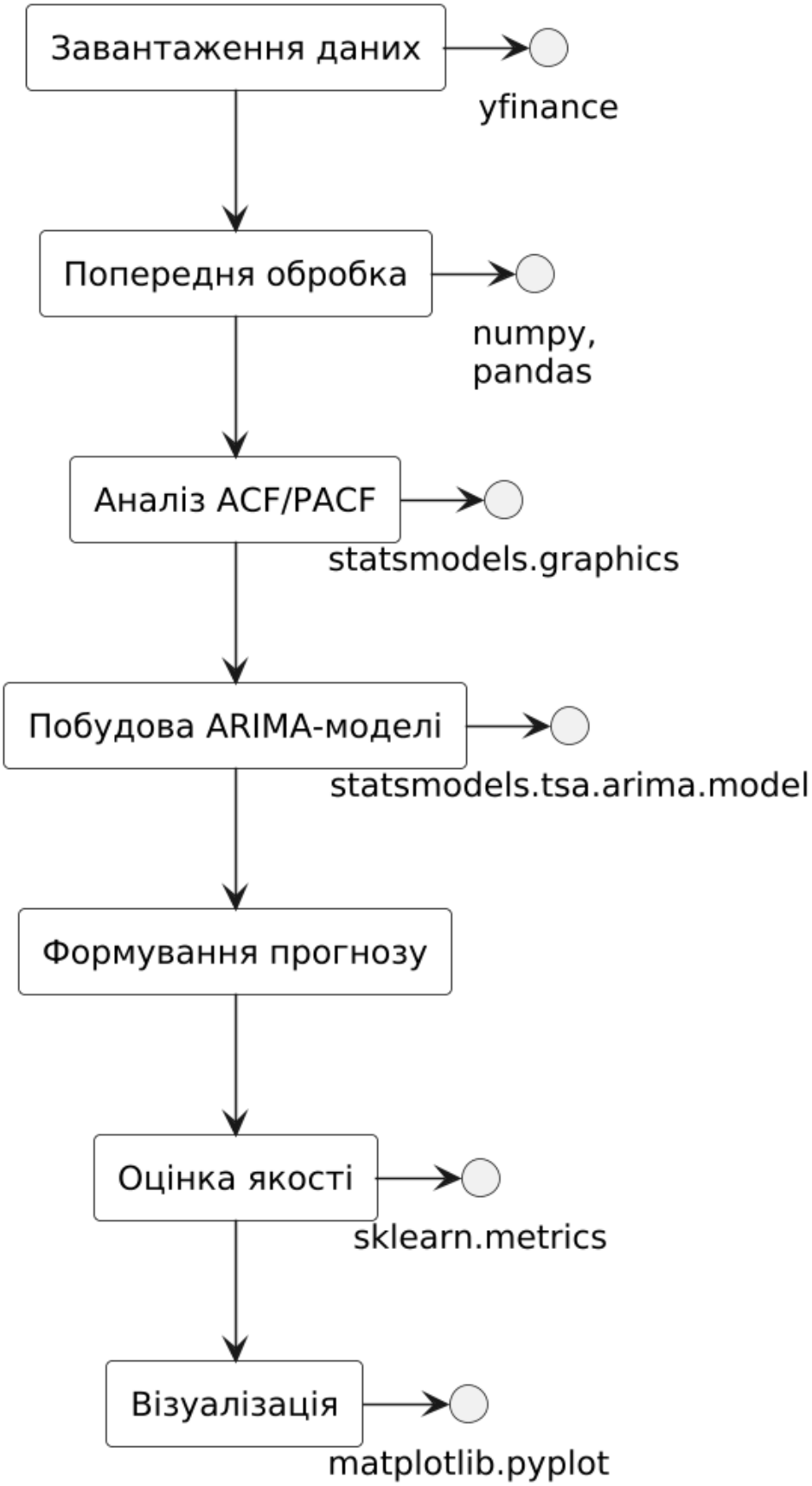

Рисунок 2.2 – Логічна схема взаємодії модулів

Дані про котирування акцій завантажуються за допомогою бібліотеки yfinance, після чого проходять попередню обробку з використанням pandas і numpy (логарифмування, диференціювання тощо). Для аналізу автокореляції застосовується statsmodels.graphics, а побудова моделі ARIMA виконується через statsmodels.tsa.arima.model. Після формування прогнозу оцінка точності здійснюється за допомогою sklearn.metrics, а результати виводяться у вигляді графіків з використанням matplotlib.pyplot.

На рисунку 2.3. наведено основні функціональні можливості системи та взаємодію користувача з модулем.

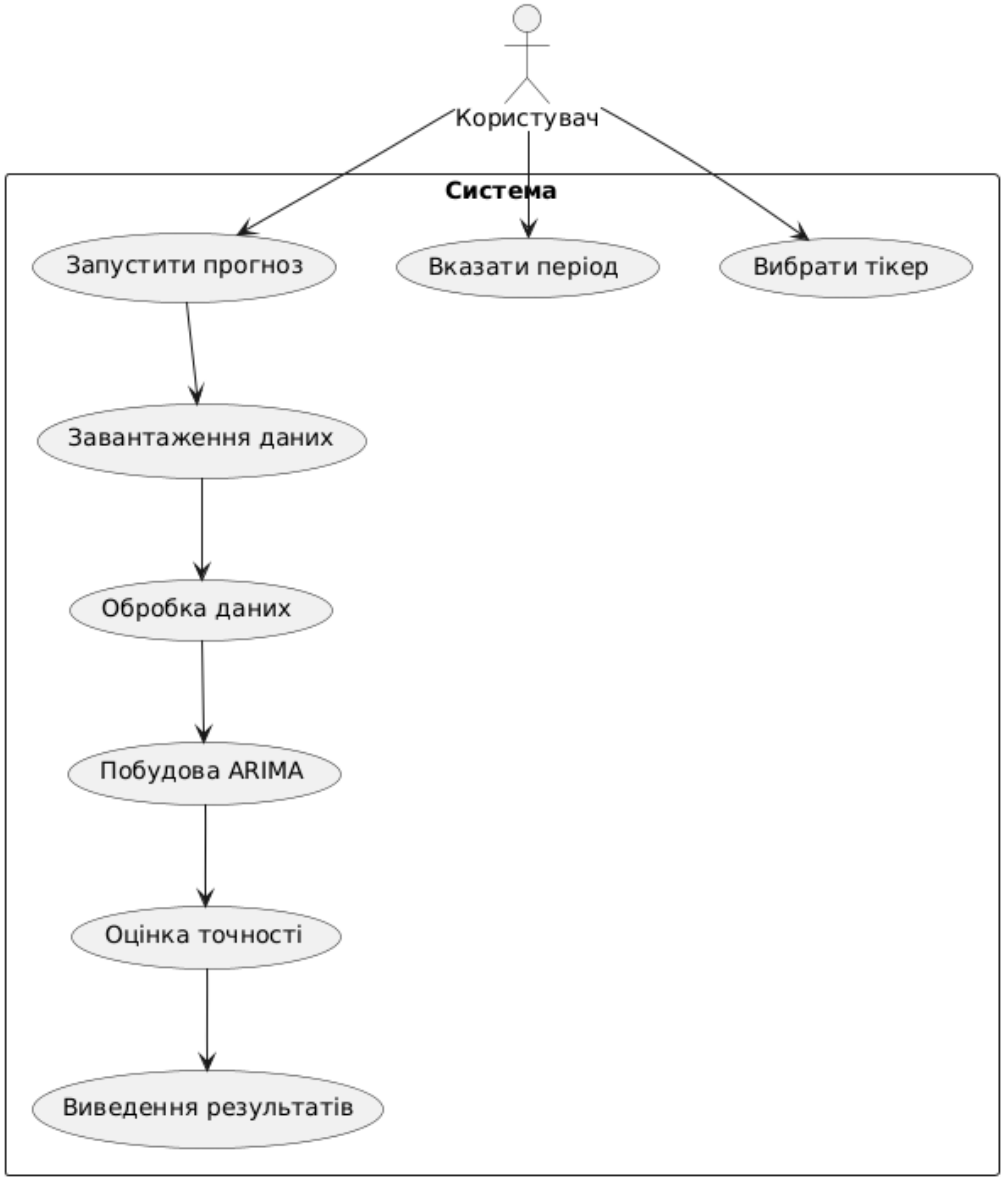

Рисунок 2.3 – Use Case діаграма програмного модуля прогнозування цін акцій

На діаграмі відображені ключові дії: вибір акції, встановлення періоду аналізу, запуск прогнозу, завантаження та обробка даних, побудова моделі, оцінка точності й відображення результатів. Така структура забезпечує прозорість логіки роботи системи та чітке розмежування відповідальностей між користувачем і програмним модулем.

Обрана архітектура програмного модуля має низку суттєвих переваг, що забезпечують її ефективність, масштабованість і відповідність сучасним вимогам до систем прогнозування часових рядів.

По-перше, модульність структури дозволяє ізолювати окремі етапи обробки даних і моделювання. Кожен функціональний блок реалізує визначене завдання, що спрощує тестування компонентів, полегшує супровід програмного забезпечення та створює передумови для подальшого розширення або вдосконалення системи без ризику порушення її загальної цілісності.

По-друге, архітектура забезпечує гнучкість, оскільки допускає заміну або модифікацію окремих алгоритмів та моделей. Наприклад, ARIMA-модель може бути без значних змін у структурі замінена на SARIMA або нейронні мережі (зокрема LSTM), що відкриває можливості для підвищення точності прогнозування відповідно до особливостей вхідних даних.

По-третє, реалізація базується на відкритому програмному забезпеченні, зокрема бібліотеках yfinance, pandas, statsmodels, scikit-learn та matplotlib. Це забезпечує прозорість розробки, вільний доступ до вихідного коду, відсутність витрат на ліцензування, а також високу адаптивність рішення до нових вимог або специфіки задач.

Таким чином, розроблена структура програмного модуля відповідає ключовим вимогам до інструментів прогнозування у фінансовій сфері. Вона забезпечує надійну основу для створення ефективного засобу підтримки прийняття рішень в інвестиційній діяльності.

2.3 Обрання засобів проєктування системи

На етапі проєктування програмного модуля прогнозування цін акцій постала необхідність у виборі таких програмних засобів, які б забезпечували зручну роботу з часовими рядами, статистичне моделювання, гнучку візуалізацію результатів та можливість побудови прикладного інтерфейсу.

Вибір інструментів здійснювався з урахуванням таких критеріїв:

- наявність функціоналу для обробки часових рядів та побудови прогнозних моделей;
- підтримка візуалізації результатів;
- відкритість, безкоштовність, підтримка спільнотою;
- можливість швидкої розробки зручного інтерфейсу користувача;
- сумісність із сервісами біржових даних.

Було розглянуто декілька платформ і мов програмування, які часто використовуються у фінансовій аналітиці. Основні з них наведено у таблиці 2.1.

У результаті аналізу наявних інструментів та з урахуванням поставлених завдань, для реалізації програмного модуля було обрано мову програмування Python у поєднанні з бібліотеками pandas, statsmodels, matplotlib та платформою Streamlit. Такий вибір зумовлений низкою переваг.

По-перше, Python забезпечує повну підтримку інструментарію для роботи зі статистичними моделями, зокрема ARIMA, що є центральною в обраному підході до прогнозування. По-друге, завдяки бібліотеці yfinance реалізується проста та надійна інтеграція з джерелом фінансових даних - Yahoo Finance, що дозволяє автоматизовано завантажувати котирування акцій.

Крім того, використання платформи Streamlit дає змогу швидко створити інтерфейс прикладного рівня, зручний для кінцевого користувача, без необхідності розробки окремого фронтенду. Це значно скоротило час розробки та спростило процес взаємодії з програмним модулем.

Слід також відзначити відкритість усіх використаних засобів, їхню безкоштовність, активну підтримку з боку спільноти розробників і наявність

великої кількості документації та навчальних матеріалів. Це дозволяє не лише ефективно реалізувати проєкт, але й забезпечує можливість його подальшої модифікації та масштабування.

Таблиця 2.1 - Порівняльний аналіз мов програмування

Мова програму-вання	**Переваги**	**Недоліки**
Python	– Величезна кількість бібліотек для аналізу даних та статистики (pandas, statsmodels, scikit-learn, yfinance) – Простота синтаксису та швидкість розробки – Гнучкий вебінтерфейс через Streamlit, Dash – Велика активна спільнота – Підтримка ARIMA, Prophet, LSTM, ETS тощо	– Відносно повільне виконання в обчислювально складних задачах – Менша продуктивність у порівнянні з C++/Java в low-level задачах
R	– Найсильніша підтримка статистичних моделей (forecast, tseries, fable) – Потужні графічні засоби (ggplot2, plotly) – Підходить для глибокого статистичного аналізу	– Менш зручний для розробки прикладних інтерфейсів – Обмежена гнучкість при побудові інтерактивних додатків – Невелика кількість сучасних фреймворків для API/GUI
Java	– Висока продуктивність та надійність – Підтримка багатопоточності – Підходить для розробки складних розподілених систем	– Високий поріг входу – Відсутність вбудованої підтримки статистичних моделей – Ускладнена робота з часовими рядами та фінансовими даними без сторонніх бібліотек
C++	– Максимальна продуктивність – Повний контроль над пам’яттю – Добре підходить для високонавантажених систем або математичних обчислень	– Дуже складна реалізація аналітичних задач – Відсутність спеціалізованих бібліотек для обробки фінансових даних – Не призначена для швидкої розробки та візуалізації

Обране програмне середовище є кросплатформеним, що дозволяє легко розгортати модуль як у локальному середовищі розробника, так і на зовнішніх хостингах або хмарних платформах.

Окремо було розглянуто декілька середовищ розробки, що використовуються для Python-проєктів. Результати порівняльного аналізу подано в таблиці 2.2.

Таблиця 2.2 - Порівняльний аналіз середовищ розробки

Критерій	**Jupyter Notebook**	**PyCharm**	**Visual Studio Code**	**Spyder**
Призначення	Аналіз, прототипування	Повноцінне IDE	Універсальне IDE	Наукові обчислення
Підтримка Python	***	***	***	***
Робота з іншими мовами	*	**	***	*
Візуалізація даних	***	**	**	**
Робота з Git	*	***	***	*
Підсвітка синтаксису	*	***	***	**
Налагодження коду	*	***	***	**
Навантаження на систему	Низьке	Високе	Середнє	Середнє
Рекомендоване використання	Дослідницькі задачі	Командна розробка	Універсальні проєкти	Навчання, аналітика
Навантаження на систему	Низьке	Високе	Середнє	Середнє
Рекомендоване використання	Аналіз даних, прототипування	Комерційна або командна розробка	Універсальне середовище для всіх етапів	Освітні та аналітичні проєкти

Як показано в таблиці 2.2, кожне з розглянутих середовищ має свою специфіку. Jupyter Notebook є зручним для початкового аналізу даних, побудови прототипів і візуалізації, однак його функціональність як повноцінного середовища розробки обмежена. Воно не забезпечує повної підтримки налагодження коду та інтеграції з системами контролю версій, тому більше підходить для дослідницьких задач.

PyCharm - це повнофункціональне середовище, яке підтримує всі етапи розробки, зокрема багатофайлові проєкти, налагодження, інтеграцію з Git та роботу з віртуальними середовищами. Однак через значне навантаження на систему і складність конфігурації воно не завжди доцільне у невеликих проєктах або при обмежених ресурсах.

Visual Studio Code поєднує переваги універсального редактора коду і гнучкого IDE. Воно підтримує розширення для роботи з Python, інтегрується з Streamlit, має вбудовану підтримку Git, працює швидко навіть на середніх системах. Саме тому Visual Studio Code було обрано як основне середовище для розробки в межах цієї роботи.

Spyder орієнтований на наукові та освітні задачі. Він має зручний інтерактивний інтерфейс та можливість швидкого запуску фрагментів коду, однак його функціонал обмежений у порівнянні з Visual Studio Code або PyCharm, особливо для задач повного циклу розробки.

Таким чином, обрані програмні засоби - Python у поєднанні з Visual Studio Code, Streamlit та спеціалізованими бібліотеками - є оптимальними для реалізації програмного модуля прогнозування: вони поєднують зручність, гнучкість, сучасність та ефективність розробки.

2.4 Висновки до розділу

У другому розділі спроєктовано програмний модуль для прогнозування цін акцій на основі часових рядів. Визначено алгоритм його роботи - від введення параметрів користувачем і завантаження даних з Yahoo Finance до побудови моделі

ARIMA, формування прогнозу, оцінки точності та виведення результатів. Модуль має чітку структуру, що складається з незалежних функціональних компонентів: отримання даних, попередньої обробки, аналізу автокореляцій, моделювання, оцінки точності та візуалізації.

Для реалізації обрано Python як мову з широкою підтримкою бібліотек для статистики, фінансової аналітики та побудови інтерфейсів. Інтерфейс створюється засобами Streamlit, що забезпечує інтерактивність і простоту у використанні. Основним середовищем розробки визначено Visual Studio Code.

Передбачено також можливість підключення додаткових модулів - новинного блоку та інвестиційного калькулятора. Усі обрані інструменти є безкоштовними, відкритими і підтримуються спільнотою. Таким чином, результати другого розділу створюють повноцінне підґрунтя для переходу до етапу практичної реалізації та тестування програмного модуля, що й буде розглянуто у наступному розділі.

3 РЕАЛІЗАЦІЯ ПРОГРАМНОГО МОДУЛЯ ПРОГНОЗУВАННЯ ЦІН АКЦІЙ ФОНДОВОГО РИНКУ

3.1 Технічна реалізація прогнозної моделі з використанням ARIMA

Процес реалізації програмного модуля прогнозування цін акцій фондового ринку здійснювався у два етапи: спочатку - як дослідницький прототип у середовищі Jupyter Notebook, а згодом - як повноцінний інтерактивний застосунок на платформі Streamlit. Такий підхід дозволив гнучко протестувати логіку алгоритму, поступово налагодити обробку даних, моделювання та візуалізацію, а потім адаптувати напрацьовану функціональність до зручного вебінтерфейсу для кінцевого користувача.

На початковому етапі розробки програмного модуля використовувалося середовище Jupyter Notebook, яке забезпечує інтерактивність, можливість покрокового виконання коду, а також миттєве відображення результатів обчислень та графіків. Основна мета цього етапу - розробка та перевірка основної логіки побудови прогнозу з використанням моделі ARIMA.

На першому кроці реалізовано завантаження історичних біржових котирувань за допомогою бібліотеки yfinance. Було обрано дані для акцій Tesla (TSLA) за останні кілька років. Дані очищуються від пропущених значень та виводяться у вигляді графіка. Приклад графіка представлений на рисунку 3.1.

На рис. 3.2 представлено фрагмент коду, відповідального за завантаження біржових даних через бібліотеку yfinance:

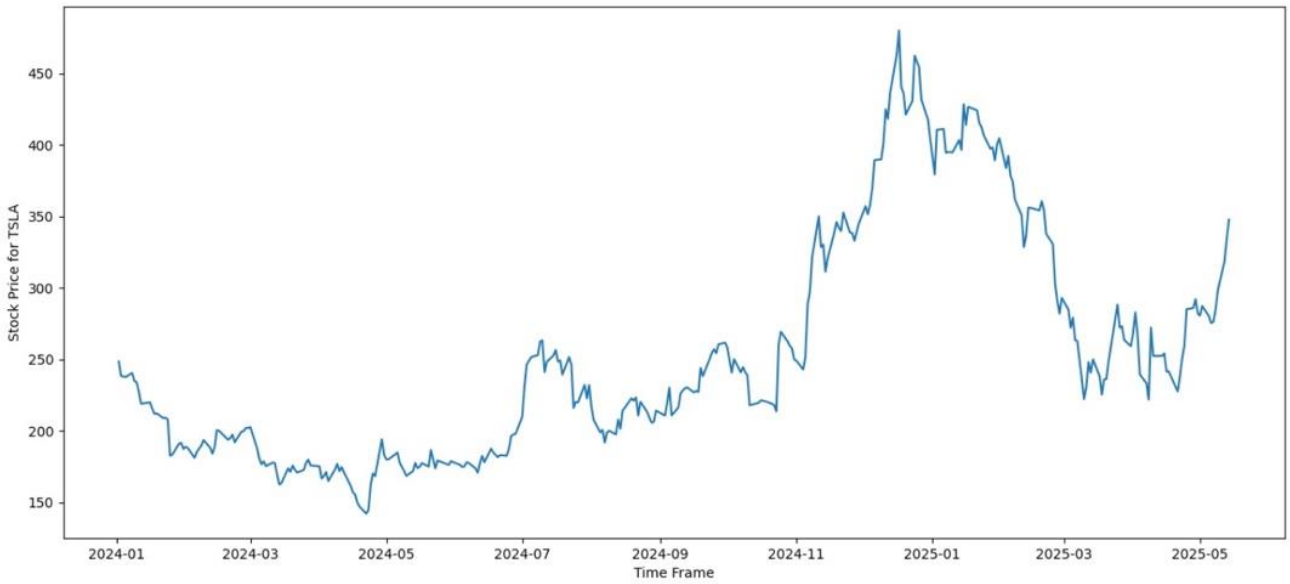

Рисунок 3.1 – Графік історичних цін акцій TSLA (Close Price)

```
# Завантаження котирувань трьох акцій
tickers = ['TSLA', 'AAPL', 'NVDA']
start_date = '2020-01-01'
end_date = '2025-05-15'

data = yf.download(tickers, start=start_date, end=end_date)['Close']
data = data.dropna()

# Вибираємо TSLA як приклад (можна замінити на 'AAPL' або 'NVDA')
TempData = data[['TSLA']].dropna().reset_index()
TempData.columns = ['Date', 'Prev Close']
TempData['Symbol'] = 'TSLA'
```

✓ 0.8s

```
YF.download() has changed argument auto_adjust default to True
[**********100%**********]  3 of 3 completed
```

Рисунок 3.2 – Фрагмент коду для завантаження історичних котирувань акцій

Для попередньої оцінки стаціонарності ряду було побудовано графіки ковзного середнього та стандартного відхилення (рисунок 3.3). Це дозволяє візуально оцінити, чи стабільна структура часового ряду з часом.

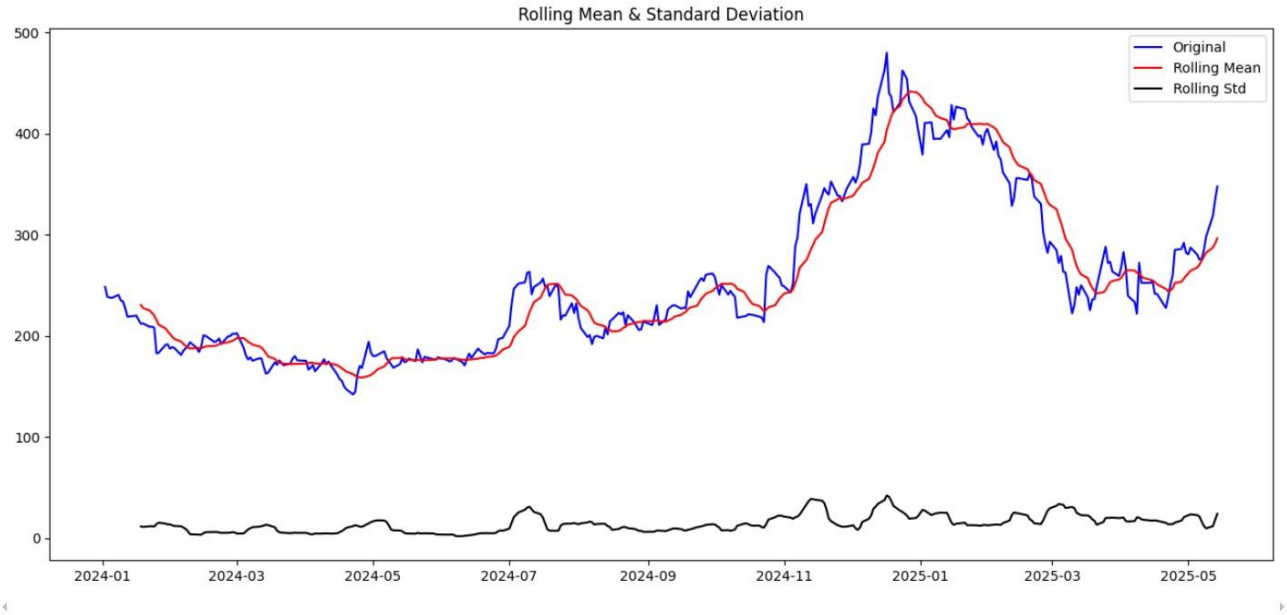

Рисунок 3.3 – Графіки ковзного середнього (Rolling Mean) та стандартного відхилення (Rolling Std)

Фрагмент коду, який виводить графіки ковзного середнього та стандартного відхилення зображено на рисунку 3.4.

```
rolLmean = StockData.rolling(12).mean()
rolLstd = StockData.rolling(12).std()

plt.figure(figsize=(16,7))
fig = plt.figure(1)
orig = plt.plot(StockData, color='blue',label='Original')
mean = plt.plot(rolLmean, color='red', label='Rolling Mean')
std = plt.plot(rolLstd, color='black', label='Rolling Std')
plt.legend(loc='best')
plt.title('Rolling Mean & Standard Deviation')
plt.show(block=False)
```

Рисунок 3.4 – Фрагмент коду для виводу графіків ковзного середнього та стандартного відхилення

Оскільки модель ARIMA потребує стаціонарного ряду, було виконано логарифмування, а далі - перше диференціювання логарифмованого ряду. Ці

трансформації дозволили зменшити тренди та стабілізувати дисперсію, що підтверджується графіком наведеним на рисунку 3.5.

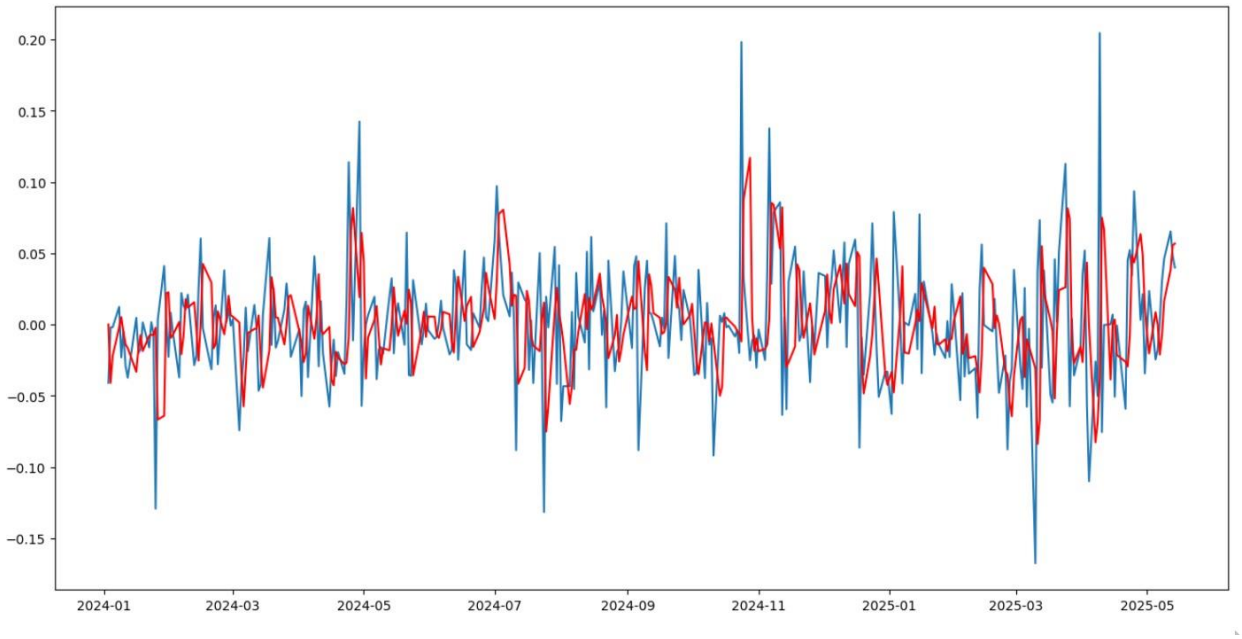

Рисунок 3.4 – Результат після логарифмування та першого диференціювання

Синя лінія на графіку показує значення часового ряду після диференціювання (тобто зміни значень, отримані як різниця між поточними та попередніми значеннями). Вона відображає поточну динаміку даних, які стали стаціонарними після застосування диференціювання. Червона лінія відображає прогнозовані значення від моделі ARIMA. Вона показує, як модель ARIMA підганяє дані, щоб наблизитися до реальних спостережень, відображаючи тренди та сезонні коливання в ряді.

Наступним кроком є аналіз автокореляції та парціальної автокореляції для визначення параметрів моделі ARIMA. Для цього використовується функції acf та pacf з бібліотеки statsmodels. ACF та PACF графіки зображені на рисунку 3.5, фрагмент коду відповідальний за вивід графіків зображено на рисунку 3.6.

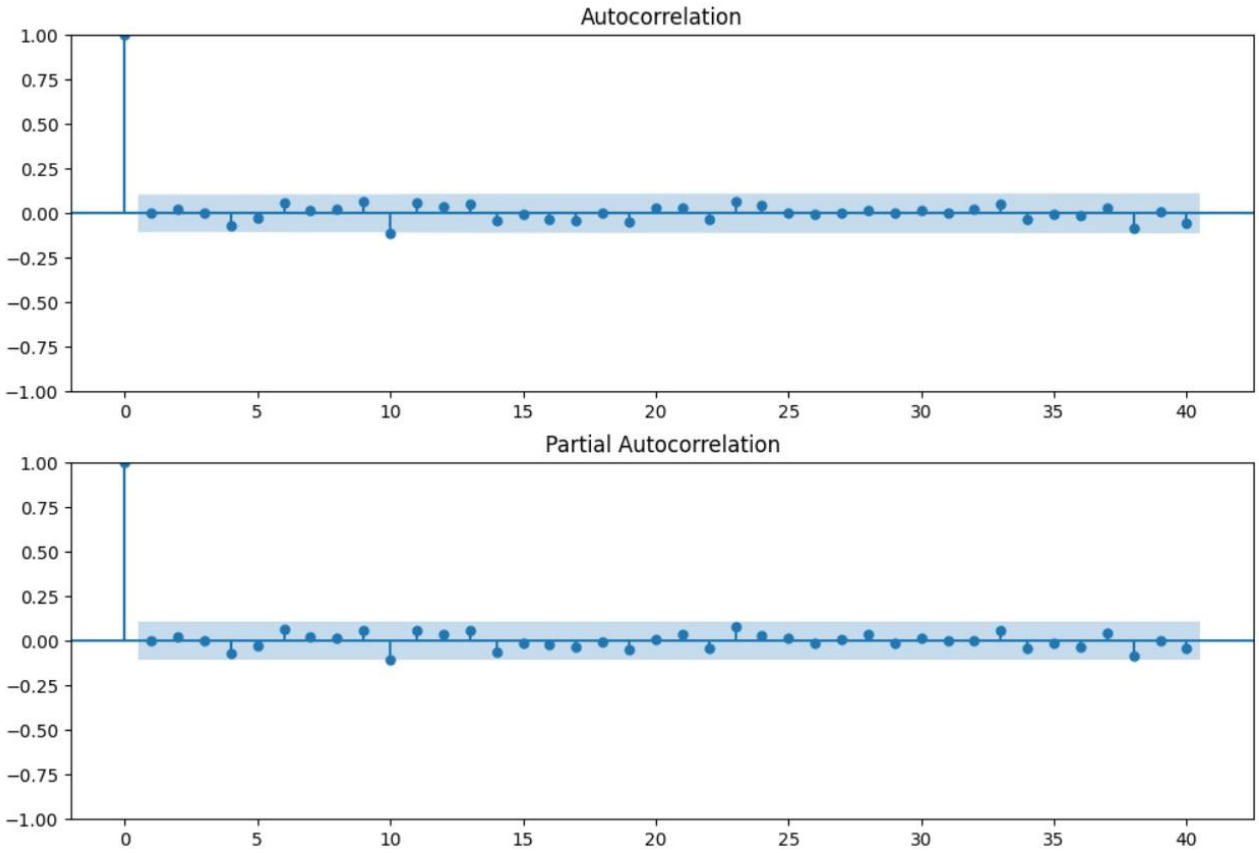

Рисунок 3.5 – Графіки ACF та PACF

```
StockData.sort_index(inplace=True)
lag_acf = acf(ts_log_diff.dropna(), nlags=20)
lag_pacf = pacf(ts_log_diff.dropna(), nlags=20)

fig = plt.figure(figsize=(12,8))
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(ts_log_diff.dropna(), lags=40, ax=ax1)
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(ts_log_diff.dropna(), lags=40, ax=ax2)
```

Рисунок 3.6 – Фрагмент коду для виводу ACF та PACF графіків

Ці графіки дозволяють оцінити кореляцію між спостереженнями на різних відстанях (лагах), що є важливим для визначення оптимальних параметрів ARIMA моделі.

На основі графіків автокореляції (ACF) та часткової автокореляції (PACF) було визначено значення параметрів p та q. Це дало змогу побудувати модель

ARIMA з параметрами (1, 1, 0), що продемонструвала задовільну якість на тестових даних. На рисунку 3.7 зображено результат формування тестового прогнозу з використанням моделі ARIMA після проведення попередньої обробки часового ряду, а на рисунку 3.8 наведено відповідний програмний код.

Рисунок 3.7 – Побудова моделі ARIMA та формування тестового прогнозу

```
ARIMA_diff_predictions = pd.Series(results_ARIMA.fittedvalues, copy=True)
ARIMA_diff_predictions_cumsum = ARIMA_diff_predictions.cumsum()

ARIMA_log_prediction = pd.Series(ts_log.iloc[0], index=ts_log.index)
ARIMA_log_prediction = ARIMA_log_prediction.add(ARIMA_diff_predictions_cumsum, fill_value=0)

plt.figure(figsize=(12,8))
predictions_ARIMA = np.exp(ARIMA_log_prediction)
plt.plot(StockData)
plt.plot(predictions_ARIMA)


mape = mean_absolute_percentage_error(StockData, predictions_ARIMA)
plt.title (f"MAPE: {mape:.4f}")
```

Рисунок 3.8 – Фрагмент коду для формування тестового прогнозу з використанням моделі ARIMA

Основна частина програмного модуля реалізована у вигляді інтерактивного веб-застосунку на базі середовища Streamlit (файл app.py). Інтерфейс побудований

таким чином, щоб забезпечити зручну взаємодію користувача з програмою без потреби прямого доступу до коду. Усі параметри задаються через інтерактивні елементи - випадаючі списки, календарі та кнопки запуску, що робить систему доступною навіть для користувачів без спеціальної технічної підготовки.

Після підтвердження працездатності моделі було реалізовано користувацький застосунок у середовищі Streamlit, який охоплює повний функціонал прогнозування, інтерфейс взаємодії з користувачем та додаткові сервіси.

3.2 Розробка інтерфейсу користувача

Інтерфейс користувача для реалізованої системи прогнозування цін акцій розроблено з використанням фреймворку Streamlit. Цей інструмент дозволяє створювати інтерактивні веб-застосунки мовою Python з мінімальним порогом входу та без потреби у класичній веб-розробці. Завдяки цьому кінцевий користувач має змогу виконувати прогнозування цін акцій у зручному браузерному середовищі.

Основний функціонал розміщено у файлі app.py. На початку програми здійснюється імпорт необхідних бібліотек. Відповідний фрагмент наведено на рисунку 3.9.

```
import warnings
warnings.filterwarnings('ignore')

import streamlit as st
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import yfinance as yf

from sklearn.metrics import mean_absolute_percentage_error
from statsmodels.tsa.arima.model import ARIMA
```

Рисунок 3.9 – Імпорт бібліотек для реалізації інтерфейсу та прогнозування

У бічній панелі інтерфейсу реалізовано блок для введення параметрів - вибору акції, дати початку та дати завершення періоду аналізу. Ці елементи створюються за допомогою функцій st.sidebar.selectbox() та st.sidebar.date_input(). Фрагмент, який відповідає за формування цих елементів, показано на рисунку 3.10.

```
# --- Інтерфейс Streamlit ---
st.title("Прогнозування цін акцій з ARIMA")
st.sidebar.header("Параметри")

ticker = st.sidebar.selectbox("Оберіть акцію", ['TSLA', 'AAPL', 'NVDA'])
start_date = st.sidebar.date_input("Дата початку", pd.to_datetime("2020-01-01"))
end_date = st.sidebar.date_input("Дата кінця", pd.to_datetime("2025-05-30"))
```

Рисунок 3.10 – Побудова панелі параметрів у Streamlit

Далі відбувається завантаження біржових котирувань з використанням бібліотеки yfinance. Обробка та підготовка даних представлена на рисунку 3.11.

```
# --- Обробка даних ---
TempData = data[[ticker]].dropna().reset_index()
TempData.columns = ['Date', 'Prev Close']
TempData['Symbol'] = ticker

StockData = TempData.dropna()
StockData.index = pd.to_datetime(StockData.Date)
StockData = StockData["Prev Close"].loc[start_date:end_date]
st.subheader(f"Історичні дані: {ticker}")
st.line_chart(StockData)
```

Рисунок 3.11 – Завантаження і очищення історичних даних акцій

Історичний ряд відображається за допомогою st.line_chart(), що дозволяє швидко оцінити динаміку цін у зазначений період.

Наступним кроком є логарифмування та диференціювання даних для приведення їх до стаціонарного вигляду, що необхідно для моделі ARIMA. Фрагмент реалізації цього етапу зображено на рисунку 3.12.

```
# --- Побудова ARIMA для історичного прогнозу ---
ts_log = np.log(StockData)
ts_log_diff = ts_log - ts_log.shift()
ts_log_diff.dropna(inplace=True)

model = ARIMA(ts_log_diff, order=(1,1,0))
results_ARIMA = model.fit()

ARIMA_diff_predictions = pd.Series(results_ARIMA.fittedvalues, copy=True)
ARIMA_diff_predictions_cumsum = ARIMA_diff_predictions.cumsum()
ARIMA_log_prediction = pd.Series(ts_log.iloc[0], index=ts_log.index)
ARIMA_log_prediction = ARIMA_log_prediction.add(ARIMA_diff_predictions_cumsum, fill_value=0)
predictions_ARIMA = np.exp(ARIMA_log_prediction)
```

Рисунок 3.12 – Попередня обробка: логарифмування та диференціювання

Після цього виконується побудова ARIMA-моделі та формування прогнозу.

Прогнозовані значення відображаються на графіку разом із фактичними даними, як показано на рисунку 3.13. Це дозволяє порівняти реальні та прогнозовані значення в одному вікні.

```
# --- Вивід графіка історичного прогнозу ---
st.subheader("Прогноз (на основі минулого)")
fig, ax = plt.subplots(figsize=(12,6))
ax.plot(StockData, label='Actual')
ax.plot(predictions_ARIMA, label='Forecast (історичний)', linestyle='--')
ax.legend()
st.pyplot(fig)
```

Рисунок 3.13 – Порівняння фактичних та прогнозованих значень на графіку

Також реалізовано побудову прогнозу на 30 днів наперед. Прогноз виводиться як у вигляді графіка, так і у вигляді табличного представлення з датами та прогнозованими цінами.

Окрім основного прогнозного модуля, інтерфейс має дві додаткові сторінки. На окремій сторінці реалізовано модуль для відображення останніх новин фондового ринку, що дозволяє користувачу швидко отримати уявлення про поточний інформаційний фон. Для цього використовується News API, який повертає список релевантних новин за ключовими словами. На рисунках 3.14 та 3.15 зображено основну частину коду, яка відповідає за отримання та відображення новин.

```
import requests
import streamlit as st

NEWS_API_KEY = 'api key'
NEWS_API_URL = 'https://newsapi.org/v2/everything'

params = {
    'q': 'stock market',  # Пошук за темою фондового ринку
    'apiKey': NEWS_API_KEY,
    'language': 'en',  # Мова новин
    'pageSize': 10,  # Кількість новин
}
```

Рисунок 3.14 – Фрагмент програмного коду завантаження сторінки новин News API

Кожна новина виводиться із заголовком, описом та гіперпосиланням на першоджерело. Користувач може оновити стрічку, натиснувши кнопку «Оновити новини».

```python
# Заголовок
st.title("📰 Останні новини фондового ринку")

# Кнопка для оновлення
if st.button("🔄 Оновити новини"):
    st.cache_data.clear()  # Очищення кешу вручну

# Завантаження новин
news_items = get_news()

# Виведення новин
if news_items:
    for item in news_items:
        title = item['title']
        url = item['url']
        description = item.get('description', 'Опис недоступний')

        st.markdown(f"### [{title}]({url})")
        st.write(description)
else:
    st.warning("Не вдалося завантажити новини. Спробуйте пізніше.")
```

Рисунок 3.15 – Фрагмент коду реалізації сторінки новин

Інша сторінка надає можливість розрахувати прибуток від інвестицій у вибрану акцію. Користувач вводить суму інвестиції, обирає тикер та діапазон дат. Програма завантажує відповідні дані, розраховує кількість куплених акцій, прибуток і прибутковість у відсотках. На рисунках 3.16 та 3.17 показано приклад реалізації логіки калькулятора.

```python
# Ввід
ticker = st.text_input("Введіть тікер акції (наприклад: AAPL, TSLA)", value="AAPL")
start_date = st.date_input("Дата початку", date(2020, 1, 1))
end_date = st.date_input("Дата завершення", date.today() - timedelta(days=1))
amount = st.number_input("Сума інвестиції ($)", min_value=10.0, value=1000.0, step=100.0)
```

Рисунок 3.16 – Фрагменти програмної реалізації інвестиційного калькулятора

Результати також виводяться у вигляді графіка зміни ціни акцій та детального фінансового звіту: початкова і кінцева ціна, кількість куплених акцій, кінцева вартість та прибуток у доларах і відсотках.

```python
if st.button("Розрахувати"):
    try:
        df = get_data(ticker, start_date, end_date)

        # Визначити доступну колонку
        if 'Adj Close' in df.columns:
            price_col = 'Adj Close'
        elif 'Close' in df.columns:
            price_col = 'Close'
        else:
            raise ValueError("Дані не містять колонок 'Adj Close' або 'Close'.")

        start_price = df[price_col].iloc[0]
        end_price = df[price_col].iloc[-1]

        shares = amount / start_price
        final_value = shares * end_price
        profit = final_value - amount
        profit_pct = (profit / amount) * 100

        st.success(f"Результати для {ticker.upper()}:")
        st.write(f"📈 Ціна на {start_date}: **${start_price:.2f}**")
        st.write(f"📉 Ціна на {end_date}: **${end_price:.2f}**")
        st.write(f"🔢 Куплено акцій: **{shares:.4f}**")
        st.write(f"💰 Кінцева вартість: **${final_value:.2f}**")
        st.write(f"📊 Прибуток: **${profit:.2f} ({profit_pct:.2f}%)**")

        st.line_chart(df[price_col])

    except Exception as e:
        st.error(f"Помилка: {e}")
```

Рисунок 3.17 – Фрагменти коду інвестиційного калькулятора з розрахунком прибутку

Таким чином, інтерфейс користувача системи є багатофункціональним, зручним та адаптованим до потреб як новачків, так і досвідчених користувачів. Інтеграція прогнозного модуля з додатковими інструментами - новинами та інвестиційним калькулятором - робить систему не лише аналітичною, а й практично корисною у процесі прийняття фінансових рішень.

3.3 Тестування роботи програмного модуля

Після завершення розробки програмного модуля було проведено тестування його функціональності, зручності використання та коректності математичних розрахунків. Тестування виконувалося у середовищі Visual Studio Code шляхом запуску інтерактивного веб-інтерфейсу, реалізованого за допомогою бібліотеки Streamlit.

Мета тестування полягала у перевірці відповідності системи поставленим вимогам, зокрема: здатності завантажувати та обробляти реальні фінансові дані, здійснювати статистичне прогнозування за допомогою ARIMA-моделі, виводити результати у зручному вигляді та забезпечувати додаткову функціональність для користувача.

Після запуску програми користувач має змогу задати параметри прогнозування: обрати біржовий тикер акції (TSLA, AAPL, NVDA), встановити часовий діапазон для аналізу, після чого система автоматично завантажує відповідні дані з Yahoo Finance. На етапі завантаження було перевірено обробку помилок (наприклад, при введенні некоректного діапазону дат або відсутності даних). На рисунку 3.15 фрагмент інтерфейсу з вибором параметрів та графіком історії котирувань.

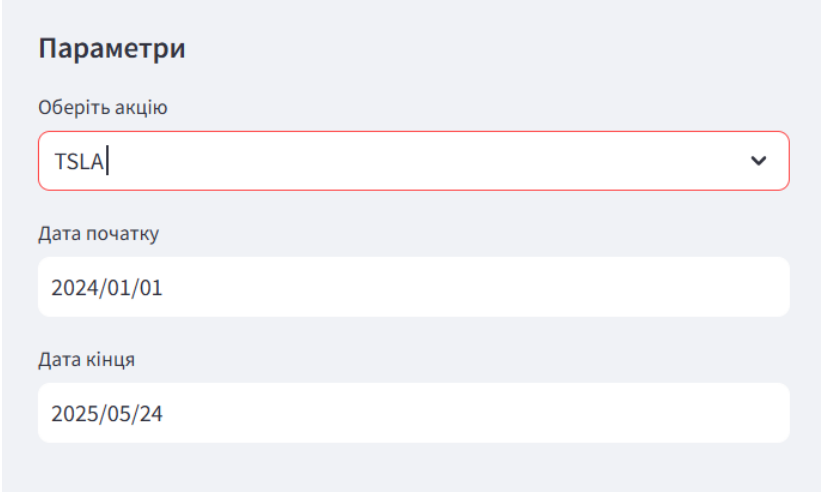

Рисунок 3.15 – Скріншот інтерфейсу з вибором параметрів та графіком історії котирувань

Після завантаження та візуалізації даних система здійснює логарифмування, диференціювання та формує модель ARIMA з фіксованими параметрами. Перевірено правильність роботи кожного етапу обробки шляхом порівняння проміжних результатів з очікуваними. Модель навчалась на логарифмічно перетворених значеннях, після чого здійснювалося відновлення реального масштабу цін. Графік тестового прогнозу зображено на рисунку 3.16.

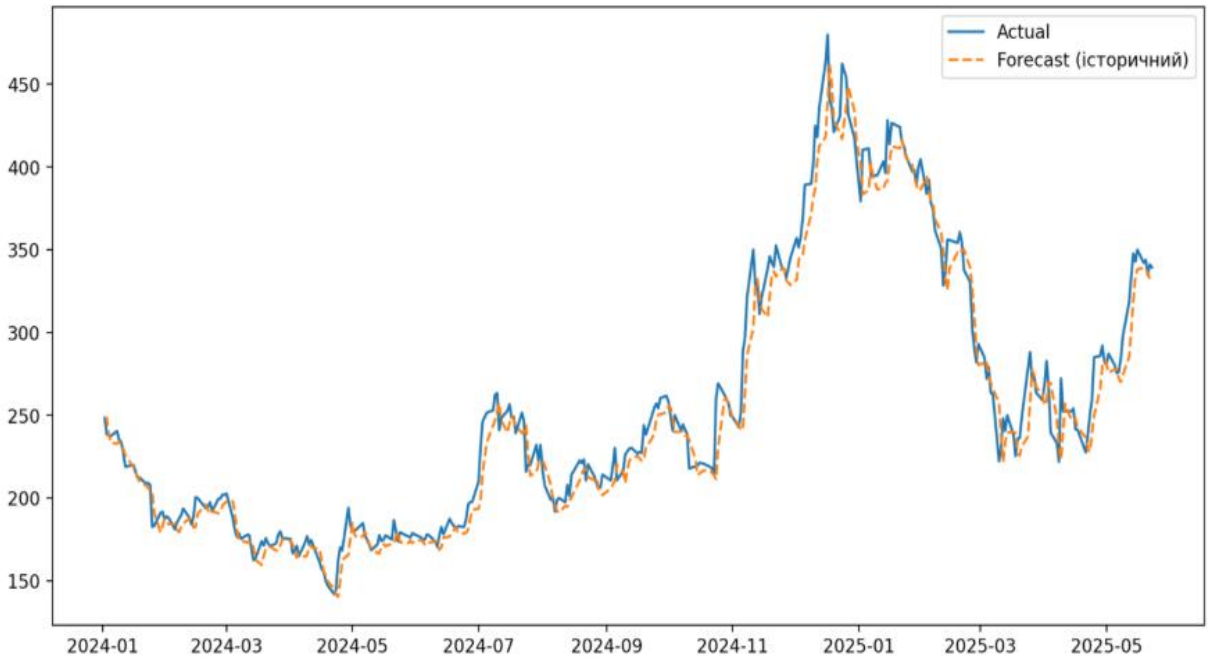

Рисунок 3.16 – Скріншот графіка з прогнозом на історичних даних

Прогноз на 30 днів будується з кумулятивним відновленням логарифмічного ряду. Результати виводяться графічно на одній діаграмі разом із історичними значеннями, як зображено на рисунку 3.17.

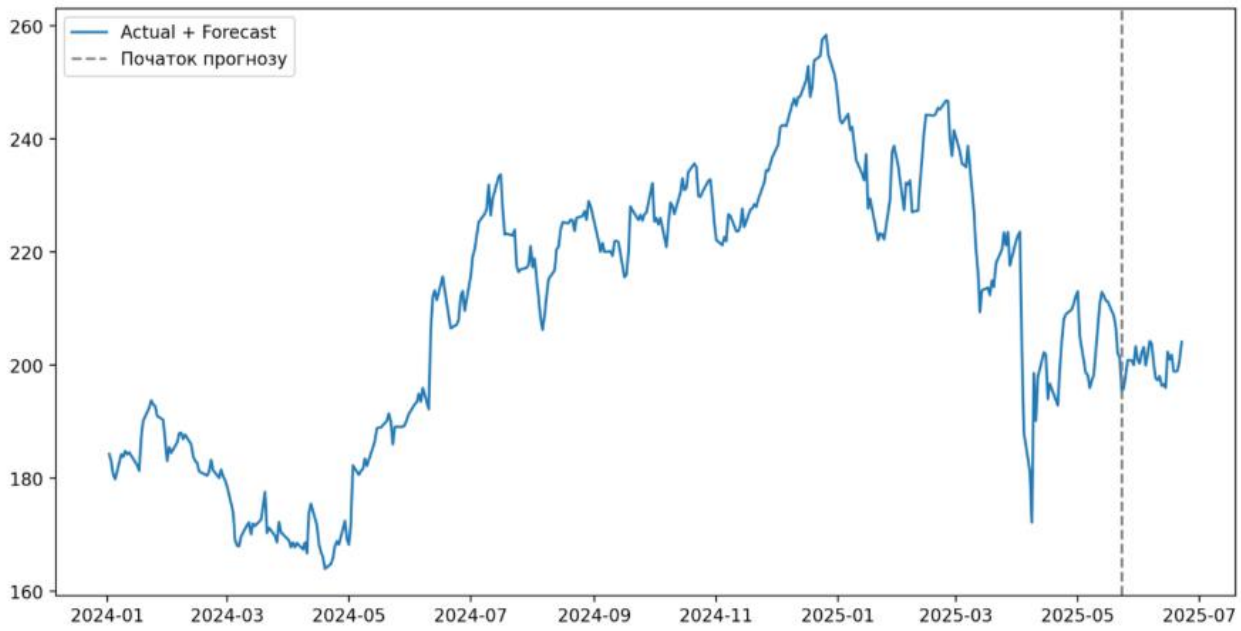

Рисунок 3.17 – Скріншот графіка з прогнозом на 30 днів уперед

Також реалізовано вивід прогнозованих значень в табличку, яку можна буде завантажити для зручності (рисунок 3.18)

Date	Forecasted Price
2025-05-24 00:00:00	195.9884
2025-05-25 00:00:00	198.3141
2025-05-26 00:00:00	200.9031
2025-05-27 00:00:00	200.8837
2025-05-28 00:00:00	200.8581
2025-05-29 00:00:00	200.0203
2025-05-30 00:00:00	203.3353
2025-05-31 00:00:00	200.9941
2025-06-01 00:00:00	200.2952
2025-06-02 00:00:00	202.2343

Рисунок 3.18 – Скріншот таблиці з прогнозом на 30 днів уперед

Також, було перевірено підключення до сервісу NewsAPI, завантаження актуальних новин за запитом "stock market" та правильність обробки відповіді. Новини відображаються із заголовком, коротким описом і гіперпосиланням. Окремо протестовано кнопку оновлення новин, що очищає кеш і завантажує нову інформацію. На рисунку 3.19 зображено скріншот модуля новин з прикладом статей.

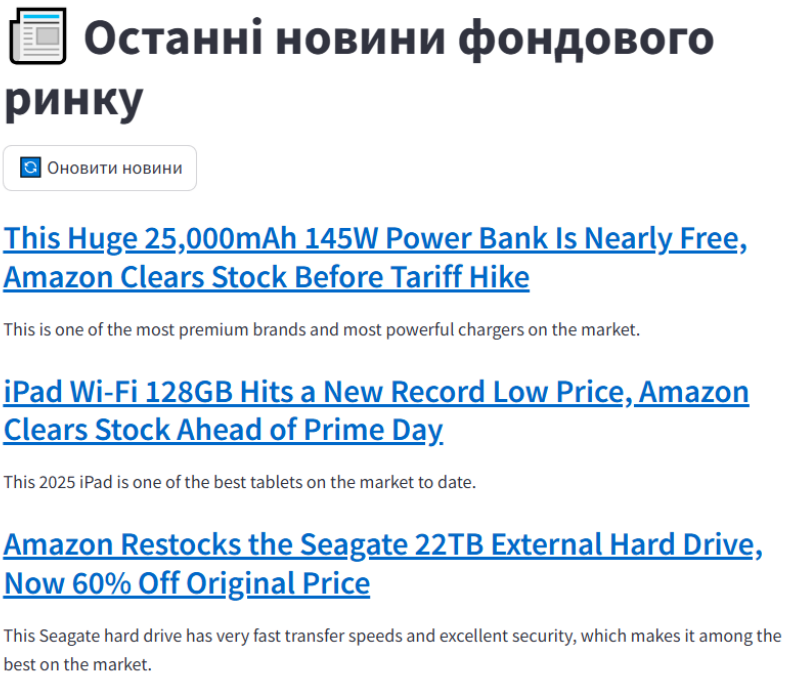

Рисунок 3.19 – Скріншот модуля новин з прикладом статей

Інвестиційний калькулятор дозволяє користувачу ввести біржовий тикер, дату початку та завершення інвестування, а також суму. Програма завантажує історичні ціни, обчислює куплену кількість акцій, вартість портфеля на кінець періоду та розраховує прибуток у грошовому та відсотковому вираженні. Проведено тестування з різними сценаріями, включаючи невалідні дати, відсутність даних та великі суми інвестицій. На рисунку 3.20 проілюстровано приклад роботи інвестиційного калькулятора.

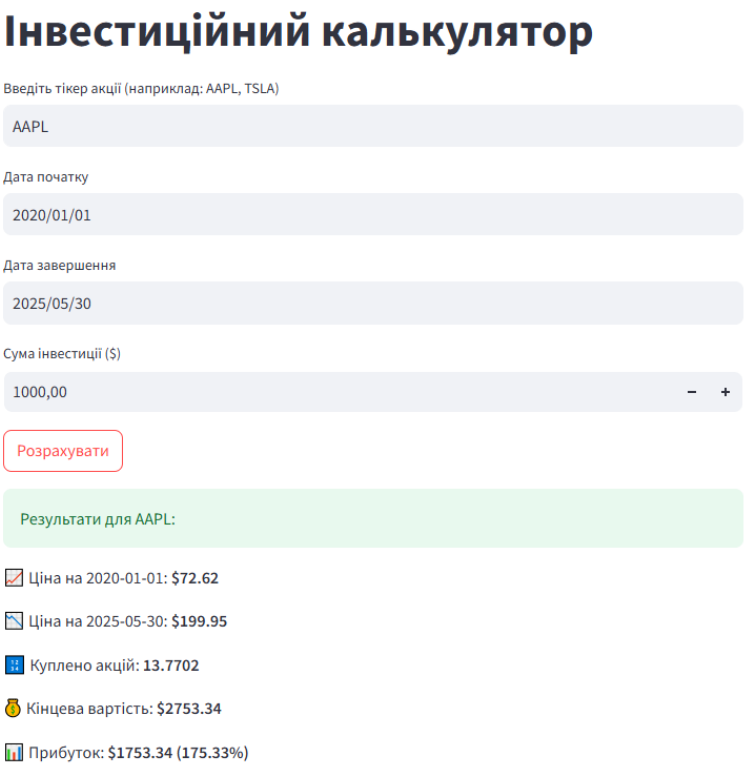

Рисунок 3.20 – Скріншот калькулятора з прикладом розрахунку

Усі функціональні блоки системи продемонстрували стабільну роботу без критичних збоїв. Всі заплановані дії виконуються коректно, інтерфейс є інтуїтивно зрозумілим, а результат прогнозування відповідає очікуваній поведінці моделі. Система адекватно реагує на зміну вхідних параметрів, забезпечуючи гнучкість та зручність при роботі.

Розроблений програмний модуль вирізняється тим, що поєднує функції прогнозування, інтерпретованості та зручної візуалізації, а також включає додаткові сервіси, які відсутні в аналогах, зокрема:

1. Сторінка з останніми фінансовими новинами - дозволяє аналізувати інформаційний фон перед прийняттям інвестиційних рішень;

2. Інтерактивний інвестиційний калькулятор - дає змогу моделювати потенційний прибуток на основі історичних даних;

3. Повністю автономна ARIMA-модель із можливістю налаштування параметрів та виводу розширеного прогнозу на 30 днів;

4. Простий веб-інтерфейс Streamlit, що не потребує попередньої технічної підготовки.

Порівняльна таблиця 3.1 демонструє функціональну відмінність між розробленим модулем та найпоширенішими програмами-аналогами.

Таблиця 3.1 – Порівняння функціональності розробленого модуля та аналогів

Функціональна можливість	**Prophet**	**QuantConnect**	**Vertex AI**	**Розроблений модуль**
Автоматичне завантаження біржових даних	+	+	+	+
Побудова прогнозу (статистична модель)	+	+ (ML)	+ (AutoML)	+ (ARIMA)
Гнучке налаштування параметрів моделі	+	+	-	+
Проста локальна інсталяція	+	-	-	+
Виведення графіків і таблиць прогнозу	+	частково	+	+
Прогноз на 30 днів	частково	+	+	+
Інвестиційний калькулятор	-	-	-	+
Вивід фінансових новин	-	-	-	+
Повна автономність моделі (локально)	+	-	-	+

Порівняння з програмами-аналогами показало, що розроблений модуль має низку важливих переваг. Його можна використовувати локально без необхідності підключення до хмарних сервісів, він містить інтеграцію з джерелами новин і дає змогу гнучко налаштовувати параметри прогнозування. Додаткові інструменти, зокрема інвестиційний калькулятор, роблять роботу з додатком більш зручною та інформативною.

Загалом, розроблена система не лише виконує основне завдання - прогнозує ціни акцій, - а й пропонує корисні додаткові функції, що допомагають

користувачеві краще орієнтуватися в ринковій ситуації та приймати зважені інвестиційні рішення.

3.4 Висновки до розділу

У третьому розділі здійснено реалізацію програмного модуля прогнозування цін акцій фондового ринку на основі моделі ARIMA, що передбачало перехід від теоретично розробленої структури до створення повнофункціонального інструменту, придатного для практичного використання.

Було представлено реалізацію системи в середовищі Python з використанням платформи Streamlit для побудови інтерактивного вебінтерфейсу. Створено зручний графічний застосунок, що дозволяє обрати акцію для аналізу, задати часовий діапазон, побудувати прогноз на 30 днів уперед, а також оцінити точність моделі. Модуль реалізує повний цикл роботи з часовими рядами: від завантаження даних з Yahoo Finance до формування прогнозу у зрозумілому форматі.

Крім основного функціоналу прогнозування, система доповнена двома додатковими модулями - новинною стрічкою, яка автоматично завантажує актуальні новини фондового ринку за допомогою API, та інвестиційним калькулятором, що дозволяє оцінити прибутковість вкладень у конкретні акції за обраний період. У порівнянні з найближчим аналогом - сервісом QuantConnect, функціональні можливості розробленого програмного модуля було розширено приблизно на 20% за рахунок реалізації двох додаткових функцій.

Також проведено тестування роботи програмного модуля. В результаті перевірки було підтверджено стабільну роботу всіх функціональних блоків, правильність математичних розрахунків, візуалізацію прогнозних значень, а також адекватну реакцію системи на зміну параметрів.

У результаті реалізовано гнучку, масштабовану та зручну систему, яка не лише прогнозує динаміку цін, але й забезпечує інтерактивну взаємодію з користувачем та додаткову аналітичну підтримку.

ВИСНОВКИ

Поставлені перед бакалаврською кваліфікаційною роботою задачі виконано в повному обсязі, а саме:

1. У процесі вивчення предметної області розглянуто структуру фондового ринку, його основні функції, учасників та особливості коливань цін, а також проаналізовано основні математичні підходи до прогнозування динамічних фінансових даних та обґрунтовано вибір моделі ARIMA.

2. Проведений аналіз існуючих програм-аналогів продемонстрував, що на ринку представлені ефективні, але вузькоспеціалізовані інструменти яким бракує достатнього функціоналу для формування обґрунтованих інвестиційних рішень

3. На етапі проєктування було сформовано логіку роботи програмного модуля та розроблено архітектуру системи з чітким розмежуванням функціональних компонентів: обробки даних, моделювання, візуалізації, формування прогнозу та взаємодії з користувачем.

4. У процесі технічної реалізації створено дослідницький прототип у середовищі Jupyter Notebook для тестування обчислювальної частини модуля та розроблено повноцінний інтерактивний застосунок у середовищі Streamlit, що надає можливість завантажувати дані, задавати параметри, переглядати прогноз та використовувати додаткові функції: модуль новин й інвестиційний калькулятор.

5. Проведене тестування підтвердило коректність роботи модуля та його відповідність поставленим завданням. У порівнянні з конкурентними системами, розроблений модуль вирізняється розширеним функціоналом за рахунок впровадження двох додаткових функцій: інвестиційного калькулятора та модуля відображення новинного фону. Таким чином, у порівнянні з найближчим конкурентом Prophet, було досягнуто розширення функціональних можливостей програми приблизно на 20%.

6. Розроблено інструкцію користувача, яка описує послідовність роботи із програмним модулем та забезпечує зручність використання розробленої системи.

Таким чином, мету бакалаврської кваліфікаційної роботи досягнуто.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Шевчук О.Ф., Краснолуцька Я.В. Особливості розробки програмного модуля прогнозування цін акцій фондового ринку. // Матеріали конференції «LII Науково-технічна конференція факультету інтелектуальних інформаційних технологій та автоматизації (2025)», Вінниця, 2025. URL: https://conferences.vntu.edu.ua/index.php/mn/mn2025/paper/view/25394 (дата звернення: 06.05.2025)

2. Шевчук О.Ф., Краснолуцька Я.В. Свідоцтво про реєстрацію авторського права на твір № 136468, дата реєстрації 22 травня 2025 р.. Комп'ютерна програма «Програмний модуль прогнозування цін акцій фондового ринку» URL: https://sis.nipo.gov.ua/uk/search/simple/ (дата звернення: 09.05.2025)

3. Мішкін, Ф. С. Гроші, банківська справа і фінансові ринки. – К.: Основи, 2020.

4. Закон України «Про цінні папери та фондовий ринок» №3480-IV від 23.02.2006.

5. Bodie, Zvi, Alex Kane, and Alan J. Marcus. Investments. – McGraw-Hill Education, 2021.

6. Chan, E. Algorithmic Trading: Winning Strategies and Their Rationale. New Jersey: Wiley, 2013.

7. Schwab, Klaus. The Fourth Industrial Revolution. – World Economic Forum, 2016.

8. Кривко, Н. Ю. Фондовий ринок: структура, функції, механізм регулювання. – Економіка та держава, 2021.

9. Савчук Т.О., Ваховський В.М. Удосконалений метод аналізу фондового ринку акцій із використанням нечіткої логіки/ Вчені записки Таврійського національного університету ім. В.І. Вернадського, серія: Технічні науки, том 31(70), №4, 2020, Київ, ISSN 2663-5941 – с.121-126

10. Shiller, Robert. Irrational Exuberance. – Princeton University Press, 2015.

11. Time Series Analysis: Definition, Types, Techniques, and When It's Used. URL: https://www.tableau.com/learn/articles/time-series-analysis#definition (дата звернення: 23.04.2025)

12. Yahoo Finance API. URL: https://finance.yahoo.com (дата звернення: 03.05.2025)

13. Яровий А.А., Шевчук О.Ф., Козловський А.В., Паночишин Ю.М., Сімончук С.В. Використання ARIMA моделей для прогнозування загального рівня злочинності в Україні. Український журнал інформаційних технологій. 2024. Т. 6 (2). С. 49-56. https://doi.org/10.23939/ujit2024.02.049 (дата звернення: 05.05.2025)

14. Hyndman, Rob J., and George Athanasopoulos. Forecasting: Principles and Practice. – OTexts, 2021.

15. Makridakis, Spyros, Evangelos Spiliotis, and Vassilios Assimakopoulos. "Statistical and Machine Learning Forecasting Methods: Concerns and Ways Forward." PLoS ONE 13.11 (2018): e0194889.

16. А. Т. Яровий, Є. М. Страхов АНАЛІЗ ЧАСОВИХ РЯДІВ / Навчальнометодичний посібник для студентів математичних та економічних спеціальностей, Освіта України, 2019, Одеса.

17. Box, George E. P., Gwilym M. Jenkins, Gregory C. Reinsel, and Greta M. Ljung. Time Series Analysis: Forecasting and Control. – Wiley, 2015.

18. The Python Tutorial. URL: https://docs.python.org/3/tutorial/index.html (дата звернення: 14.05.2025)

19. Ozer, B. Using Python for Algorithmic Trading. URL: https://towardsdatascience.com/using-python-for-algorithmic-trading-23db6059fc51 (дата звернення: 25.05.2025)

20. Методичні вказівки до виконання бакалаврської кваліфікаційної роботи для студентів денної та заочної форм навчання спеціальності 122 - «Комп’ютерні науки» / Укладачі А.А.Яровий, О.В.Сілагін, І.В.Богач. – Вінниця: ВНТУ, 2022. – 47 с.

Додаток А (обов'язковий)

Протокол перевірки кваліфікаційної роботи на наявність текстових запозичень

Назва роботи: Програмний модуль для прогнозування цін акцій фондового ринку

Тип роботи: бакалаврська кваліфікаційна робота

(бакалаврська кваліфікаційна робота / магістерська кваліфікаційна робота)

Підрозділ кафедра комп'ютерних наук

(кафедра, факультет, навчальна група)

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism 0,79 %

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне)

- ✓ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту
- ☐ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.
- ☐ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

Яровий А.А., зав. каф. КН _____
(прізвище, ініціали, посада) (підпис)

Колесницький О.К., проф. каф КН _____
(прізвище, ініціали, посада) (підпис)

Особа, відповідальна за перевірку _____ Озеранський В.С.
(підпис) (прізвище, ініціали)

З висновком експертної комісії ознайомлений(-на)

Керівник _____ Шевчук О.Ф.
(підпис) (прізвище, ініціали, посада)

Здобувач _____ Краснолуцька Я.В.
(підпис) (прізвище, ініціали)

Додаток Б (обов'язковий)
ЛІСТИНГ ПРОГРАМИ

Прототип (stock_price_forecasting.ipynb)

```
import warnings
warnings.filterwarnings('ignore')

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import yfinance as yf

from sklearn.metrics import mean_absolute_percentage_error
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.tsa.stattools import acf, pacf
from statsmodels.tsa.seasonal import seasonal_decompose
import statsmodels.api as sm

# Завантаження котирувань трьох акцій
tickers = ['TSLA', 'AAPL', 'NVDA']
start_date = '2000-01-01'
end_date = '2025-03-26'

data = yf.download(tickers, start=start_date, end=end_date)['Close']
data = data.dropna()

# Вибираємо TSLA як приклад (можна замінити на 'AAPL' або 'NVDA')
TempData = data[['TSLA']].dropna().reset_index()
TempData.columns = ['Date', 'Prev Close']
TempData['Symbol'] = 'TSLA'

StockData = TempData.dropna()
```

```
StockData.index = pd.to_datetime(StockData.Date)
StockData = StockData["Prev Close"]['2020-01-01':'2025-12-2']
StockData.describe()

# Візуалізація початкових даних
plt.figure(figsize=(16,7))
fig = plt.figure(1)
ax1 = fig.add_subplot(111)
ax1.set_xlabel('Time Frame')
ax1.set_ylabel('Stock Price for TSLA')
ax1.plot(StockData)

# Розрахунок ковзного середнього та стандартного відхилення
rolLmean = StockData.rolling(12).mean()
rolLstd = StockData.rolling(12).std()

plt.figure(figsize=(16,7))
fig = plt.figure(1)
orig = plt.plot(StockData, color='blue', label='Original')
mean = plt.plot(rolLmean, color='red', label='Rolling Mean')
std = plt.plot(rolLstd, color='black', label='Rolling Std')
plt.legend(loc='best')
plt.title('Rolling Mean & Standard Deviation')
plt.show(block=False)

# Логарифмування даних
ts_log = np.log(StockData)

plt.figure(figsize=(16,7))
fig = plt.figure(1)
plt.plot(ts_log)

# Декомпозиція за допомогою сезонного розкладу
```

```
decomposition = seasonal_decompose(ts_log, period=1, model='multiplicative')

trend = decomposition.trend
seasonal = decomposition.seasonal
residual = decomposition.resid

plt.figure(figsize=(16,7))
fig = plt.figure(1)
plt.subplot(411)
plt.plot(ts_log, label='Original')
plt.legend(loc='best')
plt.subplot(412)
plt.plot(trend, label='Trend')
plt.legend(loc='best')
plt.subplot(413)
plt.plot(seasonal, label='Seasonality')
plt.legend(loc='best')
plt.subplot(414)
plt.plot(residual, label='Residuals')
plt.legend(loc='best')

# Диференціювання для отримання стаціонарного ряду
ts_log_diff = ts_log - ts_log.shift()
plt.figure(figsize=(16,7))
fig = plt.figure(1)
plt.plot(ts_log_diff)

rolLmean = ts_log_diff.rolling(12).mean()
rolLstd = ts_log_diff.rolling(12).std()

orig = plt.plot(ts_log_diff, color='blue', label='Original')
mean = plt.plot(rolLmean, color='red', label='Rolling Mean')
std = plt.plot(rolLstd, color='black', label='Rolling Std')
```

```
plt.legend(loc='best')
plt.title('Rolling Mean & Standard Deviation')
plt.show(block=False)

# Оцінка ACF/PACF для вибору параметрів ARIMA
StockData.sort_index(inplace=True)
lag_acf = acf(ts_log_diff.dropna(), nlags=20)
lag_pacf = pacf(ts_log_diff.dropna(), nlags=20)

fig = plt.figure(figsize=(12,8))
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(ts_log_diff.dropna(), lags=40, ax=ax1)
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(ts_log_diff.dropna(), lags=40, ax=ax2)

# Побудова моделі ARIMA
ts_log_diff = ts_log_diff[~ts_log_diff.isnull()]

plt.figure(figsize=(16,8))
model = ARIMA(ts_log_diff, order=(1,1,0))
results_ARIMA = model.fit()
plt.plot(ts_log_diff)
plt.plot(results_ARIMA.fittedvalues, color='red')

# Прогнозування на основі ARIMA
ARIMA_diff_predictions = pd.Series(results_ARIMA.fittedvalues, copy=True)
ARIMA_diff_predictions_cumsum = ARIMA_diff_predictions.cumsum()

ARIMA_log_prediction = pd.Series(ts_log.iloc[0], index=ts_log.index)
ARIMA_log_prediction = ARIMA_log_prediction.add(ARIMA_diff_predictions_cumsum,
fill_value=0)

plt.figure(figsize=(12,8))
```

```
predictions_ARIMA = np.exp(ARIMA_log_prediction)
plt.plot(StockData)
plt.plot(predictions_ARIMA)
plt.title('RMSE: %.4f' % np.sqrt(sum((predictions_ARIMA - StockData)**2) / len(StockData)))

# Оцінка точності прогнозу за допомогою MAPE
mape = mean_absolute_percentage_error(StockData, predictions_ARIMA)
plt.title(f"MAPE: {mape:.4f}")

# Прогнозування для заданих інтервалів
results_ARIMA.predict(10, 20)
```

Прогноз (app.py)

```
import warnings
warnings.filterwarnings('ignore')

import streamlit as st
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import yfinance as yf

from sklearn.metrics import mean_absolute_percentage_error
from statsmodels.tsa.arima.model import ARIMA

# --- Інтерфейс Streamlit ---
st.title("Прогнозування цін акцій з ARIMA")
st.sidebar.header("Параметри")

ticker = st.sidebar.selectbox("Оберіть акцію", ['TSLA', 'AAPL', 'NVDA'])
start_date = st.sidebar.date_input("Дата початку", pd.to_datetime("2020-01-01"))
end_date = st.sidebar.date_input("Дата кінця", pd.to_datetime("2025-05-30"))
```

```
# --- Кешована функція завантаження даних ---
@st.cache_data(ttl=86400)  # кеш на 24 години
def load_data(ticker):
    data = yf.download([ticker], start='2000-01-01', end='2025-05-30')['Close']
    return data.dropna()

data = load_data(ticker)

# --- Обробка даних ---
TempData = data[[ticker]].dropna().reset_index()
TempData.columns = ['Date', 'Prev Close']
TempData['Symbol'] = ticker

StockData = TempData.dropna()
StockData.index = pd.to_datetime(StockData.Date)
StockData = StockData["Prev Close"].loc[start_date:end_date]
st.subheader(f"Історичні дані: {ticker}")
st.line_chart(StockData)

# --- Побудова ARIMA для історичного прогнозу ---
ts_log = np.log(StockData)
ts_log_diff = ts_log - ts_log.shift()
ts_log_diff.dropna(inplace=True)

model = ARIMA(ts_log_diff, order=(1,1,0))
results_ARIMA = model.fit()

ARIMA_diff_predictions = pd.Series(results_ARIMA.fittedvalues, copy=True)
ARIMA_diff_predictions_cumsum = ARIMA_diff_predictions.cumsum()
ARIMA_log_prediction = pd.Series(ts_log.iloc[0], index=ts_log.index)
ARIMA_log_prediction = ARIMA_log_prediction.add(ARIMA_diff_predictions_cumsum,
fill_value=0)
```

```
predictions_ARIMA = np.exp(ARIMA_log_prediction)

# --- Вивід графіка історичного прогнозу ---
st.subheader("Прогноз (на основі минулого)")
fig, ax = plt.subplots(figsize=(12,6))
ax.plot(StockData, label='Actual')
ax.plot(predictions_ARIMA, label='Forecast (історичний)', linestyle='--')
ax.legend()
st.pyplot(fig)

# --- MAPE ---
mape = mean_absolute_percentage_error(StockData, predictions_ARIMA)
st.markdown(f"### MAPE: `{mape:.4f}`")

n_days = 30

recent_prices = StockData[-30:]
daily_returns = recent_prices.pct_change().dropna()
mean_return = daily_returns.mean()
std_return = daily_returns.std()

last_price = StockData.iloc[-1]
forecast_values = []

for _ in range(n_days):
    simulated_return = np.random.normal(loc=mean_return, scale=std_return / 2)
    last_price = last_price * (1 + simulated_return)
    forecast_values.append(last_price)

forecast_dates = pd.date_range(start=StockData.index[-1] + pd.Timedelta(days=1), periods=n_days)
forecast_series = pd.Series(forecast_values, index=forecast_dates)
full_series = pd.concat([StockData, forecast_series])
```

```
# --- Вивід графіка з прогнозом на 30 днів ---
st.subheader("Прогноз з розширенням на 30 днів")
fig2, ax2 = plt.subplots(figsize=(12,6))
ax2.plot(full_series, label='Actual + Forecast')
ax2.axvline(x=StockData.index[-1], color='gray', linestyle='--', label='Початок прогнозу')
ax2.legend()
st.pyplot(fig2)

# --- Таблиця з прогнозом ---
forecast_table = pd.DataFrame({
    'Date': forecast_dates,
    'Forecasted Price': forecast_series.values
})
forecast_table.set_index('Date', inplace=True)
st.subheader("Таблиця прогнозу на наступні 30 днів")
st.dataframe(forecast_table)
```

Сторінка новин (1_news.py)

```
import requests
import streamlit as st

NEWS_API_KEY = '4470fa517a3749df9d28117f184235b4'
NEWS_API_URL = 'https://newsapi.org/v2/everything'

params = {
    'q': 'stock market',  # Пошук за темою фондового ринку
    'apiKey': NEWS_API_KEY,
    'language': 'en',  # Мова новин
    'pageSize': 10,  # Кількість новин
}

@st.cache_data
```

```
def get_news():
    response = requests.get(NEWS_API_URL, params=params)
    if response.status_code == 200:
        return response.json()['articles']
    else:
        st.error(f"Помилка запиту: {response.status_code} - {response.text}")
        return []

# Заголовок
st.title("📰 Останні новини фондового ринку")

# Кнопка для оновлення
if st.button("🔄 Оновити новини"):
    st.cache_data.clear()  # Очищення кешу вручну

# Завантаження новин
news_items = get_news()

# Виведення новин
if news_items:
    for item in news_items:
        title = item['title']
        url = item['url']
        description = item.get('description', 'Опис недоступний')

        st.markdown(f"### [{title}]({url})")
        st.write(description)
else:
    st.warning("Не вдалося завантажити новини. Спробуйте пізніше.")
```

Інвестиційний калькулятор (2_investment_calculator.py)

```
import streamlit as st
```

```
import yfinance as yf
import pandas as pd
from datetime import date, timedelta

st.title("Інвестиційний калькулятор")

# Ввід
ticker = st.text_input("Введіть тікер акції (наприклад: AAPL, TSLA)", value="AAPL")
start_date = st.date_input("Дата початку", date(2020, 1, 1))
end_date = st.date_input("Дата завершення", date.today() - timedelta(days=1))
amount = st.number_input("Сума інвестиції ($)", min_value=10.0, value=1000.0, step=100.0)

@st.cache_data
def get_data(ticker, start, end):
    df = yf.download(ticker, start=start, end=end)
    if df.empty:
        raise ValueError("Не знайдено даних для вказаного тикера та дат.")
    # Розплющити мультиіндекс, якщо є
    if isinstance(df.columns, pd.MultiIndex):
        df.columns = df.columns.get_level_values(0)
    return df

if st.button("Розрахувати"):
    try:
        df = get_data(ticker, start_date, end_date)

        # Визначити доступну колонку
        if 'Adj Close' in df.columns:
            price_col = 'Adj Close'
        elif 'Close' in df.columns:
            price_col = 'Close'
        else:
            raise ValueError("Дані не містять колонок 'Adj Close' або 'Close'.")
```

```
        start_price = df[price_col].iloc[0]
        end_price = df[price_col].iloc[-1]

        shares = amount / start_price
        final_value = shares * end_price
        profit = final_value - amount
        profit_pct = (profit / amount) * 100

        st.success(f"Результати для {ticker.upper()}:")
        st.write(f"📈 Ціна на {start_date}: **${start_price:.2f}**")
        st.write(f"📉 Ціна на {end_date}: **${end_price:.2f}**")
        st.write(f"🔢 Куплено акцій: **{shares:.4f}**")
        st.write(f"💰 Кінцева вартість: **${final_value:.2f}**")
        st.write(f"📊 Прибуток: **${profit:.2f} ({profit_pct:.2f}%)**")

        st.line_chart(df[price_col])

    except Exception as e:
        st.error(f"Помилка: {e}")
```

Додаток В (довідниковий)
ІНСТРУКЦІЯ КОРИСТУВАЧА

Розроблений програмний модуль має простий та інтуїтивно зрозумілий веб-інтерфейс, реалізований за допомогою фреймворку Streamlit. Для запуску і роботи з модулем користувачеві достатньо виконати такі кроки:

1. **Запуск застосунку**

У каталозі з файлом app.py потрібно виконати:

streamlit run app.py

```
PS C:\Users\user\Stock-Price-Forecasting> streamlit run app.py

  You can now view your Streamlit app in your browser.
```

Рисунок В.1 – виконання команди streamlit run app.py

Після цього в браузері автоматично відкриється вікно інтерфейсу.

2. **Опис основного функціоналу**

- **Прогнозування акцій**

На головній сторінці користувач обирає акцію (тикер) та період аналізу після чого прогноз автоматично оновлюється. На рисунку В.2 зображено вибір параметрів.

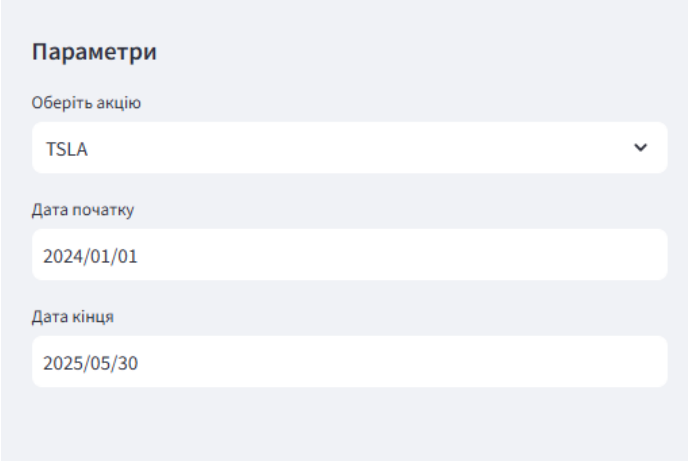

Рисунок В.2 – Вибір параметрів для прогнозу

Після обробки виводяться: графік історичних цін, прогноз на 30 днів (рис В.3), таблиця прогнозу та метрики точності (MAPE).

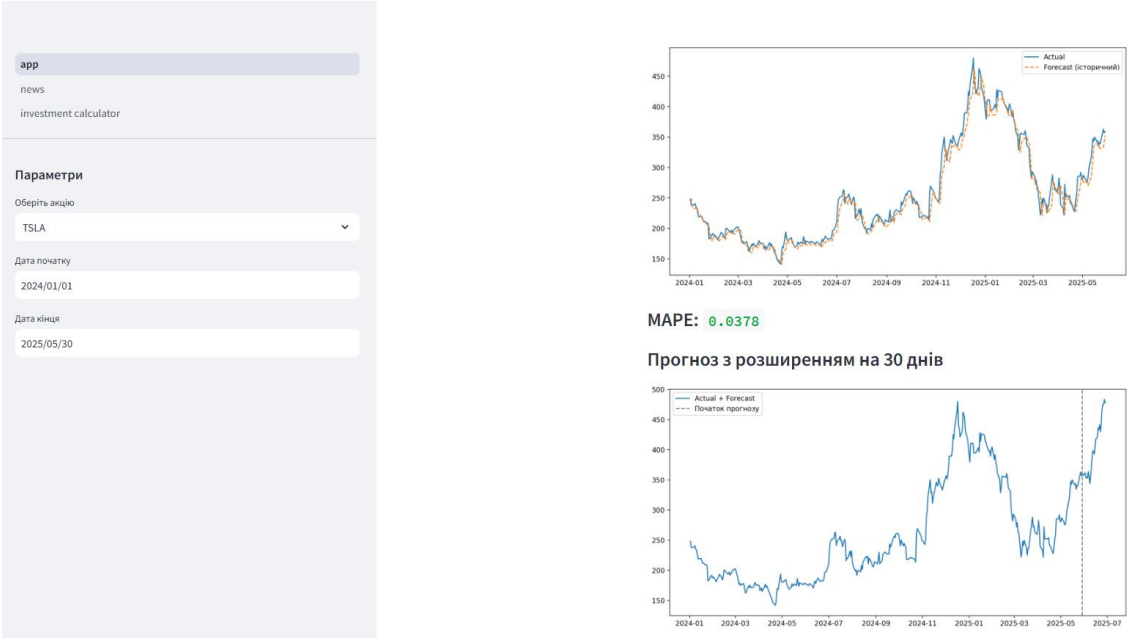

Рисунок В.3 – Сторінка прогнозу

- **Новини ринку**

На окремій сторінці (рис В.4) виводиться список актуальних фінансових новин за обраним ключовим словом. Для кожної новини подається заголовок, короткий опис і гіперпосилання на джерело. З допомогою кнопки «Оновити новини», можна завантажувати свіжі новини.

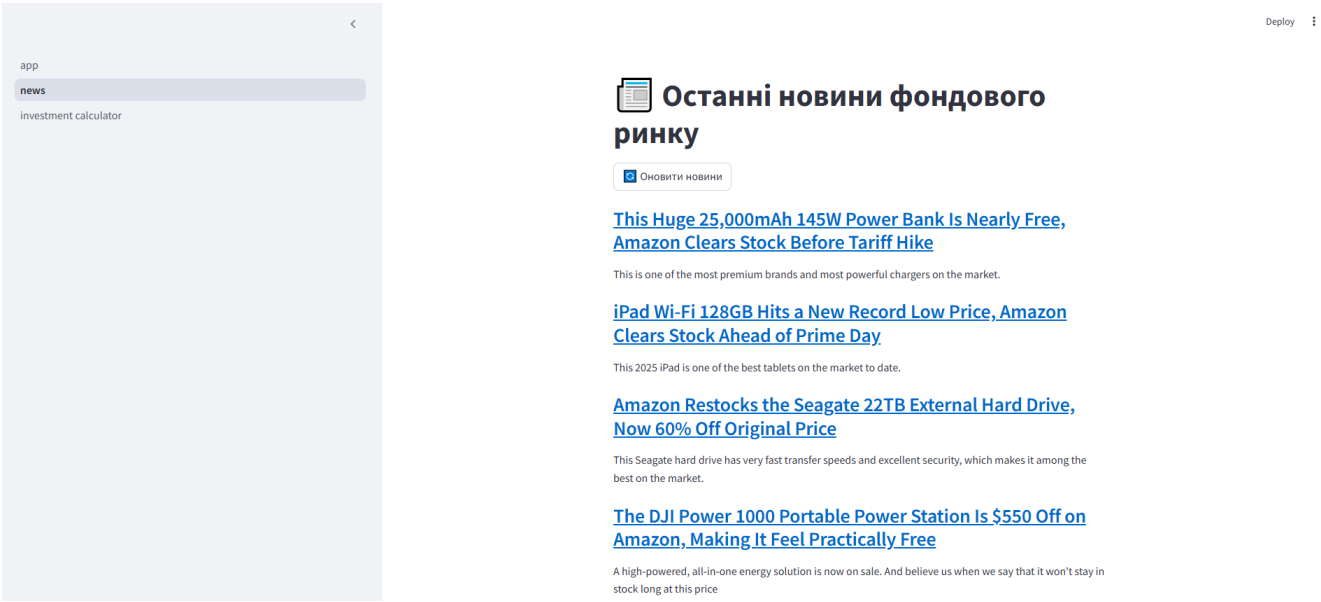

Рисунок В.4 – Сторінка новин

- **Інвестиційний калькулятор**

Користувач вводить суму інвестиції, обирає акцію та період. Програма розраховує прибуток, дохідність у %, кількість куплених акцій, а також виводить графік динаміки ціни.

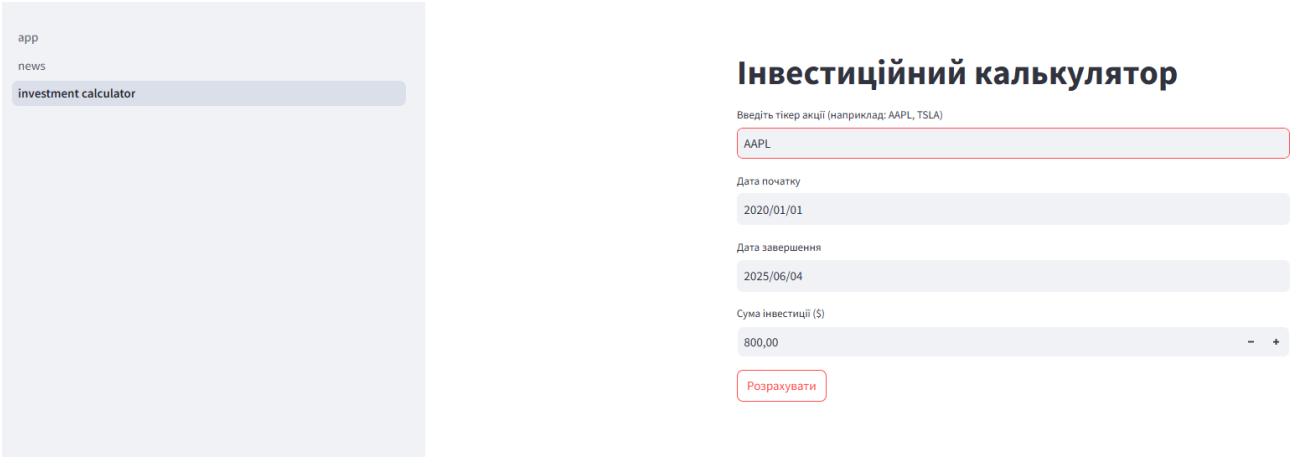

Рисунок В.4 – Сторінка інвестиційного калькулятора

- **Завершення роботи**

Щоб завершити роботу з додатком, достатньо зупинити виконання в терміналі (натиснувши Ctrl+C) або закрити браузерну вкладку з інтерфейсом.

Додаток Г (обов'язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

«ПРОГРАМНИЙ МОДУЛЬ ПРОГНОЗУВАННЯ ЦІН АКЦІЙ ФОНДОВОГО РИНКУ»

Виконала: студентка 4 курсу, групи 2КН-21б
спеціальності 122 – Комп'ютерні науки
(шифр і назва напрямку підготовки, спеціальності)

_____ Краснолуцька Я. В.
(прізвище та ініціали)

Керівник: к.ф-м.н., доц. каф. КН

_____ Шевчук О.Ф.
(прізвище та ініціали)

«03.» червня 2025 р.

Вінниця ВНТУ – 2025р.

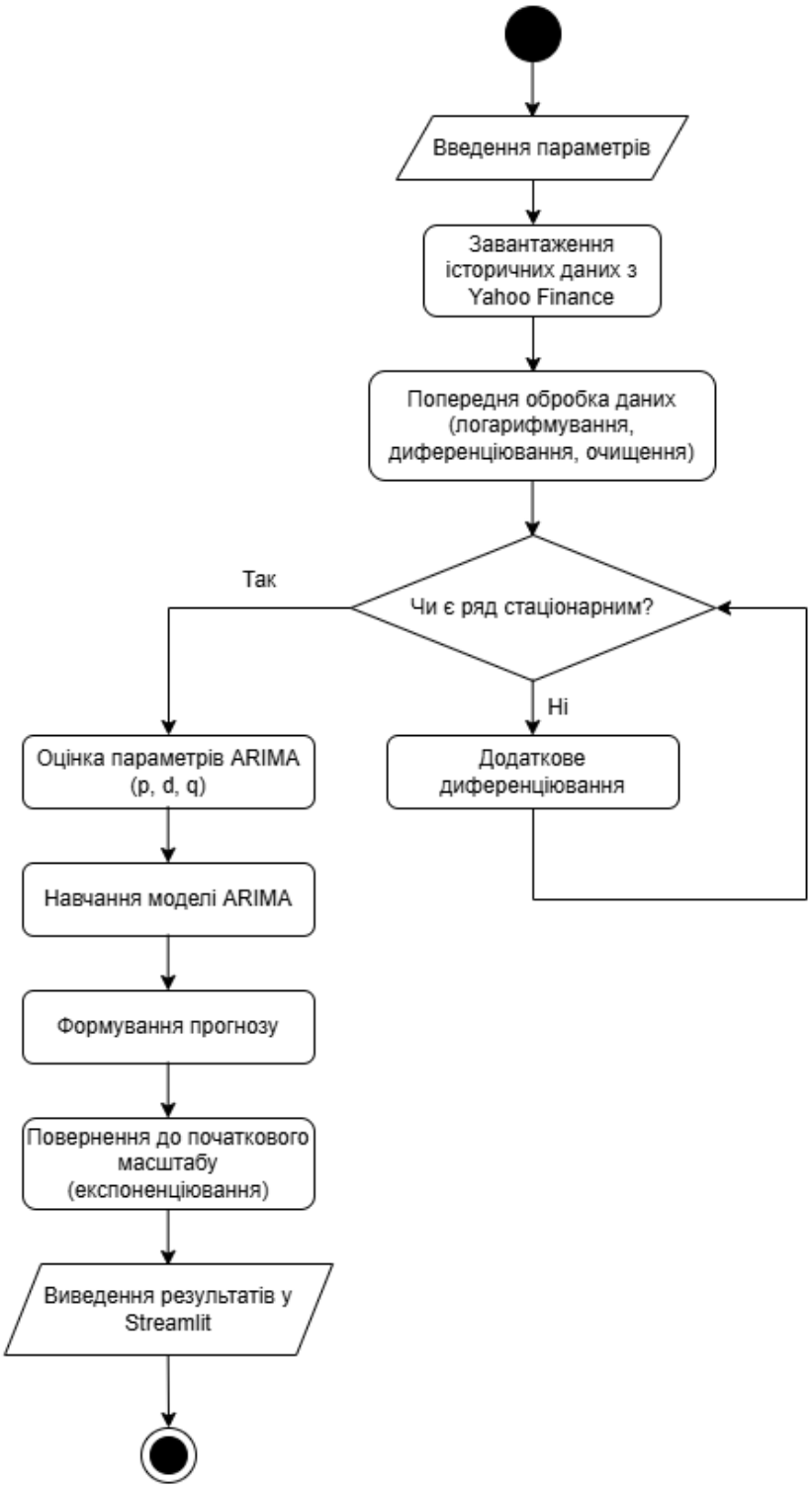

Рисунок Г.1 – Блок-схема алгоритму роботи програмного модуля

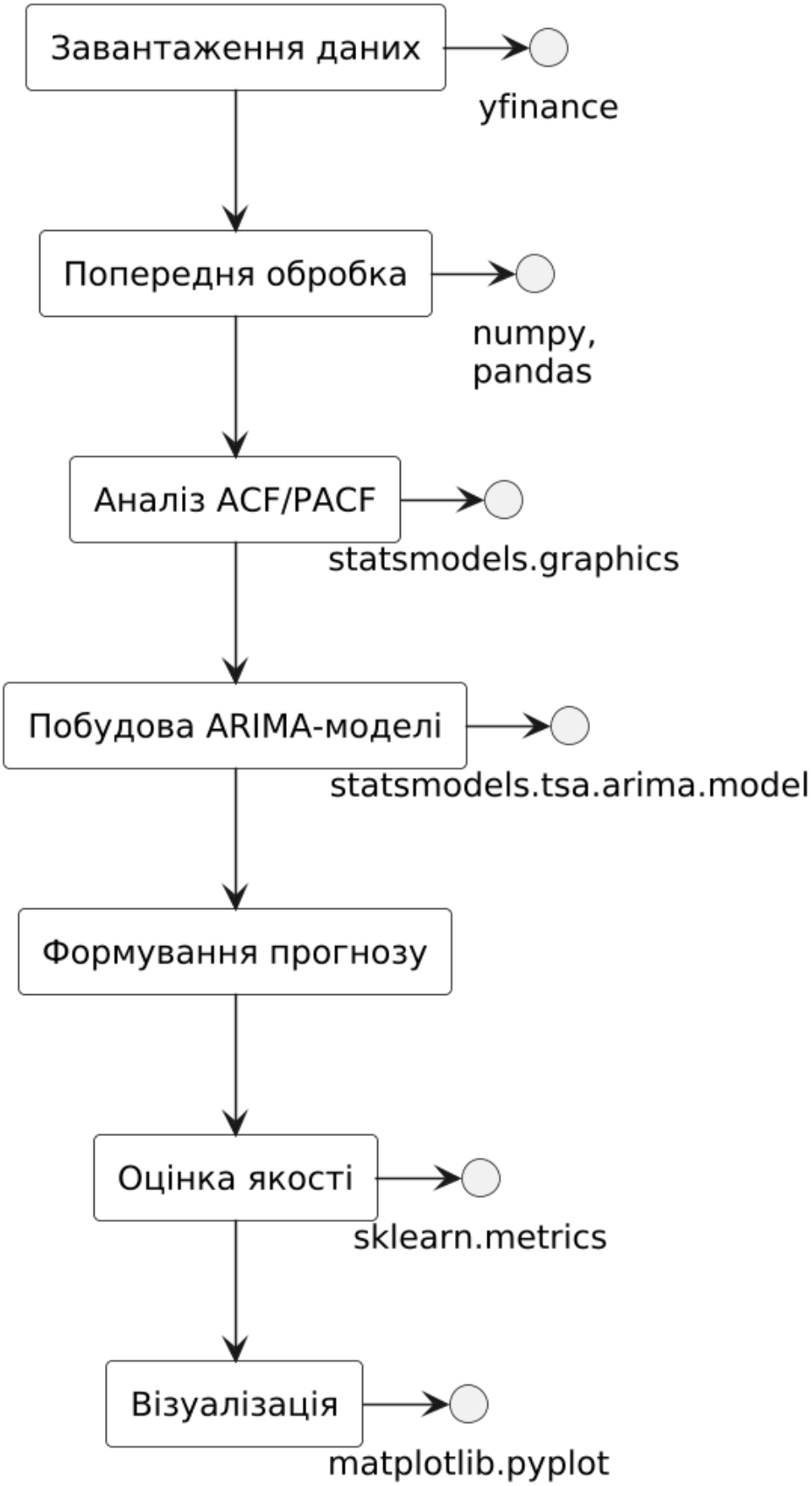

Рисунок Г.2 – Логічна схема взаємодії модулів

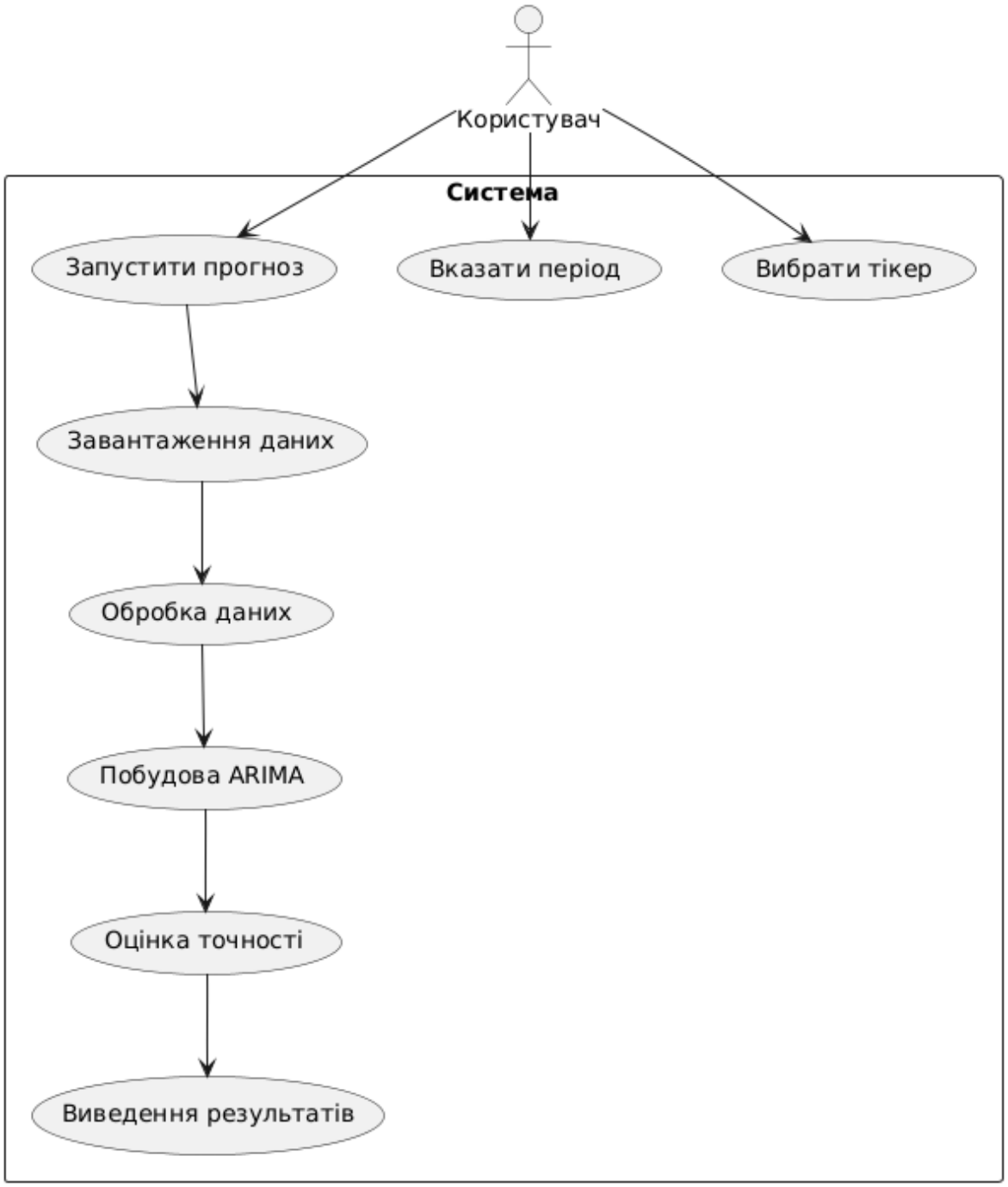

Рисунок Г.3 – Use Case діаграма програмного модуля прогнозування цін акцій

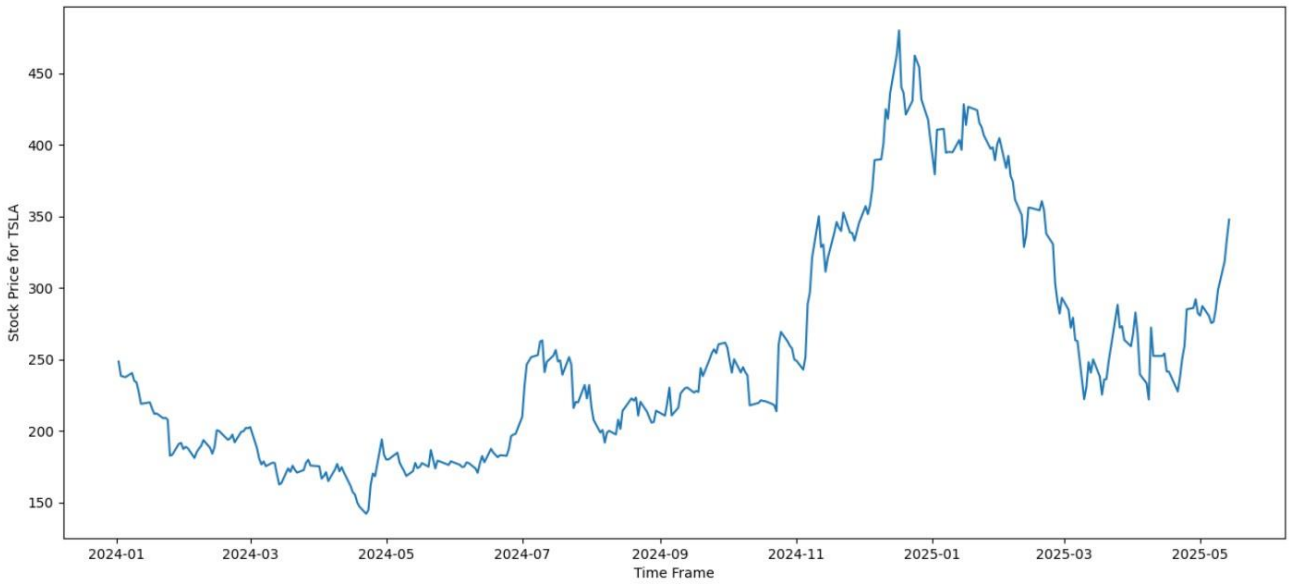

Рисунок Г.4 – Графік історичних цін акцій TSLA (Close Price)

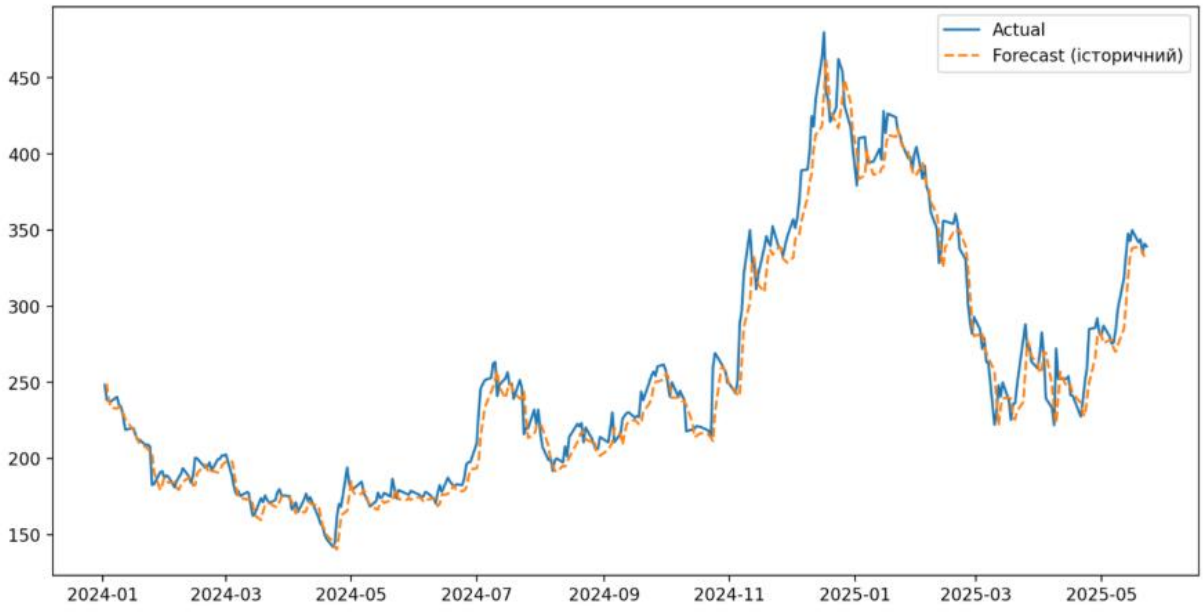

Рисунок Г.5 – Скріншот графіка з прогнозом на історичних даних

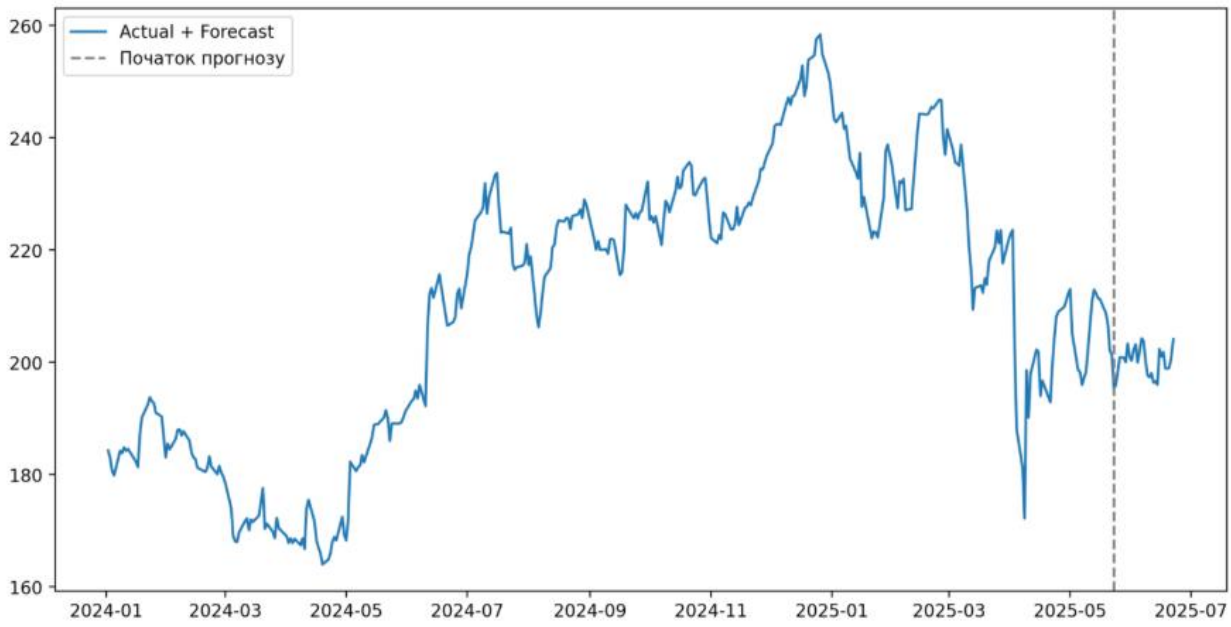

Рисунок Г.6 – Скріншот графіка з прогнозом на 30 днів уперед

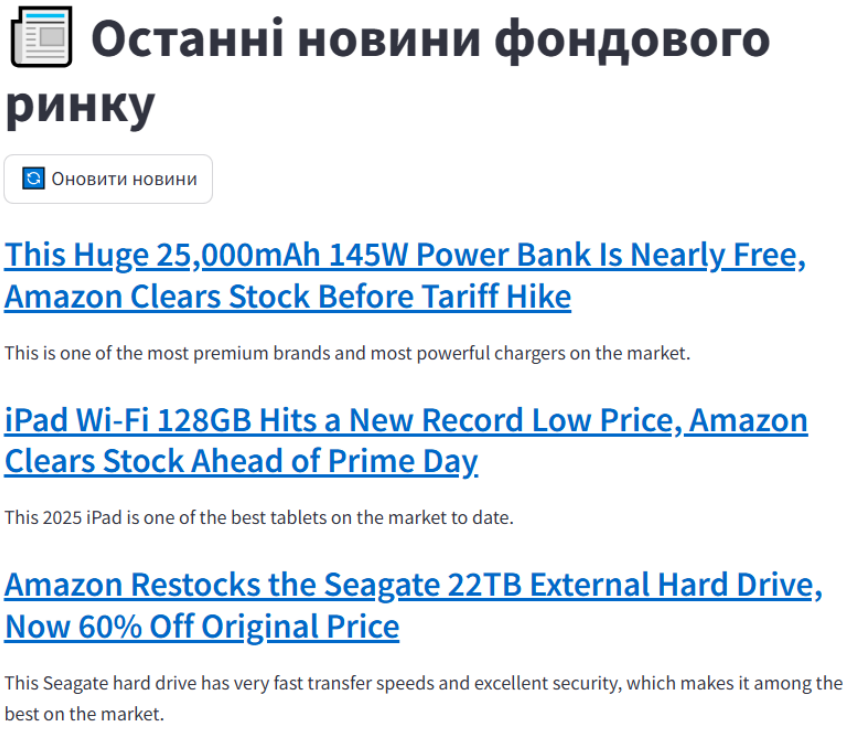

Рисунок Г.7 – Скріншот модуля новин з прикладом статей

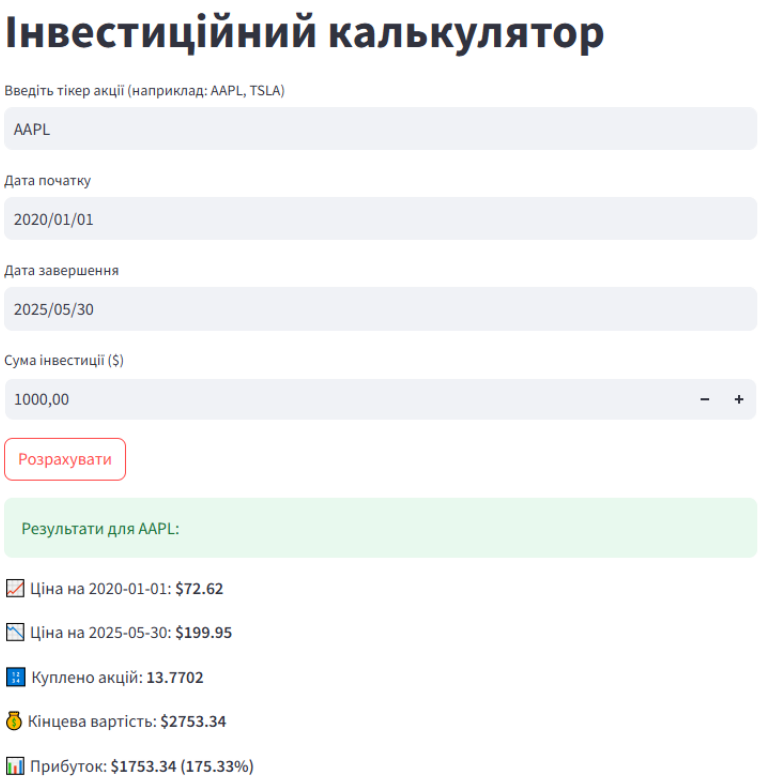

Рисунок Г.8 – Скріншот калькулятора з прикладом розрахунку

Додаток Д (довідниковий)

СВІДОЦТВО ПРО РЕЄСТРАЦІЮ АВТОРСЬКОГО ПРАВА

УКРАЇНА

СВІДОЦТВО

про реєстрацію авторського права на твір

№ 136468

Комп'ютерна програма «Програмний модуль прогнозування цін акцій фондового ринку»

(вид, назва твору)

Автор (співавтори) **Шевчук Олександр Федорович, Краснолуцька Яна Віталіївна**

(прізвище, ім'я, по батькові (за наявності), псевдонім (за наявності))

Авторські майнові права належать спільно **Шевчук Олександр Федорович, вул. Стельмаха, 21, кв. 90, м. Вінниця, 21029; Краснолуцька Яна Віталіївна, вул. Келецька, 98, м. Вінниця, 21021**

(прізвище, ім'я, по батькові (за наявності) фізичної особи / найменування юридичної особи, адреса)

Дата реєстрації 22 травня 2025 р.

Директор Державної організації «Український національний офіс інтелектуальної власності та інновацій» **Олена ОРЛЮК**

УКРНОІВІ