Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

РОЗРОБЛЕННЯ ІНФОРМАЦІЙНО-АНАЛІТИЧНОЇ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ ДЛЯ ОЦІНЮВАННЯ ІНВЕСТИЦІЙНОЇ ПРИВАБЛИВОСТІ НІШ В E-COMMERCE НА ОСНОВІ АНАЛІЗУ ВІДКРИТИХ ДАНИХ
(тема)

Виконав:
здобувач 4 року навчання,
групи ІТІНФ-22-1
Демченко М.А.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник _____
(посада, прізвище, ініціали)

Допускається до захисту

Завідувач кафедри інформатики _____ Кобилін О. А.
(підпис) (прізвище, ініціали)

2026 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту

Кафедра Інформатики

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Демченку Максиму Андрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення інформаційно-аналітичної системи підтримки прийняття рішень для оцінювання інвестиційної привабливості ніш в e-commerce на основі аналізу відкритих даних

затверджена наказом університету від 26 травня 2026 року № 435Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 19 червня 2026 р.

3. Вихідні дані до роботи науково-методична та науково-технічна література з питань електронної комерції, вебскрапінгу та теорії фінансових ризиків інвесторів; наукові статті у сфері інтелектуального аналізу ринку e-commerce; матеріали науково-практичних конференцій, дані інтернет-мережі, мова Python 3.13, бібліотеки playwright, BeautifulSoup (lxml), aiohttp, motor, pandas, matplotlib, NoSQL СУБД MongoDB, інструменти проєктування інтерфейсу customtkinter.

4. Перелік питань, що потрібно опрацювати в роботі

1. Аналіз специфіки функціонування маркетплейсу Amazon, методів моніторингу ринкового попиту, підходів до вилучення відкритих даних та критичний огляд аналітичних рішень.

2. Розробка економіко-математичної моделі оцінювання фінансових ризиків.

3. Програмна реалізація асинхронного скрапера-парсера відкритих даних, аналітичного ядра розрахунку інвестиційних метрик та настільного macOS-застосунку з інтегрованими графіками чутливості прибутку й модулем адміністрування історії в MongoDB.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій: приклад коду ASIN у каталозі, секція Buy Box та Other Sellers на сторінці товару, показник обсягу реалізації за 30 днів , концептуальна схема функціонування ІАС ППР , графік аналізу чутливості прибутку до зміни витрат логістики та маркетингу , блок-схема алгоритму проходження даних у системі, таблиця критеріїв верифікації достовірності попиту, логічна структура аналітичного документа бази даних, інтерфейс та вікна основних функціональних сценаріїв застосунку.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	13.04.2026	
2	Аналіз завдання, підбір літератури	14.04.26-18.04.26	
3	Аналіз літератури з проблеми, що вирішується	18.04.26-21.04.26	
4	Аналіз існуючих методів моніторингу ринку e-commerce та вилучення відкритих даних	23.04.26-27.04.26	
5	Формування кроків вирішення проблеми	28.04.26-03.05.26	
6	Програмна реалізація	06.04.26-23.05.26	
7	Аналіз отриманих результатів	24.05.26-01.06.26	
8	Оформлення пояснювальної записки	01.06.26-08.06.26	
9	Перевірка на нормоконтроль	10.06.26-19.06.26	
10	Перевірка на плагіат	11.06.26-30.06.26	
11	Рецензування	20.06.26-30.06.26	
12	Підготовка презентації та доповіді	20.06.26-30.06.26	
13	Занесення роботи в електронний архів	30.06.26-09.07.26	
14	Попередній захист кваліфікаційної роботи	30.06.26-09.07.26	

Дата видачі завдання 13 квітня 2026 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Яковлева О.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 76с., 5 табл., 26 рис., 0 дод., 24 джерело.

AMAZON, ВЕБСКРАПІНГ, ІНВЕСТИЦІЙНА ПРИВАБЛИВІСТЬ, IAC, МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ, ПІДТРИМКА ПРИЙНЯТТЯ РІШЕНЬ, РИЗИКИ.

Об’єктом роботи є розроблення IAC підтримки прийняття рішень для оцінювання інвестиційної привабливості товарних ніш на маркетплейсі Amazon.

Метою роботи є створення IAC для автоматизованого збору відкритих даних, розрахунку показників та формування обґрунтованого вердикту.

Під час роботи використано: економіко-математичне моделювання, методи вилучення, інтелектуального аналізу та крос-верифікації даних, моделювання NoSQL баз даних.

Практична цінність полягає в автоматизації оцінювання ризиків і прибутковості ніш та мінімізації людського фактора при розрахунках.

Розроблено IAC, що забезпечує паралельний збір ринкових метрик Amazon, розрахунок сукупних витрат і точки беззбитковості.

Область застосування: інтернет-торгівля, моніторинг глобальних маркетплейсів, управління фінансовими ризиками, торгівля на платформі Amazon.

AMAZON, IAS, INVESTMENT ATTRACTIVENESS, MATHEMATICAL MODELING, DECISION SUPPORT, RISKS, WEB SCRAPING.

The object of the work is developing a decision support IAS to assess the investment attractiveness of product niches on Amazon.

The goal is to develop an IAS for automated open data collection, metric calculation, and forming a justified verdict.

The methods used include economic-mathematical modeling, open data extraction, data mining, cross-verification, and document-oriented database modeling.

The practical value lies in automating risk and profitability assessment for niches and minimizing the human factor in calculations.

The developed IAS provides parallel collection of Amazon metrics, total costs, and break-even point calculation.

Applications: e-commerce, global marketplace monitoring, financial risk management, and retail trade on Amazon.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

API – Application Programming Interface (інтерфейс прикладного програмування, який задає правила взаємодії між програмами та автоматизованого обміну даними)

ASIN – Amazon Standard Identification Number (десятизначний унікальний буквено-цифровий код-ідентифікатор товару у внутрішньому каталозі Amazon)

BSR – Best Sellers Rank (рейтинг бестселерів Amazon, що відображає відносну кількість продажів товару в межах конкретної категорії)

Buy Box – найбільш візуально акцентована секція швидкого здійснення покупки на сторінці лістингу товару маркетплейсу

CAPTCHA – Completely Automated Public Turing test to tell Computers and Humans Apart (автоматизований тест для розрізнення комп'ютерів і людей з метою захисту від ботів)

DOM – Document Object Model (програмний інтерфейс для аналізу та обробки структури вебсторінок, що використовується під час вебскрапінгу)

DRI – Demand Reliability Index (інтегральний індекс надійності попиту, розроблений для верифікації реальної ринкової активності позиції)

E-commerce – електронна комерція (сфера глобальної роздрібної інтернет-торгівлі на транскордонних платформах

EAN – European Article Number (європейський міжнародний стандартний товарний код та штрихкод продукту)

ETL – Extract, Transform, Load (процес вилучення, трансформації та нормалізації даних перед їхньою фінансовою обробкою)

FBA – Fulfillment By Amazon (модель логістичного обслуговування, за якої зберігання, обробка замовлень та доставка здійснюються складами Amazon)

FBM – Fulfillment By Merchant (модель логістичного обслуговування, де пакування та безпосереднє відправлення замовлення покупцю виконується продавцем)

HTML – HyperText Markup Language (стандартна мова розмітки вебсторінок, вихідний код якої аналізується під час вебскрапінгу)

HTTP – HyperText Transfer Protocol (асинхронний протокол передачі гіпертексту для виконання прямих мережевих запитів до серверів маркетплейса)

ID – Identifier (унікальний цифровий або буквено-цифровий ідентифікатор об’єкта або аналітичного звіту в базі даних)

ISBN – International Standard Book Number (міжнародний універсальний стандартний книжковий номер)

JSON – JavaScript Object Notation (текстовий формат структурованого обміну даними, що використовується в API та базах даних)

NoSQL – клас систем керування базами даних, які не використовують жорсткі реляційні табличні схеми та є документоорієнтованими)

PPC Pay Per Click (модель інтернет-реклами, де оплата здійснюється за кожен клік користувача, основа маркетингових витрат системи)

REST – Representational State Transfer (архітектурний стиль взаємодії компонентів вебсервісів, на якому побудовано офіційний інтерфейс маркетплейсу)

Review Velocity – швидкість або інтенсивність приросту органічних відгуків покупців про товар за фіксований звітний період

ROI – Return on Investment (рентабельність інвестицій, ключовий відсотковий показник ефективності використання капіталу)

SP-API – Selling Partner API (офіційний сучасний REST-інтерфейс та безпечний шлюз для доступу розробників до даних Amazon)

UPC – Universal Product Code (американський міжнародний універсальний товарний код)

User-Agent – ідентифікаційний текстовий рядок клієнтського застосунку браузера, що використовується для імітації поведінки людини)

ІАС – інформаційно-аналітична система

ППР – підтримка прийняття рішень

СУБД – система управління базами даних

Unit-економіка – метод моделювання фінансової ефективності бізнесу через детальний розрахунок витрат і прибутку на одну одиницю товару

ВСТУП

Розвиток глобальних маркетплейсів, таких як Amazon, докорінно змінив підходи до ведення роздрібного бізнесу [1]. Сьогодні успіх у сфері e-commerce залежить не від інтуїції, а від точності аналізу великих масивів даних. Проблема більшості підприємців полягає у неможливості швидко обробити тисячі товарних позицій, розрахувати реальну конкуренцію та фінансові ризики вручну. Це призводить до помилок у виборі товарів, неефективного використання капіталу та швидкого виходу з ринку через невраховані витрати на логістику та маркетинг.

Предметною областю даного дослідження є процеси автоматизації аналізу ринкових ніш у транскордонній електронній торгівлі. Це охоплює методи збору даних (скрапінг), моніторинг динаміки цін, оцінку ліквідності товарів та розрахунок граничної вартості закупівлі для забезпечення рентабельності. Використання інформаційно-аналітичних систем дозволяє автоматизувати конкурентний аналіз, виявляти слабкі місця в асортименті опонентів та будувати прогнозні моделі прибутку.

Акуальність роботи поялгає у розробленні автоматизованого інструментарію для оцінювання інвестиційної привабливості ніш, що дозволяє підприємцям приймати обґрунтовані рішення на основі аналізу відкритих даних, мінімізувати фінансові ризики та точно розраховувати поріг входу в ринок e-commerce.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА МЕТОДІВ МОНІТОРИНГУ РИНКУ E-COMMERCE

1.1 Специфіка функціонування маркетплейсу Amazon

Amazon – найбільша торговельна платформа у світі, на якій працюють понад 3 мільйонів активних продавців [2]. Асортимент платформи налічує понад 350 мільйонів товарних позицій, а щоденний обсяг логістичних операцій перевищує 4 мільйони відправлень. Окрім торгівлі, корпорація отримує прибуток через екосистему супутніх сервісів: платні підписки (Amazon Prime), дистрибуцію цифрового контенту (аудіокниги, стрімінгові сервіси) та хмарні обчислення.

Детальний аналіз архітектури платформи дозволяє виділити ключові механізми роботи платформи. Кожен товар у каталозі має унікальний номер ASIN (Amazon Standard Identification Number). Цей десятизначний буквено-цифровий код виконує роль внутрішнього ID продукту. На відміну від міжнародних стандартів, таких як ISBN, UPC або EAN, код ASIN є специфічним виключно для внутрішнього каталогу Amazon [3]. На рисунку 1.1 представлено ASIN код на сторінці товару.

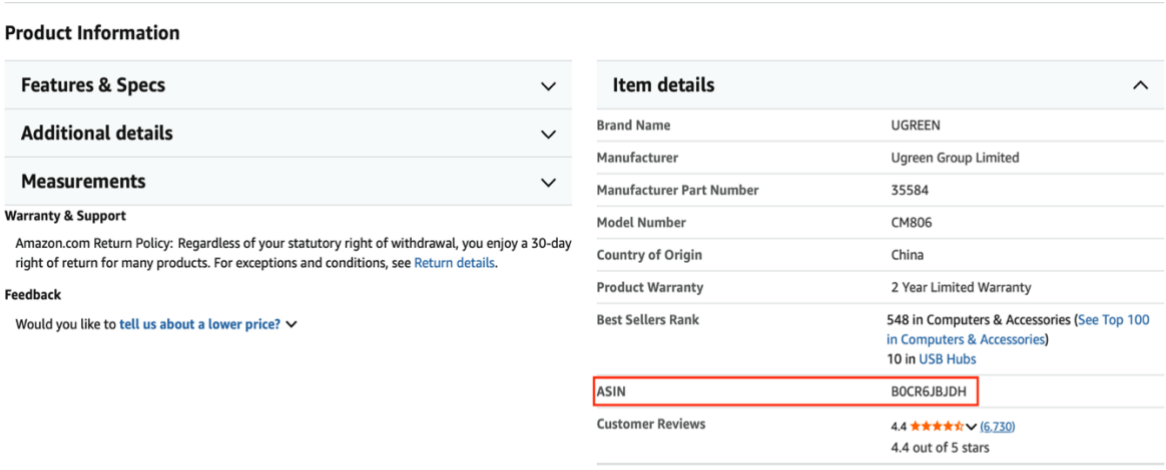

Рисунок 1.1 – Приклад ASIN

Використання ASIN забезпечує точність пошукових запитів та оптимізацію клієнтського досвіду. Завдяки цій системі ідентифікації інформаційно-аналітична система може здійснювати однозначне зіставлення даних під час моніторингу цін.

Для взаємодії з продавцями Amazon пропонує дві основні моделі логістичного обслуговування:

– FBA (Fulfillment By Amazon) – модель, за якої зберігання, обробка замовлень та доставка здійснюються зі складів Amazon. Продавець делегує логістичні процеси платформі, що дозволяє автоматизувати масштабування бізнесу. Ключовою перевагою моделі FBA є більша лояльність алгоритмів Amazon до продавця;

– FBM (Fulfillment By Merchant) – модель, де маркетплейс використовується лише як вітрина та платіжний шлюз. Пакування, маркування та безпосереднього відправлення товару клієнту залишаються в зоні відповідальності продавця [4].

Додатковим бонусом FBA є статус «Prime», що відкриває доступ до багатомільйонної аудиторії підписників і гарантує безкоштовну дводенну доставку. Це створює суттєву перевагу у швидкості сервісу та довірі з боку покупця.

Попри очевидні переваги FBA, значна частина продавців обирає модель FBM через такі чинники:

– відсутність щомісячних комісій за зберігання та зборів за обробку повернень;

– мінімізація фінансових втрат у разі зміни внутрішніх тарифів Amazon;

– можливість використання єдиного складського запасу для торгівлі на інших майданчиках (eBay, Etsy, Walmart).

Якщо продавець працює за системою FBA, у нього є можливість отримати «Buy Box». Buy Box – це велика секція, яка розташована в тій області, куди клієнти дивляться під час здійснення покупки (рис. 1.2). Близько

90% продажів здійснюються через цей блок і лише 10% – через список «Other Sellers», володіння «Buy Box» є визначальним фактором конкурентоспроможності [5]. Це зумовлено структурою сторінки лістингу.

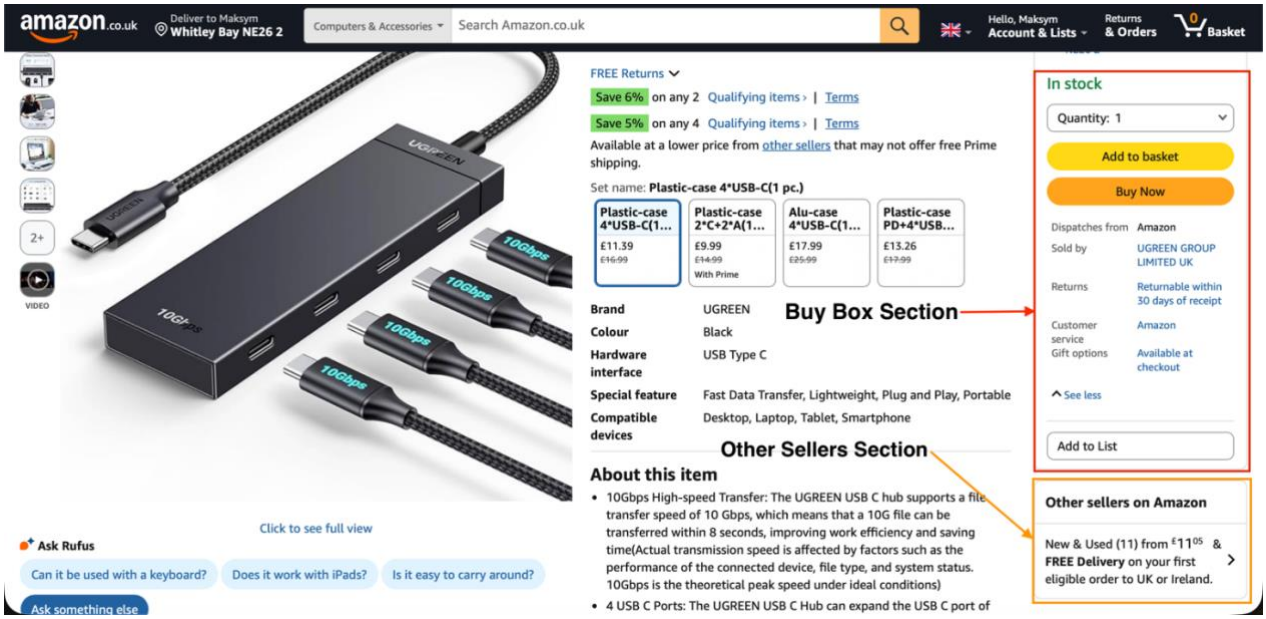

Рисунок 1.2 – Секція «Buy Box» та «Other Sellers» на сторінці товару

Секція Buy Box є найбільш візуально акцентованою та зручною для швидкого оформлення замовлення.

Окрему увагу слід приділити алгоритмам розподілу Buy Box. Це динамічний розділ, що включає кнопки «Add to Cart» та «Buy Now». Якщо покупець не змінює продавця вручну, продаж отримує саме той учасник, який на цей момент займає місце в Buy Box.

1.2 Показники ринкового попиту та методи оцінювання конкурентного середовища

Кількість відгуків, динаміка цін та середній рейтинг є формою «соціального доказу», що безпосередньо корелює з рівнем довіри споживачів

та обсягами продажів. У контексті аналізу ніші ці показники виступають як «бар'єри входу».

Amazon BSR (Best Sellers Rank) – це рейтинг бестселерів. Amazon призначає BSR продукту, який вказує на його показники продажів у конкретній категорії.

Amazon BSR використовує порядкову шкалу. Нижчий BSR на продукті вказує на те, що конкретний товар має високі продажі. Продукт з BSR 1 є найкращим у цій категорії, оскільки він отримує величезні продажі.

BSR – це дуже важливе число, яке має більшість продуктів Amazon. Цей номер надається продукту, коли він досягає принаймні одного продажу. Однак кінцева мета BSR полягає в тому, щоб показати, наскільки добре даний продукт продається на Amazon. Чим нижчий BSR продукту, тим вищі його продажі. Цю метрику Amazon надає кожному, хто переглядає їх веб-сайт або додаток, щоб ви знали, які продукти є найбільш продаваними на Amazon [6]. Жодна інша платформа для продажу не має нічого подібного, що дозволяє будь-кому, будь то продавцям чи покупцям, знати в будь-який момент часу, які продукти є найбільш продаваними в певній категорії. Рейтинг BSR розраховується на рівні категорії та підкатегорії. На рисунку 1.3 представлено рейтинг BSR на сторінці товару.

Кількість відгуків і середній рейтинг – найголовніші показників вартості входу в нішу. Якщо у топ-10 продавців більше ніж 5000 відгуків, то зайти в нішу практично неможливо. Це означає, що доведеться витратити величезні бюджети на рекламу, щоб стати помітним і при тому не факт, що буде створюватись конкуренція. Необхідно шукати ніші, де в топі є хоча б 2–3 лістинги з менш ніж 100–300 відгуками – це «вікно можливостей».

Ще одним показником є показник якості задоволення попиту. Ніші з низьким середнім рейтингом, наприклад менше 4.2 зірок при високому попиті є потенційно привабливими для розроблення та виведення на ринок якіснішого продукту.

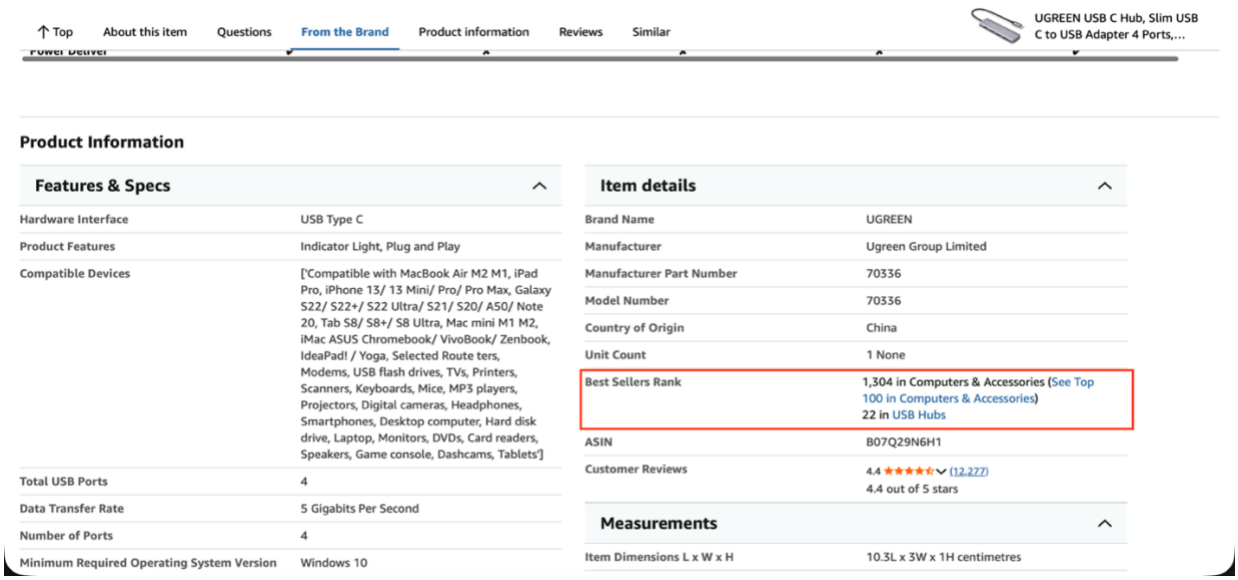

Рисунок 1.3 – Best Sellers Rank

Аналіз насиченості дозволяє оцінити ступінь монополізації ринку великими гравцями. Для оцінювання використовується підхід, близький до розрахунку індексу Герфіндаля-Гіршмана, але адаптований під топ-20 результатів пошукової видачі [7]. Якщо 3-5 основних брендів контролюють понад 60% обсягу продажів у ніші, такий ринок вважається висококонцентрованим. Для нового продавця це означає ризик агресивного витіснення з боку лідерів.

Натомість висока частка небрендованих товарів у топі свідчить про низьку лояльність покупців до конкретних марок, що спрощує вхід для нових інвесторів.

Також одним із показників для оцінки є ціна. Цей параметри є динамічним, він змінюється під впливом алгоритмів. Оцінювання інвестиційної привабливості на основі лише поточної ціни є методологічною помилкою, оскільки вона не враховує акційні періоди та цінові війни. Система повинна порівнювати поточну ціну з медіанним значенням за тривалий період. Це дозволяє уникнути входу в нішу під час тимчасового демпінгу. Висока волатильність ціни свідчить про нестабільність ніші або агресивну конкуренцію, що робить прогнозування чистого прибутку менш точним.

1.3 Методи та підходи до вилучення відкритих даних

Для проведення аналізу даних спочатку необхідно отримати їх із вебсторінки, що містить потрібну інформацію, для цього існує два робочих методи:

– Web scraping – це перетворення у структуровані дані інформації з вебсторінок, які призначені для перегляду людиною за допомогою браузера [8]. Виконується за допомогою комп'ютерних програм, що імітують поведінку людини в інтернеті, або з'єднуючись з вебсервером напряму по протоколу HTTP, або управляючи повноцінним веббраузером;

– API (Application Programming Interface) – це посередник між програмами, який задає правила «спілкування», програма лише надсилає запит та отримує відповідь від програми до якої звертається використовуючи лише її інтерфейс [9].

Для забезпечення точності аналітичних звітів існує Amazon Selling Partner API (SP-API). Це сучасний REST-інтерфейс, який є офіційним шлюзом для доступу до даних маркетплейсу. На відміну від Web scraping, SP-API використовує стандарти які гарантують безпечне з'єднання, але вимагають реєстрації розробника в Amazon Developer Central [10, 11]. Через API можна отримувати дані, які важко або неможливо стабільно дістати через HTML-код сторінки, такі як:

– точний розрахунок комісій Amazon на основі габаритів товару;

– дані про ціни конкурентів у режимі реального часу, включаючи стан Buy Box;

– повна технічна специфікація товару за його ASIN.

Використання API дозволяє верифікувати дані, зібрані скрапером. Якщо вебскрапінг дає нам картинку очима покупця, то API надає структуровані JSON-дані, які легше обробляти алгоритмам. Це значно знижує ризик помилки

при розрахунку ROI, оскільки отримуються офіційні цифри щодо податків та логістичних зборів безпосередньо від платформи.

У контексті розроблення інформаційно-аналітичної системи, вебскрапінг має низку критичних переваг над офіційним API:

– не потрібно чекати на схвалення від Amazon, скрапер може почати збір даних миттєво, що критично для етапу розроблення та тестування;

– офіційні ліміти API не дозволяють швидко проаналізувати тисячі товарів;

– деякі візуальні елементи, детальна ієрархія категорій або специфічні віджети маркетингових акцій легше дістати через аналіз DOM-структури сторінки, ніж через структуровані відповіді API [12];

– скрапер бачить сторінку точно так само, як її бачить покупець у цю секунду, що дозволяє миттєво фіксувати зміну ціни в Buy Box або появу нових відгуків, тоді як дані в API можуть оновлюватися із затримкою.

Не дивлячись на переваги вебскрапінгу, він також має достатньо недоліків, які можуть стати критичними. Amazon володіє однією з найдосконаліших систем захисту від автоматизованого збору даних у світі. Оскільки прямий вебскрапінг створює навантаження на сервери та дозволяє конкурентам отримувати стратегічну інформацію, платформа використовує багаторівневі механізми блокування ботів. До недоліків можна віднести:

– при виявленні аномальної кількості запитів з однієї адреси, Amazon тимчасово або назавжди вносить її до «чорного списку»;

– система захисту аналізує рухи курсора, швидкість кліків та заголовки запитів. Якщо система розпізнає бота, з’являється CAPTCHA;

– Amazon зчитує відбиток браузера (версія ОС, шрифти, розширення екрана).

Але, не зважаючи на те, що SP-API є офіційним інструментом, який не може бути заблокованим, він також має низку суттєвих недоліків, які обмежують його використання як єдиного джерела даних, до прикладу:

– щоб отримати доступ до API, потрібно зареєструватися як професійний розробник. Amazon вимагає детальну інформацію про бізнес який робить запит, політику безпеки даних і може тижнями перевіряти застосунок;

– Amazon використовує алгоритм для обмеження частоти запитів. Якщо система буде аналізувати тисячі товарів одночасно, то буде накладено ліміт;

– деякі критично важливі дані (наприклад, детальна інформація про покупців або специфічні звіти по замовленнях) доступні лише за умови, що продавець надав застосунку прямий доступ. Для «зовнішнього» аналізу ніш API часто віддає менше інформації, ніж можна «побачити» на живій сторінці товару;

– дані в API не завжди оновлюються в реальному часі. Іноді BSR або ціна в API можуть відставати від реальних показників на сайті на кілька годин.

Узагальнене порівняння методів показано у таблиці 1.1.

Таблиця 1.1 – Порівняння Selling Partner API та вебскрапінгу

Критерій	**Selling Partner API**	**Вебскрапінг**
Доступ	Складний	Відкритий
Швидкість запитів	Суворо обмежена	Масштабована
Точність	Висока	Середня, можливі неточності
Ризик блокування	Відсутній	Високий
Вартість	Безкоштовно	Потребує невеликих витрат

1.4 Економіко-математичні моделі оцінювання інвестиційної привабливості

Теоретичне обґрунтування інвестиційної привабливості в e-commerce базується на синтезі фінансового аналізу та методів математичної статистики.

У міжнародній практиці аналізу маркетплейсів фундаментальним методом є Reverse Sourcing [13]. На відміну від класичного підходу, де ціна

продажу формується на основі собівартості, ця модель працює у зворотному напрямку:

– за вихідну точку береться встановлена ринком ціна у конкретній ніші;

– від неї послідовно віднімаються фіксовані та змінні витрати (комісії, логістика, податки);

– результатом є «гранична вартість закупівлі», яка визначає, чи є сенс взагалі входити в дану нішу.

Для математичного опису привабливості товару використовують два ключові показники – чиста маржа та рентабельність інвестицій (ROI).

Чиста маржа відображає відсоток прибутку в кожній одиниці доходу після врахування всіх витрат і визначається наступним співвідношенням:

$$\text{Чиста маржа} = \frac{\text{P-TC}}{\text{P}} \times 100\%, \quad (1.1)$$

де P – ціна реалізації;

TC – повна собівартість.

Ефективність використання залученого капіталу інвестора відображає показник рентабельності інвестицій, розрахунок якого здійснюється таким чином:

$$\text{ROI} = \frac{\text{Чиста маржа}}{\text{Інвестиційні витрати}} \times 100\%. \quad (1.2)$$

У теорії e-commerce прийнятним вважається рівень ROI, отриманого в (1.2), від 30%, що дозволяє перекривати можливі коливання цін та витрати на маркетинг.

Також існують наступні методологічні підходи до верифікації попиту:

– використання прямих кількісних індикаторів реалізації;

– метод верифікації нижньої межі;

– калібрування прогнозних моделей.

Сучасні алгоритми Amazon публічно відображають приблизні кількісні показники реалізації за останні 30 днів. Не зважаючи на те, що ці показники неточні, вони все ж таки є цінними для аналізу. На рисунку 1.4 представлені кількісні показники на сторінці товару.

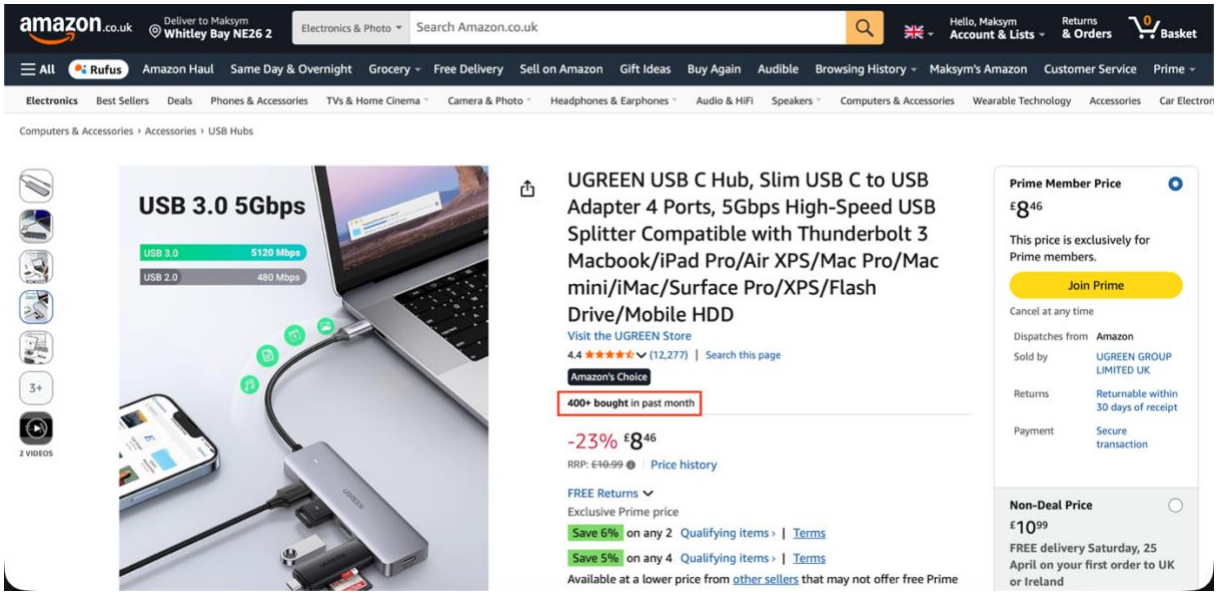

Рисунок 1.4 – Показник реалізації за останні 30 днів

На відміну від динамічного BSR, який змінюється щогодини, показник реалізації за останні 30 днів дає стабільну нижню межу місячного попиту. Це дозволяє мінімізувати помилки прогнозування, що виникають через різкі коливання рейтингу (наприклад, після короткочасних рекламних кампаній).

1.5 Критичний аналіз існуючих аналітичних рішень та програмних продуктів

На сучасному ринку e-commerce існують комерційні платформи для аналізу Amazon, такі як Helium 10, Jungle Scout та Keepa [14,15]. Основною проблемою цих сервісів є те, що вони є закритими. Підприємець не може

бачити розраховані моделі та повинен довіряти цим сервісам лише на основі їх слів.

До основних недоліків можна віднести:

– висока вартість підписки;

– відсутність гнучкого розрахунку Unit-економіки;

– часто не враховуються локальні показники, такі як логістика, податкові витрати та змінні маркетингові витрати;

– як правило системи не є гнучкими, немає можливості аналізувати показники які можуть змінитись і проект перетвориться на збитковий;

– затримка даних через те, що ці системи працюють на офіційних арі які в свою чергу працюють із затримкою.

Схожі системи існуть, але їх обмеження не дозволяють підприємцям стовідсотково довіряти їм.

1.6 Постановка задачі

Оцінювання інвестиційної привабливості товарних ніш на глобальних маркетплейсах є критично важливим завданням для забезпечення рентабельності проєктів у сфері e-commerce. Враховуючи динамічність ринку Amazon, Ebay, Etsy та інших маркетплейсів, а також високу щільність конкурентного середовища, виникає необхідність в автоматизації процесів збору та інтелектуальної обробки відкритих даних за для зменшення шансу на невдачу під час роботи з капіталом. Тому ставиться завдання створити інформаційно-аналітичну систему підтримки прийняття рішень, що дозволить здійснювати комплексний моніторинг ринкових показників та моделювати фінансові результати входу в нішу.

Об’єктом роботи є розроблення ІАС підтримки прийняття рішень для оцінювання інвестиційної привабливості товарних ніш на маркетплейсі Amazon.

Метою роботи є створення ІАС для автоматизованого збору відкритих даних, розрахунку показників та формування обґрунтованого вердикту.

Для досягнення мети необхідно вирішити такі завдання:

– проаналізувати специфіку функціонування маркетплейсу Amazon та визначити ключові метрики конкурентоспроможності товарів;

– дослідити та обґрунтувати методи вилучення відкритих даних (Web Scraping та API) в умовах протидії антифрод-системам;

– розробити економіко-математичну модель оцінювання інвестиційної привабливості, що інтегрує розрахунок ROI та метод верифікації нижньої межі попиту;

– спроєктувати архітектуру інформаційно-аналітичної системи та реалізувати її програмний модуль;

– провести тестування системи на реальних товарних нішах та оцінити точність сформованих прогнозів і фінансових моделей.

2 ПРОЄКТУВАННЯ ТА ОБҐРУНТУВАННЯ ВИБОРУ ЗАСОБІВ РЕАЛІЗАЦІЇ ІНФОРМАЦІЙНО-АНАЛІТИЧНОЇ СИСТЕМИ

Проєктування системи здійснюється з урахуванням специфіки апаратного забезпечення на базі архітектури Apple Silicon (macOS), що зумовлює вибір інструментарію, здатного ефективно використовувати ресурси Unix-подібних систем для паралельної обробки даних та високошвидкісного вебскрапінгу. Метою є побудова логіко-математичної моделі системи, яка трансформує неструктуровану інформацію маркетплейсу у верифіковані інвестиційні рішення.

2.1 Формалізація задачі підтримки прийняття рішень

Функціонування розроблюваної ІАС базується на методології зворотного інжинірингу прибутковості. Основна мета системи полягає не просто у зборі статистичних даних, а у формуванні «захисного бар'єру» для капіталу інвестора шляхом превентивного виявлення збиткових товарних позицій [16]. Задача підтримки прийняття рішень у даному контексті формалізується як пошук максимально допустимої ціни закупівлі ($P_{\max\_buy}$), за якої проєкт залишається рентабельним у межах щільного ринкового середовища.

Для подальшого визначення критеріїв оцінювання, формалізації завдання та розуміння самої суті ІАС розглянемо практичний кейс. Маємо людину (інвестора), у якої є капітал у розмірі 10 000$. Це гроші, які людина довго відкладала і тепер хоче вкласти у товарний бізнес. Інвестору подобається ідея продавати павербанки фірми Belkin, оскільки він сам користується цим пристроєм і впевнений у надійності бренду.

Проте тут виникає критичний нюанс: інвестор не хоче просто купити товар, він хоче заробити. Будь-яке вкладення капіталу – це ризик залишитися з неліквідним товаром на руках або, що ще гірше, піти в мінус. Тому головна вимога до системи – це мінімізація ризиків там, де це можна прорахувати математично.

У інвестора є ринок збуту (маркетплейс), де він бачить усіх майбутніх конкурентів, їхні ціни та кількість відгуків. Припустимо, він знаходить постачальника, який пропонує павербанки оптом по 15$ за штуку, в той час як на ринку вони продаються по 20$. На перший погляд, прибуток у 5$ з одиниці здається очевидним, але саме тут криється «інвестиційна пастка».

Інвестор відправляє посилання на конкретну позицію в ІАС, і система розписує наступний сценарій:

– система автоматично аналізує та розраховує приховані витрати, розраховує логістику, реферальну комісію маркетплейсу, податки, витрати на упаковку, середній відсоток браку в ніші та багато інших витрат;

– система оцінює вартість реклами. Якщо конкуренти з тисячами відгуків продають по 20$, то нашому інвестору, щоб його взагалі помітили, доведеться витрачати умовно по 2-3$ на залучення кожного покупця;

– проаналізувавши всі ці фактори, ІАС видає вердикт: щоб продавати по 20$ і мати хоча б мінімальний прибуток, ціна закупівлі має бути не 15$, а, наприклад, 8$.

Таким чином, маючи пропозицію від постачальника по 15$, інвестор завдяки системі бачить реальну картину: якщо він вкладе свої 10 000$ зараз, то замість прибутку він отримає або «нуль», або взагалі залишиться з купою павербанків, які неможливо реалізувати в плюс. Система працює як «холодний душ», який оберігає капітал від необдуманих рішень.

Всі ці розрахунки можна зробити вручну, але це довго, складно і завжди є ризик людської помилки – забути про один дрібний податок або неправильно порахувати габарити, що в масштабах партії на 10 000$ призведе до

фінансового краху. Задача ІАС – автоматизувати цей процес, видавши конкретні цифри, звіти та висновки, мінімізуючи людський фактор. На рисунку 2.1 представлено діаграму функціонування ІАС ППР.

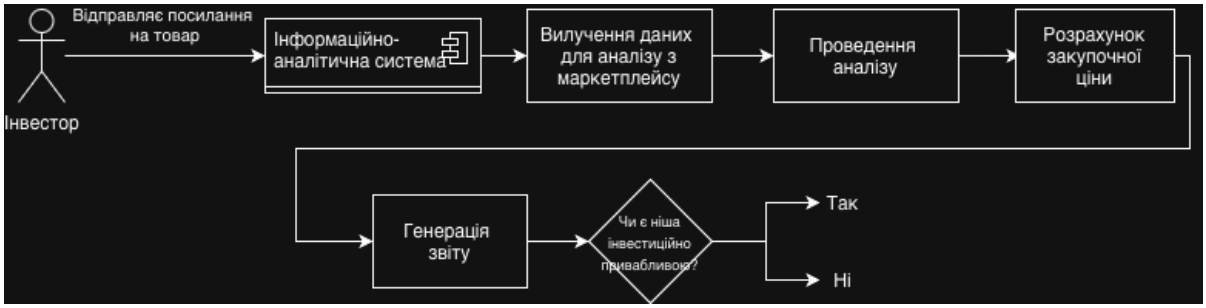

Рисунок 2.1 – Концептуальна схема функціонування ІАС ППР

На рисунку 2.1 представлено логічну послідовність процесів, що реалізуються в інформаційно-аналітичній системі під час проведення інвестиційного аналізу товарної позиції. Кожен етап схеми є критично важливим для формування об’єктивного фінансового висновку.

Процес починається з передачі користувачем прямого посилання на товар. На відміну від автоматизованих веб скраперів, дана система ідентифікує конкретний об’єкт за його посиланням, що дозволить більш влучно аналізувати конкретний товар у конкретного продавця, ця система дозволить збирати дані відповідно до побажань користувача.

Далі система автоматично запускає збір усієї необхідної інформації. Система звертається до сторінки товару та вилучає:

- поточну ціну;
- показники попиту;
- технічні параметри (вага, розміри пакування);
- кількість активних конкурентів.

На цьому етапі ІАС автоматично враховує регіональну специфіку та коригує валютні одиниці.

Система проводить комплексний аналіз усіх сирих показників. Це «аналітичний центр» системи, де зібрані дані перетворюються на показники Unit-економіки. Тут відбувається розрахунок усіх неявних витрат:

– комісії маркетплейсу;
– логістичних зборів;
– оціночних витрат на рекламу;
– податкових відрахувань.

Повертаючись до прикладу з павербанками Belkin, саме на цьому етапі система виявляє, що реальний дохід з одиниці товару значно нижчий за різницю між ціною закупівлі та ціною продажу.

На наступному етапі система математично розраховує граничну ціну закупки. Це найбільш інноваційний компонент системи. Замість того, щоб просто констатувати факт прибутку, ІАС здійснює зворотій розрахунок. Вона визначає ціну, вище якої закупівля товару стає збитковою. Якщо інвестор бачить пропозицію від постачальника за 15$, а система розрахувала граничну ціну у 8$, це стає головним аргументом проти здійснення інвестиції саме у цього постачальника.

Фінальний етап передбачає формування структурованого аналітичного висновку з усіма розрахунками. Логічний блок «Чи є ніша інвестиційно привабливою?» працює на основі порівняння розрахованого ROI з пороговим значенням користувача. Вердикт «Так» видається лише у випадку, коли прогнозований чистий залишок капіталу після всіх операційних циклів відповідає очікуванням інвестора та ніша не є занадто переповненою.

Задача оцінювання привабливості товару (A) може бути сформульована як знаходження цільової функції F(A), що базується на векторі вхідних параметрів.

Множина ключових характеристик товарної позиції, які інформаційно-аналітична система вилучає шляхом автоматизованого вебскрапінгу сторінки лістингу, має наступний вигляд:

$$S=\{P_m, W,V,R,B,Q\}, \quad (2.1)$$

де P_m – ринкова ціна;

W, V – вага та об’єм товару відповідно;

R – кількість відгуків конкуртентів;

B – показник Best Sellers Rank;

Q – підтверджений обсяг продажів за останні 30 днів.

Головним завданням формалізації математичного апарату системи є обчислення цільового параметра, який гарантує інвестору досягнення заданої норми рентабельності. Розрахунок даного параметра здійснюється за допомогою виразу:

$$P_{target}=P_m-(C_{fba}(W,V)+C_{ref}(P_m)+C_{ads}+C_{tax}+C_{other})-\frac{K\times ROI_{target}}{N}, \quad (2.2)$$

де $C_{fba}(W, V)$ – функція витрат на логістику, що залежить від габаритів;

C_{ref} – реферальна комісія;

K – обсяг капіталу інвестора;

B – показник Best Sellers Rank;

N – кількість одиниць товару, яку планується закупити.

Для систематизації всіх чинників, що впливають на прийняття рішення, сформуємо таблицю 2.1, яка описує структуру даних, що обробляються IAC.

Таблиця 2.1 – Опис параметрів прийняття інвестиційних рішень.

Параметр	**Позначення**	**Вплив на рішення**	**Джерело даних**
1	**2**	**3**	**4**
Ринкова ціна	P_m	Визначає верхню межу виручки	Вебскрапінг сторінки
Логістичний збір	C_{fba}	Зменшує маржу залежно від фізичних розмірів	Розрахунок за тарифами
Маркетинговий тиск	C_{ads}	Визначає вартість входу в нішу	Аналіз конкуренції

Продовження таблиці 2.1

1	2	3	4
Попит	Q	Верифікує реальність оборотності капіталу	Вебскрапінг сторінки
Цільовий прибуток	П	Визначає доцільність вкладення капіталу	Запит користувача

На основі розрахованих показників система формує інтегральний висновок привабливості $I \in \{0,1\}$, де:

– I=1 (Інвестиція рекомендована), якщо $ROI_{real} \geq ROI_{target}$ та $P_{buy} \leq P_{target}$;

– I=0 (Інвестиція ризикована), якщо $ROI_{real} < ROI_{target}$ або виявлено критичне домінування брендів-конкурентів.

2.2 Математична модель оцінювання фінансових ризиків

Оцінювання фінансових ризиків при інвестуванні в товарні ніші базується на математичному моделюванні ймовірності дефіциту. У межах функціонування IAC ризик розглядається як багатофакторна величина, що залежить від змін ринкової ціни, точності розрахунку операційних витрат та ефективності рекламних кампаній [17]. Основна задача даного підрозділу – формалізувати умови, за яких інвестиція перетворюється на збиткову.

Основна задача – формалізувати умови, за яких початковий капітал інвестора піддається загрозі втрати через непередбачені витрати. Для виявлення потенційних фінансових ризиків обчислювальне ядро системи здійснює розрахунок сукупних операційних витрат на реалізацію однієї одиниці товару за наступною залежністю:

$$C_{total} = C_{landed} + C_{fba} + C_{ref} + C_{storage} + C_{ads} + C_{tax} + C_{risk}, \quad (2.3)$$

де: C_{total} – сумарні операційні витрати на реалізацію однієї одиниці товару;

C_{landed} – фактична ціна закупівлі товару у постачальника;

C_{fba} – витрати на логістику, що розраховуються на основі ваги та габаритів;

C_{ref} – реферальна комісія маркетплейсу;

$C_{storage}$ – вартість зберігання одиниці товару на складі з урахуванням сезонних коефіцієнтів;

C_{ads} – прогнозні витрати на маркетинг, необхідні для підтримки обсягу продажів;

C_{tax} – сумарні податкові зобов'язання;

C_{risk} – фінансовий резерв на покриття відсотка браку, логістичних втрат та повернень.

Саме показник C_{total} дозволяє системі провести діагностику: якщо ціна реалізації на ринку складає 20$, а розрахований C_{total} дорівнює 19$, то інвестиція є високоризикованою, оскільки будь-яке коливання ринку призведе до збитків.

Точка беззбитковості роздрібної вартості позиції базується на сформованому векторі операційних витрат і визначається математичним співвідношенням:

$$P_{be}=\frac{C_{buy}+C_{fba}+C_{storage}+C_{ads}+C_{risk}}{1-(\alpha_{ref}+\alpha_{tax})}, \quad (2.4)$$

де P_{be} – розрахункова точка беззбитковості;

α_{ref} – коефіцієнт реферальної комісії;

α_{tax} – коефіцієнт податкових відрахувань.

Головним індикатором стійкості капіталу для інвестора в розроблюваній комп'ютерній моделі виступає запас міцності маржі, який показує допустиму

глибину падіння роздрібної ціни до моменту виходу бізнес-моделі в зону збитків:

$$M_{safety}=\frac{P_m-P_{be}}{P_m}\times 100\%, \quad (2.5)$$

де M_{safety} – запас міцності маржі у відсотках;

P_m – поточна середня ціна.

Якщо користувач не має ціни від постачальника, система трансформує базовий баланс для безпосереднього знаходження цільової ціни закупівлі:

$$P_{target}=P_m\times(1-\alpha_{ref}-\alpha_{tax}-ROI_{desired})-(C_{fba}+C_{storage}+C_{ads}+C_{risk}), \quad (2.6)$$

де P_{target} – ідеальна ціна закупівлі, яку інвестор має виставити як умову постачальнику;

$ROI_{desired}$ – бажаний рівень рентабельності інвестицій, заданий користувачем [18].

Для автоматизації видачі висновків система має використовувати універсальну шкалу класифікації ризиків (табл. 2.2).

Таблиця 2.2 – Універсальна класифікація рівнів фінансового ризику інвестиційного об’єкта.

Рівень ризику	**Значення M_{safety}**	**Вплив на інвестиційний капітал**	**Рекомендація системи**
1	**2**	**3**	**4**
Низький	>30%	Висока стійкість до ринкових коливань	Рекомендовано до інвестування
Помірний	15-30%	Можливе зниження ціни при зростанні витрат на рекламу кампанію	Потребує періодичного спостереження

Продовження таблиці 2.2

1	2	3	4
Високий	5-15%	Висока імовірність нульової рентабельності	Не рекомендується до інвестування без зниження C_{buy}
Критичний	<5%	Висока імовірність часткової або повної втрати інвестиційного капіталу	Заборонено для інвестування

Завершальним етапом математичного є аналіз чутливості. Система симулює зміну вартості залучення клієнта та коливання ринкової ціни на інтервалі 20% [19]. Це дозволяє інвестору побачити не просто статичну цифру, а динамічний прогноз стабільності його капіталу (рис. 2.2).

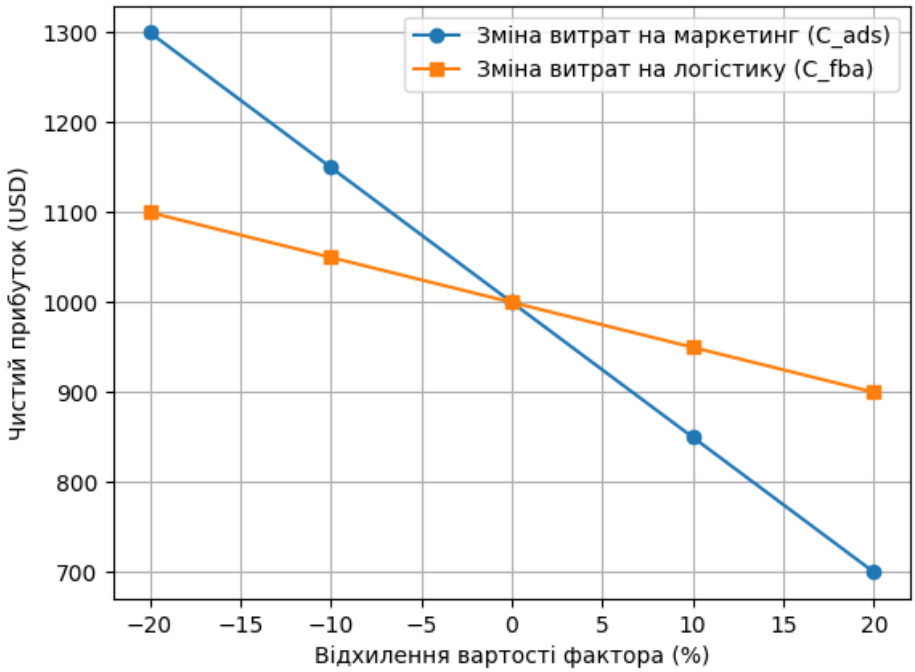

Рисунок 2.2 – Графік аналізу чутливості прибутку до зміни вартості логістики та маркетингу.

Графік візуалізує «запас міцності» проєкту через аналіз чутливості прибутку до зміни витрат:

– синя лінія має стрімкий нахил. Це свідчить про високу чутливість бізнес-моделі до вартості реклами. Зростання рекламних витрат на 20% призводить до падіння прибутку на 30%;

– помаранчева лінія більш полога. Зміна тарифів на доставку та зберігання менше впливає на фінальний результат, що робить цей фактор прогнозованим та стабільним.

Головним ризиком для даної інвестиції є волатильність рекламного аукціону.

2.3 Алгоритмічне моделювання системних зв’язків

Основна складність алгоритмічного моделювання системних зв’язків у межах розроблюваної IAC полягає в необхідності поєднання високої швидкості обробки неструктурованих даних із забезпеченням стійкості до зовнішніх обмежень маркетплейсу. Цей етап є переходом від абстрактних математичних моделей до їхньої програмної реалізації. Побудована алгоритмічна модель базується на принципах модульної декомпозиції, де кожен функціональний блок виконує специфічну роль у загальному процесі трансформації вхідного посилання на товар у фінальний інвестиційний звіт.

Функціонування системи починається з ініціалізації модуля мережевої взаємодії, який на логічному рівні керує сесіями та браузерними контекстами. Оскільки сучасні маркетплейси використовують складні системи детекції автоматизованих запитів, алгоритм скрапінгу не може бути лінійним. Він реалізує логіку людської поведінки, що включає динамічну ротацію заголовків запитів та генерацію псевдовипадкових затримок між ітераціями. Час

регулювання очікування перед кожним наступним зверненням до сервера описується виразом:

$$T_{wait} = T_{base} + random(0, T_{jitter}), \quad (2.7)$$

де T_{wait} – час затримки перед запитом;

T_{base} – базовий інтервал очікування для запобігання блокування;

T_{jitter} – випадкова величина для нівелювання автоматичного процесу.

Після успішного отримання сирих даних у форматі HTML-коду, алгоритм переходить до фази аналізу та нормалізації. Цей етап є критично важливим для забезпечення точності подальших розрахунків. Система автоматично розпізнає ключові атрибути товару, такі як поточна ціна, вага, габарити пакування та категорія. Важливою частиною алгоритму є етап на якому система порівнює отримані значення з історичними медіанами для даної категорії, щоб виключити аномалії, спричинені помилками лістингу або технічними збоями на сторінці. Якщо критично важливий параметр відсутній або має некоректний тип даних, алгоритм ініціює резервний сценарій повторного запиту.

Логічна структура системи замикається на аналітичному ядрі, яке функціонує як багатокритеріальний фільтр прийняття рішень. Дані, що пройшли етап нормалізації, подаються на вхід блоку фінансового аналізу, де вони інтегруються з математичними моделями, розробленими у попередніх підрозділах. Алгоритм послідовно перевіряє товар на відповідність критеріям «виживання» капіталу, а саме:

- розраховується точка беззбитковості;
- оцінюється запас міцності маржі;
- визначається цільова ціна закупівлі.

Кожен етап цієї перевірки є логічним розгалуженням, якщо на будь-якому кроці ризик перевищує встановлений поріг, то алгоритм припиняє

обчислення та формує негативний вердикт, що дозволяє економити ресурс для обчислення та час інвестора.

Для візуалізації описаних системних зв’язків та логіки проходження даних розроблено концептуальну блок-схему алгоритму (рис. 2.3). Схема відображає ієрархію процесів від моменту введення користувачем цільового URL-посилання до моменту запису сформованого аналітичного звіту в базу даних. Особлива увага в алгоритмі приділяється асинхронності виконання операцій. Завдяки використанню асинхронних примітивів програмування, основні часові витрати припадають на мережеве очікування, яке відбувається паралельно для кількох об’єктів аналізу. Це дозволяє системі масштабуватися та обробляти великі масиви даних без експоненціального зростання часу очікування результату.

Завершальним етапом є формування об’єкта звіту, який містить не лише цифри, а й логічні обґрунтування для кожної рекомендації. Алгоритм створює з всіх результатів перевірок чітку структуру. Таким чином, побудована алгоритмічна модель забезпечує цілісність системи. Система поєднує технічні аспекти збору даних із бізнес-логікою інвестиційного аналізу.

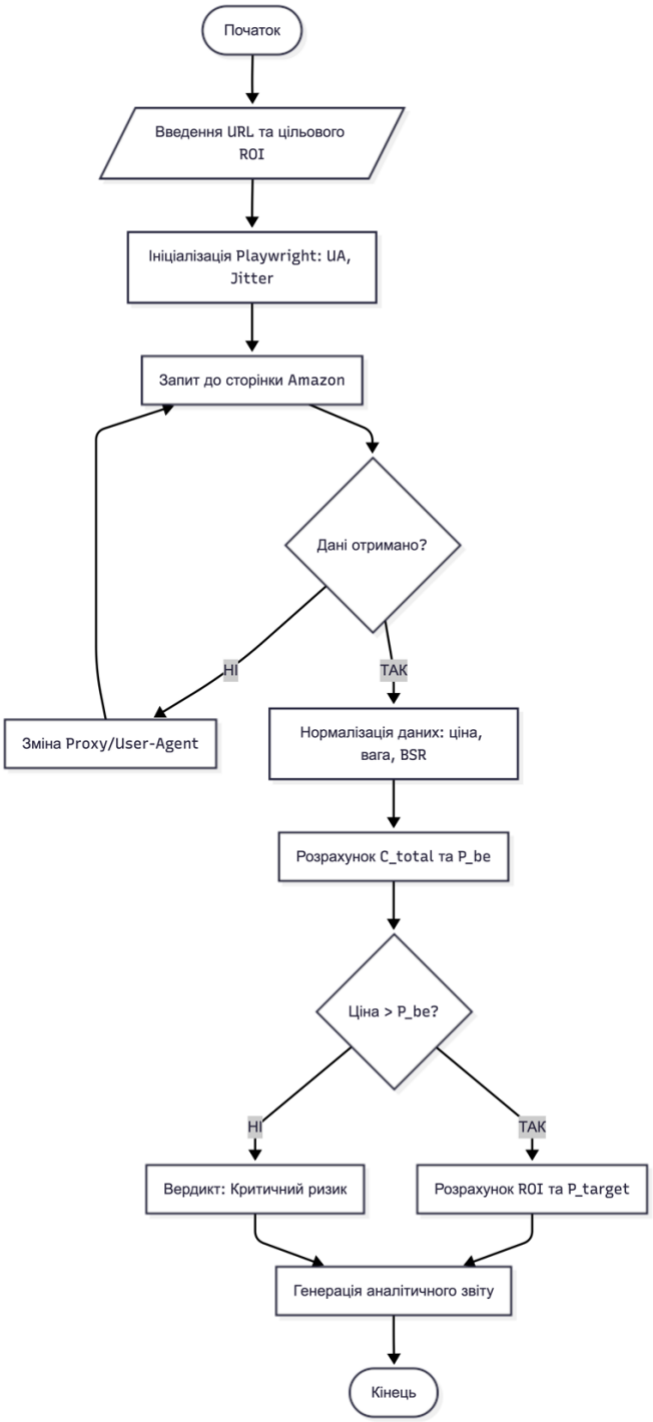

Рисунок 2.3 – Блок-схема алгоритму функціонування ІАС.

2.4 Розроблення методу багатокритеріальної крос-верифікації

Оскільки прямі дані про обсяги продажів на маркетплейсах часто є закритими або піддаються маніпуляціям з боку продавців є необхідність у

розробленні методу багатокритеріальної крос-верифікації попиту. Основна логіка полягає у формуванні інтегрального індексу надійності попиту. Індекс базується на кореляції між динамікою Best Sellers Rank, швидкістю появи нових відгуків та зміною залишків товару на складі. Математична формалізація методу передбачає розрахунок індексу надійності попиту за наступною багатофакторною залежністю:

$$DRI=\omega_{bsr}\times\varphi(B)+\omega_{rev}\times\Delta R+\omega_{v}\times Q_{ext}, \quad (2.8)$$

де DRI – інтегральний індекс надійності попиту;

$\varphi(B)$ – функція відповідності рангу BSR реальному обсягу продажів у конкретній категорії;

ΔR – швидкість приросту органічних відгуків за звітний період;

Q_{ext} – верифікований показник зовнішніх джерел або плашок маркетплейса;

ω_{bsr}, ω_{rev}, ω_{v} – вагові коефіцієнти значущості кожного параметра, визначені шляхом експертного оцінювання.

Першим етапом реалізації методу є аналіз стійкості BSR. Алгоритм IAC проводить моніторинг його волатильності. Якщо високий ранг (B) не підкріплюється відповідним приростом відгуків (ΔR), система ідентифікує таку активність як потенційний «фрод». Крос-верифікація в даному контексті дозволяє відсіяти товарні позиції, які знаходяться на етапі штучного просування. Це оберігає інвестора від входу в нішу з непідтвердженою ліквідністю. Другим аспектом методу є оцінювання органічного інтересу через аналіз Review Velocity. Алгоритм порівнює заявлену кількість продажів із кількістю нових відгуків, враховуючи середній коефіцієнт конверсії в ніші. У випадку виявлення значного дисонансу, коли кількість продажів за даними маркетплейса зростає експоненціально, метод автоматично знижує показник DRI до критичного рівня. Це дозволяє системі верифікувати «якість» попиту та відокремити реальні купівельні наміри від технічних маніпуляцій з

алгоритмами ранжування. Для систематизації результатів верифікації та формування фінального висновку в системі використовується таблиця відповідності крос-параметрів (табл. 2.3), яка дозволяє класифікувати попит за ступенем його достовірності.

Таблиця 2.3 – Критерії верифікації достовірності ринкового попиту.

Параметри	**Стан верифікації**	**Висновок**	**Рівень довіри IAC**
Високий BSR + стабільний ΔR	Верифіковано	Органічний трафік	Високий
Високий BSR + низький ΔR	Аномалія	Імовірно штучна накрутка	Низький
Низький BSR + високий ΔR	Невідповідність	Можливий дефіцит товару	Середній
Низький BSR + низький ΔR	Підтверджено	Відсутність інтересу в ніші	Високий (негативний)

Застосування розробленого методу багатокритеріальної крос-верифікації дозволяє перетворити IAC з простого збирача даних на повноцінний інструмент підтримки прийняття рішень з функцією фільтрації. Універсальність методу забезпечується можливістю адаптації вагових коефіцієнтів під різні категорії товарів. Таким чином, розроблений підхід забезпечує високу точність прогнозування оборотності капіталу.

2.5 Моделювання інтегрального показника інвестиційної привабливості

Для формування остаточного вердикту IAC необхідно звести всі розраховані параметри до єдиного показника. Це реалізується через побудову моделі інтегрального індексу привабливості. Розрахунок інтегрального індексу привабливості об’єкта здійснюється у вигляді:

$$I_{total}=\omega_1\times\overline{ROI}+\omega_2\times M_{safety}+\omega_3\times DRI, \quad (2.8)$$

де I_{total} – інтегральний показник привабливості об’єкта (від 0 до 1);

$\overline{ROI}$ – нормалізований показник рентабельності інвестицій;

M_{safety} – показник запасу міцності маржі;

DRI – верифікований індекс надійності попиту;

ω_1, ω_2, ω_3 –вагові коефіцієнти, що визначають пріоритетність фінансових та ринкових чинників для інвестора.

Процес моделювання передбачає використання матриці рішень. В ній кожна гілка алгоритму оцінює ймовірність успіху за різними сценаріями розвитку ринку. Використання асинхронних обчислень дозволяє проводити багатоваріантне моделювання.

Дана модель дозволяє системі не просто порівнювати цифри, а оцінювати комбіновний ефект факторів, наприклад, товар із середнім ROI, але дуже високим запасом міцності та надійним попитом, може бути пріоритетнішим за товар із високим ROI, але критичними ризиками блокування. Це забезпечує високий рівень інтелектуальної підтримки прийняття рішень. Допомогає мінімізувати вплив емоційних чинників на інвестора.

2.6 Моделювання логічної структури бази даних

Моделювання логічної структури бази даних є завершальним етапом проєктування інформаційно-аналітичної системи, що забезпечує цілісність, довготривале зберігання та швидкий доступ до верифікованих інвестиційних даних. Враховуючи специфіку вхідної інформації, яка отримується шляхом вебскрапінгу та має високий ступінь неструктурованості, для реалізації ІАС було обрано модель даних на базі СУБД MongoDB [23]. Такий підхід дозволяє

уникнути жорстких схем, властивих реляційним базам, та забезпечує можливість легкого розширення структури при додаванні нових аналітичних метрик або зміні форматів даних маркетплейса. Логічна структура бази даних проєктується таким чином, щоб мінімізувати надмірність інформації, зберігаючи при цьому можливість швидкої генерації історичних звітів без повторного звернення до зовнішніх серверів маркетплейса. Центральним елементом логічної структури є колекція «Investment_Reports», яка акумулює результати роботи аналітичного та алгоритмічного модулів. Кожен документ у цій колекції являє собою цілісний інвестиційний зріз для конкретної товарної позиції. Для забезпечення високої швидкодії системи на базі архітектури Apple Silicon, структура документа оптимізована для операцій читання та містить вкладені об’єкти, що описують різні аспекти життєвого циклу товару. Основні поля логічної моделі документа можна представити наступним переліком:

– ID_report – унікальний ідентифікатор звіту в системі;

– Product_Metadata – блок технічних даних;

– Market_Snapshot – об’єкт, що містить поточну ринкову ціну, показник BSR та обсяг продажів;

– Financial_Metrics – розраховані значення сумарних витрат, точки беззбитковості та цільової ціни закупівлі;

– Risk_Assessment – результати крос-верифікації попиту та визначений рівень фінансового ризику;

– Timestamp – мітка часу створення запису для відстеження динаміки змін.

Особлива увага при моделюванні приділяється зв’язкам між колекціями. Незважаючи на NoSQL-природу обраної СУБД, логічна архітектура передбачає взаємодію між основними сутностями: «Users», «Projects» та «Items». Колекція користувачів зберігає індивідуальні налаштування інвестора, такі як бажаний рівень рентабельності та валюта розрахунків, що є критично важливим для коректного функціонування аналітичного ядра в

різних регіональних юрисдикціях. Це дозволяє системі персоналізувати фінансові звіти, автоматично підставляючи податкові ставки та логістичні тарифи, специфічні для конкретного профілю користувача.

Для забезпечення ефективного пошуку та аналізу великих масивів даних, логічна модель передбачає створення індексів за ключовими полями, такими як артикул товару та категорія. Це дозволяє реалізувати функцію порівняльного аналізу, коли система може миттєво зіставити поточні показники товару з середніми значеннями в ніші, що зберігаються в базі. Такий підхід до моделювання структури бази даних перетворює її з простого сховища на активний компонент підтримки прийняття рішень, який дозволяє інвестору не лише бачити поточний стан окремої позиції, а й відстежувати глобальні тренди зміни цін та прибутковості в обраному сегменті ринку. Таким чином, розроблена логічна структура бази даних повністю відповідає вимогам універсальності, масштабованості та надійності, закладеним на етапі формалізації задачі.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ІНФОРМАЦІЙНО-АНАЛІТИЧНОЇ СИСТЕМИ

3.1 Обґрунтування вибору середовища програмної реалізації

Ефективність функціонування розроблюваної інформаційно-аналітичної системи підтримки прийняття рішень безпосередньо залежить від обґрунтованого вибору технологічного стеку проєктування. Основним критерієм при аналізі інструментів була їх спроможність забезпечувати високу швидкість обробки неблокуючих операцій введення-виведення, оскільки паралельний аналіз великої кількості товарних лістингів та взаємодія з віддаленими серверами створюють значне навантаження на мережу. Оскільки розгортання комп’ютерної моделі здійснюється на базі сучасної апаратної архітектури Apple Silicon під керуванням операційної системи macOS, мовою розроблення було обрано Python версії 3.13. Середовище функціонування системи повністю ізольовано у віртуальному просторі Miniconda, що гарантує абсолютну стабільність версій підключених модулів та упереджує виникнення системних конфліктів інтерпретатора. Вибір мови Python обумовлений його високою архітектурною гнучкістю, нативною інтеграцією з асинхронним ядром та максимальною продуктивністю при обчислювальній обробці складних математичних масивів даних на ARM-процесорах [20].

Для розроблення та реалізації графічного інтерфейсу користувача в роботі використано сучасну високорівневу бібліотеку CustomTkinter, яка підтримує апаратне прискорення відрисування віджетів на рівні операційної системи. Візуалізація результатів фінансового моделювання інтегрована безпосередньо в інтерфейсну оболонку за допомогою пакету matplotlib та конвеєра Pillow. На відміну від класичного Tkinter, обраний інструментарій надає об’єкти з вбудованим згладжуванням кутів та можливістю масштабування. Пакет Pillow забезпечує безперебійне декодування та

підготовку завантажених бінарних масивів медіаконтенту товарів. Matplotlib інтегрується у вікна застосунку, що дозволяє рендерити динамічні векторні графіки без втрати швидкодії.

Підсистема автоматизованого збору ринкових даних побудована на базі асинхронного фреймворку playwright.async_api, що дозволяє запускати браузер Chromium у фоновому режимі без критичного перевантаження оперативної пам’яті комп’ютера. Парсинг завантажених вебсторінок реалізовано за допомогою бібліотеки BeautifulSoup. Швидкісне паралельне завантаження медіафайлів через клієнт aiohttp з обходом SSL-обмежень сокетів. Вибір Playwright замість Selenium обумовлений його здатністю взаємодіяти з низькорівневим протоколом Chrome DevTools Protocol, що забезпечує миттєву ініціалізацію ізольованих контекстів сторінок. Обробка сирого HTML-коду здійснюється синтаксичним аналізатором BeautifulSoup.

Для збереження історії розрахунків було обрано NoSQL СУБД MongoDB із використанням офіційного асинхронного драйвера Motor. Експорт аналітичних звітів у зовнішні файли для інвестора реалізовано через структури DataFrame бібліотеки pandas. Вибір NoSQL-архітектури обґрунтований високою мінливістю відкритих даних маркетплейсу, оскільки лістинги різних товарних категорій мають унікальні специфікації, які складно нормалізувати в класичних реляційних таблицях. Драйвер Motor дозволяє інтегрувати транзакції створення, зчитування та видалення BSON-документів безпосередньо в загальний цикл подій програми, завдяки чому робота з базою даних виконується паралельно з мережевим скрапінгом, не викликаючи затримок графічної оболонки [24]. Бібліотека pandas використовується як інструмент фінальної серіалізації, що трансформує масиви документів у таблиці з подальшим збереженням у форматі CSV із підтримкою універсального кодування UTF-8-SIG для коректного відображення кирилиці.

3.2 Технічне завдання

Головною метою розроблення інформаційно-аналітичної системи є повна автоматизація всього циклу інвестиційного аналізу. На основі математичних моделей та аналізу бізнес-процесів електронної комерції, розроблювана комп’ютерна програма повинна успішно вирішувати такий комплекс технічних завдань:

– забезпечити стійкий асинхронний скрапінг, який в автоматичному режимі точно визначає регіональну валюту розрахунку, коректно склеює дробові частини ціни, та вилучає фізичні габарити товару виключно з первинних таблиць технічних специфікацій без захоплення аналогічних даних конкурентних;

– розробити обчислювальне фінансове ядро з динамічною системою адаптивних тарифів;

– інтегрувати алгоритм глибокої юніт-економіки, здатний розраховувати чисту собівартість на основі вартості товару на виробничій фабриці, тарифів логістичної компанії та офіційних митних зборів;

– створити функціонал «Проєкції капіталу», який автоматично резервує жорстку квоту у розмірі 15% від введеного інвестором загального бюджету на первинний рекламний запуск партії, розраховує максимально можливу кількість одиниць продукції до закупівлі на залишок фінансових ресурсів та прогнозує фінальний чистий грошовий потік з диференціацією витрат на живі гроші для старту;

– реалізувати надійну взаємодію між графічним інтерфейсом користувача та мережевими завданнями збору інформації для повного запобігання блокуванню операційної системи, забезпечуючи плавність роботи інтерфейсу додатка за будь-яких умов зв’язку.

3.3 Програмна реалізація

В основі програмної архітектури розробленої інформаційно-аналітичної системи лежить принцип об’єктно-орієнтованої модульної декомпозиції. Кодова база проєкту логічно розділена на три незалежні підсистеми, кожна з яких інкапсульована в окремому програмному модулі на диску:

– main.py – графічний інтерфейс користувача та координація міжпотокової взаємодії;

– scraper.py – підсистема асинхронного збору даних та первинного HTML-парсингу;

– engine.py – математичне ядро, фінансове моделювання та взаємодія з базою даних MongoDB.

Для розв’язання поширеної проблеми зависання графічного інтерфейсу застосунку під час виконання тривалого мережевого очікування, у файлі main.py було спроєктовано та реалізовано спеціалізований клас-обгортку AsyncLoopThread. Цей клас створює ізольований фоновий потік операційної системи із власним нескінченним циклом, куди безпечно передаються всі важкі асинхронні завдання.

Лістинг 3.1 Вирішення проблеми зависання графічного інтерфейсу:

```
class AsyncLoopThread:
    def __init__(self):
        self.loop = asyncio.new_event_loop()
        self._thread = threading.Thread(target=self._run_loop, daemon=True)
        self._thread.start()

    def _run_loop(self) -> None:
        asyncio.set_event_loop(self.loop)
        self.loop.run_forever()
```

```
    def submit(self, coro):
        return asyncio.run_coroutine_threadsafe(coro, self.loop)

    def stop(self) -> None:
        self.loop.call_soon_threadsafe(self.loop.stop)
```

Важливим аспектом архітектурної реалізації головного модуля main.py є розробка компонування інтерфейсу користувача за допомогою фреймворку CustomTkinter. Для забезпечення адаптивності та правильного масштабування віджетів на дисплеях з різною роздільною здатністю було використано сітковий менеджер геометрії з явним проектуванням вагових коефіцієнтів рядків та колонок. Головне вікно програми структуровано у вигляді оболонки, де за перемикання між ізольованими аналітичними екранами відповідає логічний метод select_frame. Дане рішення дозволяє динамічно керувати простором пам’яті, приховуючи неактивні компоненти. Нижче наведено фрагмент програмного коду, який відповідає за ініціалізацію головного вікна застосунку, побудову навігаційного сайдбару та логіку безпечного перемикання робочих фреймів.

Лістинг 3.2 Компонування інтерфейсу користувача:

```
def __init__(self):
        super().__init__()
        self.title("Margix — Аналіз інвестицій Amazon")
        self.geometry("1400x900")
        self.minsize(1300, 800)

        self.async_loop = AsyncLoopThread()
        self.engine = InvestmentEngine()

        self.current_scraped_data: Optional[dict] = None
```

```
        self.current_metrics: Optional[dict] = None
        self.history_reports: list = []

        self.grid_columnconfigure(1, weight=1)
        self.grid_rowconfigure(0, weight=1)

        self.frames = {}
        self._build_sidebar()
        self._build_all_frames()

        self.select_frame("scraper")
        self._load_history_async()
        self.protocol("WM_DELETE_WINDOW", self._on_close
def select_frame(self, name):
        for key, btn in self.nav_buttons.items():
            btn.configure(fg_color="#3498DB" if key == name else "transparent",
text_color="white" if key == name else "black")

        for key, frame in self.frames.items():
            if key == name:
                frame.grid(row=0, column=1, sticky="nsew", padx=15, pady=15)
            else:
                frame.grid_forget()

        if name == "plan" and self.current_metrics:
            self._update_plan_ui(self.current_metrics)
```

Описана програмна реалізація ініціалізації забезпечує повне зв’язування графічних подій з обчислювальними потоками. Метод select_frame не просто змінює візуальні стани віджетів, а й виконує умовну перевірку, якщо

користувач переходить на вкладку «План Дій», система автоматично ініціює виклик _update_plan_ui для рендерингу оновлених фінансових метрик, що унеможливлює відображення застарілих даних попередніх сесій.

Модуль вебскрапінгу scraper.py імітує поведінку реального користувача на сторінці маркетплейсу, виконуючи мікроскролінг, випадкову ротацію заголовків User-Agent та впроваджуючи адаптивні часові затримки для захисту від передчасного розриву з'єднання [21]. Критично важливим елементом підсистеми є унікальний алгоритм зчитування ціни:

– програма здійснює первинний пошук цілісного текстового значення вартості у прихованому вузлі з класом a-offscreen, який містить повний рядок разом із валютним знаком;

– у разі відсутності тегу, система активує резервний математичний алгоритм, витягуючи цифри окремо з класів a-price-whole та a-price-fraction, очищаючи їх від нечислових символів за допомогою регулярних виразів та склейки в єдиний тип float;

– вилучення фізичних параметрів ваги та об'єму виконується лише після примусового видалення рекламних блоків, спонсорованих слайдерів та таблиць порівняння;

– такі міри як фунти, унції та дюйми автоматично конвертуються у стандартну метричну систему через математичні коефіцієнти переведення.

Також модуль вебскрапінгу відповідає за обхід захисних механізмів маркетплейсу та нормалізацію неструктурованного HTML-коду сторінки. Оскільки маркетплейс Amazon застосовує динамічне формування DOM-дерева, де ціла частина ціни та копійки розділені по різних тегах, стандартні методи парсингу призводять до втрати дробових значень. Для розв'язання цієї проблеми розроблено алгоритм точної екстракції та склейки фракцій ціни. Алгоритм спочатку шукає прихований тег a-offscreen, а у разі його відсутності програмно об'єднує значення з класів a-price-whole та a-price-fraction.

Лістинг 3.3 Обхід захисних механізмів Amazon:

```
def _extract_price(soup: BeautifulSoup) -> float:
    price_blocks = soup.find_all("span", attrs={"class": re.compile(r"\ba-price\b")})
    for block in price_blocks:
        offscreen = block.find("span", attrs={"class": "a-offscreen"})
        if offscreen:
            val = _parse_float(offscreen.get_text(strip=True))
            if val is not None:
                return val

    whole_span = soup.find("span", attrs={"class": "a-price-whole"})
    if whole_span:
        parent = whole_span.find_parent("span", attrs={"class": re.compile(r"\ba-price\b")})
        search_root = parent if parent else whole_span.parent
        whole_text = whole_span.get_text(strip=True)
        whole_digits = re.sub(r"[^\d]", "", whole_text)

        fraction_text = ""
        if search_root:
            fraction_span = search_root.find("span", attrs={"class": "a-price-fraction"})
            if fraction_span:
                fraction_text = re.sub(r"[^\d]", "", fraction_span.get_text(strip=True))

        if whole_digits:
            merged = f"{whole_digits}.{fraction_text}" if fraction_text else whole_digits
            val = _parse_float(merged)
```

```
        if val is not None:
            return val
```

Поряд з вилученням цінових параметрів, критично важливим завданням підсистеми scraper.py є збір фізичних характеристик об’єкта. Головною перешкодою є наявність на сторінці великої кількості стороннього HTML-шуму, рекламних блоків конкурентів, слайдерів та споживчих таблиць порівняння схожих товарів, з яких регулярні вирази можуть помилково витягнути чужі габарити. Для забезпечення чистоти вибірки в систему було впроваджено метод деструктуризації шуму _clean_soup_from_competitors, який примусово видаляє сторонні гілки DOM-дерева за допомогою функції decompose(), після чого виконується точковий пошук специфікацій у цільових контейнерах.

Лістинг 3.4 Збір фізичниї характеристик об’єкта:

```
def _clean_soup_from_competitors(soup: BeautifulSoup) -> None:
    """Видаляє з HTML дерева таблиці порівняння та рекламні блоки."""
    bad_ids = re.compile(r"HLCXComparisonTable|sims-
consolidated|sp_detail|askATFLink|hero-quick-promo|aplus")
    for bad_div in soup.find_all(["div", "table"], id=bad_ids):
        bad_div.decompose()

def _extract_weight(soup: BeautifulSoup) -> Optional[float]:
    _clean_soup_from_competitors(soup)
    keywords = ["item weight", "package weight", "product weight", "weight",
"gewicht"]
    def _find_in_elements(elements) -> Optional[float]:
        for el in elements:
            text = el.get_text(separator=" ", strip=True).lower()
            if any(k in text for k in keywords):
```

```
            match =
re.search(r"([\d]+(?:[.,]\d+)?)\s*(kg|kilograms?|kilogramm|g|grams?|gramm|lb|p
ounds?|oz|ounces?)", text)
            if match:
                raw_value = float(match.group(1).replace(",", "."))
                unit = match.group(2)
                if unit.startswith("g") and not unit.startswith("kg"): return
round(raw_value / 1000.0, 3)
                elif unit.startswith("lb") or unit.startswith("pound"): return
round(raw_value * 0.453592, 3)
                elif unit.startswith("oz") or unit.startswith("ounce"): return
round(raw_value * 0.0283495, 3)
                return round(raw_value, 3)
        return None

    for container in primary_containers:
        res = _find_in_elements(container.find_all(["tr", "li"]))
        if res is not None: return res

    return _find_in_elements(soup.find_all(["tr", "li", "td", "span"]))
```

Функція _extract_weight демонструє імплементацію вбудованої матриці конвертації імперських одиниць вимірювання в метричну систему. У разі виявлення текстових паттернів, що вказують на фунти або унції, обчислювальний алгоритм автоматично нормалізує значення, переводячи їх у кілограми, що забезпечує уніфікацію вхідних даних для математичного ядра Unit-економіки. Аналогічний підхід з регулярними виразами застосовано і в методі _extract_volume для обчислення об’єму в літрах на основі лінійних розмірів сторін упаковки.

Серцем розробленої системи є математичне ядро InvestmentEngine у файлі engine.py, яке відповідає за повну нормалізацію та розрахунок параметрів досліджуваного товару. Головною інновацією обчислювального ядра є інтегрований модуль «Проєкції капіталу». Алгоритм не просто вираховує сухі накладні витрати на одну штуку, а й моделює комплексний розподіл виділеного інвестором інвестиційного бюджету. Програма в автоматичному режимі відраховує фіксовану квоту на маркетинговий запуск і просування картки, обчислює максимально можливий обсяг першої закупівлі товарної маси в штуках та детально розписує фінальний рух грошових коштів з урахуванням логістики, митних зборів брокера та комісійних утримань маркетплейсу Amazon.

Лістинг 3.5 Моделювання комплексного розподілу виділеного інвестором інвестиційного бюджету:

```
    if capital and capital > 0:
        cap_marketing_budget = capital *
CAPITAL_MARKETING_ALLOCATION
        purchasing_power = capital - cap_marketing_budget

        unit_cost = landed if c_exw is not None else p_target
        if unit_cost > 0:
            cap_units = int(purchasing_power / unit_cost)
            cap_revenue = cap_units * pm
            cogs = cap_units * unit_cost
            if c_exw is not None:
                cap_goods_cost = cap_units * c_exw_val
                cap_shipping_cost = cap_units * c_shipping_val
                cap_customs_cost = cap_units * c_customs

            cap_amazon_fees = cap_units * (pm * ALPHA_REF)
```

```
cap_fba_fees = cap_units * c_fba
cap_storage_fees = cap_units * c_storage

amazon_fees_and_vars = c_fba + c_storage + c_risk + amazon_fees
total_vars = cap_units * amazon_fees_and_vars
        cap_gross = cap_revenue - cogs
cap_net = cap_revenue - cogs - total_vars - cap_marketing_budget
```

Фінальним етапом обробки результатів симуляції є їх візуалізація засобами вбудованого аналітичного бекенду. Для реалізації графічного аналізу чутливості прибутку та наочного відображення часток собівартості в інтерфейс системи було інтегровано функції взаємодії з бібліотекою matplotlib. Метод _render_charts динамічно очищає холст графічного вікна і будує дві ілюстрації: лінійні тренди поведінки чистого прибутку при коливаннях ринкових факторів у діапазоні 20% та кругову діаграму розподілу роздрібної ціни на логістичні й маркетингові складові.

Лістинг 3.6 Візуалізація графічного аналізу:

```
def _render_charts(self, m: dict) -> None:
    self.ax_line.clear()
    base_profit = (m['pm'] - m['c_total']) if m.get('c_landed') is not None else (m['pm'] * (m['desired_roi'] / 100))
    variations = [-20, -10, 0, 10, 20]

    profit_fba = [base_profit - (m['c_fba'] * (v/100)) for v in variations]
    profit_pm = [base_profit + (m['pm'] * (v/100)) for v in variations]

    self.ax_line.plot(variations, profit_pm, marker='o', color='#2ECC71', label='Зміна ціни продажу')
```

```
        self.ax_line.plot(variations, profit_fba, marker='s', color='#E74C3C',
label='Стрибок логістики (FBA)')

        self.ax_line.set_title("Чутливість чистого прибутку (на 1 од.)",
fontsize=11, fontweight='bold')
        self.ax_line.set_xlabel("Відхилення (%)", fontsize=9)
        self.ax_line.set_ylabel(f"Чистий прибуток ({m['currency']})", fontsize=9)
        self.ax_line.axvline(0, color='gray', linestyle='--', alpha=0.5)
        self.ax_line.legend(fontsize=8)
        self.ax_line.grid(True, linestyle=":", alpha=0.6)
        self.canvas_line.draw()

        self.ax_pie.clear()
        if m.get('c_exw') is not None:
            labels = ['Товар (EXW)', 'Доставка', 'Мито', 'FBA Логістика', 'Комісія
Amazon', 'Маркетинг (PPC)', 'Чистий Прибуток']
            sizes = [m['c_exw'], m.get('c_shipping', 0), m.get('c_customs', 0),
m['c_fba'] + m['c_storage'], m['pm'] * 0.15, m['c_ads'], max(base_profit, 0)]
            colors = ['#BDC3C7', '#34495E', '#95A5A6', '#F39C12', '#9B59B6',
'#E74C3C', '#2ECC71']
        else:
            labels = ['Закупівля (P_target)', 'FBA Логістика', 'Комісія Amazon',
'Маркетинг', 'Чистий Прибуток']
            sizes = [m['p_target'], m['c_fba'] + m['c_storage'], m['pm'] * 0.15,
m['c_ads'], max(base_profit, 0)]
            colors = ['#BDC3C7', '#F39C12', '#9B59B6', '#E74C3C', '#2ECC71']

        valid_data = [(l, s, c) for l, s, c in zip(labels, sizes, colors) if s > 0]
        if valid_data:
            v_labels, v_sizes, v_colors = zip(*valid_data)
```

```
self.ax_pie.pie(v_sizes, labels=v_labels, colors=v_colors,
autopct='%1.1f%%', startangle=140, textprops={'fontsize': 8})
self.ax_pie.set_title("Розподіл роздрібної ціни", fontsize=11,
fontweight='bold')
self.canvas_pie.draw()
```

Динамічний виклик self.canvas_line.draw() та self.canvas_pie.draw() змушує FigureCanvasTkAgg виконати миттєве перемальовування піксельної матриці віджету всередині головного вікна програми. Завдяки умовній конструкції фільтрації масивів valid_data, кругова діаграма захищена від логічних збоїв відображення, вона будується виключно на базі додатних значень витрат, що запобігає графічному спотворенню секторів.

3.4 Інструкція користувача

Для початку роботи з інформаційно-аналітичною системою користувач повинен запустити виконання головного файлу структури за допомогою команди python main.py. На екрані з’явиться візуальне вікно застосунку, розділене на ліве вертикальне навігаційне меню та праву інформаційну зону (рис. 3.1.). Покроковий процес повної експлуатації системи користувачем складається з декількох обов’язкових етапів.

Етап збору даних. У лівому бічному меню необхідно обрати першу вкладку «Скрейпер». У верхнє текстове поле введення користувач вставляє скопійоване URL-посилання на товарну позицію з маркетплейсу Amazon і натискає кнопку «Автоматичний збір даних» (рис. 3.2.). Програма запускає фоновий Playwright-контекст, збирає назву, поточну Buy Box ціну лістингу, ранг продажів BSR, кількість відгуків та обсяг продажів за 30 днів, паралельно вираховуючи реальні вагу та об’єм пакування і завантажуючи головне фото. Хід процесу відображається на нижньому статус-рядку.

Етап конфігурації фінансової моделі. Після успішного завершення збору даних користувачу необхідно перейти до другої вкладки меню «Аналіз та Калькулятор». У лівому меню відобразяться витягнуті вага та об’єм, які за необхідності можна відкоригувати вручну у разі помилки під час витягування даних (рис. 3.3.). За наявності додаткових даних користувач може заповнити комерційні параметри такі як ціну товару від постачальника, поточний тариф логістичної компанії за доставку, перемикнути режим нарахування ціни доставки, ввести відсоток митної декларації, бажану норму рентабельності інвестицій у полі «Бажаний ROI, %» та суму наявного інвестиційного капіталу. Програма може надати користувачу звіт лише на основі самого факту наявності товару та його базових показників з сторінки лістингу, але для більш детального аналізу та подальшого планування все ж необхідно вказати додаткові дані.

Етап симуляції Unit-економіки. Натискання великої кнопки «Симулювати Економіку» миттєво запускає обчислювальні алгоритми математичного ядра. У правій частині вікна калькулятора заповнюються картки ключових показників, а також оновлюється колір оцінки попиту (рис. 3.4.). Велике центральне поле «ВЕРДИКТ СИСТЕМИ» генерує розгорнутий текстовий аналіз ризиків та доцільності фінансування проєкту (рис. 3.5.).

Етап прийняття рішення та адміністрування. На основі вердикту користувач приймає остаточне інвестиційне рішення та натискає одну з трьох кольорових кнопок на панелі дій: «Розробити План», «В Пошук або «В Архів» (рис. 3.6.). Система автоматично маркує фінансовий документ відповідним статусом та записує його у сховище бази даних MongoDB.

Етап аналізу та планування капіталу. Користувач переходить до вкладки «Графіки попиту», де аналізує векторний лінійний графік чутливості та кругову діаграму розподілу ціни (рис. 3.7.). Після цього у вкладці «План Дій» стає доступним деталізований текстовий покроковий план, який чітко розписує розподіл стартового капіталу на закупівлю, доставку, мито та маркетинг (рис. 3.8.).

Управління базою товарів. У вкладці «База Товарів» користувач взаємодіє з історією за допомогою зручних вкладок фільтрації. Виділивши будь-який товар у списку, його можна повторно завантажити в калькулятор для зміни параметрів, повністю видалити з бази даних та інтерфейсу за допомогою кнопки «Видалити» або вивантажити всю історію у зовнішній табличний файл Excel за допомогою кнопки «Експортувати базу (CSV)» (рис. 3.9.).

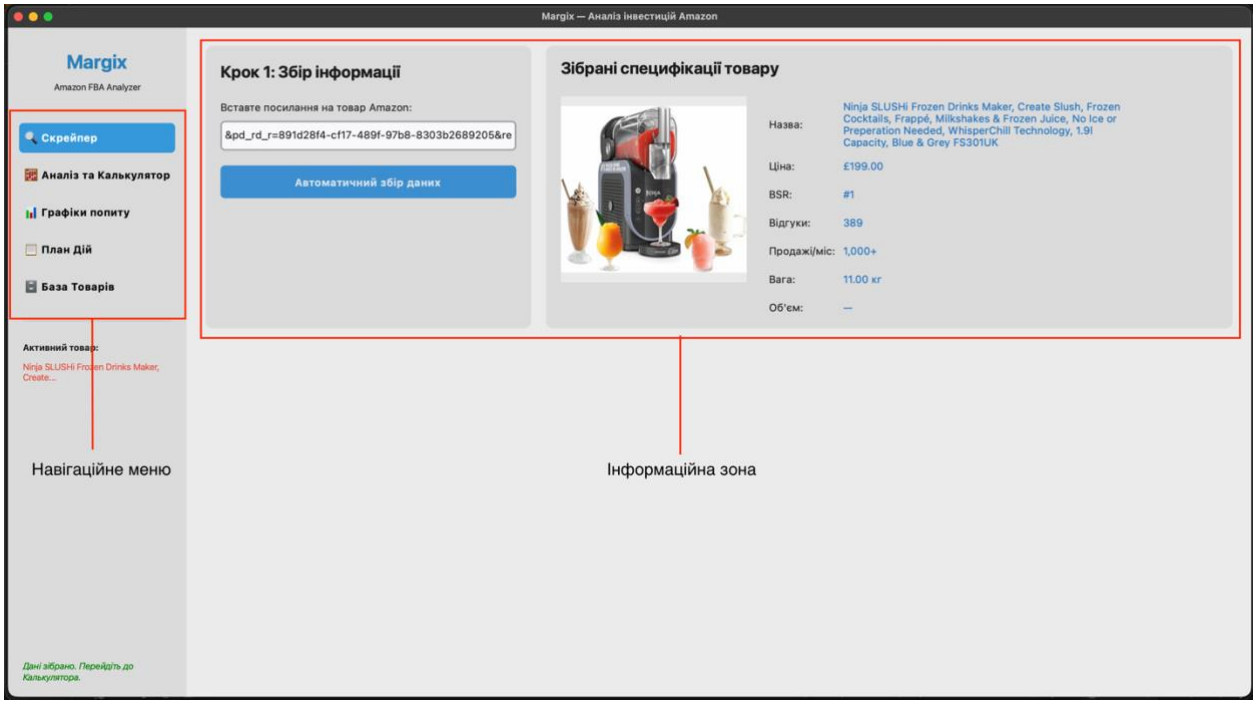

Рисунок 3.1 – Інтерфейс застосунку.

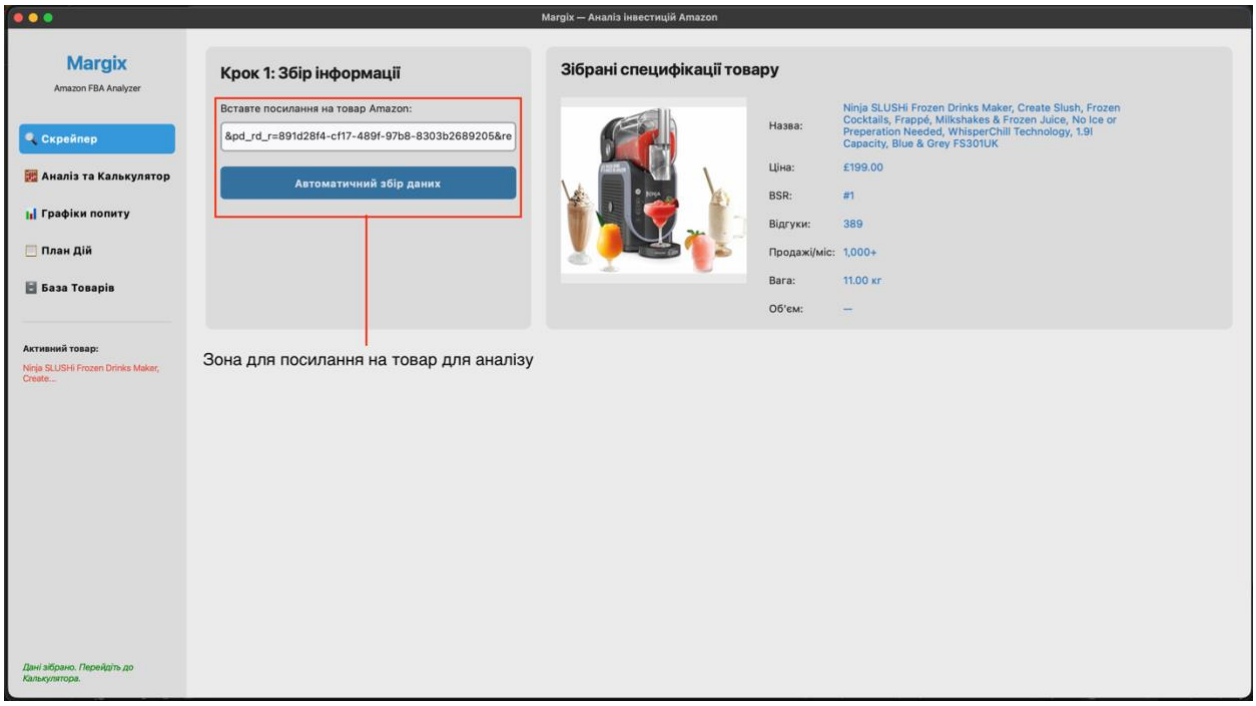

Рисунок 3.2 – Вкладка «Скрейпер» та поле для введення URL-посилання.

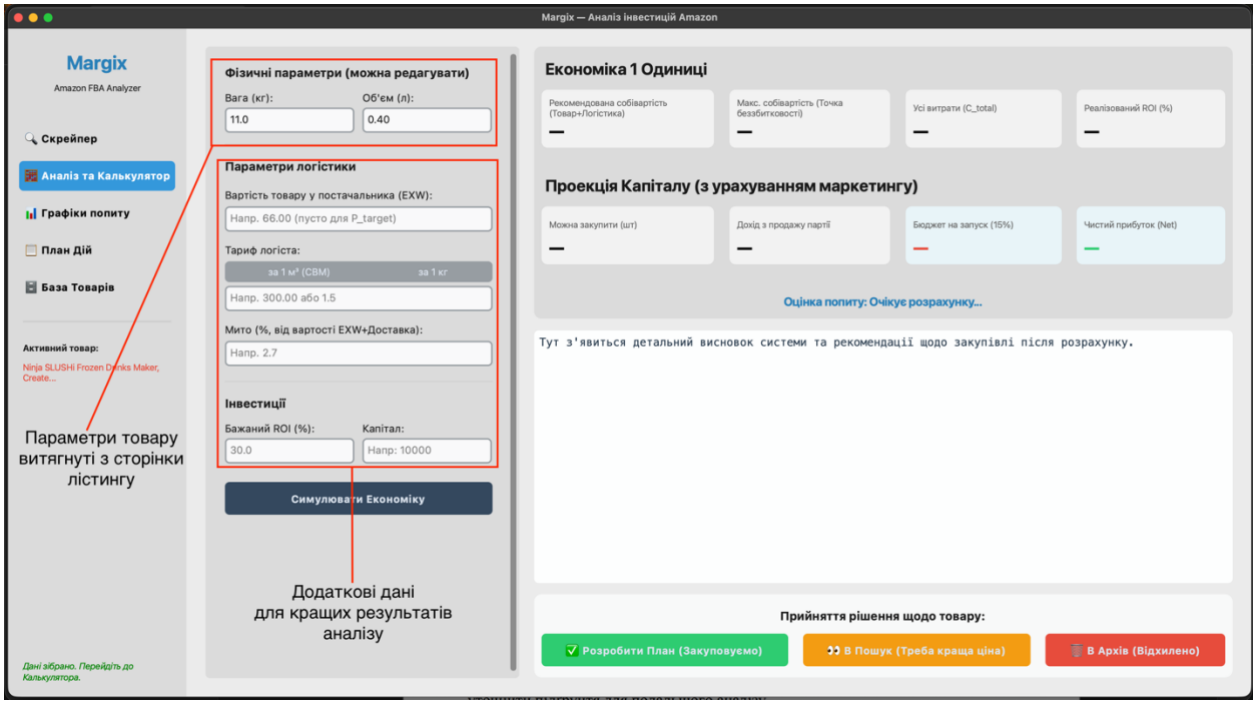

Рисунок 3.3 – меню «Аналіз та Калькулятор» та меню для додаткових комерційних параметрів.

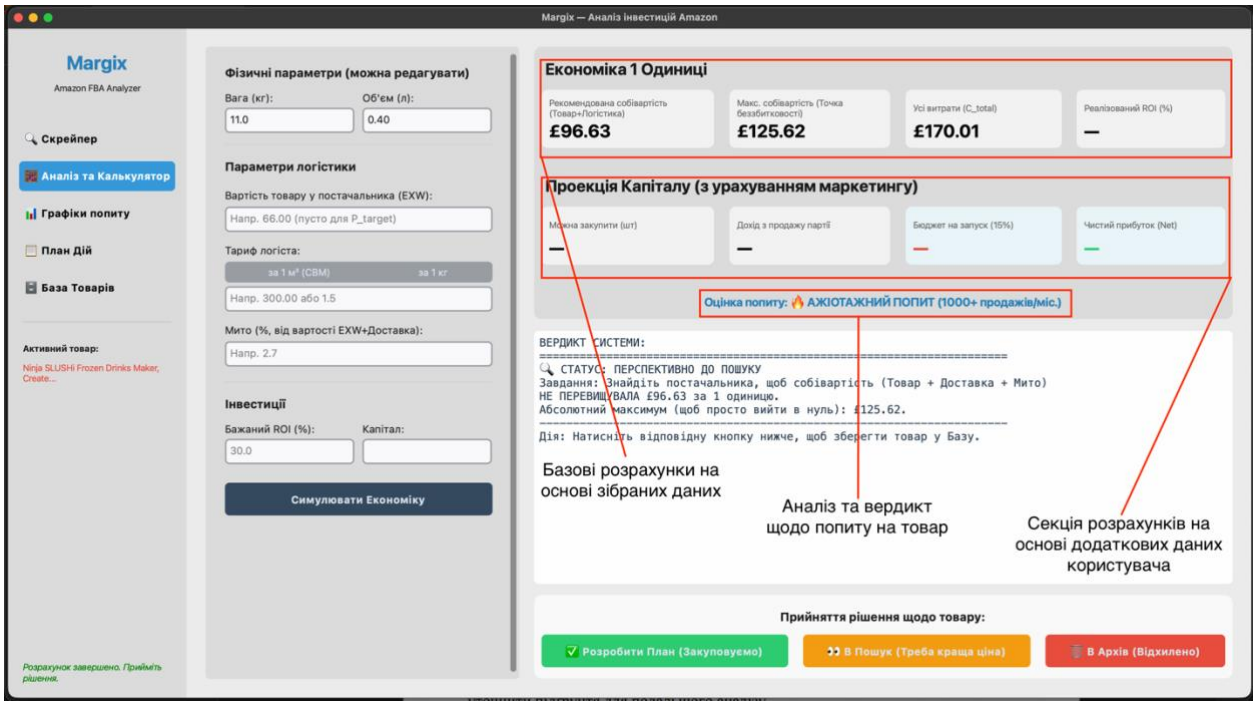

Рисунок 3.4 – Картки ключових показників.

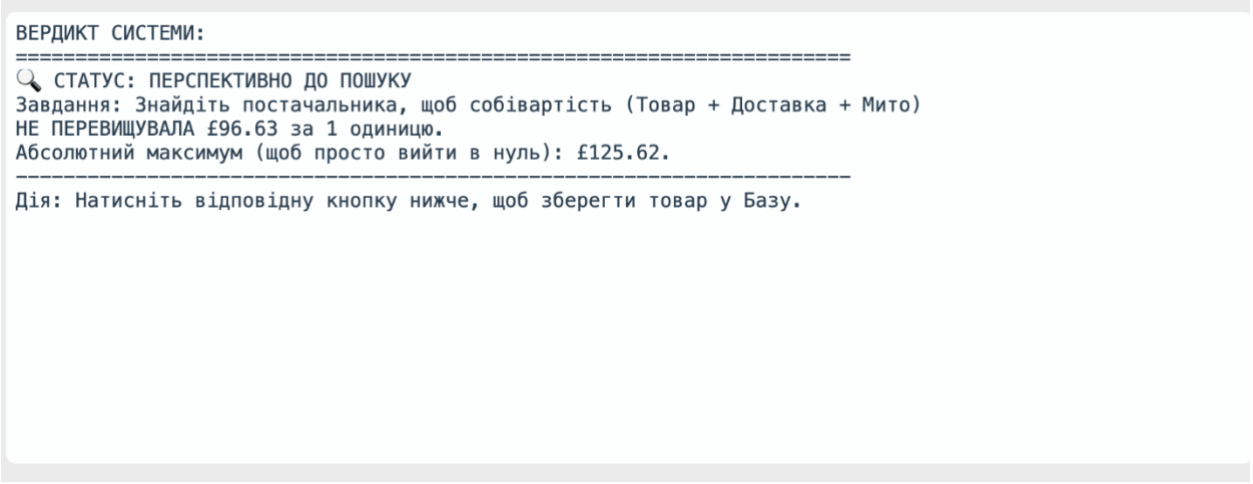

Рисунок 3.5 – Динамічний вердикт системи відповідно до показників аналізу.

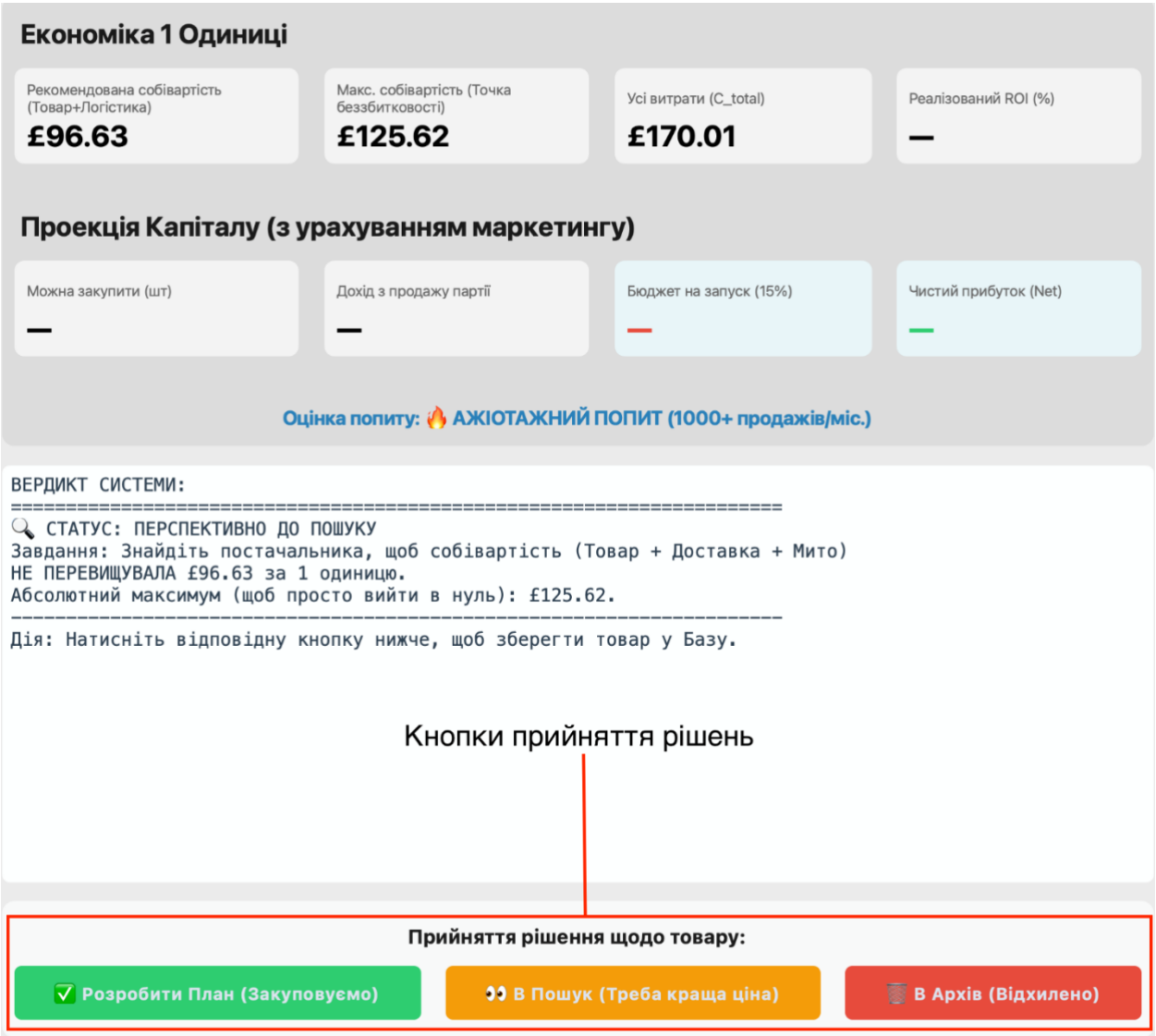

Рисунок 3.6 – Кнопки для прийняття рішення для обраного товару.

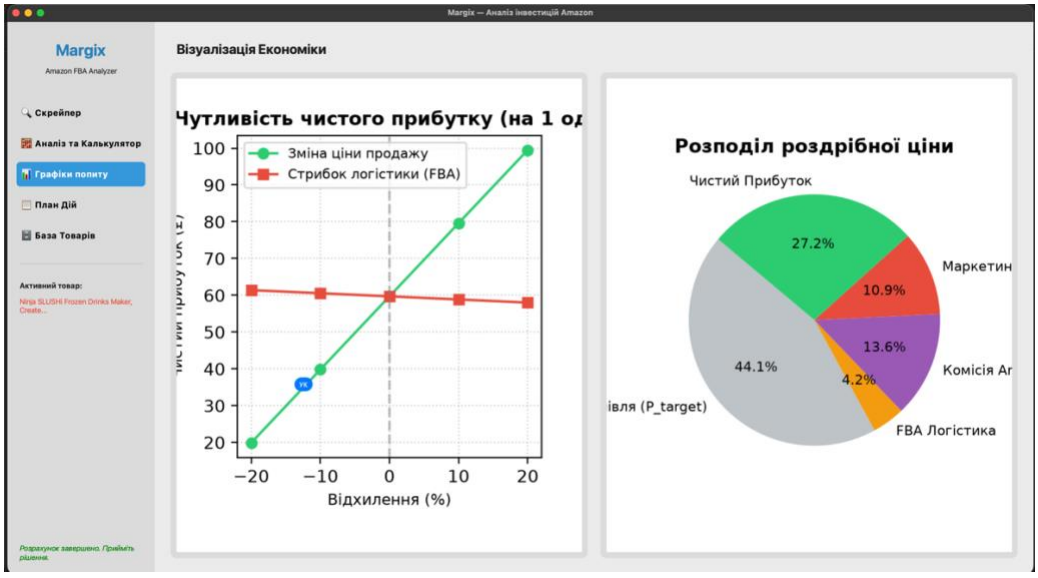

Рисунок 3.7 – Графік чутливості та кругова діаграма розподілу ціни.

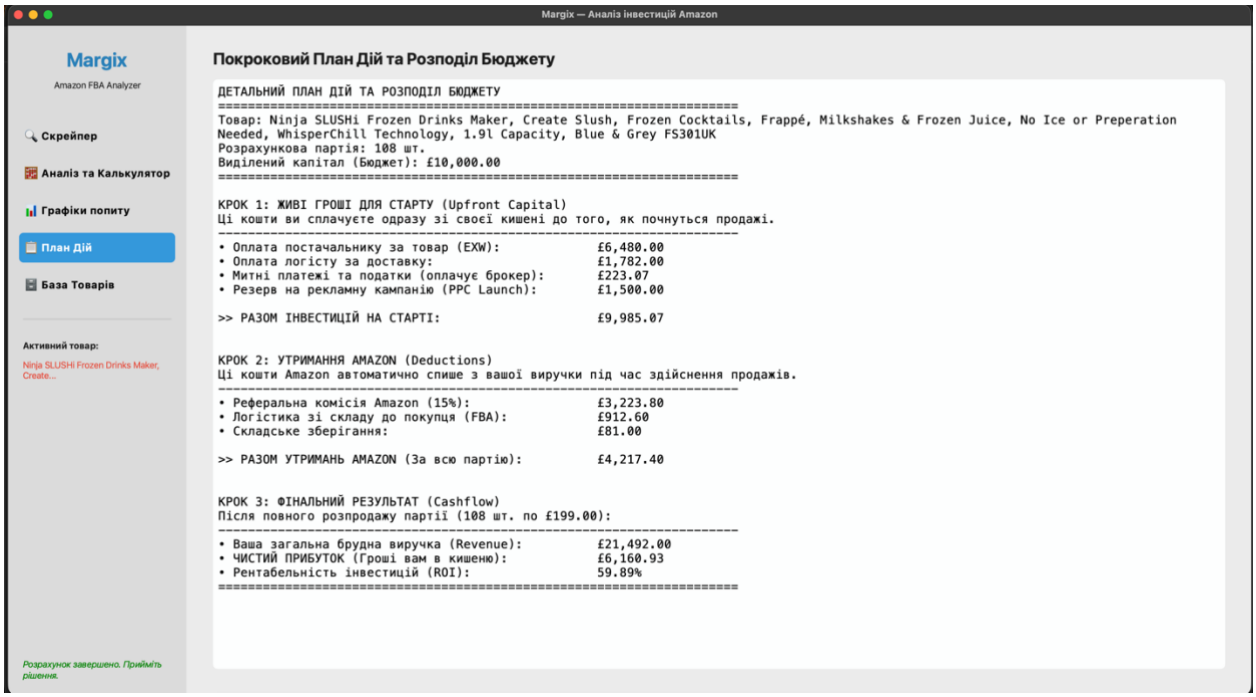

Рисунок 3.8 – План дій рекомендований системою.

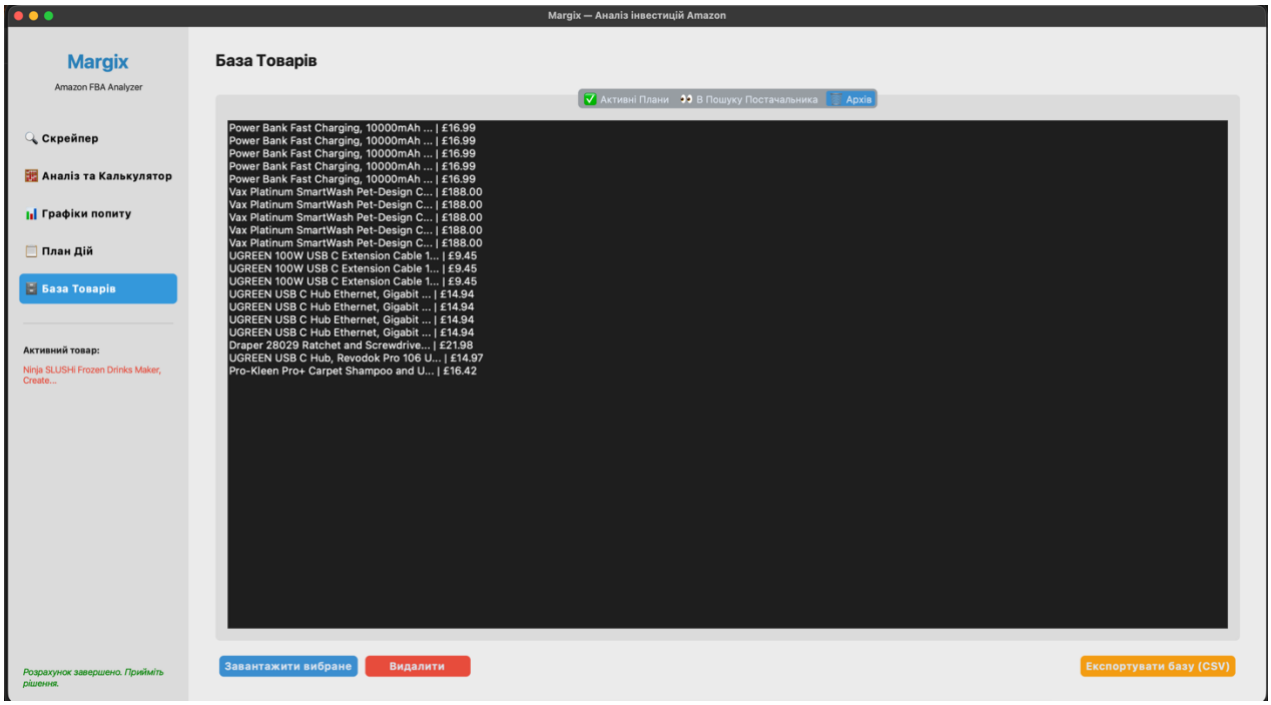

Рисунок 3.9 – База товарів та функції управління.

3.5 Тестування розробленої моделі

Експериментальна перевірка алгоритмів здійснювалася на реальних лістингах Amazon з метою тестування точності розпізнавання даних та коректності фінансової моделі.

У наведеному прикладі розглядається вже знайомий кейс з павербанками. Першим етапом є повний аналіз товару та збір даних. Оскільки програма виконує зворотній аналіз, спочатку необхідно обрати конкурента, який продає павербанки (рис. 3.10).

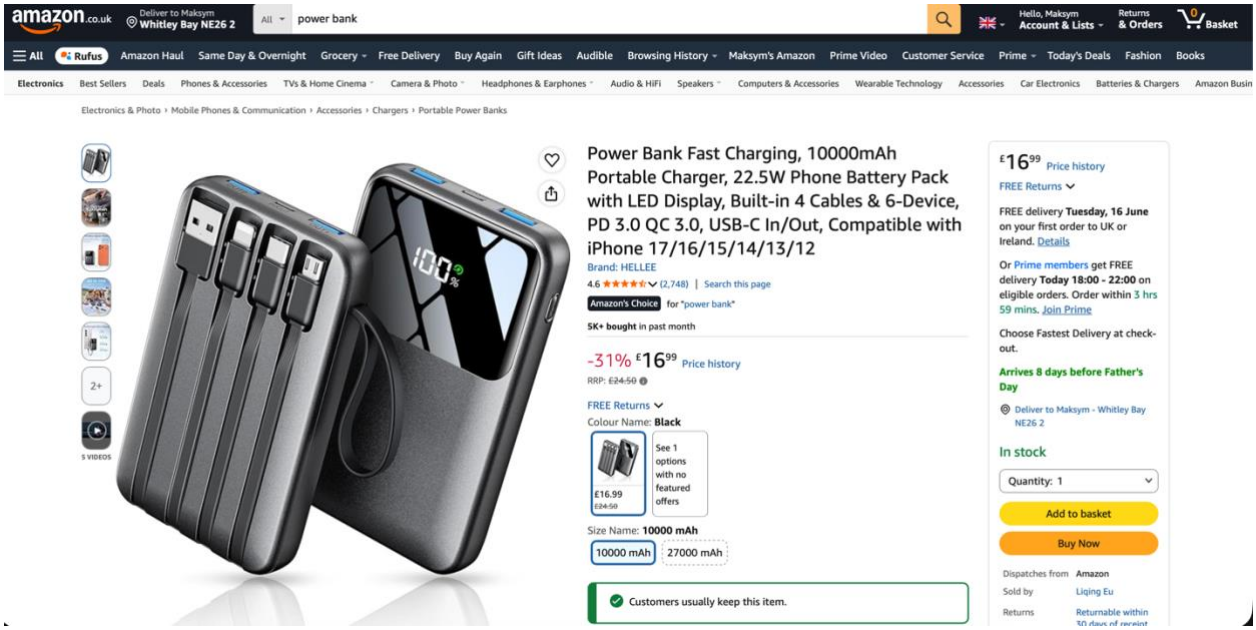

Рисунок 3.10 – Обраний конкурент для проведення аналізу.

Оскільки система робить все автоматично, єдине, що потрібно зробити, – це скопіювати посилання на конкретну позицію та вставити її у відповідне поле програми. Система витягує всі дані, що є на сторінці лістингу та відображає їх у боковому меню (рис. 3.11.). У результаті програма успішно витягнула ціну, визначила валюту та інші критично важливі показники.

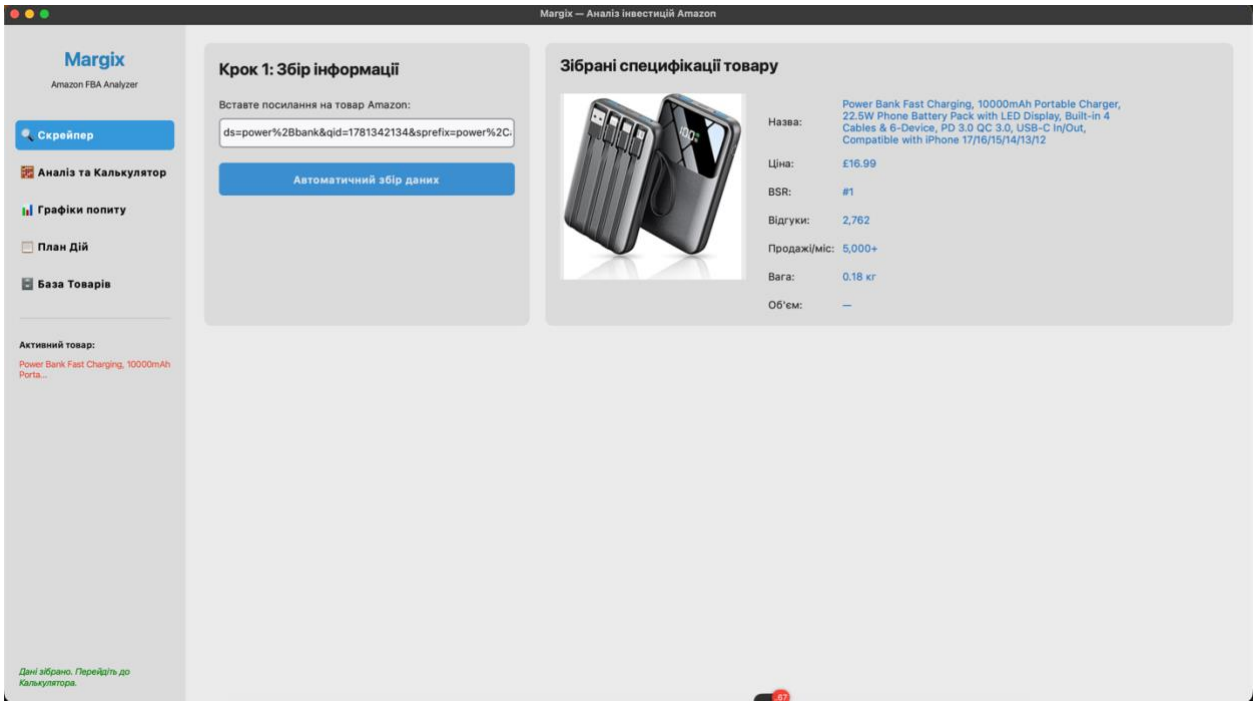

Рисунок 3.11 – Результат роботи вебскрапера.

Другим етапом є фінансовий аналіз ніші, програма дозволяє обрати декілька способів аналізу відповідно до даних які є у користувача. Щоб аналіз був максимально реалістичним, було знайдено постачальника (рис. 3.12.), а також ціни логістів на доставку до Англії. В результаті при вкладенні £5000 система розраховує на чистий прибуток у розмірі £3490, всі метрики та результати аналізу показано на рисунку 3.13.

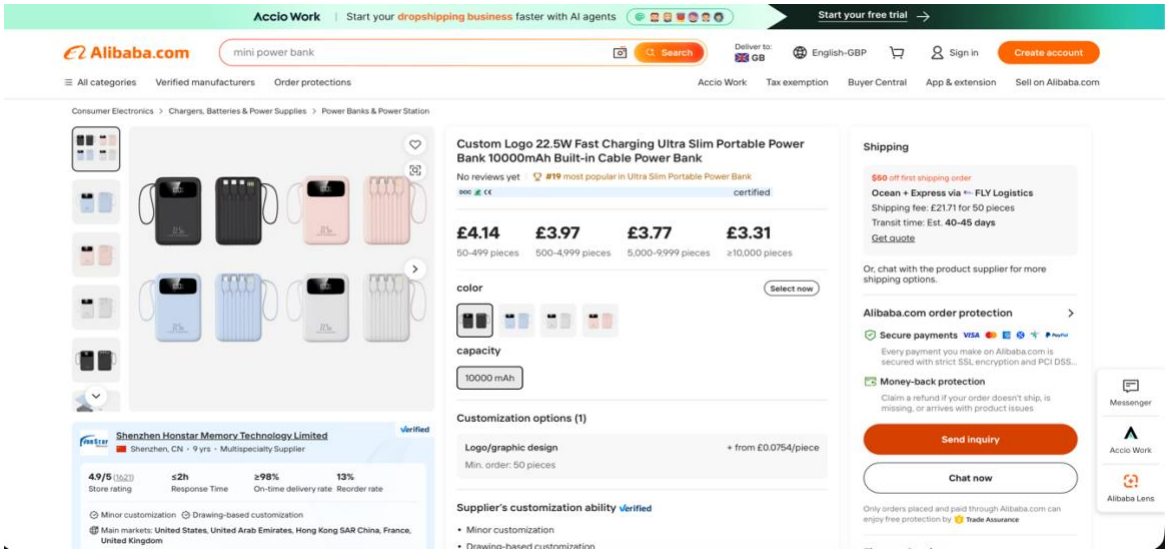

Рисунок 3.12 – Пропозиція від постачальника на ідентичні павербанки.

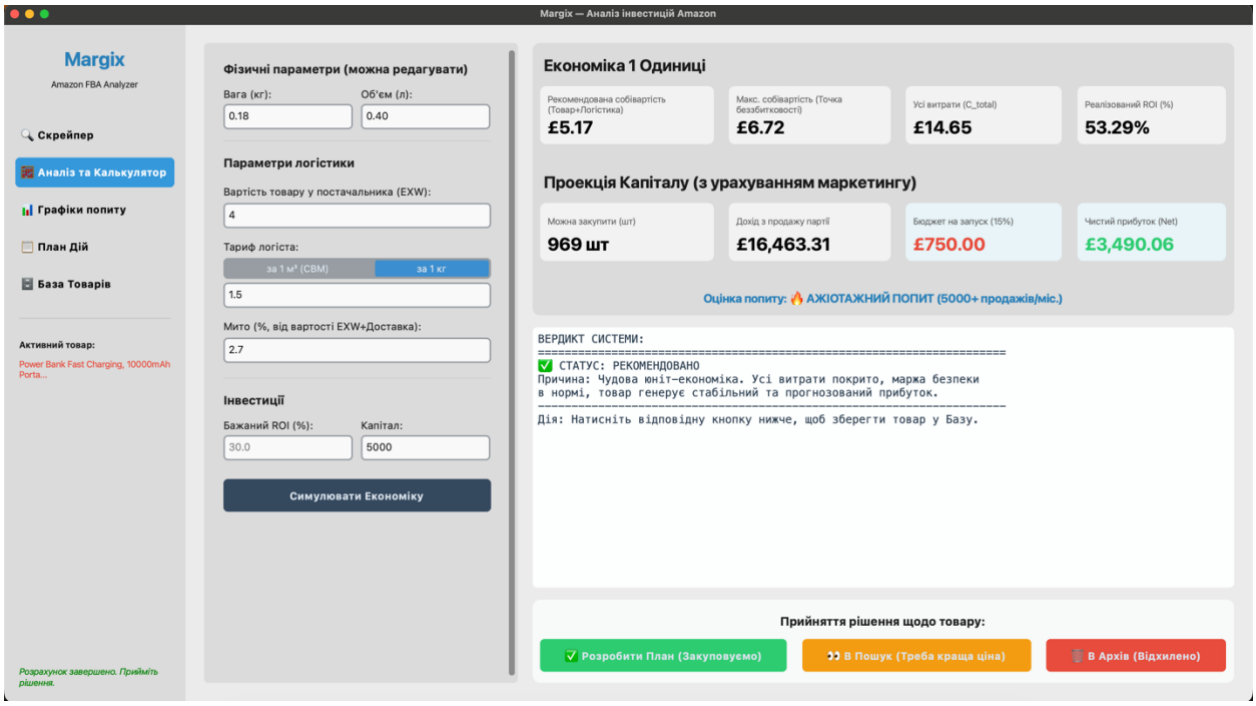

Рисунок 3.13 – Результат роботи функції аналізу та калькулятора.

Система робить вердикт за яким можна одразу зрозуміти чи необхідно вкладати гроші у товар від цього постачальника. У разі, якщо постачальник продає товар за ціною вище, ніж у прикладі, та ціна не є ідеальною – система видає попередження та рекомендації щодо можливих покращень (рис. 3.14.).

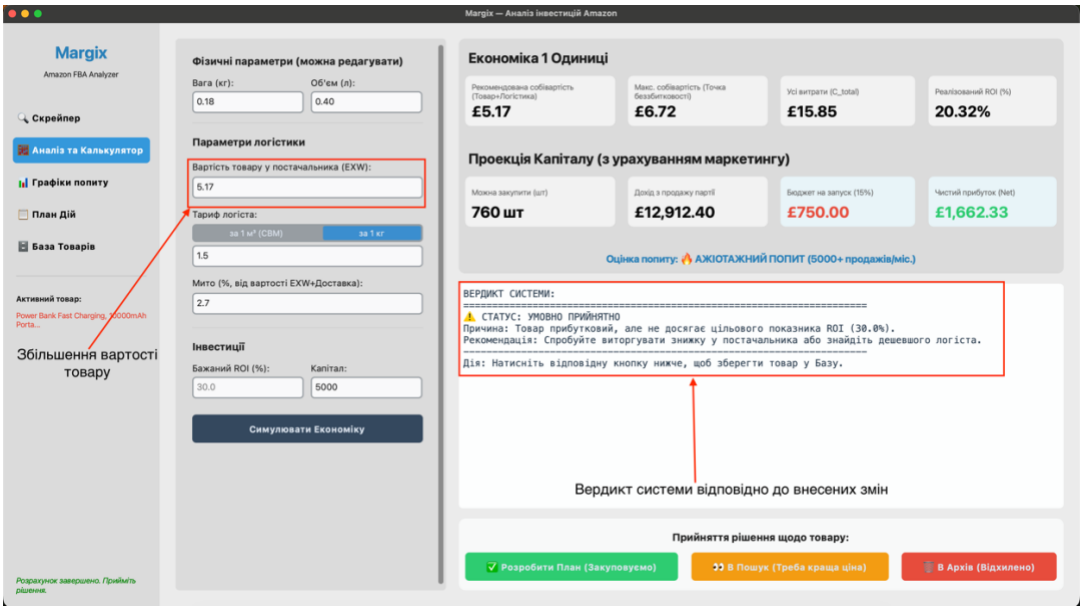

Рисунок 3.14 – Попередження системи у разі збільшення ризику втрати капіталу.

У випадку, коли постачальник продає товар за невигідною ціною, або ціна на логістичні послуги у купі з іншими факторами перевищує точку беззбитковості, система забороняє вкладати капітал та обґрунтовує чому саме (рис. 3.15.).

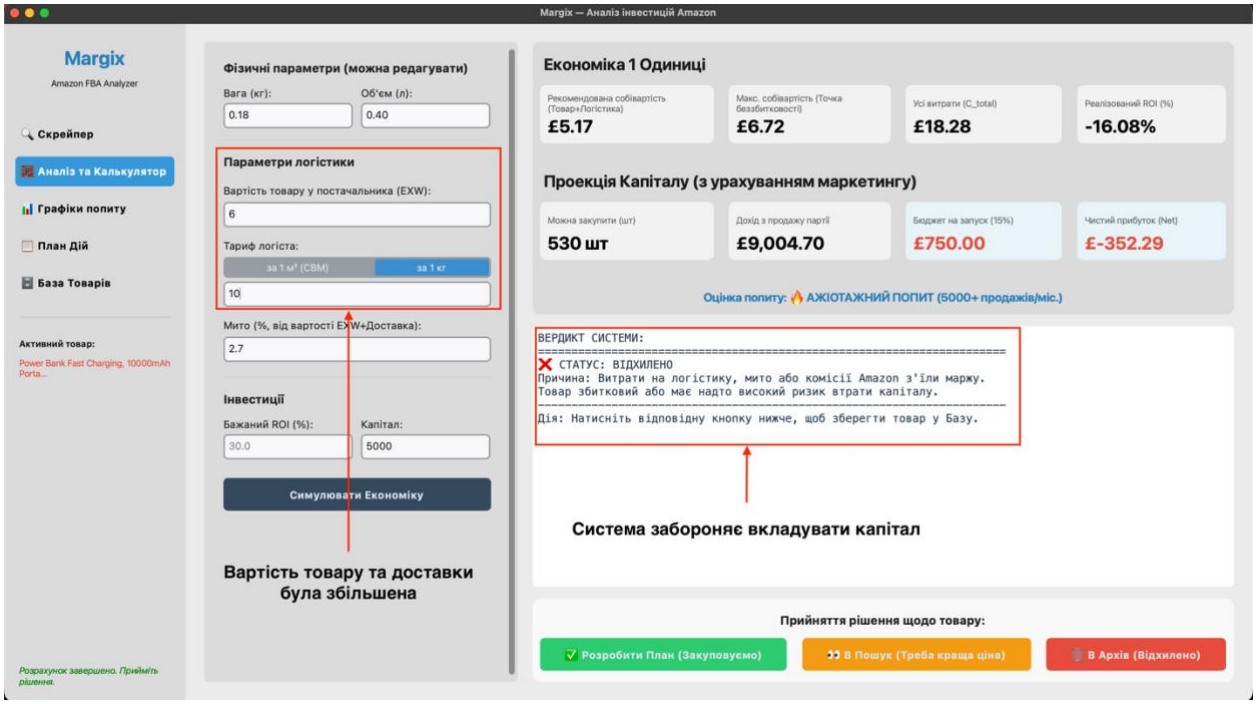

Рисунок 3.15 – Заборона системи у разі збитковості проєкту.

У випадках, коли проєкт все ж таки є прибутковим, але реалізований ROI знаходиться на низьких значеннях, система також забороняє вкладати капітал, оскільки бізнес може стати збитковим у разі будь-яких коливань ринку або непередбачуваних витрат (рис. 3.16).

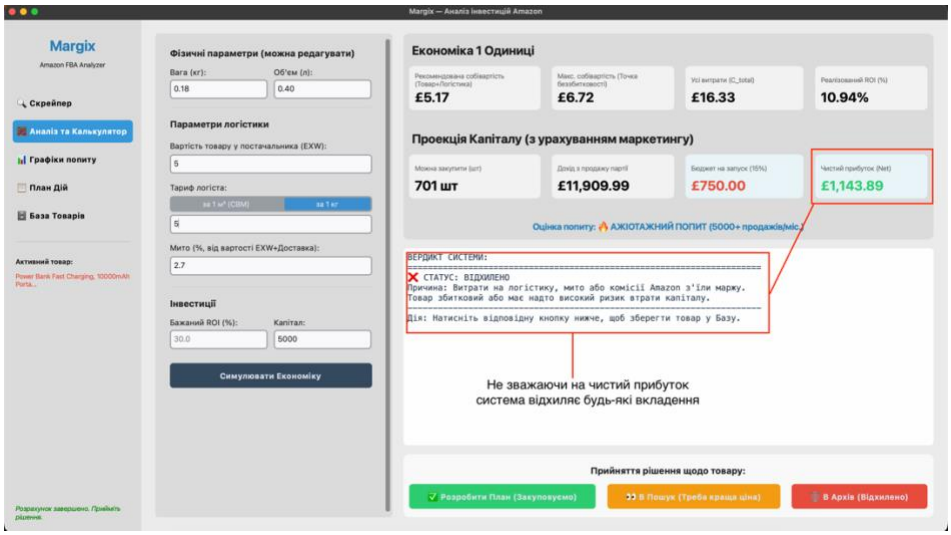

Рисунок 3.16 – Заборона системи у разі високих ризиків втрати капіталу.

Після виконаного аналізу у користувача є можливість ознайомитись з графіком чутливості чистого прибутку та круговою діаграмою розподілу роздрібної ціни (рис. 3.17.) та перейти до прийняття рішення щодо товару.

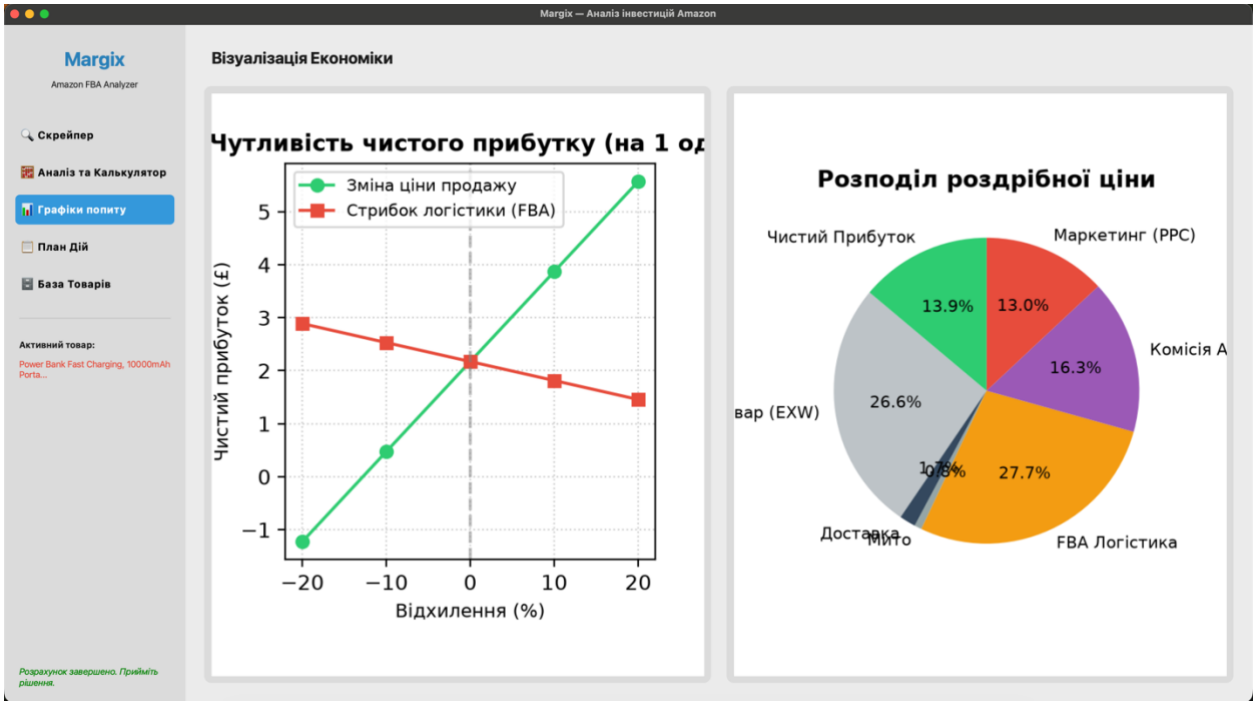

Рисунок 3.17 – Графік попиту та діаграма розподілу роздрібної ціни.

Останнім етапом є прийняття рішення користувачем. У разі вибору кнопки «Розробити план (Закуповуємо)» система робить детальний план та

розписує всі етапи та рух капіталу крок за кроком (рис. 3.18.), додаючи при цьому товар та всі дані аналізу у відповідну папку у меню «База товарів» (рис. 3.19). У випадку, коли користувач обирає інші кнопки, система відправляє товар у відповідну до кнопки категорію, при цьому не генеруючи план.

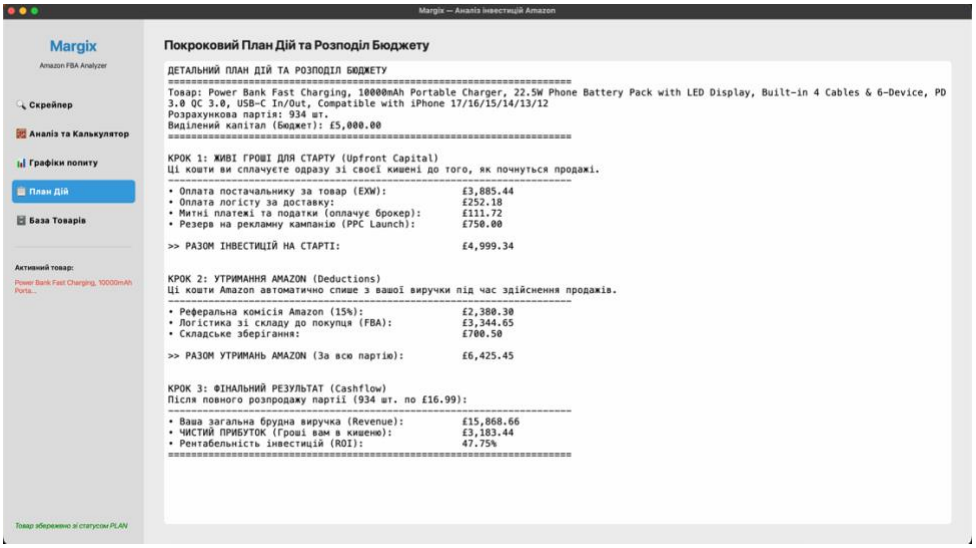

Рисунок 3.18 – Згенерований план дій відповідно до обраного товару.

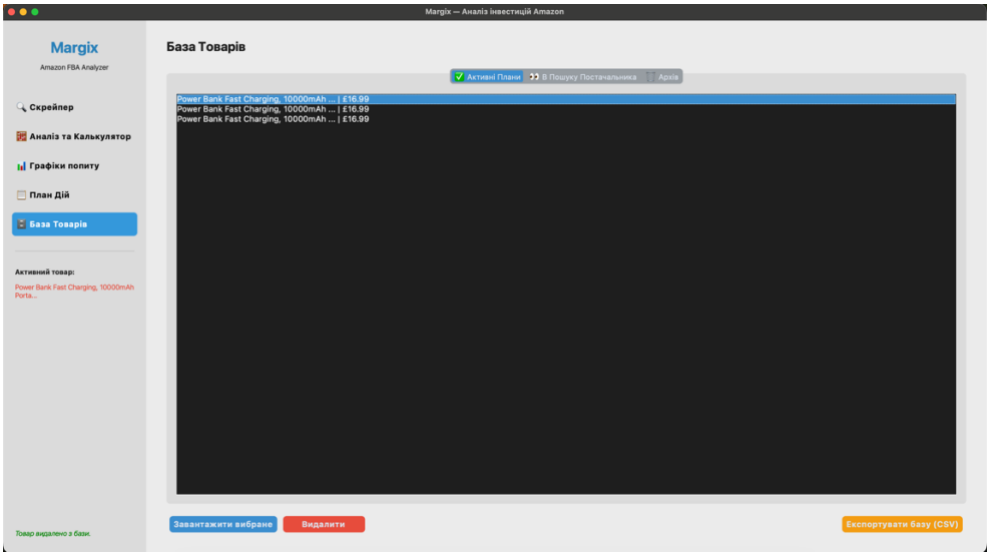

Рисунок 3.19 – Згенерований план дій відповідно до обраного товару.

На сторінці з вибором товару користувач також має можливість експортувати всю базу в форматі CSV, завантажити позицію для внесення змін або видалити товар з бази даних.

Для комплексної оцінки якості розробленого програмного забезпечення та підтвердження практичної цінності системи було проведено серію верифікаційних випробувань, результати яких зведені у загальну матрицю тестування функціональних сценаріїв комп'ютерної моделі (табл. 3.1).

Таблиця 3.1 – Матриця результатів тестування функціональних модулів

№	Функціональний сценарій	Вхідні дані	Очікуваний результат	Фактичний результат	Статус
1	2	3	4	5	6
1	Веб-скрапінг та парсинг лістингу	URL сторінка павербанка	Вилучення назви, ціни з копійками, BSR, відгуків, ваги	Дані вилучено у повному обсязі.	Успішно
2	Автоматичне визначення регіональної валюти	Домен суфікса .co.uk у структурі URL	Автоматична зміна маски знаку валюти на символ фунта (£)	Всі вікна, картки та текстовий звіт переведено на знак £	Успішно
3	Адаптація тарифів для сегменту LOW	Об'єкт з Buy Box ціною ≤15.0	Зниження базових ставок FBA та активоване обмеження TACoS	Увімкнено режим LOW, FBA = £1.62, маркетинг обмежено	Успішно
4	Симуляція глибокої Unit-економіки	Введення EXW = £5.00, тариф доставки, мито 5%	Точний розрахунок повної собівартості та прибутку	Розраховано собівартість, чистий прибуток £3490	Успішно
5	Генерація проєкції капіталу партії	Введення бюджету £5,000	Резерв 15% на PPC, розрахунок кількості штук	Виділено £750 на PPC, розраховано партію та план дій	Успішно
6	Взаємодія з NoSQL сховищем MongoDB	Кліки по кнопках рішень	Запис документа у БД, оновлення списку історії	Документ збережено, сайдбар історій миттєво оновлено	Успішно

Продовження таблиці 3.1

1	2	3	4	5	6
7	Перманентне видалення записів з СУБД	Виділення позиції, клік по кнопці «Видалити»	Виклик асинхронного методу delete_report, очищення UI	Запис видалено з MongoDB, список очищено без зависань UI	Успішно
8	Табличний експорт бази даних	Клік по кнопці «Експортувати базу (CSV)»	Серіалізація через pandas у файл CSV з кодуванням UTF-8-SIG	Файл згенеровано, кириличні вердикти відображаються коректно	Успішно

3.6 Ілюстрація роботи застосунку

Графічний користувацький інтерфейс розробленої десктопної системи підтримки прийняття рішень спроєктовано з використанням сучасних принципів ергономіки та UX-дизайну десктопних застосунків на macOS. Компонування елементів керування та панелей відображення розраховано на забезпечення максимальної швидкості сприйняття фінансової інформації користувачем.

Програма реалізована у вигляді двокомпонентного вікна розміром 1400×900 пікселів, де ліву частину займає постійний навігаційний сайдбар шириною 250 пікселів, а праву аналітичний дашборд, конфігурація якого змінюється залежно від обраного пункту меню. Сайдбар містить назву програми, кнопки вибору режимів із вбудованою підсвіткою активного стану, текстову мітку назви активного у даний момент товару та нижній текстовий індикатор асинхронних операцій ядра.

У вкладці «Скрейпер» права область ділиться на два рівні фрейми. Ліва панель містить інструменти введення посилання та кнопку запуску Playwright. Права панель являє собою блок специфікацій, де центральне місце займає холст tk.Canvas для рендерингу зображення. Справа від фотографії розміщено

чітку вертикальну таблицю текстових міток, які відображають спарсені числові показники лістингу із застосуванням великого шрифту для забезпечення високої контрастності читання.

Візуальний простір вкладки «Аналіз та Калькулятор» розподілено між лівою панеллю введення ручних параметрів, побудованою на базі скрол-фрейму, та правою аналітичною зоною. Права зона містить два ряди плиткових карток з великими жирними цифровими індикаторами, під якими розміщено текстовий рядок якісної оцінки попиту та велике поле «ВЕРДИКТ СИСТЕМИ». Нижня частина фрейму обладнана трьома великими кольоровими кнопками дій для маршрутизації документа.

Вкладка «Графіки попиту» містить інтегроване полотно розбите на лінійний графік чутливості прибутку з двома різнокольоровими лініями маркерів та кругову діаграму розподілу роздрібної ціни, яка секторно відображає частку кожного накладного розходу в структурі вартості. Вкладка «План Дій» представлена текстовим полем із детальним фінансовим розбором. Вкладка «База Товарів» містить горизонтальний блок вкладок для фільтрації історій та нижній ряд кнопок адміністрування бази даних і виклику експорту.

3.7 Формулювання шляхів подальшого розвитку отриманих результатів

Розроблена інформаційно-аналітична система підтримки прийняття рішень має значний потенціал для подальшого масштабування та функціонального розширення в рамках автоматизації бізнес-процесів електронної комерції. Основним вектором подальшого розвитку отриманих результатів є інтеграція в обчислювальне фінансове ядро engine.py модулів прогнозної аналітики на основі машинного навчання. Впровадження легковагових регресійних моделей (наприклад, алгоритмів Random Forest або XGBoost) безпосередньо у фоновий потік програми дозволить системі не просто зчитувати поточний ранг BSR товару, а й прогнозувати його динаміку

та коливання попиту на 30, 60 та 90 днів уперед на основі ретроспективного аналізу історії відгуків та графіків завантаження аналогічних ніш.

Другим важливим напрямком модернізації системи є архітектурне розширення підсистеми скрапінгу scraper.py для підтримки мультіплатформенного паралельного збору даних. Додавання нових асинхронних модулів парсингу дозволить ІАС ППР одночасно збирати інформацію не лише з європейських та американських майданчиків Amazon, але й крос-платформенно аналізувати оптові ціни на B2B-платформах (таких як Alibaba, 1688, Global Sources) та роздрібні показники на альтернативних маркетплейсах (eBay, Walmart). Це дозволить реалізувати повністю автоматизований алгоритм віртуального пошуку постачальників, коли система за одним кліком самостійно знаходить ідентичний товар на оптовому ринку Китаю, розраховує повну вартість його доставки (Landed Cost) та миттєво порівнює з роздрібною ціною Buy Box в Англії чи США.

Третім вектором розвитку є розширення інтерфейсних та комунікаційних можливостей додатка в main.py. Перспективним є впровадження системи хмарної синхронізації локальної бази даних MongoDB з розподіленими хмарними сховищами (наприклад, MongoDB Atlas), що дозволить інвестору мати безперебійний доступ до збережених планів закупівлі з різних пристроїв. Також доцільним є інтеграція API-інтерфейсів сучасних банківських сервісів та логістичних платформ для автоматичного оновлення курсів валют у реальному часі та динамічного підтягування актуальних митних та фрахтових тарифів, що підніме точність симуляцій юніт-економіки до абсолютного максимуму.

ВИСНОВКИ

У рамках кваліфікаційної роботи було розроблено та реалізовано інформаційно-аналітичну систему підтримки прийняття рішень для оцінювання інвестиційної привабливості ніш в e-commerce на основі аналізу відкритих даних маркетплейсу Amazon.

Для досягнення поставленої мети вирішено такі завдання:

– досліджено попит на інформаційно-аналітичні системи та тенденції розвитку інтелектуального аналізу відкритих даних у сфері e-commerce;

– проаналізовано існуючі програмні рішення для моніторингу товарних ніш та визначено їх функціональні обмеження;

– досліджено можливості використання асинхронних headless-контекстів браузерів та засобів парсингу для автоматизованого вилучення інформації з лістингів в умовах захисту від роботів;

– розроблено алгоритми точної екстракції дробової частини ціни та автоматичного визначення регіональної валюти за доменним суфіксом посилання;

– сформовано комплексну економіко-математичну модель юніт-економіки з розщепленням повної собівартості на ціну заводу, логістичний фрахт та митні збори;

– розроблено підхід до адаптивного моделювання операційних витрат і маркетингового тиску для запобігання від’ємної маржинальності у низькобюджетних товарних сегментах;

– спроєктовано та реалізовано функціонал «Проєкції капіталу» з автоматичним резервуванням 15% бюджету на первинний рекламний запуск партії товару;

– обґрунтовано та побудовано багатопотокову модель неблокуючої взаємодії користувацького інтерфейсу з асинхронним циклом подій;

– розроблено робочий прототип застосунку під керуванням macOS із плитковою матрицею фінансових показників та вкладками інтерактивного аналізу;

– реалізовано підсистему довгострокового збереження, фільтрації за бізнес-статусами та перманентного видалення звітів у NoSQL базі даних MongoDB.

Проаналізовано існуючі аналітичні продукти для оцінювання комерційного потенціалу товарів та сучасні підходи до вебскрапінгу динамічних вебсторінок. За результатами дослідження обрано асинхронне ядро Playwright спільно з компіляційним синтаксичним аналізатором BeautifulSoup як основу стабільного вилучення неструктурованих даних лістингів, стійку до змін верстки маркетплейсу Amazon.

Розроблено обчислювальне ядро nvestmentEngine для проведення фінансових симуляцій, розрахунку роздрібної точки беззбитковості, запасу міцності маржі та граничної ціни закупівлі у постачальника. Для оцінювання ринкового попиту товару використано алгоритм розрахунку індексу надійності попиту DRI на основі рангу BSR, обсягу продажів та відгуків, з автоматичною активацією моделі симуляції за середньою медіаною ніші у разі часткового блокування сторінки.

Розроблено настільний macOS-застосунок для комплексного оцінювання інвестиційної привабливості товарів з модулем автоматизованого збору інформації за єдиним посиланням. Застосунок реалізовано на основі фреймворку CustomTkinter з інтеграцією розрахункових модулів на Python та асинхронного драйвера Motor для MongoDB, що дозволило повністю ліквідувати ручне заповнення специфікацій, автоматизувати аудит собівартості Landed Cost та уникнути зависань вікон програми завдяки багатопоточному класу AsyncLoopThread. За результатами експериментального тестування на реальних ринкових лістингах система

продемонструвала високу точність обчислень, гнучкість адаптації тарифів та адекватність сформованих вердиктів.

Реалізовано інтегроване візуальне полотно на базі бібліотеки Matplotlib, що будує графіки чутливості прибутку до коливань витрат у діапазоні 20% та кругові діаграми розподілу роздрібної ціни, а також впроваджено модуль табличного експорту за допомогою pandas, що дозволило інвестору здійснювати миттєвий візуальний аудит капіталу та вивантажувати всю історію бази даних у загальноприйнятий формат CSV з кодуванням UTF-8-SIG.

Результати роботи можуть бути використані для подальшого розвитку інтелектуальних систем підтримки прийняття рішень у сфері міжнародної електронної комерції, оптимізації ланцюгів постачання та вдосконалення предиктивних алгоритмів фінансового аналізу комерційних ризиків.

Результати роботи апробовано у вигляді тези доповіді під час IX Міжнародної студентської конференції «Міждисциплінарні наукові дослідження та перспективи їх розвитку».

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Плескач, В. Л., & Зайцева, Н. В. (2021). Електронна комерція: підручник Знання.

2. Что такое Амазон: подробное описание маркетплейса : вебсайт. Хабр. URL: https://habr.com/ru/companies/pochtoy/articles/406783/

3. Що таке ASIN (Amazon Standard Identification Number) : вебсайт. Sellerlogic Wiki. URL: https://www.sellerlogic.com/ru/wiki/asin/.

4. Що вибрати: FBA чи FBM? : вебсайт. Блог про Amazon. URL: https://gul.in.ua/chto-vybrat-fba-ili-fbm/.

5. What is the Amazon Buy Box? : вебсайт. Helium 10. URL: https://www.helium10.com/blog/what-is-the-amazon-buy-box/.

6. What is Amazon Best Sellers Rank (BSR)? : вебсайт. Helium 10. URL: https://www.helium10.com/blog/what-is-amazon-best-sellers-rank/.

7. Косцов, М. П., & Іванов, О. В. (2023). Модифікація класичних індексів концентрації ринку для аналізу цифрових платформ та маркетплейсів. Економіка та управління підприємствами.

8. Web scraping : вебсайт. Вікіпедія: вільна енциклопедія. URL: https://uk.wikipedia.org/wiki/Web_scraping.

9. Що таке API та як воно працює : вебсайт. HOSTiQ. URL: https://hostiq.ua/blog/ukr/what-is-api/.

10. Гавловський, С. В. (2022). Інтерфейси прикладного програмування в системах електронного бізнесу. КНУТД.

11. Amazon API Gateway : вебсайт. Amazon Web Services. URL: https://aws.amazon.com/ru/api-gateway/.

12. Web Scraper - Free Web Scraping Tool : вебсайт. URL: https://webscraper.io.

13. Кларк, Д. (2024). Метод зворотного сорсингу (Reverse Sourcing) як основа побудови ефективної бізнес-моделі на міжнародних торгових майданчиках. Міжнародний економічний журнал.

14. Keepa - Amazon Price Tracker : вебсайт. URL: https://keepa.com/.

15. Jungle Scout: Amazon Product Research Tool & Finder : вебсайт. URL: https://www.junglescout.com/.

16. Бідюк, П. І., & Повгач, О. А. (2024). Системний аналіз та моделювання процесів прийняття рішень в умовах невизначеності. Комп'ютерні дослідження та інформаційні технології.

17. Вітлінський, В. В., & Великоіваненко Г. І. (2021). Ризикологія в економіці та підприємництві: монографія. КНЕУ.

18. Бланк, І. О. (2022). Інвестиційний менеджмент: підручник. Ельга; Ніка-Центр.

19. Saltelli, A., Ratto, M., & Andres, T. (2024). Global Sensitivity Analysis: The Primer. John Wiley & Sons.

20. Спрингер, М. (2025). Асинхронне програмування та оптимізація мережевих запитів високої частоти в Unix-подібних системах. Програмування та комп'ютерні системи.

21. Ричардсон, Л., & Руби, С. (2023). RESTful Web Services: веб-скрапінг та імітація поведінки користувача для запобігання блокувань. Старий Лев.

22. Чоо, Ю. Х., & Кім, С. (2023). Метод багатокритеріальної крос-верифікації даних в інформаційно-аналітичних системах електронної комерції. Журнал системних наук та кібернетики.

23. Садаладж, П. Дж., & Фаулер, М. (2022). NoSQL дистильовано. Короткий путівник за світом нереляційних баз даних. Діалектика.

24. Кузнецов, О. В., & Федоренко, Д. М. (2024). Оптимізація логічних схем документо-орієнтованих баз даних для високошвидкісних аналітичних звітів. Вісник ХНУРЕ. Серія: Інформаційні системи.