Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

магістра
(назва освітньо-кваліфікаційного рівня)

студента *Іванова Дениса Вікторовича*
(ПІБ)

академічної групи *122м-22-2*
(шифр)

спеціальності *122 Комп'ютерні науки*
(код і назва спеціальності)

освітньої програми *«122 Комп'ютерні науки»*
(назва освітньої програми)

на тему: *«Розробка та дослідження ефективності прогнозування доходів фінансовою інформаційною системою»*

_____ *Д.В. Іванов*

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	*доц. Приходченко С. Д.*			
спеціального розділу	*доц. Приходченко С. Д.*			

Рецензент				

Нормоконтролер	*Доц. Гуліна І.Г.*			

Дніпро
2023

Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
Завідувач кафедри
Програмного забезпечення комп'ютерних систем
(повна назва)

М.О. Алексєєв
(підпис) (прізвище, ініціали)

« » 2023 року

ЗАВДАННЯ
на виконання кваліфікаційної роботи

спеціальності *122 Комп'ютерні науки*
(код і назва спеціальності)

студенту *122м-22-2* *Іванова Дениса Вікторовича*
(група) (прізвище та ініціали)

Тема кваліфікаційної роботи *«Розробка та дослідження ефективності прогнозування доходів фінансовою інформаційною системою»*

1 ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Наказ ректора НТУ «Дніпровська політехніка» від 09 .10. 2023 р. № 1227-с

2 МЕТА ТА ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Об'єкт дослідження – прогнозування доходів фінансовою інформаційною системою

Предмет дослідження – ефективність прогнозування доходів програмно.

Мета НДР – визначення методів ефективного прогнозування та ефективності прогнозування доходів.

Вихідні дані для проведення роботи – теоретичні та експериментальні дослідження, дані отримані під час використання сучасних засобів та методів прогнозування, використовуючи реальні дані.

3 ОЧІКУВАНІ НАУКОВІ РЕЗУЛЬТАТИ

Новизна запропонованих рішень та очікуваних результатів полягає у дослідженні ефективності прогнозування доходів фінансовою інформаційною системою та можливість її покращити.

Практична цінність отриманих результатів полягає в виявленні методу для ефективного прогнозування майбутніх доходів.

4 ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Результати досліджень повинні бути представлені у формі, що дозволяє їх безпосереднє використання в реальних проєктах, без необхідності додаткового аналізу інструментів та визначення чинників для ефективного виконання процесу прогнозування.

5 ЕТАПИ ВИКОНАННЯ РОБІТ

Найменування етапів робіт	**Строки виконання робіт (початок – кінець)**
Розробка UI компоненти для інформаційної системи	01.09.2023-01.10.2023
Розробка та пошук оптимальних методів для прогнозування доходів	01.10.2021-14.11.2023
Аналіз отриманих результатів внаслідок дослідження ефективності прогнозування наявними методами	15.11.2021-10.12.2023

6 РЕАЛІЗАЦІЯ РЕЗУЛЬТАТІВ ТА ЕФЕКТИВНІСТЬ

Економічний ефект від реалізації результатів роботи очікується позитивним завдяки швидшому аналізу наявних даних маркетплейсу. Впровадження програми, яка автоматизовано аналізує доходи та витрати, а також прогнозує прибуток на майбутній рік, може призвести до значного економічного ефекту для підприємства. Ось кілька ключових аспектів ефективності, які може забезпечити ця програма. Інформаційна система надає

можливість точно визначити фінансові цілі та розробляти оптимальні стратегії для досягнення їх. Це сприяє оптимізації розподілу бюджету, забезпечуючи ефективне використання ресурсів.

Соціальний ефект від реалізації результатів роботи очікується позитивним завдяки підвищенню швидкості проведення аналізу економічної складової маркетплейсу. Здатність точно прогнозувати прибуток дозволяє підприємствам раціонально розподіляти ресурси та інвестувати в потенційно прибуткові сфери. Це сприяє загальному економічному зростанню та розвитку бізнес-середовища.

7 ДОДАТКОВІ ВИМОГИ

Завдання видав _____ *Приходченко С.Д.*
(підпис) (прізвище, ініціали)

Завдання прийняв до виконання _____ *Іванов Д.В.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *25.10.2023 р.*

Термін подання кваліфікаційної роботи до ЕК *21.12.2023*

РЕФЕРАТ

Пояснювальна записка: 80 с., 26 рис., 2 додатки, 25 джерел.

Об’єкт досліджень: прогнозування доходів фінансовою інформаційною системою

Предмет досліджень: ефективність прогнозування доходів програмно.

Мета роботи: визначення методів ефективного прогнозування та ефективності прогнозування доходів.

Методи дослідження: проведення математичних обчислень щодо ефективності прогнозування.

Наукова новизна: дослідження ефективності прогнозування доходів фінансовою інформаційною системою та можливість її покращити.

Практична цінність: виявленні ефективного методу для хорошого прогнозування майбутніх доходів.

Область застосування: Дослідження в галузі прогнозування доходів знаходить широке застосування в фінансовій сфері та бізнес-аналізі, де точні та ефективні методи аналізу дозволяють досягати значущих переваг.

Економічний ефект: підвищенню швидкості проведення аналізу економічної складової маркетплейсу.

Значення роботи та висновки: проведені аналіз показує, що для системи щ, що має багато змінних, метод прогнозування лінійної регресією є оптимальним.

Прогнозні припущення про розвиток досліджень: перспективним напрямком розвитку цієї роботи може стати удосконалення програмного продукту для роботи з іншими об’ємами даних.

Список ключових слів: ЛІНІЙНА РЕГРЕСІЯ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, МАШИННИЙ АНАЛІЗ, ПРОГНОЗУВАННЯ

ABSTRACT

Explanatory note: 80 pp., 26 figures, 2 appendix, 25 sources.

Research Object: income forecasting by financial information system

Research Subject: the effectiveness of programmatic revenue forecasting.

Research Objective: determination of methods of effective forecasting and the effectiveness of revenue forecasting.

Research Methods: performing mathematical calculations on the effectiveness of forecasting.

Scientific Novelty: a study of the effectiveness of income forecasting by the financial information system and the possibility of its improvement.

Practical Value: identifying an effective method for good forecasting of future income.

Application Area: research in the field of revenue forecasting is widely used in finance and business analysis, where accurate and efficient methods of analysis allow significant benefits to be achieved.

Economic Impact: increasing the speed of analysis of the economic component of the marketplace.

Significance and Conclusions: the conducted analysis shows that for a system with many variables, the method of forecasting by linear regression is optimal.

Future Research Assumptions: a promising direction for the development of this work may be the improvement of a software product for working with other volumes of data.

Keywords: LINEAR REGRESSION, SOFTWARE, MACHINE ANALYSIS, PREDICTION.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ПЗ -	Програмне забезпечення
ІС -	Інформаційна система
ОС -	Операційна система
БД -	База даних
CSS -	Cascading Style Sheets
SQL -	Structured Query Language
HTML -	Hyper Text Markup Language
JS -	JavaScript
ПК -	Персональний комп’ютер
ФІС -	Фінансова інформаційна система
API -	Application Programming Interface
BSON -	Binary Javascript Object Notation
UI -	User interface

ЗМІСТ

ВСТУП

У сучасному світі, де бізнес та економіка швидко розвиваються, а конкуренція на ринку набуває все більшого значення, аналіз та прогнозування фінансових показників стають важливими складовими для успішної діяльності підприємств та маркетплейсів. Один із важливих аспектів, що впливає на прийняття стратегічних рішень у сфері бізнесу, полягає в аналізі зростання чи спадання потенційних доходів.

Об'єктом досліджень даної магістерської роботи є показники зростання чи спадання потенційних доходів. Ця тема стає особливо актуальною в умовах постійних змін на ринку, коливань попиту та пропозиції, а також зростаючої конкуренції.

Предметом досліджень є математичні моделі та методи розрахунку поточних та потенційних прибутків. Вивчення цих аспектів дозволить ефективніше аналізувати і прогнозувати фінансову стабільність та перспективи розвитку різних галузей економіки.

Метою даної науково-дослідної роботи є підвищення ефективності аналізу та прогнозування релевантності маркетплейсів. Ця мета важлива для підтримки прийняття рішень на основі об'єктивних даних та забезпечення конкурентних переваг у бізнесі.

Вихідні дані для проведення досліджень включають в себе теоретичні та експериментальні дослідження, а також різноманітні методи обробки експериментальних даних. Ці дані стануть основою для розробки та валідації математичних моделей, які допоможуть досягти поставленої мети. У цій магістерській роботі ми спробуємо детально розглянути важливі аспекти аналізу та прогнозування фінансових показників, використовуючи різні математичні підходи та методи. Ми сподіваємося, що результати наших досліджень сприятимуть покращенню стратегічного управління та прийняттю обгрунтованих рішень в галузі бізнесу та економіки.

Ця магістерська робота скерована на вдосконалення аналітичних інструментів та методів, що допоможуть сучасним підприємствам бути більш

конкурентоспроможними та успішними на ринку.

Робота складатиметься з трьох розділ та висновків до них.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Загальні відомості з предметної галузі

Фінансова інформаційна система (ФІС) стала неодмінною частиною сучасного управління організаціями та підприємствами у всіх галузях бізнесу. Розробка та дослідження доходів ФІС відіграють ключову роль у забезпеченні ефективного контролю фінансів, прийнятті обгрунтованих управлінських рішень та досягненні фінансового успіху.

Дана робота включає в себе глибокий аналіз різноманітних аспектів, які впливають на фінансовий облік і управління.

Основні напрями дослідження та розробки включають в себе:

- архітектуру ФІС, тобто при розробці ФІС важливо ретельно проаналізувати архітектурні особливості системи та визначити оптимальну структуру, яка забезпечить ефективний облік та аналіз фінансової інформації. Це може включати в себе вибір технологічних платформ, дизайн баз даних, та інші технічні аспекти.
- ФІС повинна бути пов'язана з іншими інформаційними системами, які використовуються в організації. Це може включати взаємодію з системами управління ресурсами підприємства (ERP), системами управління виробництвом, торгівлі та іншими. Дослідження способів інтеграції є важливою частиною розробки ФІС.
- ФІС має надавати можливості для глибокого аналізу фінансових даних та відображення ключових показників, що впливають на фінансовий стан організації. Дослідження методів аналізу фінансової інформації та розробка відповідних інструментів стають важливими завданнями.
- з огляду на чутливу природу фінансової інформації, розробка ФІС повинна передбачати використання високих

стандартів безпеки даних та контролю доступу. Це включає в себе захист від несанкціонованого доступу, а також резервне копіювання та відновлення даних в разі аварій.

Один із ключових аспектів дослідження - оцінка впливу ФІС на фінансовий результат організації. Це включає в себе аналіз показників рентабельності, прибутковості та ефективності фінансових операцій під впливом впроваджених змін.

Зараз ФІС розглядається як стратегічний інструмент для організацій будь-якого розміру та сфери діяльності. Вона допомагає підприємствам збільшити ефективність операцій, оптимізувати фінансовий контроль і мінімізувати ризики.

1.2 Мета розробки та сфера застосування

Мета створення нашої інформаційної системи полягає в тому, щоб революціонізувати підхід до фінансового аналізу та прогнозування доходів для підприємств у всіх сферах діяльності. Я прагну створити інтелектуальну і надійну інформаційну систему, яка допомагатиме бізнесам краще розуміти, аналізувати і передбачати свої фінансові результати.

Сфера застосування нашої системи широка та багатогранна:

- наша інформаційна система допоможе фінансовим аналітикам та фахівцям в галузі прогнозування отримувати доступ до високоякісних інструментів для аналізу фінансових даних. Вони зможуть визначити тенденції, ідентифікувати ключові фактори та робити зважені прогнози
- бізнес-лідери та менеджери зможуть використовувати нашу систему для розробки більш обґрунтованих стратегій та бюджетів, спираючись на точні прогнози доходів.
- власники підприємств та фінансові керівники зможуть краще керувати фінансами, мінімізувати ризики та оптимізувати фінансовий результат.
- інвестори та фонди зможуть використовувати нашу систему для прийняття обґрунтованих рішень щодо інвестицій у компанії.

– система може бути використана у будь-якій галузі, включаючи фінанси, торгівлю, виробництво, технології, охорону здоров'я та інші, для поліпшення управління та фінансової стратегії.

Дана інформаційна система буде інструментом, що допомагає підвищити рівень аналізу та точності прогнозування доходів. Вона розроблена з метою покращення фінансової стійкості, зростання прибутковості та конкурентоспроможності підприємств, що використовують її можливості.

1.3 Підстава для розробки

Відповідно до ОПП, згідно навчального плану та графіків навчального процесу, в кінці навчання студент виконує кваліфікаційну роботу (проект).

Тема роботи узгоджується з керівником проекту, випускаючою кафедрою, та затверджується наказом ректора.

Отже, підставами для розробки (виконання кваліфікаційної роботи) є:

– ОПП за спеціальністю 122 «Комп’ютерні науки»;
– навчальний план та графік навчального процесу;
– наказ ректора Національного технічного університету «Дніпровська політехніка» №1227-с від 09.10.2023;
– завдання на дипломний проект на тему «Розробка та дослідження ефективності прогнозування доходів фінансовою інформаційною системою».

1.4 Постановка задачі

Система, яку ми розробляємо в рамках магістерської роботи, має на меті відповісти на потреби сучасного бізнесу у надійному та точному прогнозуванні доходів.

Ось докладні вимоги до функціонального забезпечення цієї системи:

- система повинна бути здатна збирати фінансові дані з різних джерел, включаючи рахунки, фінансові звіти та джерела зовнішньої інформації. Дані повинні автоматично інтегруватися в єдину систему для подальшого аналізу
- система повинна надавати інструменти для обробки та аналізу фінансових даних. Це включає в себе фільтрацію, сортування, агрегацію даних та визначення ключових показників для подальшого прогнозування.
- основною функцією системи є прогнозування доходів. Вона повинна використовувати методи машинного навчання, такі як нейронні мережі, для створення точних прогнозів на основі наявних даних.
- щоб забезпечити зручність користувачів, система повинна мати інтуїтивний та зрозумілий користувацький інтерфейс. Він повинен надавати можливість завантаження даних, вибору параметрів аналізу та відображення результатів прогнозування.
- система повинна бути легко розгорнута на серверах або у хмарному середовищі. Розробник повинен також надати можливість підтримки та оновлення системи для користувачів.
- надавати докладну та зрозумілу документацію, яка описує архітектуру системи, методи прогнозування та інструкції користувача.
- система має надавати користувачам рекомендації на основі аналізу та прогнозів. Також, генерувати аналітичні звіти для допомоги прийняття управлінських рішень.
- забезпечити високий рівень безпеки для фінансових даних. Механізми шифрування та доступу повинні бути належним чином налаштовані.
- система повинна гарантувати високу продуктивність та швидкий доступ до результатів аналізу та прогнозів.
- надати пропозиції щодо подальшого розвитку системи та можливостей для вдосконалення її функціональності.

1.5 Висновки

В першому розділі було розглянуто загальні відомості про ФІС, її переваги та тенденції, обрано основні напрямки дослідження.

На основі цих даних була сформована мета та сфера застосування нашої системи. Також ми обрали з великого спектру сучасних технологій саму ті, що більше всього підпадають під наші потреби та задачі, та є актуальними та іноваційними в даний час.

В останньому підрозділі була розроблена постановка задачі, що відповідає на потреби сучасного бізнесу у надійному прогнозуванні доходів.

РОЗДІЛ 2
ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

2.1 Вимоги до функціонального забезпечення

Веб-орієнтований фінансовий додаток, створений для надання швидкої та ефективної аналітики продажів, виявляється невід'ємною частиною сучасного бізнес-середовища. Його функціональні можливості охоплюють широкий спектр інструментів, спрямованих на поліпшення управлінської стратегії та прийняття обґрунтованих рішень.

Основні функції цього інноваційного веб-додатка включають:

– ефективне ведення бази даних, що дозволяє зберігати та керувати інформацією про продукти, клієнтів та інші важливі дані магазину;
– інтуїтивний та деталізований перегляд ключових показників, таких як обсяг продажів, середній чек та кількість клієнтів, що дозволяє вчасно реагувати на зміни в бізнес-процесах;
– використання графіків та діаграм для ілюстрації динаміки продажів протягом різних періодів, що сприяє легкому сприйняттю та аналізу даних;
– використання передових аналітичних методів для точного прогнозу майбутнього розвитку бізнесу на основі акумульованих даних;
– можливість легко експортувати накопичені дані для подальшого детального аналізу чи обробки за допомогою інших інструментів;
– гнучку можливість переглядати аналітику та прогнози не лише на комп'ютері, але і на різних мобільних пристроях, забезпечуючи доступність в будь-який момент та в будь-якому місці.

2.2 Опис застосованих математичних методів

Для обчислення ключових показників таких як обсяг продажів, середній чек

та кількість клієнтів було використано прості математичні операції.

Для прогнозування доходів було обрано лінійну регресію (рис. 2.2).

Лінійна регресія [1] визначається математичною формулою:

$$Y = mx + b \quad (2.1)$$

де кожен компонент відіграє свою унікальну роль у моделюванні залежності між змінними:

– y визначає залежну змінну, тобто те, що ми намагаємося передбачити чи пояснити;
– x представляє незалежну змінну, яка слугує вхідним фактором для прогнозу;
– m - це нахил (коефіцієнт), що визначає, як змінюється залежна змінна y відповідно до змін x;
– b представляє точку перетину, показуючи значення y, коли x рівне нулю.

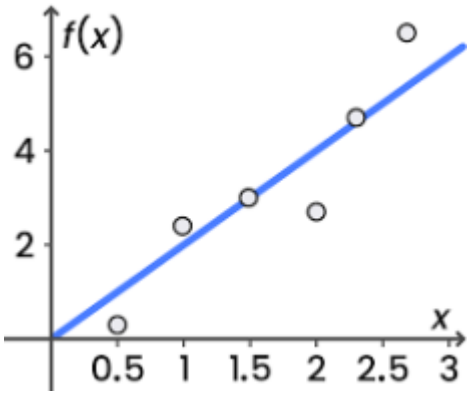

Рис. 2.1. Приклад лінійної регресії

У своїй суті лінійна регресія використовує ці компоненти для створення лінійного відображення між двома змінними. Параметри m та b налаштовуються так, щоб модель максимально точно відображала реальні дані. Цей метод є ефективним інструментом у визначенні та кількісному описі зв'язків між різними змінними в наукових, економічних та інших галузях [2].

2.3 Опис використаних технологій та мов програмування

Для реалізації фінансової інформаційної системи було використано наступні мови програмування:

– JavaScript;

– TypeScript.

Серед технологій було використано:

– React;

– Tailwind CSS;

– база даних MongoDB;

– Material-UI;

– grid-template-areas;

– Express.js;

– Node.js.

2.3.1 Огляд використаних мов програмування

Мова програмування JavaScript

JavaScript [3] представляє собою високорівневу, об'єктно-орієнтовану мову програмування, призначену для розробки веб-додатків. Ініційована для покращення взаємодії з користувачем на стороні клієнта веб-сайтів, згодом JavaScript також став широко використовуватися на стороні сервера завдяки платформі Node.js (рис. 2.2).

JavaScript є однією з найпопулярніших мов програмування у світі. Її застосовують практично на всіх веб-сайтах для реалізації інтерактивності, асинхронних запитів, анімацій та інших функціональностей. Фреймворки, такі як React, Angular і Vue.js, базуються на JavaScript, підсилюючи його популярність [4].

Рис. 2.2. Рейтинг JavaScript проти інших мов

Можливості мови:

– JavaScript підтримує об'єктно-орієнтований підхід, що дозволяє створювати об'єкти з власними властивостями та методами.
– Використовуючи асинхронний підхід, JavaScript оптимізує роботу з операціями вводу/виводу та мережевими запитами, уникаючи блокування виконання коду.
– Змінні можуть змінювати свій тип під час виконання програми, що характеризується динамічною типізацією.
– JavaScript може бути використана для розробки різноманітних додатків, включаючи веб-сайти, веб-додатки, розширення для браузерів, мобільні додатки і навіть серверні застосунки.

Функції мови:

– JavaScript служить скриптовою мовою для взаємодії з HTML-документами та керування веб-сторінками на стороні клієнта.
– За допомогою обробників подій, JavaScript реагує на дії користувача, такі як кліки мишею чи введення з клавіатури.
– JavaScript може змінювати структуру та зовнішній вигляд веб-сторінки,

взаємодіючи з Document Object Model (DOM).

– Використовуючи технології, такі як AJAX, JavaScript може асинхронно взаємодіяти з сервером, уникаючи перезавантаження всієї сторінки.
– JavaScript використовується для створення різноманітних анімацій та динамічних ефектів на веб-сторінці.

Мова програмування TypeScript

TypeScript [5] - це мова програмування, розширення JavaScript, яке надає статичну типізацію та інші розширені можливості для розробки великих та складних веб-додатків. Розроблена і підтримується Microsoft, TypeScript володіє високою ефективністю та інструментами для полегшення роботи розробників.

TypeScript завоювала широку популярність у спільноті розробників завдяки своїй здатності покращувати структуру коду та попереджувати помилки завдяки статичній типізації [6]. Ця мова широко використовується в проектах, що базуються на Angular та React.

Можливості мови:

– TypeScript дозволяє визначати типи змінних та інших об'єктів, що сприяє виявленню помилок на етапі розробки та полегшує супровідний процес.
– Підтримка класів та інтерфейсів у TypeScript дозволяє розробникам створювати чітко структуровані та об'єктно-орієнтовані додатки.
– TypeScript використовує неймспейси та модулі для організації та розділення коду, що полегшує роботу над великими проектами.
– Автоматичне визначення типів дозволяє TypeScript визначати типи без явного їх вказування, що робить код більш конкретним та зрозумілим.
– Код на TypeScript може бути легко скомпільований у чистий JavaScript, що робить його сумісним з усіма браузерами та середовищами виконання.

Функції мови:

– TypeScript дозволяє визначати типи, зменшуючи ймовірність помилок та полегшуючи супровід коду.
– З підтримкою класів та інтерфейсів, TypeScript сприяє впровадженню об'єктно-орієнтованих підходів у розробці.

– TypeScript може бути доданий до існуючих JavaScript-проектів, поступово впроваджуючи статичну типізацію та інші переваги.
– TypeScript підтримує різноманітні інструменти для рефакторингу коду та аналізу, що полегшує роботу з великими проектами.
– З моменту свого виникнення TypeScript отримала широку підтримку спільноти, а також розвинуту екосистему, що включає різноманітні бібліотеки та інструменти.

2.3.2 Огляд використаних технологій

У розробці фінансової інформаційної системи в рамках даної кваліфікаційної роботи використовується бібліотека React [7], розроблена компанією Facebook. React, яка ґрунтується на концепції компонентів, дозволяє ефективно створювати різноманітні інтерфейси користувача та керувати станами компонентів.

Завдяки різноманітним компонентам та інструментам, використовуваним у процесі розробки, забезпечується можливість створення складних програм, при цьому зберігаючи стан в об'єктній моделі документу (DOM). Крім того, React використовується для розробки мобільних додатків з використанням React Native – відкритої "native" бібліотеки для мобільних додатків.

Важливо відзначити, що React не є повноцінним фреймворком, але представляє собою гнучке рішення, що дозволяє використовувати багато інших бібліотек для досягнення конкретних цілей у розробці. В процесі взаємодії з React розробнику надається можливість зосередитися на написанні якісної логіки на рівні програми, використовуючи вже прийняті дизайнерські рішення.

Незважаючи на те, що фреймворки мають свої переваги у вигляді заздалегідь визначених дизайнерських рішень, які полегшують роботу розробника, вони також володіють певними недоліками, особливо для досвідчених розробників, які працюють з великими кодовими базами.

React відкриває можливості для створення різноманітних інтерфейсів користувача та ефективного управління станами компонентів [8]. Важливо

враховувати, що React служить для конкретної області розробки та не включає всі необхідні інструменти, які можуть знадобитися у традиційних фреймворках JavaScript. Отже, вибір інших бібліотек з екосистеми React залишається на розсуд розробника, і нові інструменти постійно виходять на ринок, доповнюючи його функціональність.

Ця технологія веб-розробки, яка базується на React, налічує безліч переваг, роблячи її однією з найпопулярніших у світі. Ось кілька ключових аспектів:

- велика спільнота розробників - важливий фактор у комерційній розробці. Запитання та проблеми можна швидко вирішити завдяки активній спільноті, що полегшує та прискорює процес розробки.
- React відзначається збереженням зворотної сумісності, що робить його оновлення API майже безболісним. Навіть при великих змінах відбувається попередження, що спрощує перенесення коду.
- можливість створювати багаторазові компоненти дозволяє розробникам створювати маленькі фрагменти інтерфейсу та використовувати їх у будь-якій частині програми. Це сприяє швидкій і гнучкій розробці складних інтерфейсів користувача.
- React служить основою для багатьох фреймворків, таких як Gatsby та Next.js, які дозволяють створювати серверні додатки React та генерувати SEO-дружелюстові веб-сайти. Це полегшує створення швидких та оптимізованих додатків.
- використання віртуального DOM допомагає у підтримці абстракції React та покращує продуктивність. Порівнюючи зміни в абстракції програми та відображаючи лише змінені частини, React підвищує ефективність та швидкість програм.

Візуальна компонента в даній кваліфікаційній роботі розроблена за допомогою Tailwind CSS [5]. Tailwind CSS - це сучасний інструмент для розробки веб-інтерфейсів, який базується на концепції utility-first CSS. Замість написання власних стилів, ви використовуєте набір готових класів для визначення стилів елементів. Можливості та Функції:

– Tailwind CSS відзначається підходом "utility-first", що означає використання набору класів для визначення стилів. Наприклад, замість написання окремого CSS правила для кожного стилю, ви можете просто додати класи прямо в HTML-коді.
– Tailwind CSS надає широкий спектр класів для стилізації тексту, рамок, кольорів, відступів, розмірів та інших елементів. Кожен клас логічно іменований, що полегшує їх використання та розуміння.
– Tailwind дозволяє вам легко створювати адаптивні дизайни за допомогою класів, які дозволяють змінювати стилі в залежності від розміру екрану.
– робочі простори (Workspaces) Tailwind дозволяють створювати та використовувати власні компоненти, визначаючи їх стилі, щоб полегшити управління дизайном проекту.
– Tailwind може бути розширений за допомогою додаткових плагінів, які додають нові класи або змінюють налаштування за замовчуванням.
– можливість автоматичної оптимізації та зменшення розміру CSS-файлів, що покращує продуктивність веб-сайту.
– Tailwind надає можливість легко реалізувати режим темної теми за допомогою готових класів.Tailwind CSS надає розробникам ефективний інструмент для швидкого та чистого написання CSS, спрощуючи стилізацію веб-сайтів і покращуючи робочий процес.

MongoDB [9] - це документ-орієнтована система керування базами даних (NoSQL), яка використовує BSON (бінарний формат JSON) для зберігання та обміну даними. MongoDB відрізняється від традиційних реляційних баз даних, орієнтуючись на гнучку та масштабовану схему. ЇЇ було обрано для реалізації в кваліфікаційній роботі через її можливості та функції, такі як:

– MongoDB дозволяє зберігати документи різної структури в одній колекції, надаючи гнучкість у роботі з даними.
– здатність до горизонтального масштабування дозволяє легко розширювати базу даних, додаючи нові сервери для обробки збільшеної навантаженості.

– MongoDB підтримує індексацію для швидкого пошуку даних та оптимізації виконання запитів.
– база даних зберігає дані у вигляді документів у форматі, схожому на JSON, що полегшує роботу з даними та їх розуміння.
– можливість створювати резервні копії даних та реплікувати їх для забезпечення високої доступності та надійності системи.
– MongoDB надає багато вбудованих операторів та можливостей для виконання різноманітних операцій з даними, таких як сортування, фільтрація та агрегація.
– підтримка геопросторових запитів дозволяє працювати з геоданими та виконувати запити, що ґрунтуються на просторових параметрах.
– використання мови запитів, подібної до JavaScript, для взаємодії з базою даних, робить розробку більш інтуїтивною.

MongoDB відкриває широкі можливості для зберігання та обробки даних у великих та розподілених системах, де гнучкість та масштабованість є важливими факторами [10].

Для швидкого та ефективного створення інтерфейсу користувача було обрано бібліотеку Material-UI (рис. 2.3). Material-UI [11] - це популярна бібліотека інтерфейсу користувача (UI) для React, яка базується на дизайн-мові Google Material Design. Вона надає набір готових React компонентів та стилів, що відповідають концепціям Material Design [12].

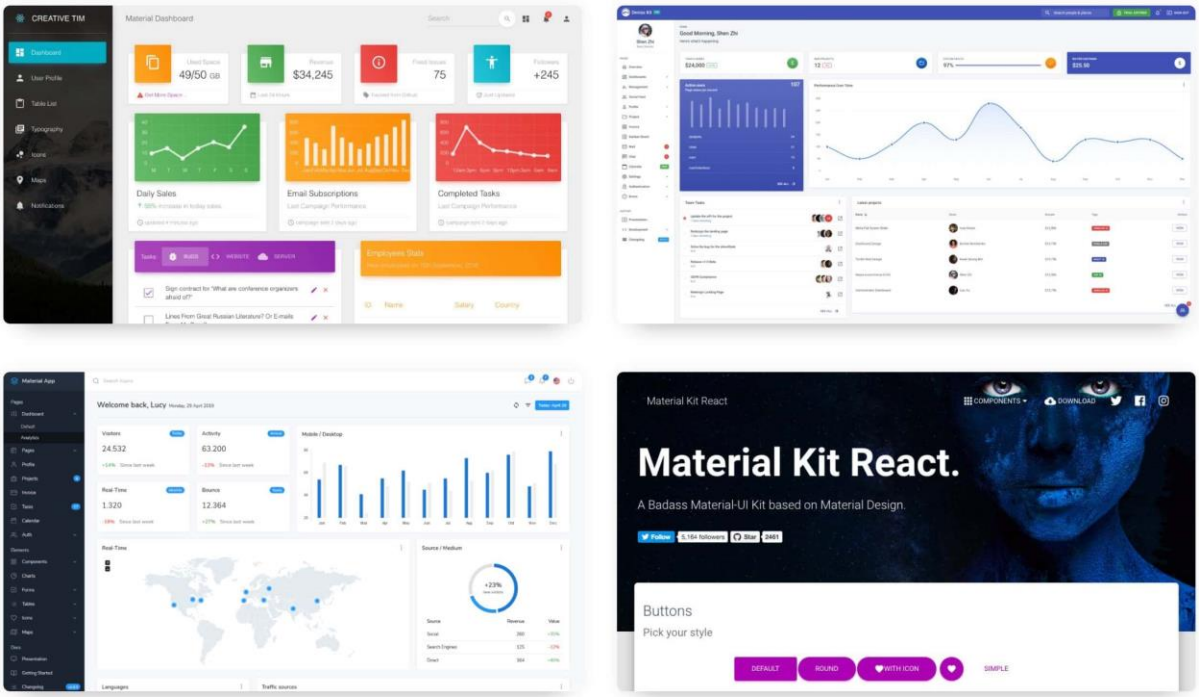

Рис. 2.3. Приклад користувальницького інтерфейсу створеного за допомогою Material-UI

Основними перевагами, що підштовхнули мене до використання цієї технології в даній роботі були:

– Material-UI постачається з великою кількістю готових компонентів, таких як кнопки, текстові поля, таблиці, карточки та інші, що полегшує створення стильного та функціонального інтерфейсу.
– бібліотека відповідає принципам Material Design, які включають в себе використання тіней, анімацій та кольорової палітри для створення зручного та привабливого дизайну.
– Material-UI дозволяє легко використовувати теми для керування виглядом та стилями компонентів, що робить простим зміну зовнішнього вигляду всього додатку.
– Матеріал-дизайн компоненти можна легко пристосовувати до різних екранів та пристроїв для забезпечення адаптивного дизайну.
– значна активна спільнота та регулярні оновлення роблять Material-UI надійним та підтримуваним інструментом для розробки.

– оскільки Material-UI побудована на React, вона ідеально інтегрується з React-проектами, дозволяючи вам легко використовувати компоненти та функції React.
– Material-UI можна використовувати не лише для веб-розробки, а й для створення інтерфейсів на платформах, таких як React Native для мобільних додатків.
– має вбудовану підтримку для значків Material Design Icons, що розширює можливості роботи з іконками.

Шукаючи метод ефективного позиціювання елементів в даній фінансовій інформаційній системі, було знайдено властивість grid-template-areas [13] (рис. 2.4).

Grid-template-areas - це властивість у CSS Grid Layout, яка дозволяє визначити макет сітки, задаючи іменовані області (areas), до яких потім можна вирізати елементи [14].

Рис. 2.4. Демонстрація роботи grid-template-areas

За допомогою даного підходу біло розділено сторінку на потрібні частини.

Рис. 2.5. Схема розбиття сторінки для ПК

При розширенні екрану користувача менше ніж 1200 пікселів схема розбиття сторінки матиме наступний вигляд:

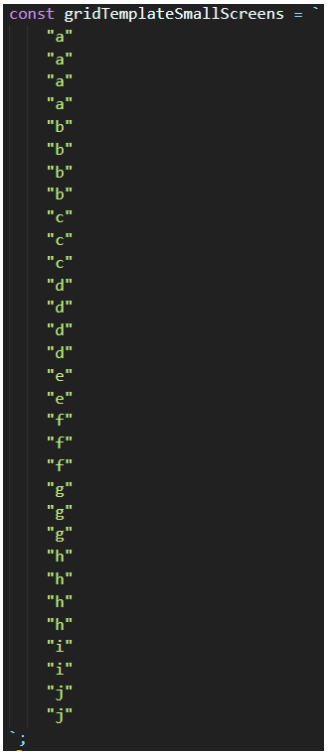

Рис. 2.6. Схема розбиття сторінки для пристроїв з розширення менше 1200 пікселів

Для написання серверної частини я обрав Express [15]. Express.js є фреймворком для розробки веб-додатків на мові програмування JavaScript, який базується на середовищі виконання Node.js. Серед переваг та можливостей Express.js я відмітив:

- Express.js пропонує простий та лаконічний синтаксис, який полегшує розробку веб-додатків. Він дозволяє швидко створювати маршрути, обробники та інші компоненти додатку.
- Express надає потужні можливості маршрутизації, що дозволяє вам легко визначати, як додаток відповідає на конкретні запити.
- система middleware в Express дозволяє вам виконувати різноманітні функції на кожному етапі обробки запиту. Це робить реалізацію функцій, таких як логування, аутентифікація та обробка помилок, більш ефективною.
- Express підтримує різноманітні шаблонізатори (наприклад, EJS, Pug), що дозволяє легко генерувати HTML на серверному боці.
- Express є легко розширюваним, і ви можете використовувати різні модулі та додатки для розширення його функціональності.
- Express чудово підтримує створення RESTful API, що робить його популярним в розробці мікросервісів та серверних додатків [16].

Серверну частину фінансової інформаційної системи було написано з використанням виконавчого середовища Node.js [17].

Node.js - це відкрите виконавче середовище, спроектоване для виконання JavaScript-коду на серверному боці. Його основна філософія базується на асинхронності та ефективному використанні ресурсів для побудови високопродуктивних та масштабованих додатків. Ось деякі унікальні аспекти [18] та особливості Node.js, що були виявлені в ході виконання кваліфікаційної роботи:

- Node.js використовує однопотоковий цикл подій, що дозволяє ефективно обробляти багато одночасних з'єднань без блокування потоків.
- однією з ключових переваг є використання двигуна V8 [19]., розробленого Google, що забезпечує високу швидкодію виконання JavaScript-коду.

– NPM є потужним інструментом для управління залежностями та пакетами. Велика кількість пакетів дозволяє розробникам легко використовувати готові рішення.
– вбудована підтримка асинхронності спрощує роботу з введенням/виведенням та запитами до бази даних, забезпечуючи високу ефективність.
– Node.js підтримує модульну структуру, дозволяючи розробникам створювати чистий та організований код. Розширюваність забезпечує можливість використовувати додаткові пакети та модулі.
– додатки, побудовані на Node.js, можуть легко реалізовувати зв'язок в режимі реального часу за допомогою технологій, таких як WebSocket.
– Node.js підтримується на різних операційних системах, що дозволяє розробникам створювати кросплатформенні додатки.
– існує багато фреймворків для розробки веб-додатків на Node.js, таких як Express.js, які пропонують різні рівні абстракції та можливості для розробників [20].

2.4 Опис структури системи та алгоритмів її функціонування

2.4.1 Структура і алгоритми системи

Структура фінансової інформаційної системи

Інформаційна система складається з наступних елементів:

– Головна сторінка з детальної інформацією про ключові показники маркетплейсу (рис. 2.7).
–

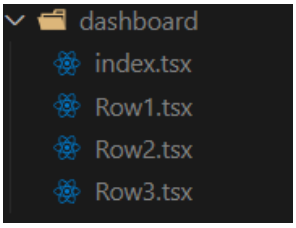

Рис. 2.7. Вміст головної сторінки

– Сторінка з прогнозами. На рис. 2.8 показано компоненти, що входять до складу сторінки.

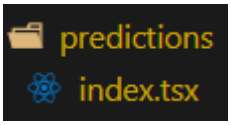

Рис. 2.8. Вміст сторінки з прогнозами

Алгоритм роботи фінансової інформаційної системи

В рамках виконання кваліфікаційної роботи була створена динамічна фінансова інформаційна система, яка володіє унікальними характеристиками та функціональністю. Основним інструментарієм для реалізації цього проекту став React - високоефективний та популярний бібліотечний інструмент для розробки інтерфейсів користувача[21].

Система маршрутизації нашої інформаційної системи налаштована за допомогою компоненту BrowserRouter, що є частиною бібліотеки React Router [22]. Цей підхід дозволяє створити динамічні та зручні маршрути для різних сторінок вашого веб-додатка. Маршрутизація складається з наступних сторінок:

```
<Routes>
    <Route path="/" element={<Dashboard />} />
    <Route path="/predictions" element={<Predictions />} />
</Routes>
```

2.4.2 Структура бази даних

В кваліфікаційній роботі було використано MongoDB. MongoDB, визнана як впливова документоорієнтована система керування базами даних, відмінюється від традиційних реляційних баз даних. Вона використовує нативний формат BSON (Binary JSON) для представлення даних, що дозволяє зберігати інформацію у вигляді динамічних, структурованих документів.

Основними сутностями є ключові показники, транзакції та продукти (рис.

2.9).

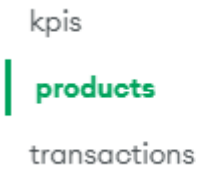

Рис. 2.9. Основні сутності в MongoDB

Таблиця KPIS (рис. 2.10) відповідає за зберігання інформації про ключові показники ефективності:

– totalProfit – загальний прибуток;
– totalRevenue – загальна виручка;
– totalExpenses – загальні витрати;
– expensesByCategory – витрати за категоріями;
– dailyData – щоденні дані;
– monthlyData – щомісячні дані.

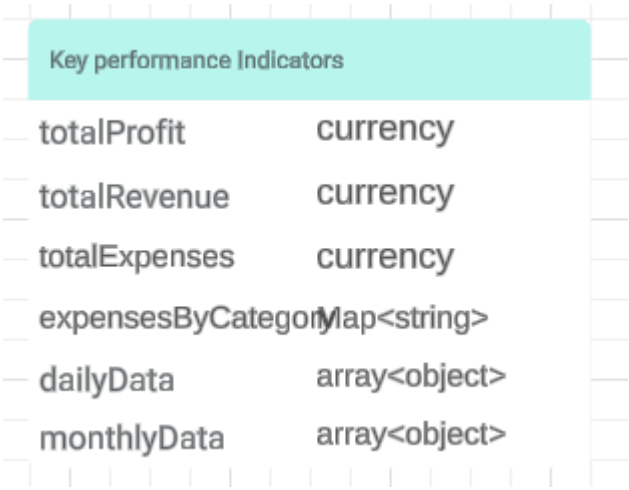

Рис. 2.10. ER-діаграма ключових показників

Таблиця products [23] (рис. 2.11) відповідає за зберігання інформації про продукти:

– id - первинний ключ, ідентифікатор товару;
– price – ціна;

– expense –витрати;
– transactions – транзакції;

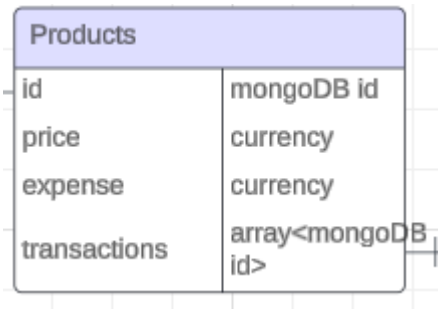

Рис. 2.11. ER-діаграма продуктів

Таблиця transactions (рис. 2.12) відповідає за зберігання інформації про транзакції:

– id - первинний ключ, ідентифікатор товару;
– buyer – покупець;
– amount – сума;
– productsID – id продукту в MongoDB;

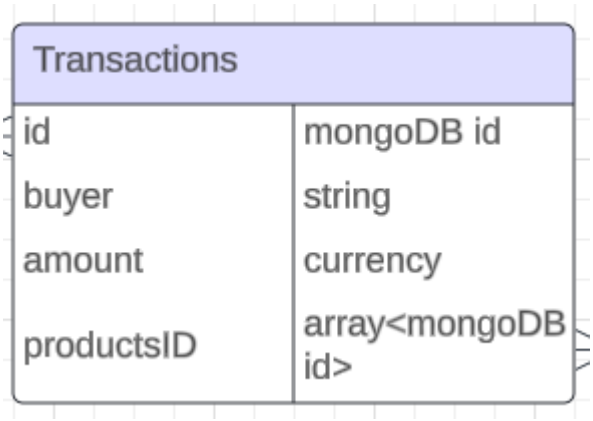

Рис. 2.12. ER-діаграма транзакцій

Нижче приведена загальна ER-діаграма інформаційної системи (рис. 2.13).

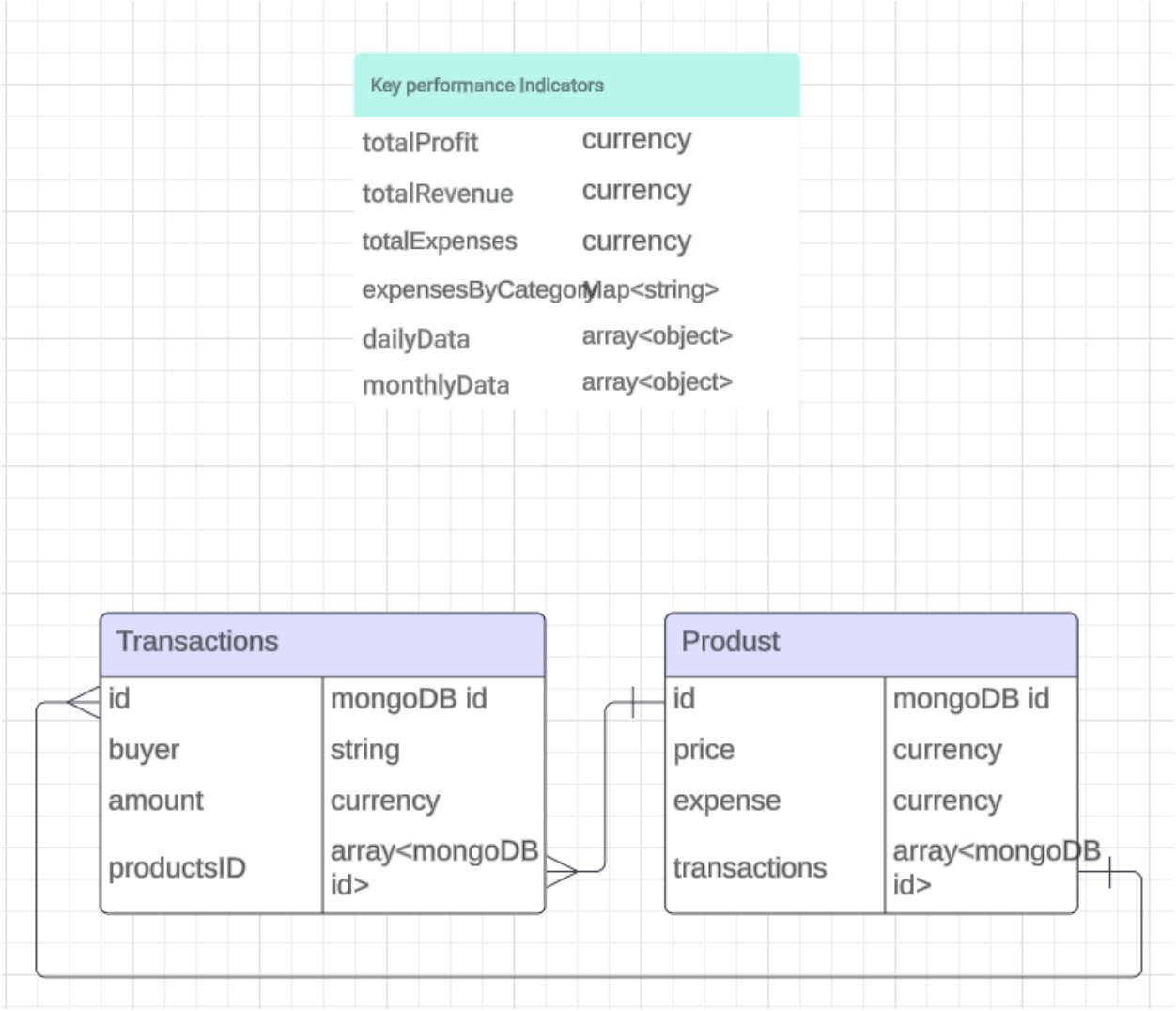

Рис. 2.13. Загальна ER-діаграма

2.5 Обґрунтування та організація вхідних та вихідних даних програми

Програмне забезпечення фінансової інформаційної системи отримує інформацію шляхом завантаження її з MongoDB системи (рис. 2.14).

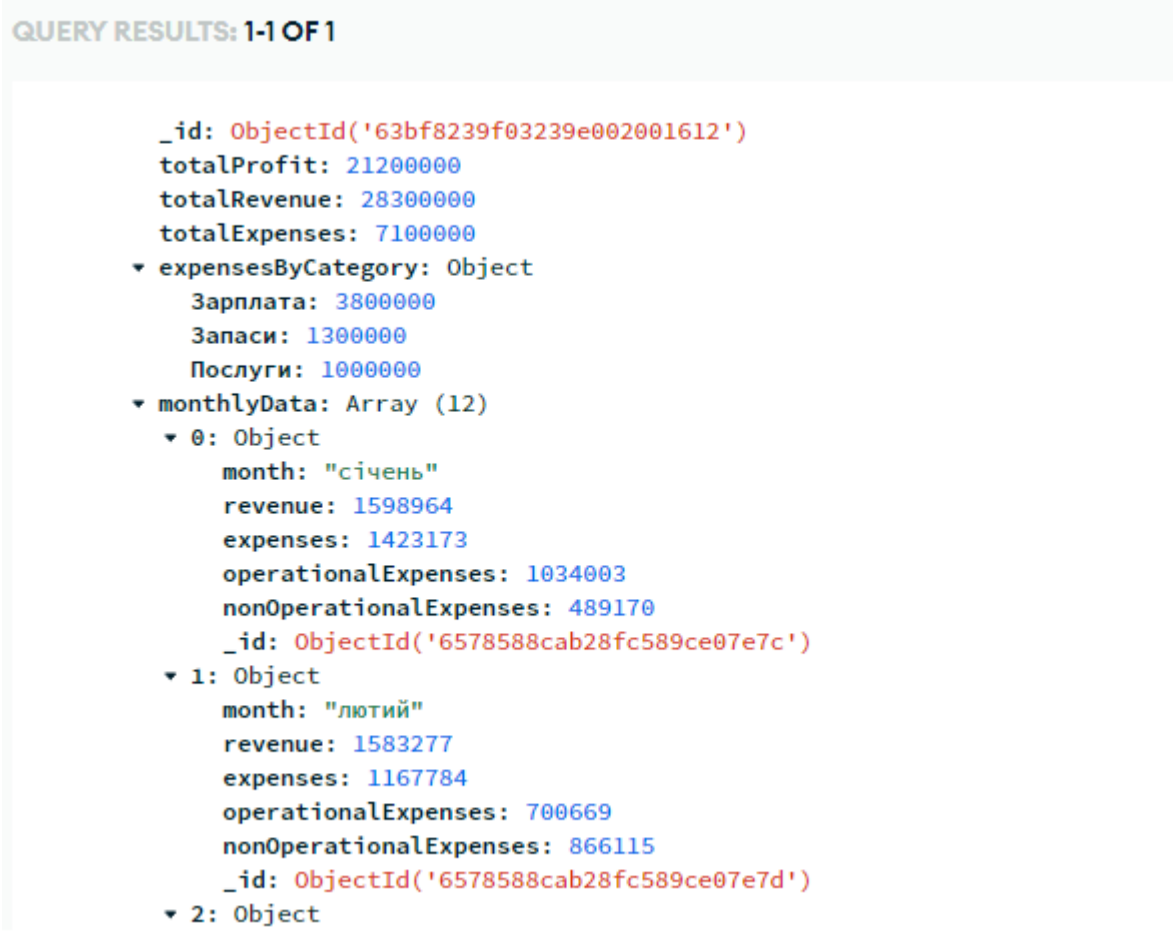

Рис. 2.14. Приклад зберігання даних в MongoDB

Щоб оновити дані в нашій колекції слід виконати ряд дій:

– підключившись до нашої бази даних, ми викликаємо метод dropDatabase(), що повністю очищує вміст нашої бази;
– далі методом insertMany() в колекцію, до якої нам потрібно додати дані, записуємо масив документів;

Такий підхід є корисним при ініціалізації бази даних з тестовими даними або при очищенні та оновленні бази даних в деяких сценаріях розробки. Нижче наведено лістинг коду даної операції.

```
Mongoose
.connect(process.env.MONGO_URL, {
})
.then(async () => {
    app.listen(PORT, () => console.log(`Server Port: ${PORT}`));
```

```
    await mongoose.connection.db.dropDatabase();
    KPI.insertMany(kpis);
    Product.insertMany(products);
    Transaction.insertMany(transactions);
})
```

На основі внесених, чи, якщо потрібно, отриманих даних з MongoDB можна проводи наступні операції для аналізу ефективності маркетингу.

2.6 Опис розробленої системи

Фінансову інформаційну систему було розроблено та протестовано на ПК з такими характеристиками та гарнітурою:

− процесор: Процесор Intel Core i7-11700KF 3.6 GHz / 16 MB;

− Материнська плата MSI MPG B550 Gaming Plus;

− модулі пам’яті: Kingston Fury DDR4-2666 16384 MB PC4-21300;

− SSD диск Kingston FURY Renegade 1TB M.2 2280 NVMe PCIe Gen 4.0 x4 3D TLC NAND;

− Блок живлення Gigabyte GP-UD1000GM;

− монітор: 27" Samsung Curved C27R500 Dark Silver;

− Комп’ютерна миша;

− Клавіатура.

Під час виконання даної кваліфікаційної роботи були використані наступні програмні продукти:

– Visual Studio Code;

– Recharts;

Visual Studio Code

Visual Studio Code [24] (рис. 2.15) - це не просто текстовий редактор, а потужний інструмент для розробки програмного забезпечення, який поєднує в собі простоту використання та вражаючий функціонал. Завдяки своїй легкості та гнучкості, VSCode завоював величезну популярність серед розробників усіх рівнів.

Ось кілька аспектів, які роблять VSCode неперевершеним інструментом для програмістів:

- VSCode підтримує широкий спектр мов програмування, починаючи від популярних, таких як JavaScript, Python і Java, до менш відомих. Розширення для різних мов і фреймворків дозволяють налаштовувати середовище розробки під конкретні потреби проекту.
- VSCode завантажується швидко і працює легко навіть на менш потужних комп'ютерах. Зручний інтерфейс, можливість розгортання зі скороченими клавішами та простота використання роблять його приємним для користування.
- VSCode базується на відкритому коді, що дозволяє розробникам вносити власні покращення та розширювати функціонал. Загальнодоступні розширення в бібліотеці розширень дозволяють розширювати функціонал для практично будь-якого випадку використання.
- вбудована підтримка Git дозволяє легко взаємодіяти з репозиторіями, відслідковувати зміни та виконувати коміти безпосередньо з інтерфейсу редактора.
- вбудовані засоби відлагодження дозволяють легко виправляти помилки та аналізувати виконання коду без використання зовнішніх інструментів.
- VSCode володіє потужними інструментами для підсвічування синтаксису та автоматичного завершення коду, що полегшує роботу з великими кодовими базами та зменшує ймовірність помилок.

VSCode — це не просто інструмент для написання коду, а повноцінна розробницька платформа, яка сприяє ефективній та приємній роботі розробників у будь-якому проекті [25].

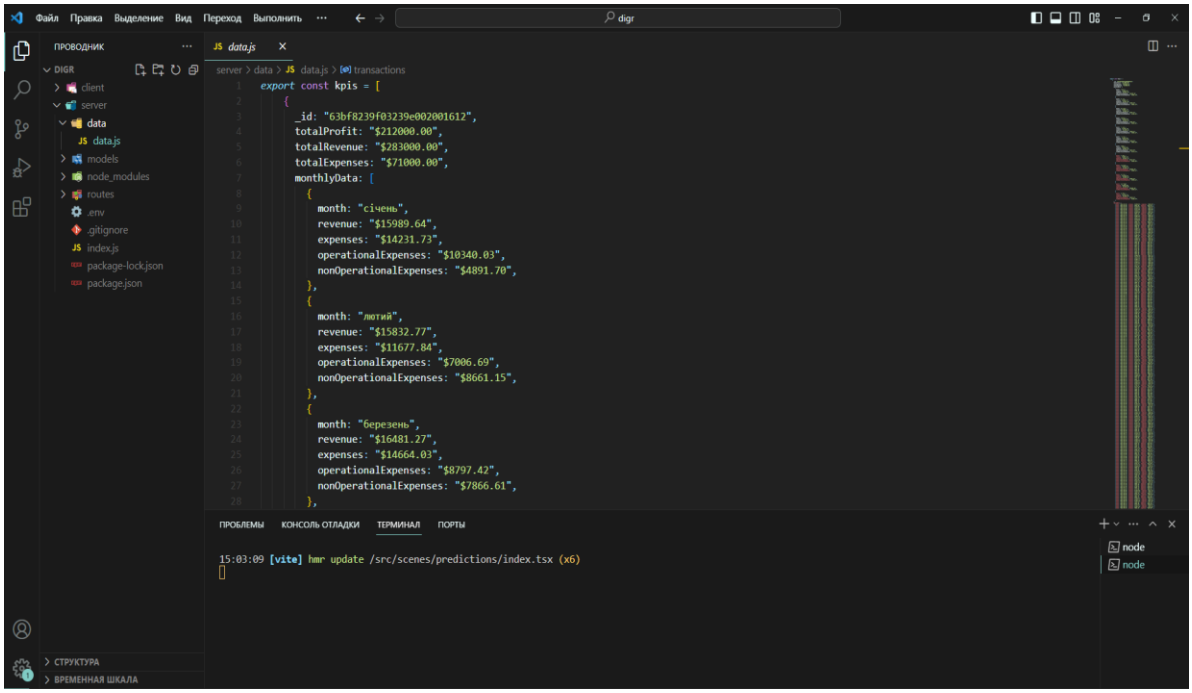

Рис. 2.15. Visual Studio Code

Recharts

Recharts відзначається своєрідністю в сфері візуалізації даних для розробників, що працюють з React. Ця бібліотека надає широкі можливості для створення динамічних та привабливих графіків у веб-додатках. Основними переваги при роботі над кваліфікаційною роботою стали:

– Recharts добре вписується в екосистему React, використовуючи компонентний підхід. Це дозволяє швидко і просто інтегрувати графіки та діаграми в React-проєкти без значних зусиль.
– бібліотека підтримує широкий спектр типів графіків, таких як лінійні, стовпчасті, кругові та інші. Це надає розробникам можливість вибрати найкращий тип графіка для конкретного сценарію візуалізації даних.
– Recharts дозволяє додавати анімації до графіків, що полегшує сприйняття та робить візуалізацію динамічною. Зміни у вхідних даних або параметрах можуть автоматично відображатися на графіку без необхідності перезавантаження сторінки.
– Recharts підтримує адаптивний дизайн, що робить його ідеальним вибором для розробки мобільних та планшетних додатків. Графіки

можуть пристосовуватися до різних розмірів екранів, забезпечуючи оптимальний вигляд на будь-якому пристрої

– Recharts пропонує зручний інтерфейс конфігурації, де багато опцій можна налаштовувати за допомогою простого API. Це дозволяє розробникам швидко змінювати вигляд та поведінку графіків.

2.6.1 Виклик та завантаження програми

Інформаційна система не призначена для широкого загального користування, а замість цього вона виступає як допоміжний інструмент для меркетплейсів. Головною метою цієї системи є забезпечення меркетплейсів зручним та ефективним засобом для відстеження їхньої статистики, обліку продажів та аналізу маркетингових тенденцій.

Меркетплейси, використовуючи цю інформаційну систему, отримують можливість глибокого аналізу своєї діяльності. Вони можуть спостерігати за динамікою продажів, визначати найбільш прибуткові товари чи послуги, а також вивчати ефективність своїх маркетингових кампаній.Ця система дозволяє здійснювати точні та об'єктивні оцінки різних аспектів діяльності меркетплейсів. Інтерфейс користувача розроблений таким чином, щоб надавати зручний доступ до ключової інформації, забезпечуючи зрозуміле та ефективне сприйняття статистичних даних.

Окрім того, ця інформаційна система може бути налаштована під індивідуальні потреби кожного меркетплейсу, надаючи можливість вибору та налаштування параметрів звітності, що сприяє визначенню конкретних потреб та стратегій для підвищення ефективності бізнесу. Такий підхід робить цю інформаційну систему необхідним інструментом для успішного управління та розвитку меркетплейсів.

З метою ефективного комерційного використання, маркетплейс повинен розглядати можливість використання серверного обладнання для зберігання та обробки великого обсягу даних. Це важливий крок, оскільки надійне зберігання і доступ до даних є ключовим елементом успішної функціонування маркетплейсу.

Для виклику інформаційної системи, розробленої для вашого маркетплейсу, необхідно встановити Node.js на основний комп'ютер підприємства. Node.js відповідає за запуск серверної частини додатка, забезпечуючи швидку та ефективну обробку запитів користувачів.

Також важливим елементом є створення та налаштування колекції в базі даних MongoDB. MongoDB - це документоорієнтована база даних, яка надає гнучкість у зберіганні та опрацюванні даних вашого маркетплейсу.

Після цього необхідно викликати серверну частину, що може бути здійснено через консоль, прописавши команду "npm run". Це запустить сервер Node.js на визначеному порту, в даному випадку - порті 1337, який передбачено в файлі .env (рис. 2.16).

Забезпечуючи правильне налаштування та запуск сервера, ви готові до обробки запитів і взаємодії із вашим маркетплейсом.

Рис. 2.16. Порт запуску сервера

Цей комплексний підхід до налаштування інфраструктури дозволяє вашому маркетплейсу працювати ефективно та забезпечує надійність обробки даних для ваших користувачів.

Для відкриття UI компоненти слід зробити те саме в клієнтській частині. Vite локально запуститься на порті 5173 (рис. 2.17).

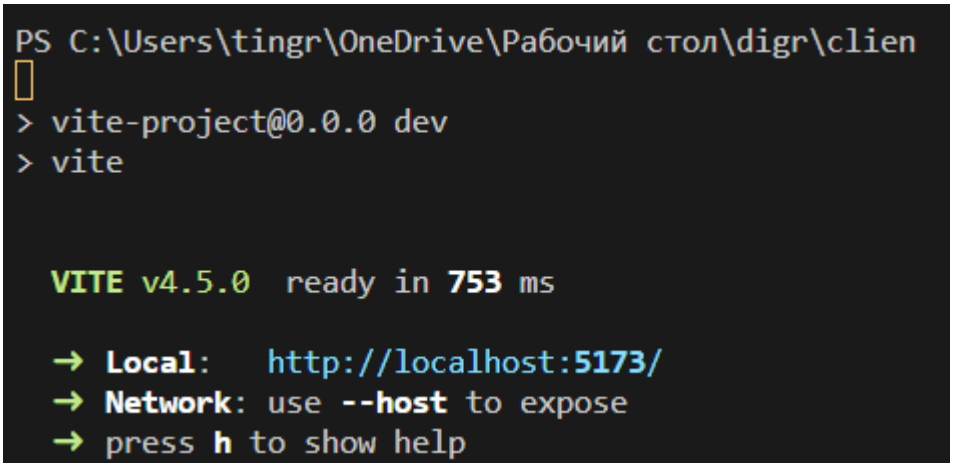

Рис. 2.17. Порт запуску клієнтської частини

При здобутті віддаленого доступу до інформаційної системи варто використовувати хост-сервери, що надають широкі можливості для забезпечення безперервного та ефективного функціонування. Процес встановлення та налаштування такого віддаленого з'єднання включає кілька ключових етапів:

- Початковим кроком є реєстрація домену для вашого сайту. Важливо обрати відповідний та легко запам'ятовуваний домен. Іншими словами, це адреса, за допомогою якої ваші користувачі будуть звертатися до вашого маркетплейсу.
- Вибір Хостинг-Постачальника:Оберіть надійного хостинг-постачальника, орієнтуючись на його функціональні можливості, рівень обслуговування та вартість послуг. Багато провайдерів також пропонують послуги реєстрації доменів, спрощуючи вам весь процес.
- Після придбання хостинг-послуг ви отримаєте доступ до адміністративної панелі. Це інтерфейс, який дозволяє керувати різними аспектами вашого хостинг-акаунту. Зазвичай це веб-інтерфейс, такий як cPanel або подібний.
- Налаштуйте з'єднання з базою даних через адміністративний інтерфейс хостингу. Це важливий крок, оскільки база даних відповідає за зберігання та управління інформацією, яка використовується вашою інформаційною системою.
- Встановіть DNS-записи для направлення домену на ваш хостинг-сервер. Це дозволяє користувачам звертатися до вашого маркетплейсу за

допомогою домену.

Після успішно зроблених кроків ви можете використовувати вашу інформаційну систему, просто вводячи дійсну адресу у вікно браузера. Це відкриє доступ до функціональності маркетплейсу та його можливостей для віддаленого користування.

Загальною метою цього процесу є створення надійного та ефективного середовища для взаємодії користувачів з вашим маркетплейсом незалежно від їхнього місця знаходження.

2.6.2 Опис інтерфейсу користувача

При переході на головну сторінку сайту, маркетолог чи будь-яка особа, яка займається аналітикою продажів, відкриє для себе важливий інтерфейс, який дозволяє глибоко аналізувати та контролювати ключові показники ефективності (рис. 2.18).

Перш за все, структурований та інтуїтивно зрозумілий дизайн головної сторінки створює зручне середовище для взаємодії. Маркетолог миттєво отримує доступ до основних аналітичних звітів, які відображають зведену інформацію про продажі, прибуток та інші стратегічно важливі метрики.

Контекстно-орієнтовані панелі та графіки надають можливість швидкого отримання уявлення про загальну динаміку. Наявність інтерактивних елементів дозволяє здійснювати глибокий аналіз конкретних періодів чи продуктових категорій.

Підтримка персоналізованих звітів та можливість налаштовувати панелі спостереження забезпечують індивідуальний підхід до відображення важливої інформації. Маркетолог може вибрати та організувати дані так, як це найбільше відповідає його потребам та стратегії.

Такий підхід до розміщення інформації на головній сторінці створює ефективне робоче середовище для аналізу продажів та прийняття обґрунтованих рішень в сфері маркетингу.

Рис. 2.18. Головна сторінка інформаційної системи

Головна сторінка сайту містить у своїй структурі логотип, панель приладів та самі графіки з аналітикою.

На візуальному зображенні, яке відображається на першому графіку (рис. 2.19), розкривається фінансова динаміка маркетплейсу протягом року. Цей графік включає в себе два основні аспекти - доходи та витрати.

Доходи, представлені на графіку, вказують на прибуток, отриманий маркетплейсом протягом кожного місяця упродовж року. Ця інформація надає чіткий огляд про те, як виручка розвивається від місяця до місяця, а також дозволяє виявити можливі тренди та підходи до стратегії прибутковості.

З іншого боку, витрати на графіку вказують на витрати, які маркетплейс поніс

протягом року. Ця інформація є важливою для аналізу ефективності управління ресурсами та визначення факторів, що впливають на фінансову стійкість.

Шляхом віднімання витрат від доходів кожного місяця отримуємо значення прибутку. Цей аналітичний підхід дозволяє точно визначити фінансовий результат кожного періоду та виявити ті місяці, коли прибуток був найбільшим чи, можливо, найменшим. Це корисна інформація для прийняття управлінських рішень та вдосконалення стратегій фінансового управління.

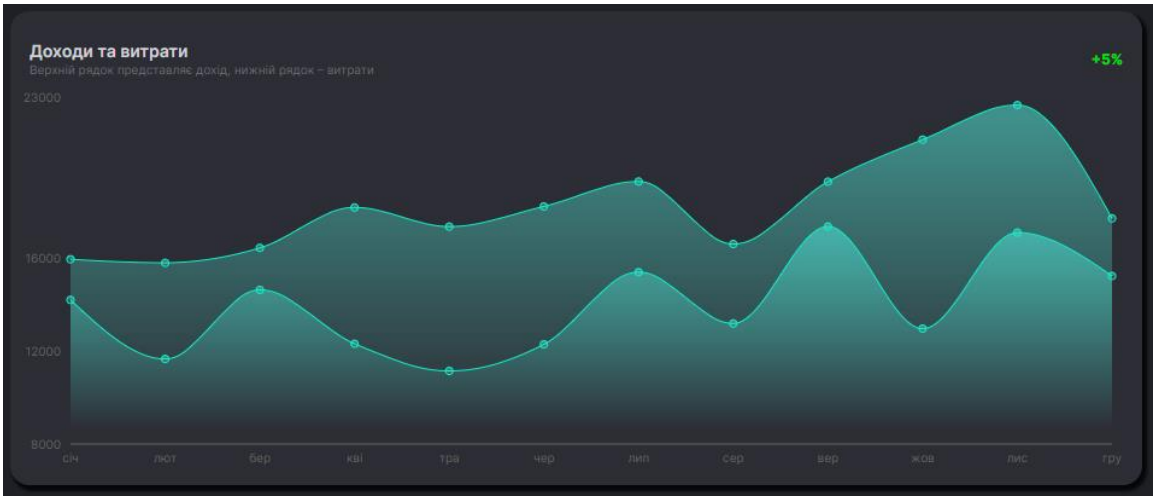

Рис. 2.19. Графік доходів та витрат

На наступному графіку ми можемо побачити прибутки та доходи компанії (рис. 2.20)

Рис. 2.20. Графік прибутку та доходу

Графік помісячних доходів виявляється ключовим інструментом у стратегічному прогнозуванні майбутніх продажів маркетплейсу (рис. 2.21). Цей важливий аналітичний інструмент не лише відображає історичні зміни у виручці протягом років, але також дозволяє здійснювати передбачення та визначати тенденції, які можуть впливати на майбутні доходи компанії.

Ключові аспекти графіка помісячних доходів:

– Графік надає можливість оцінити історичні тенденції і зміни в доходах. Це дозволяє виявити місяці або періоди, коли доходи були найвищими чи, навпаки, спостерігалися зниження.
– Аналізуючи графік помісячних доходів, можна ідентифікувати сезонні впливи, що сталися у певні періоди року. Це важливо для належного планування ресурсів та стратегій маркетингу.
– Графік служить основою для прогнозування майбутніх доходів та визначення потенційних виручок. Це дозволяє приймати обгрунтовані рішення стосовно запасів, маркетингових кампаній та інших стратегічних кроків.
– Аналізуючи графік, можна виявити тренди, які можуть бути корисні для пристосування бізнес-стратегії до змін на ринку. Виявлення позитивних або негативних тенденцій дозволяє підготувати відповідні заходи.

Рис. 2.21. Графік помісячних доходів

Власне вивчення операційних та неопераційних витрат визначає ключовий аспект в розробці стратегії стабільного та ефективного росту маркетплейсу. Цей

процес відіграє визначальну роль у формуванні бізнес-політики та дозволяє приймати обгрунтовані рішення для досягнення успіху та стійкості на ринку.

Аналіз операційних витрат дозволяє маркетплейсу розуміти, як ефективно використовуються ресурси в рамках щоденної діяльності. Це включає в себе витрати на робочу силу, технологічні рішення, логістику та інфраструктуру. Зрозуміння цих аспектів дозволяє оптимізувати операційні процеси, підвищувати продуктивність та знижувати витрати.

Неопераційні витрати включають аспекти, які не є безпосередньо пов'язаними з виробництвом або обслуговуванням товарів чи послуг. Це може включати витрати на маркетинг, дослідження та розвиток, адміністрування та інші стратегічні напрямки. Розуміння неопераційних витрат допомагає ефективно розподіляти ресурси для підтримки маркетингових ініціатив, вдосконалення продукту та побудови партнерських відносин.

Загальне вивчення обох видів витрат надає повний обсяг знань про фінансовий стан маркетплейсу, сприяє прийняттю стратегічних рішень для забезпечення ефективного та стійкого росту (рис. 2.22).

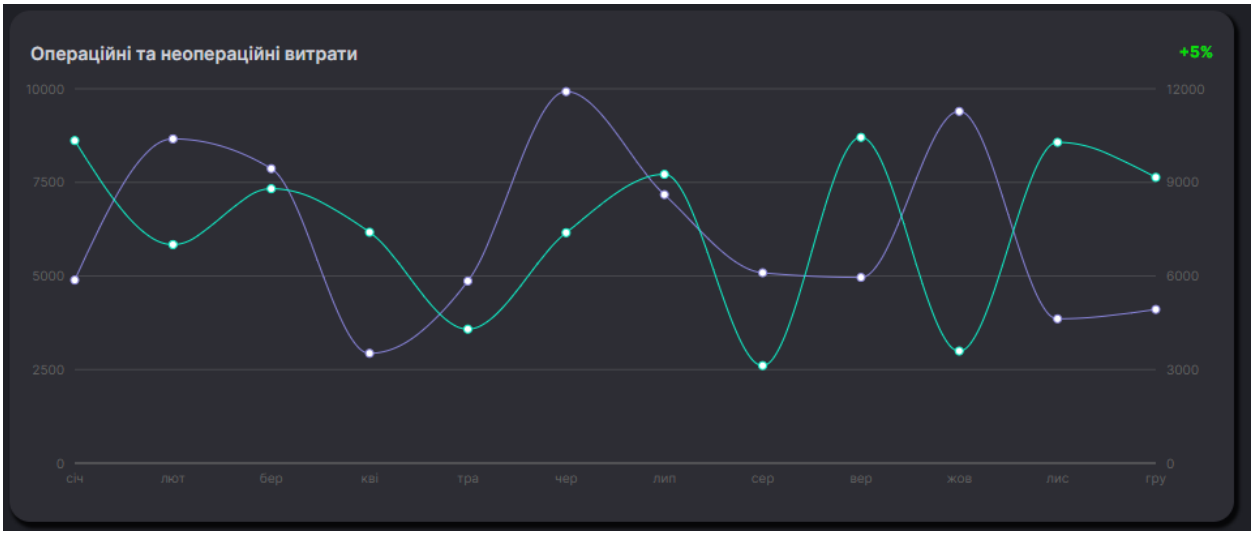

Рис. 2.22. Графік операційних та неопераційних витрат

Наступний можна побачити графік цін на товари проти їх собівартості (рис. 2.23).

Графік цін на товари проти витрат відіграє ключову роль у візуалізації та

аналізі взаємозв'язку між ціноутворенням на продукти та супутніми витратами. Цей інструмент надає чітку інсайтову перспективу на те, як цінова політика впливає на загальні фінансові показники бізнесу.

Основні аспекти графіка Ціни на товари проти витрат:

– графік дозволяє визначити маржинальність продуктів, порівнюючи ціну на товари з витратами на їх виробництво чи закупівлю. Визначення ефективної маржинальності є важливим для оптимізації прибутковості.
– аналізуючи графік, можна виявити, як зміни цін впливають на загальні витрати та, відповідно, на фінансовий стан бізнесу.
– графік слугує джерелом інформації для визначення ефективності використання різних стратегій ціноутворення та їх впливу на витрати.
– шляхом порівняння цін та витрат можна визначити товари чи послуги, які мають найвищу прибутковість. Це дозволяє бізнесу концентруватися на стратегіях, спрямованих на максимізацію прибутків.
– графік допомагає визначити, чи є цінова політика конкурентоспроможною, а також впливає на рішення щодо адаптації цін відносно ринкових тенденцій.

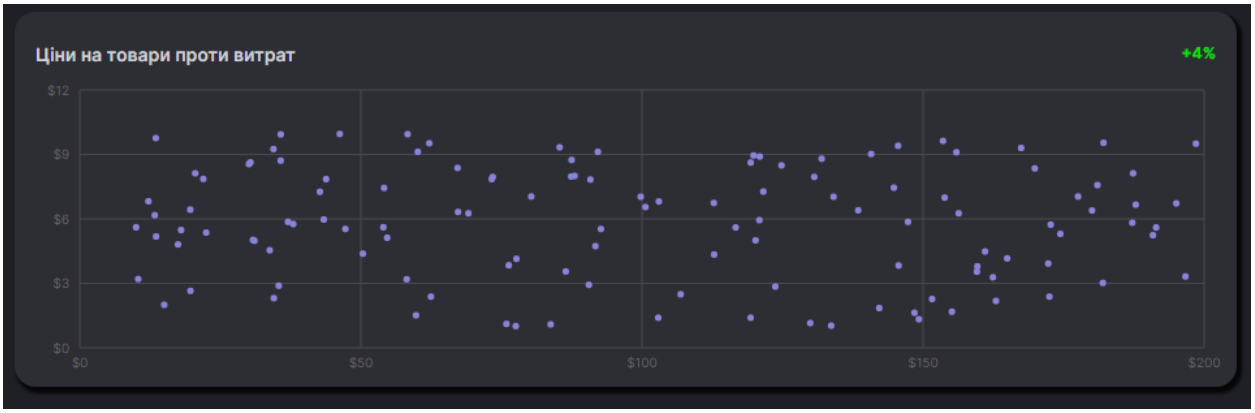

Рис. 2.23. Графік цін на товари проти їх витрат

Графік списку товарів (рис. 2.24) становить важливий засіб візуалізації та аналізу ключових характеристик продуктів. Його унікальність полягає у можливості відстеження ідентифікаторів товарів, вартості їх придбання та собівартості, а також надає можливість вивчення різноманітності асортименту продуктів компанії.

Основні аспекти та переваги графіка списку товарів:

– графік надає чітке представлення ідентифікаторів кожного товару, що спрощує внутрішнє відстеження та управління продуктами.
– за допомогою графіка можна порівнювати ціни продуктів з їх вартістю, що є ключовим фактором для оптимізації маржинальності.
– графік дозволяє вивчати різноманітність товарів, що пропонуються компанією, і виявляти найбільш та найменш прибуткові частини асортименту.
– на основі графіка можна приймати стратегічні рішення щодо розвитку асортименту, фокусуючись на тих товарах, які мають високий попит та прибутковість.
– графічне відображення дозволяє вчасно виявляти зміни в цінах та собівартості, допомагаючи в управлінні фінансовими показниками продуктів.

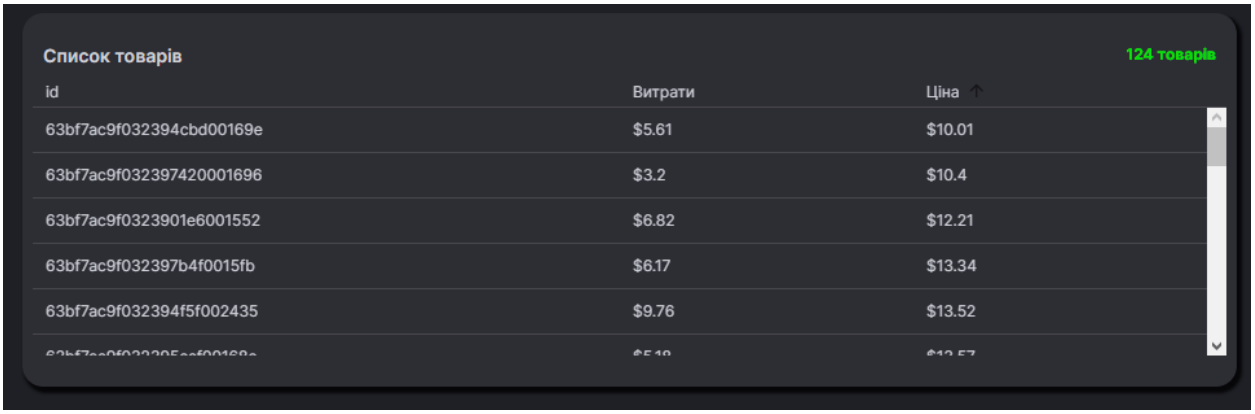

Рис. 2.24. Список товарів

Графік транзакцій (рис. 2.25) є інструментом, що відображає важливі аспекти взаємодії з покупцями, надаючи візуальне представлення унікальних ідентифікаторів транзакцій, імен клієнтів, сум продажу та кількості покупок кожного клієнта.

Основні характеристики та переваги графіка транзакцій:

– графік надає можливість однозначно відстежувати кожну транзакцію за її унікальним ідентифікатором, що сприяє систематизації та безпеці обліку.
– включення імен клієнтів в графік дозволяє здійснювати аналіз покупців,

їх звичок та передпочень, що може бути корисним при формуванні стратегій продажів.

– графічне представлення сумарної вартості та кількості покупок кожного клієнта дозволяє визначити ключові показники їхнього внеску у загальний обсяг продажів.
– графік сприяє вивченню зв'язків між сумою покупок та кількістю транзакцій, допомагаючи визначити поведінкові та фінансові тенденції клієнтів.
– аналізуючи графік транзакцій, бізнес може оптимізувати обслуговування клієнтів, враховуючи їхню активність та вартість для компанії.

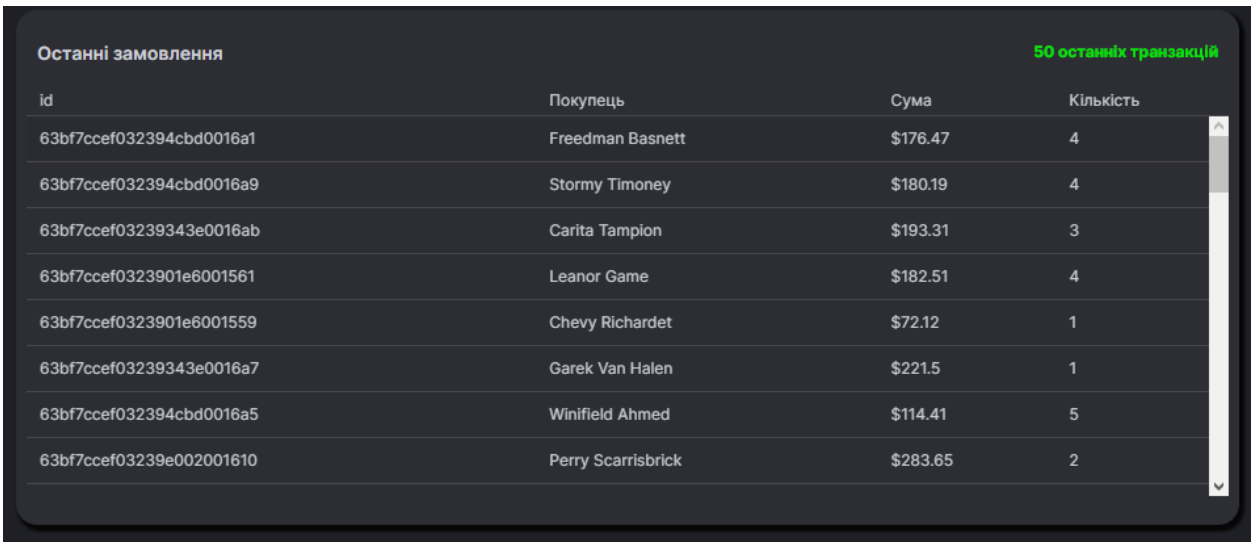

Рис. 2.25. Список транзакцій

РОЗДІЛ 3
ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ ПРОГНОЗУВАННЯ

3.1 Метод прогнозування лінійною регресією

В ході виконання кваліфікаційної роботи було обрано метод лінійної регресії для прогнозування майбутніх доходів на основі наявних даних маркетплейсу з метою впровадження ефективної стратегії фінансового управління та розвитку (рис. 3.1.). Лінійна регресія дозволяє побудувати математичну модель, яка відображає лінійну залежність між різними факторами та доходами, що дозволяє здійснювати прогнози на майбутнє.

Використання лінійної регресії для прогнозування доходів маркетплейсу наділяє нашу аналітичну стратегію рядом ключових переваг:

- лінійна регресія вирізняється своєю простотою та зрозумілістю, що дозволяє ефективно використовувати її, навіть не володіючи глибокими знаннями в математиці.
- кожен коефіцієнт у рівнянні лінійної регресії несе конкретне значення, що спрощує інтерпретацію впливу кожної змінної на дохід маркетплейсу.
- лінійна регресія виявляється ефективною на перших етапах аналізу, коли швидко необхідно отримати загальне уявлення про ситуацію та розвиток подій.
- модель лінійної регресії може легко розширюватись для врахування додаткових факторів, що дозволяє адаптувати її до змінюючихся умов ринку та бізнес-середовища.
- за допомогою отриманої моделі можна чітко визначити напрямок впливу різних факторів на дохід, що надає підставу для обґрунтованих та стратегічно обраної управлінської політики.

Обираючи лінійну регресію, ми зосереджуємося на її потужності для точного та практичного прогнозування доходів маркетплейсу, забезпечуючи стабільний фінансовий розвиток та досягаючи стратегічних цілей у сучасному бізнес-середовищі.

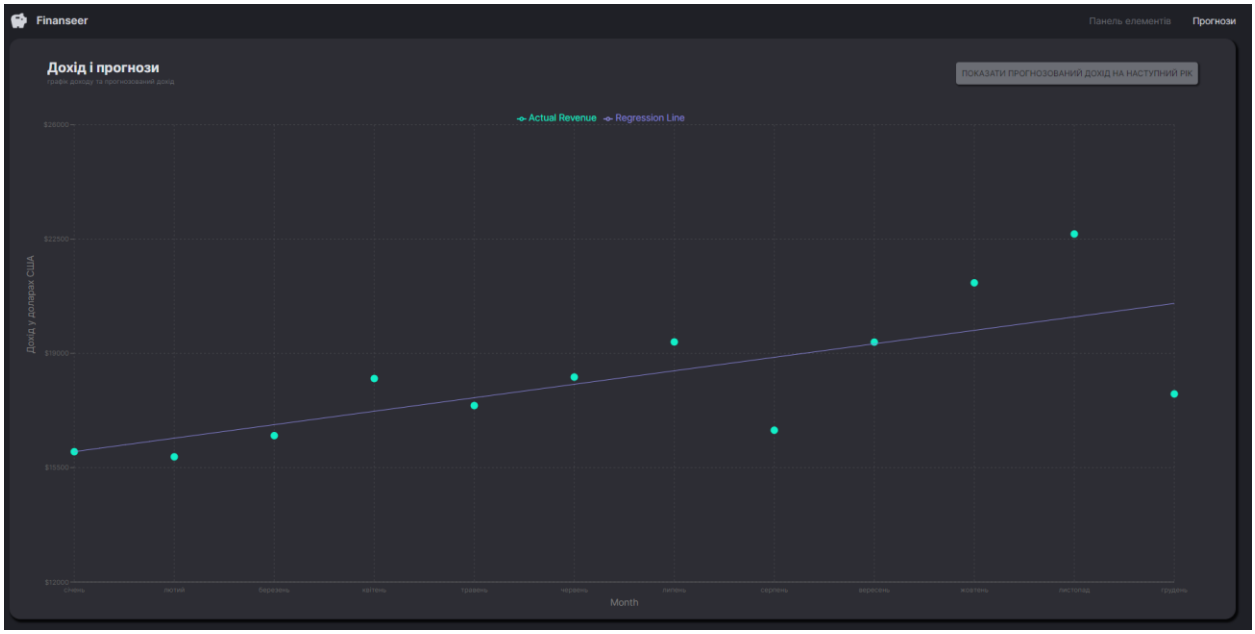

Рис. 3.1. Метод лінійної регресії

Лінійна регресія спрямована на знаходження лінії, яка на найкращий спосіб відповідає набору точок. Отже, розв'язок лінійної регресії визначає значення для m і b так, що f(x) найкращим чином наближається до y.

У спробах лінійної регресії ми досліджуємо декілька випадкових кандидатів, щоб знайти оптимальні значення m і b. Цей метод дозволяє нам наблизити лінію до набору точок, максимально враховуючи їхню взаємодію.

Враховуючи складність завдання, лінійна регресія виявляється потужним інструментом для прогнозування та аналізу взаємозв'язків у даних.

3.2 Методи дослідження ефективності лінійної регресії

Існує безліч методів визначення ефективності роботи лінійної регресії. Найбільш розповсюдженими є:

– Коефіцієнт детермінації (R-squared);
– Середньоквадратична помилка (MSE) і Середньоабсолютна помилка (MAE);
– F-статистика;
– Стандартна помилка коефіцієнтів;

– Діагностика викидів;
– Критерій Акаіке (AIC) та Критерій Шварца (BIC).

Коефіцієнт детермінації (R-squared)

Коефіцієнт детермінації, позначений як R-squared або R^2, є метрикою, яка вимірює відсоток варіації залежної змінної, яку може пояснити модель регресії. Його значення лежить в діапазоні від 0 до 1, де 0 вказує на те, що модель не пояснює жодної варіації, а 1 вказує на повну відповідність між моделлю та даними.

R-squared обчислюється як квадрат кореляційного коефіцієнта між фактичними та прогнозованими значеннями.

Формула виглядає так:

$$R^2=(Cor * (X,Y))^2 \quad (3.1)$$

Переваги:

– простота та Зрозумілість, тобто R-squared є легким у використанні та зрозумілим для широкого кола користувачів без глибоких математичних знань.
– значення R-squared вказує, наскільки точно модель відображає варіації в даних. Чим вище R-squared, тим краще модель підходить до даних.

Недоліки:

– R-squared зростає із додаванням нових змінних до моделі, навіть якщо ці змінні не мають суттєвого впливу. Це може призвести до переоцінки ефективності моделі.
– R-squared не надає інформації щодо статистичної значущості коефіцієнтів моделі.
– модель може мати високий R-squared, але бути чутливою до викидів, що погіршує її прогностичні можливості.

Хоча R-squared є важливим показником для оцінки ефективності моделі регресії, важливо використовувати його разом із іншими метриками та проводити додатковий аналіз для комплексного оцінювання моделі.

Середньоквадратична помилка (MSE) і Середньоабсолютна помилка (MAE)

MSE є метрикою, яка вимірює середньоквадратичне значення квадратів відхилень між фактичними та прогнозованими значеннями. Вона використовується для кількісної оцінки точності моделі.

Середньоквадратична помилка (MSE) є метрикою, що використовується для вимірювання точності моделі прогнозування у контексті лінійної регресії. Ця метрика враховує квадратичні відхилення між фактичними і прогнозованими значеннями, покладаючи більший ваговий коефіцієнт на більші відхилення.

Щоб обчислити MSE, для кожного набору фактичних y_i та відповідних прогнозованих $y\hat{}_i$ значень виконуються такі кроки:

– розрахунок різниці між фактичним і прогнозованим значеннями.
– піднесення цих різниць до квадрата.
– обчислення середньої арифметичної для всіх отриманих квадратичних відхилень.

MSE виражається формулою:

$$MSE=1/n\sum^{n}_{i=1}(y_i - y_i)^2 \quad (3.2)$$

де:

n - кількість спостережень (кількість пар фактичних і прогнозованих значень).

y_i- фактичне значення залежної змінної для і-го спостереження.

$y\hat{}_i$- прогнозоване значення залежної змінної для і-го спостереження.

$\sum$ - сума значень виразу для кожногоі від 1 до n.

Переваги:

– Більші помилки мають великий вплив на значення MSE, що дозволяє виявляти важливі аномалії.

Недоліки:

– Великі відхилення мають значний вплив, що робить MSE чутливим до викидів.
– Оскільки MSE виражається в квадратних одиницях, важко є інтерпретувати його значення в контексті оригінальної шкали даних.

Методика розрахунку MSE надає можливість оцінювати точність прогнозів моделі лінійної регресії, враховуючи квадратичні відхилення і покращуючи якість моделі з плином часу.

Середньоабсолютна помилка (MAE) є метрикою, використовуваною для вимірювання точності моделі прогнозування, зокрема, в контексті лінійної регресії.

MAE враховує абсолютні відхилення між фактичними і прогнозованими значеннями, що дозволяє визначити середню величину помилок без врахування їхнього напрямку.

Формула для розрахунку MAE виглядає наступним чином:

$$MAE=n/1\sum_{i=1}^{n}|y_i-y^\wedge_i| \quad (3.3)$$

де:

n - кількість спостережень,

y_i- фактичне значення,

$y^\wedge_i$- прогнозоване значення.

Переваги:

– MAE не залежить від знаку помилок, що робить його корисним для визначення середньої величини відхилень незалежно від їхнього напрямку.
– MAE менше чутливий до великих відхилень або викидів, оскільки не використовує квадратичні значення помилок.

Недоліки:

– так як MAE враховує абсолютні значення помилок, великі помилки можуть мати менший вплив на метрику порівняно з MSE.
– MAE не нормалізує значення помилок, тому важко порівнювати

результати для різних моделей на різних шкалах.

MAE надає можливість оцінити середню абсолютну величину відхилень між фактичними і прогнозованими значеннями, дозволяючи здійснити кількісну оцінку точності моделі лінійної регресії без врахування напрямку помилок.

F-статистика

F-статистика в лінійній регресії є важливим інструментом для оцінки відповідності моделі даним. Вона визначає, наскільки значущо впливає включення змінних у модель, тобто, чи вони дійсно додають інформацію та покращують її адекватність.

За допомогою F-статистики можна провести тест на значущість моделі в цілому. Значення F-статистики порівнює дисперсію поясненої змінної моделі з дисперсією непоясненої частини. Чим вище значення F-статистики, тим більше підстав вважати модель значущою.

Формула F-статистики визначається відношенням середньої квадратичної суми залишкових (непояснених) величин до середньої квадратичної суми регресії (поясненої моделлю). Це важливе відношення, яке враховує якість пояснення моделлю варіації у вхідних змінних порівняно з варіацією, яка залишається непоясненою.

Формула для розрахунку F-статистики виглядає наступним чином

$$F=SSR/k/SSE/(n-k-1) \quad (3.4)$$

де:

SSR - сума квадратів регресії, яка вимірює відхилення прогнозованих значень від середнього;

k - кількість регресорів (змінних);

SSE - залишкова сума квадратів, яка вимірює відхилення спостережень від прогнозованих значень;

n - загальна кількість спостережень.

Що стосується переваг та недоліків, F-статистика надає можливість

визначити статистичну значущість моделі в цілому та її адекватність. Однак важливо пам'ятати, що вона може бути чутливою до розміру вибірки, і для великих вибірок може виявитися статистично значущою, навіть якщо ефекти невеликі.

Також, F-статистика повинна використовуватися в контексті інших статистичних показників для повноцінної оцінки моделі лінійної регресії. Це важливий інструмент для того, щоб переконатися в обґрунтованості та ефективності моделі.

Стандартна помилка коефіцієнтів

Стандартна помилка коефіцієнтів визначає, наскільки точно оцінки коефіцієнтів регресії в лінійній моделі відображають реальні відносини між змінними. Це важливий параметр, який вказує на розкид фактичних значень залежної змінної навколо регресійної прямої.

Формула для обчислення стандартної помилки коефіцієнта має вигляд:

$$SE(\hat{\beta}_j) = \frac{\sqrt{MSE}}{SST_j*(1-R_j^2)} \quad (3.5)$$

де:

$SE(\hat{\beta}_j)$ - стандартна помилка оцінки коефіцієнта β^j,

MSE - середньоквадратична помилка моделі,

SST_j- загальна сума квадратів відхилень xj від їх середнього значення,

R_j^2- коефіцієнт детермінації, який вимірює відсоток варіації відхилень у від її середнього значення, який пояснюється моделлю.

Переваги стандартної помилки коефіцієнтів полягають у тому, що вона надає інформацію про стабільність та достовірність оцінок коефіцієнтів. Однак слід враховувати, що цей підхід передбачає нормальний розподіл помилок та лінійні та адитивні взаємозв'язки між змінними.

Недоліком є чутливість до викидів та ненормальності в розподілі помилок, що може впливати на точність її оцінки. Також важливо пам'ятати, що стандартна помилка коефіцієнтів вимірює тільки точність оцінювання, а не самого коефіцієнта.

Діагностика викидів

Діагностика викидів у лінійних регресійних моделях грає ключову роль в забезпеченні точності та надійності результатів аналізу. Нижче представлено деякі методи та переваги виявлення викидів:

- використання цього графіку дозволяє вам визначити, як залишкові значення відхиляються від прогнозованих. Зрозуміло, що точки, які значно відхиляються від нуля, можуть свідчити про викиди.
- перевірка нормальності залишкових значень може бути важливою. Аномальне розподілення може свідчити про наявність викидів.
- оцінка впливу кожного спостереження на регресійну пряму. Значення з великим leverage можуть вказувати на наявність викидів, які мають великий вплив на модель.
- використовується для виявлення викидів, особливо тих, які мають значний вплив на параметри регресії.
- проведення тестів на нормальність та стійкість дисперсії залишкових значень, що може допомогти виявити аномалії в їх розподілі.

Переваги виявлення викидів полягають в здатності до уточнення регресійної моделі, поліпшенні точності та надійності прогнозів. Однак важливо розуміти, що жоден метод не є універсальним, і їх слід використовувати комплексно для досягнення найкращих результатів.

Критерій Акаіке (AIC) та Критерій Шварца (BIC).

Критерій Акаїке (AIC) та Критерій Шварца (BIC) — це інформаційні критерії, які використовуються для оцінки якості моделей. Вони базуються на понятті інформаційної втрати та враховують баланс між точністю моделі та її складністю.

Критерій Акаіке (AIC):

- AIC визначається як сума двох компонентів - кількості параметрів моделі та коефіцієнта втрати інформації. Метою є мінімізація AIC.
- AIC штрафує складні моделі, але дозволяє трошки більше гнучкості, ніж BIC.

– Зазвичай використовується в ситуаціях, де кількість спостережень велика порівняно з кількістю параметрів.

Критерій Шварца (BIC):

– BIC також враховує кількість параметрів, але штраф більш суворий, ніж у AIC. Метою є мінімізація BIC.
– BIC штрафує складні моделі сильніше, що може призводити до більш консервативного вибору моделі.
– Використовується в ситуаціях, де передбачається, що модель повинна бути простішою, або коли кількість спостережень обмежена.

Обидва критерії є інструментами для вибору моделей, але їхній вибір може залежати від конкретного контексту дослідження та припущень. Важливо розуміти, що жоден критерій не є універсальним, і вибір між AIC та BIC може визначатися конкретною задачею та характеристиками даних.

3.3 Визначення ефективності прогнозу

Для визначення ефективності прогнозу спершу слід знайти коефіцієнт детермінації.

Формула для R-squared виглядає так:

$$R^2 = 1 - \frac{SSE}{SST} \tag{3.6}$$

Знаходимо SSE, тобто залишкову суму квадратів для кожного спостереження, шляхом віднімання фактичного значення від прогнозованого (рис 3.1).

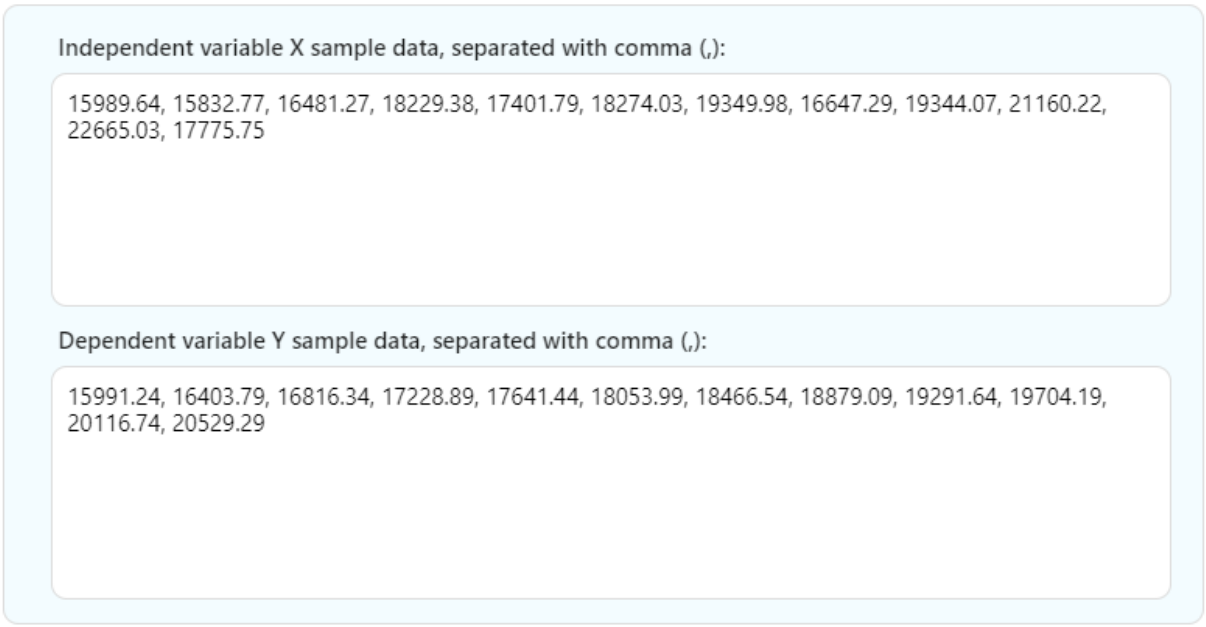

Рис. 3.1. Список фактичних та прогнозованих значень

Сума всіх квадратів значень із таблиці для осі X визначається як:

$$SS_{XX} = \sum_{i=1}^{n} X_i^2 - \frac{1}{n}(\sum_{i=1}^{n} X_i)^2 \quad (3.7)$$

Підставивши значеня в формулу отримаємо:

$$4050233034.766 - \frac{1}{12}(219151.22)^2 = 47962000,$$

Сума всіх квадратів значень із таблиці для осі Y визначається як:

$$SS_{YY} = \sum_{i=1}^{n} Y_i^2 - \frac{1}{n}(\sum_{i=1}^{n} Y_i)^2 \quad (3.8)$$

Підставивши значеня в формулу отримаємо:

$$4025585577.3002 - \frac{1}{12}(219123.18)^2 = 24338000,$$

Загальна сума квадратів для обох осей має вигляд:

$$SS_{XY} = \sum_{i=1}^{n} X_i Y_i - \frac{1}{n}(\sum_{i=1}^{n} X_i)(\sum_{i=1}^{n} Y_i) \quad (3.9)$$

Підставивши значеня в формулу отримаємо:

$$4026156754.8008 - \frac{1}{12}(219151.22)(219123.18) = 24397000,$$

Обчислимо нахил прямої за формулою:

$$\hat{\beta}_1 = \frac{SS_{XY}}{SS_{XX}} \tag{3.10}$$

Підставивши значеня в формулу отримаємо нахил прямої:

$$\frac{24397000}{4796000} = 0.50869,$$

Точки перетину обчислимо за формулою

$$\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 * \bar{X} \tag{3.11}$$

Підставивши значеня в формулу отримаємо точки перетину:

$$18260 - 0.50869 * 18263 = 8970.3,$$

Загальна сума квадрата дорівнює:

$$SS_{Total} = SS_{YY} = 24558000 \tag{3.12}$$

Крім того, регресійна сума квадратів обчислюється як:

$$SS_{Error} = SS_{Total} - SS_{Regression} \tag{3.13}$$

Підставивши значеня в формулу отримаємо:

$$24338000 - 12411000 = 11928000$$

Отже SSE для нашої лінійної регресії дорівнює 11928000, загальна ж сума

квадратів дорівнює 24558000.

Підставившии результати в формулу коефіцієнта детермінації отримаємо:

$$R^2 = 1 - \frac{11928000}{24558000} = 1 - 0485 = 0.515$$

Отже, коефіцієнт детермінації R^2 дорівнює 0.515 або 51.5%.

Оцінка того, чи є 51.5% задовільним коефіцієнтом детермінації для інформаційної системи прогнозування доходів підприємства потребує уважного врахування специфіки завдання та обставин аналізу. Коефіцієнт детермінації визначає відсоток варіації в залежній змінній (у нашому випадку, доходах) за допомогою розглядуваних змінних (прогнозованих факторів).

На перший погляд, 51.5% може здатися середнім значенням, проте в контексті прогнозування доходів, особливо в умовах невизначеності та впливу багатьох факторів, такий результат може бути прийнятним. Прогнозування фінансових показників завдяки ряду факторів може бути важливим в управлінні підприємством, і навіть часткове узгодження з фактичними результатами може допомогти в прийнятті обґрунтованих стратегічних рішень.

Звісно, ефективність моделі не обмежується лише одним числовим значенням, і важливо враховувати специфіку бізнес-середовища, мету прогнозування, а також інші методи та показники. В разі необхідності можна використовувати додаткові моделі чи оптимізувати існуючу для отримання кращих результатів.

З даного числа коефіцієнта детермінації слід зазначити, що маркетологам та аналітикам слід більш детально розробляти стратегії продажів аби униктути великих розбіжностей в щомісячних показниках.

Коефіцієнт детермінації на рівні 51.5% свідчить про те, що більшість варіації в доходах підприємства може бути пояснена використаною моделлю лінійної регресії. У даному контексті, маркетологам та аналітикам важливо брати до уваги цей результат при розробці стратегій продажів.

З огляду на те, що понад половина зміни в доходах пов'язана з урахованими факторами, можливо виявляти та розуміти ключові аспекти, що впливають на

фінансові результати. Такий рівень пояснення дозволяє зробити висновки про те, як різноманітні фактори впливають на вибрані економічні показники.

Згідно з цим, маркетологи та аналітики можуть використовувати цей коефіцієнт детермінації для того, щоб більш ефективно розробляти стратегії продажів та уникати значних розбіжностей у щомісячних показниках. Важливо не лише визначити впливові фактори, але й взяти до уваги їхню динаміку та змінюваність для більш точного прогнозування та оптимізації продажів на підприємстві.

ВИСНОВКИ

Мета цієї кваліфікаційної роботи полягала у створенні та дослідженні фінансової інформаційної системи, яка відповідала б сучасним вимогам та розвитку ринку маркетплейсів.

Актуальність цієї теми обумовлена стрімким ростом кількості маркетплейсів та необхідністю ефективної обробки та аналізу статистики продажів.У рамках розробки цієї інформаційної системи був створений інтуїтивний інтерфейс, який дозволяє зручно та швидко працювати з різноманітними фінансовими даними. Особливий акцент було зроблено на обробці актуальних даних, використанні метрик та графіків для зручного аналізу та виведення інсайтів.

Що стосується технічної реалізації, інформаційна система була розроблена за допомогою найактуальніших та популярних інструментів розробки. Це гарантує довгострокове обслуговування та можливість масштабування та вдосконалення системи для вирішення конкретних завдань у майбутньому.

Отже, розроблена інформаційна система стане надійним інструментом для ефективного управління фінансовими даними маркетплейсу, забезпечуючзручність та точність аналізу, а також готовність пристосовуватися до зростаючих потреб та вимог сучасного бізнесу.

Розроблена система має важливе практичне значення, оскільки вона спрощує процес прогнозування доходів маркетплейсу на майбутній рік на основі наявних даних. Вона стає невід'ємним інструментом для маркетологів та аналітиків, надаючи їм засоби для ефективного планування та визначення стратегій.

У ході розробки цієї кваліфікаційної роботи були вирішені численні завдання. Була проведена глибока аналітика предметної галузі, що дозволило врахувати ключові аспекти при розробці системи. Створено інтуїтивний інтерфейс користувача для зручності роботи з даними та взаємодії з системою.

Додатково, була налагоджена база даних, спроектована для ефективного зберігання та обробки інформації. Реалізовано метод прогнозування доходів на майбутній рік, який є ключовим елементом системи та визначає її основну

функціональність.

Під час розробки програмного продукту було проведено важливі обчислення та аналіз. Зокрема, було виявлено коефіцієнт детермінації, який надає маркетологам інформацію про ефективність їхніх рішень та служить важливим критерієм для розвитку та оптимізації маркетплейсу.

В цілому, розробка та впровадження цієї інформаційної системи вносять суттєвий внесок у практичний аспект управління фінансовими ресурсами маркетплейсу та забезпечують маркетологам та аналітикам необхідні інструменти для прийняття обґрунтованих рішень.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Jim Frost: Regression Analysis: An Intuitive Guide for Using and Interpreting Linear Models : Statistics By Jim Publishing, 2020. -355 pages – Bibliogr.: s. 284–346.
2. Andrew F. Hayes: Introduction to Mediation, Moderation, and Conditional Process Analysis: A Regression-Based Approach (Methodology in the Social Sciences Series) : The Guilford Press, 2022. -732 pages – Bibliogr.: s. 280-344.
3. David Flanagan: The Definitive Guide: Master the World's Most-Used Programming Language : O'Reilly Media, 2020. -704 pages – Bibliogr.: s. 560–578.
4. Jon Duckett: JavaScript and jQuery: Interactive Front-End Web Development: Wiley, 2014. -640 pages – Bibliogr.: s. 140–279.
5. Josh Goldberg: Learning TypeScript: Enhance Your Web Development Skills Using Type-Safe JavaScript: O'Reilly Media, 2022. -318 pages – Bibliogr.: s. 270–316.
6. Stefan Baumgartner: TypeScript Cookbook: Real World Type-Level Programming: O'Reilly Media, 2023. -419 pages – Bibliogr.: s. 256-344.
7. Robin Wieruch: he Road to React: Your journey to master plain yet pragmatic React.js: Independently published, 2018. -292 pages – Bibliogr.: s. 234–292.
8. Loren Klingman: React Programming: The Big Nerd Ranch Guide (Big Nerd Ranch Guides): Big Nerd Ranch Guides, 2023. -420 pages – Bibliogr.: s. 145-207.
9. Kartik Bhat: Ultimate Tailwind CSS Handbook: Build sleek and modern websites with immersive UIs using Tailwind CSS: Orange Education Pvt Ltd, 2023. -408 pages – Bibliogr.: s. 230-287.
10. Badal Tripathy: Tailwind CSS: Orange Education Pvt Ltd, 2023. -141 pages – Bibliogr.: s. 93-123.
11. Shannon Bradshaw: MongoDB: The Definitive Guide: O'Reilly Media, 2019. -511 pages – Bibliogr.: s. 345-393.
12. Daniel Coupal: MongoDB Data Modeling and Schema Design: Technics

Publications 2023. -354 pages – Bibliogr.: s. 208-245.
13. Tom Greever: Articulating Design Decisions: O'Reilly, 2015. -278 pages – Bibliogr.: s. 108-132.
14. Jenifer Tidwell: Designing Interfaces: Patterns for Effective Interaction Design: O'Reilly, 2020. -598 pages – Bibliogr.: s. 346-408.
15. Eric A. Meyer: CSS: The Definitive Guide: Web Layout and Presentation 5th Edition: O'Reilly, 2023. -1126 pages – Bibliogr.: s. 450-478.
16. Ben Frain: Responsive Web Design with HTML5 and CSS: O'Reilly, 2023. -384 pages – Bibliogr.: s. 246-290.
17. Ethan Brown: Web Development with Node and Express: Leveraging the JavaScript Stack 2nd Edition: O'Reilly, 2019. -340 pages – Bibliogr.: s. 305-338.
18. Rick L.: Express.js: Guide Book on Web framework for Node.js: O'Reilly, 2016. -99 pages – Bibliogr.: s. 23-67.
19. Sebastian Springer: Node.js: The Comprehensive Guide to Server-Side JavaScript Programming (Rheinwerk Computing): O'Reilly, 2022. -834 pages – Bibliogr.: s. 236-342.
20. Thomas Hunter II: Distributed Systems with Node.js: Building Enterprise-Ready Backend Services 1st Edition: O'Reilly, 2016. -377 pages – Bibliogr.: s. 114-231.
21. Steven Spadotto: React Router Ready: Learn React Router with React and TypeScript: O'Reilly, 2023. -148 pages – Bibliogr.: s. 23-112.
22. Sagar Ganatra: React Router Quick Start Guide: Routing in React applications made easy: Packt Publishing, 2018. -158 pages – Bibliogr.: s. 46-78.
23. Serge Gershkovich: Data Modeling with Snowflake: A practical guide to accelerating Snowflake development using universal data modeling techniques: Packt Publishing, 2023. -324 pages – Bibliogr.: s. 126-132.
24. Alessandro Del Sole: Visual Studio Code Distilled: Evolved Code Editing for Windows, macOS, and Linux: Apress, 2023. -353 pages – Bibliogr.: s. 110-207.
25. Bruce Johnson: Visual Studio Code: End-to-End Editing and Debugging Tools for Web Developers: Wiley, 2019. -192 pages – Bibliogr.: s. 76-145.
26. Patrick Mulder: Node.js for Embedded Systems: Using Web Technologies to

Build Connected Devices 1st Edition: O'Reilly, 2016. -266 pages – Bibliogr.: s. 34-127.

ДОДАТОК А

КОД ПРОГРАМИ

REACT

Навігаційна панель

```
import { useState } from 'react';
import SavingsIcon from '@mui/icons-material/Savings';
import { Link } from "react-router-dom";
import { Box, Typography, useTheme } from "@mui/material";
import FlexBetween from '@/components/FlexBetween';

type Props = {}

const Navbar = (props: Props) => {
   const { palette } = useTheme();
   const [selected, setSelected] = useState("dashboard");
 return (<FlexBetween
 mb="0.25rem"
 p="0.5rem 0rem"
 color={palette.grey[300]}
 >

   <FlexBetween gap="0.75rem">
      <SavingsIcon sx={{ fontSize: "28px" }} />
      <Typography  variant="h4" fontSize="16px">Finanseer</Typography>
   </FlexBetween>


   <FlexBetween gap="2rem">
      <Box sx={{ "&:hover": { color: palette.primary[100] }}}>
         <Link
            to="/"
            onClick={() => setSelected("dashboard")}
            style={{
            color: selected === "dashboard" ? "inherit" : palette.grey[700],
            textDecoration: "inherit",
            }}
         >
            Панель елементів
         </Link>
      </Box>
      <Box sx={{ "&:hover": { color: palette.primary[100] }}}>
         <Link
            to="/predictions"
            onClick={() => setSelected("predictions")}
            style={{
            color: selected === "predictions" ? "inherit" : palette.grey[700],
            textDecoration: "inherit",
            }}
         >
            Прогнози
         </Link>
      </Box>
   </FlexBetween>
```

```
  </FlexBetween> )
}

export default Navbar;
```

index.js
Конфігурація бекенду

```
import express from "express";
import bodyParser from "body-parser";
import mongoose from "mongoose";
import cors from "cors";
import dotenv from "dotenv";
import helmet from "helmet";
import morgan from "morgan";
import kpiRoutes from "./routes/kpi.js";
import productRoutes from "./routes/product.js";
import transactionRoutes from "./routes/transaction.js";
import KPI from "./models/KPI.js";
import Product from "./models/Product.js";
import Transaction from "./models/Transaction.js";
import { kpis, products, transactions } from "./data/data.js";


dotenv.config();
const app = express();
app.use(express.json());
app.use(helmet());
app.use(helmet.crossOriginResourcePolicy({ policy: "cross-origin" }));
app.use(morgan("common"));
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));
app.use(cors());



app.use("/kpi", kpiRoutes);
app.use("/product", productRoutes);
app.use("/transaction", transactionRoutes);


const PORT = process.env.PORT || 9000;
mongoose
  .connect(process.env.MONGO_URL, {
  })
  .then(async () => {
    app.listen(PORT, () => console.log(`Server Port: ${PORT}`));
    await mongoose.connection.db.dropDatabase();
    KPI.insertMany(kpis);
    Product.insertMany(products);
    Transaction.insertMany(transactions);
  })
  .catch((error) => console.log(`${error} did not connect`));
```

Theme.ts
Налаштування кольорів,абзаців та шрифтів

```
export const tokens = {
  grey: {
    100: "#f0f0f3",
    200: "#e1e2e7",
    300: "#d1d3da",
    400: "#f6caff",
    500: "#b3b6c2",
    600: "#8f929b",
    700: "#6b6d74",
    800: "#48494e",
    900: "#242427",
  },
  primary: {
    // light green
    100: "#d0fcf4",
    200: "#a0f9e9",
    300: "#52f1e1",
    400: "#41f2d3",
    500: "#12efc8",
    600: "#0ebfa0",
    700: "#0b8f78",
    800: "#e6e7dd",
    900: "#043028",
  },
  secondary: {
    // yellow
    100: "#fcf0dd",
    200: "#fae1bb",
    300: "#f7d299",
    400: "#f5c377",
    500: "#00e600",
    600: "#c29044",
    700: "#916c33",
    800: "#614822",
    900: "#302411",
  },
  tertiary: {
    // purple
    500: "#8884d8",
  },
  background: {
    light: "#2d2d34",
    main: "#1f2026",
  },
};

// mui theme settings
export const themeSettings = {
  palette: {
    primary: {
      ...tokens.primary,
      main: tokens.primary[500],
      light: tokens.primary[400],
    },
    secondary: {
```

```
      ...tokens.secondary,
      main: tokens.secondary[500],
    },
    tertiary: {
      ...tokens.tertiary,
    },
    grey: {
      ...tokens.grey,
      main: tokens.grey[500],
    },
    background: {
      default: tokens.background.main,
      light: tokens.background.light,
    },
  },
  typography: {
    fontFamily: ["Inter", "sans-serif"].join(","),
    fontSize: 12,
    h1: {
      fontFamily: ["Inter", "sans-serif"].join(","),
      fontSize: 32,
    },
    h2: {
      fontFamily: ["Inter", "sans-serif"].join(","),
      fontSize: 24,
    },
    h3: {
      fontFamily: ["Inter", "sans-serif"].join(","),
      fontSize: 20,
      fontWeight: 800,
      color: tokens.grey[200],
    },
    h4: {
      fontFamily: ["Inter", "sans-serif"].join(","),
      fontSize: 14,
      fontWeight: 600,
      color: tokens.grey[300],
    },
    h5: {
      fontFamily: ["Inter", "sans-serif"].join(","),
      fontSize: 12,
      fontWeight: 400,
      color: tokens.grey[500],
    },
    h6: {
      fontFamily: ["Inter", "sans-serif"].join(","),
      fontSize: 10,
      color: tokens.grey[700],
    },
  },
};
```

Types.ts
Визначення набору інтерфейсів

```
export interface ExpensesByCategory {
```

```
  salaries: number;
  supplies: number;
  services: number;
}

export interface Month {
  id: string;
  month: string;
  revenue: number;
  expenses: number;
  nonOperationalExpenses: number;
  operationalExpenses: number;
}

export interface Day {
  id: string;
  date: string;
  revenue: number;
  expenses: number;
}

export interface GetKpisResponse {
  id: string;
  _id: string;
  __v: number;
  totalProfit: number;
  totalRevenue: number;
  totalExpenses: number;
  expensesByCategory: ExpensesByCategory;
  monthlyData: Array<Month>;
  dailyData: Array<Day>;
  createdAt: string;
  updatedAt: string;
}

export interface GetProductsResponse {
  id: string;
  _id: string;
  __v: number;
  price: number;
  expense: number;
  transactions: Array<string>;
  createdAt: string;
  updatedAt: string;
}

export interface GetTransactionsResponse {
  id: string;
  _id: string;
  __v: number;
  buyer: string;
  amount: number;
  productIds: Array<string>;
  createdAt: string;
  updatedAt: string;
```

```
}
```

Row1.tsx
Створення первого рядка grit-teamplate-areas

```
import BoxHeader from "@/components/BoxHeader";
import DashboardBox from "@/components/DashboardBox";
import { useGetKpisQuery } from "@/state/api";
import { useTheme } from "@mui/material";
import { useMemo } from "react";
import {
  ResponsiveContainer,
  CartesianGrid,
  AreaChart,
  BarChart,
  Bar,
  LineChart,
  XAxis,
  YAxis,
  Legend,
  Line,
  Tooltip,
  Area,
} from "recharts";

const Row1 = () => {
  const { palette } = useTheme();
  const { data } = useGetKpisQuery();

  const revenue = useMemo(() => {
    return (
      data &&
      data[0].monthlyData.map(({ month, revenue }) => {
        return {
          name: month.substring(0, 3),
          revenue: revenue,
        };
      })
    );
  }, [data]);

  const revenueExpenses = useMemo(() => {
    return (
      data &&
      data[0].monthlyData.map(({ month, revenue, expenses }) => {
        return {
          name: month.substring(0, 3),
          revenue: revenue,
          expenses: expenses,
        };
      })
    );
  }, [data]);

  const revenueProfit = useMemo(() => {
    return (
```

```
    data &&
    data[0].monthlyData.map(({ month, revenue, expenses }) => {
      return {
        name: month.substring(0, 3),
        revenue: revenue,
        profit: (revenue - expenses).toFixed(2),
      };
    })
  );
}, [data]);

return (
  <>
    <DashboardBox gridArea="a">
      <BoxHeader
        title="Доходи та витрати"
        subtitle="Верхній рядок представляє дохід, нижній рядок – витрати"
        sideText="+5%"
      />
      <ResponsiveContainer width="100%" height="100%">
        <AreaChart
          width={500}
          height={400}
          data={revenueExpenses}
          margin={{
            top: 15,
            right: 25,
            left: -10,
            bottom: 60,
          }}
        >
          <defs>
            <linearGradient id="colorRevenue" x1="0" y1="0" x2="0" y2="1">
              <stop
                offset="5%"
                stopColor={palette.primary[300]}
                stopOpacity={0.5}
              />
              <stop
                offset="95%"
                stopColor={palette.primary[300]}
                stopOpacity={0}
              />
            </linearGradient>
            <linearGradient id="colorExpenses" x1="0" y1="0" x2="0" y2="1">
              <stop
                offset="5%"
                stopColor={palette.primary[300]}
                stopOpacity={0.5}
              />
              <stop
                offset="95%"
                stopColor={palette.primary[300]}
                stopOpacity={0}
              />
```

```
        </linearGradient>
      </defs>
      <XAxis
        dataKey="name"
        tickLine={false}
        style={{ fontSize: "10px" }}
      />
      <YAxis
        tickLine={false}
        axisLine={{ strokeWidth: "0" }}
        style={{ fontSize: "10px" }}
        domain={[8000, 23000]}
      />
      <Tooltip />
      <Area
        type="monotone"
        dataKey="revenue"
        dot={true}
        stroke={palette.primary.main}
        fillOpacity={1}
        fill="url(#colorRevenue)"
      />
      <Area
        type="monotone"
        dataKey="expenses"
        dot={true}
        stroke={palette.primary.main}
        fillOpacity={1}
        fill="url(#colorExpenses)"
      />
    </AreaChart>
  </ResponsiveContainer>
</DashboardBox>
<DashboardBox gridArea="b">
  <BoxHeader
    title="Прибуток і дохід"
    subtitle="Верхній рядок представляє дохід, нижній рядок – прибуток"
    sideText="+5%"
  />
  <ResponsiveContainer width="100%" height="100%">
    <LineChart
      width={500}
      height={400}
      data={revenueProfit}
      margin={{
        top: 20,
        right: 0,
        left: -10,
        bottom: 55,
      }}
    >
      <CartesianGrid vertical={false} stroke={palette.grey[800]} />
      <XAxis
        dataKey="name"
        tickLine={false}
```

```
          style={{ fontSize: "10px" }}
        />
        <YAxis
          yAxisId="left"
          tickLine={false}
          axisLine={false}
          style={{ fontSize: "10px" }}
        />
        <YAxis
          yAxisId="right"
          orientation="right"
          tickLine={false}
          axisLine={false}
          style={{ fontSize: "10px" }}
        />
        <Tooltip />
        <Legend
          height={20}
          wrapperStyle={{
            margin: "0 0 10px 0",
          }}
        />
        <Line
          yAxisId="left"
          type="monotone"
          dataKey="profit"
          stroke={palette.tertiary[500]}
        />
        <Line
          yAxisId="right"
          type="monotone"
          dataKey="revenue"
          stroke={palette.primary.main}
        />
      </LineChart>
    </ResponsiveContainer>
  </DashboardBox>
  <DashboardBox gridArea="c">
    <BoxHeader
      title="Щомісячні доходи"
      subtitle="Відображає доходи по місяцях"
      sideText="+5%"
    />
    <ResponsiveContainer width="100%" height="100%">
      <BarChart
        width={500}
        height={300}
        data={revenue}
        margin={{
          top: 17,
          right: 15,
          left: -5,
          bottom: 58,
        }}
      >
```

```
          <defs>
            <linearGradient id="colorRevenue" x1="0" y1="0" x2="0" y2="1">
              <stop
                offset="5%"
                stopColor={palette.primary[300]}
                stopOpacity={0.8}
              />
              <stop
                offset="95%"
                stopColor={palette.primary[300]}
                stopOpacity={0}
              />
            </linearGradient>
          </defs>
          <CartesianGrid vertical={false} stroke={palette.grey[800]} />
          <XAxis
            dataKey="name"
            axisLine={false}
            tickLine={false}
            style={{ fontSize: "10px" }}
          />
          <YAxis
            axisLine={false}
            tickLine={false}
            style={{ fontSize: "10px" }}
          />
          <Tooltip />
          <Bar dataKey="revenue"  fill="url(#colorRevenue)" />
        </BarChart>
      </ResponsiveContainer>
    </DashboardBox>
  </>
 );
};

export default Row1;
```

ДОДАТОК Б

ПЕРЕЛІК ФАЙЛІВ НА ДИСКУ

Ім’я файлу	Опис
Пояснювальні документи	
Кваліфікаційна_робота_Іванова.docx	Пояснювальна записка до кваліфікаційної роботи. Документ Word.
Кваліфікаційна_робота_Іванова.pdf	Пояснювальна записка до кваліфікаційної роботи в форматі PDF
Код_Іванов.docx	Код програми, який не ввійшов до пояснювальної записки
Програма	
Program.rar	Архів. Містить коди програми і откомпільовану програму
Презентація_Іванов.pptx	Презентація кваліфікаційної роботи