ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«**Архітектура додатку на основі штучного інтелекту для комерційного аналізу за допомогою LLM-агентів**»

на здобуття освітнього ступеня магістр
зі спеціальності 126 Інформаційні системи та технології
(код,найменування спеціальності)
освітньо-професійної програми Інформаційні системи та технології
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Денис СОСІН
(підпис) *(ім'я, ПРІЗВИЩЕ здобувача)*

Виконав:
здобувач вищої освіти
група ІСДМ-62 Денис СОСІН
(ім'я, ПРІЗВИЩЕ)

Керівник Віктор САГАЙДАК
PhD. *(ім'я, ПРІЗВИЩЕ)*

Рецензент:
науковий ступінь, вчене звання *(ім'я, ПРІЗВИЩЕ)*

Київ 2025

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

AI - штучний інтелект (Artificial Intelligence)

API - програмний інтерфейс додатку (Application Programming Interface)

ASIN - стандартний ідентифікаційний номер Amazon (Amazon Standard Identification Number)

CSV - значення, розділені комами (Comma-Separated Values)

F1-міра - гармонійне середнє між точністю та повнотою (F1-score)

GDPR - загальний регламент захисту даних (General Data Protection Regulation)

HTML - мова розмітки гіпертексту (HyperText Markup Language)

JSON - об'єктна нотація JavaScript (JavaScript Object Notation)

KPI - ключові показники ефективності (Key Performance Indicators)

MAE - середня абсолютна помилка (Mean Absolute Error)

MAPE - середня абсолютна процентна помилка (Mean Absolute Percentage Error)

NLP - обробка природної мови (Natural Language Processing)

PDF - портативний формат документа (Portable Document Format)

PostgreSQL - об'єктно-реляційна система управління базами даних

R^2 - коефіцієнт детермінації (коефіцієнт детермінації)

REST - передача репрезентативного стану (Representational State Transfer)

RMSE - корінь із середньоквадратичної помилки (Root Mean Square Error)

SQL - мова структурованих запитів (Structured Query Language)

TLS - протокол захисту транспортного рівня (Transport Layer Security)

UI - інтерфейс користувача (User Interface)

Unicode - стандарт кодування символів, що підтримує більшість писемностей світу

ВСТУП

Сучасне бізнес-середовище, особливо в умовах глобалізації та цифровізації, характеризується експоненціальним зростанням обсягів даних, високою динамікою ринкових змін і загостренням конкуренції. Ці тенденції особливо чітко проявляються в сфері електронної комерції, де успіх компанії значною мірою залежить від здатності оперативно аналізувати поведінку клієнтів, виявляти приховані закономірності та адаптуватися до змін. Однак традиційні підходи до аналізу даних і прогнозування часто не в змозі врахувати всю складність взаємозв'язків між якісними (текстовими) та кількісними (числовими) даними, що призводить до неточних рішень і втрати конкурентних переваг.

Відгуки клієнтів, зокрема на майданчиках типу Amazon, містять цінну, але неструктуровану інформацію: вони відображають сприйняття товару, виявляють проблемні зони, вказують на переваги перед конкурентами. Одночасно з цим дані про продажі, хоча й мають високу структурованість, самі по собі не враховують контекстуальних факторів, які можуть значно вплинути на попит. Сучасні системи штучного інтелекту та машинного навчання надають потенціал для об'єднання цих двох типів даних, але переважна більшість існуючих рішень реалізовані або як окремі інструменти аналізу тексту (наприклад, VADER, BERT-based sentiment analyzers), або як моделі прогнозування часових рядів (ARIMA, Prophet, LSTM). Лише у поодиноких дослідженнях, зокрема в працях Liu et al. (2021) та Chen & Zhang (2023), робляться спроби інтегрувати NLP-інсайти в прогнозні моделі, проте без формування універсальної, гнучкої та бізнес-орієнтованої архітектури.

Для України ця проблема є особливо актуальною: вітчизняні онлайн-ритейлери та експортно-орієнтовані бренди, які працюють на міжнародних маркетплейсах, потребують доступних, локалізованих інтегрованих рішень, які враховують особливості ринку, мови (у тому числі підтримку української) та обмежені ресурси середнього бізнесу. Наразі в українській науково-прикладній практиці майже відсутні комплексні системи, які б поєднували AI-аналіз відгуків і точне прогнозування попиту в єдиній архітектурі.

Тому розробка такого інструменту не лише має наукову новизну, але й має високу соціальну та економічну значущість для розвитку вітчизняного цифрового бізнесу.

Метою даної кваліфікаційної роботи є розробка, реалізація та оцінка ефективності інтегрованої системи для AI-аналізу відгуків покупців та прогнозування майбутніх продажів у сфері електронної комерції.

В процесі дослідження вирішувалися наступні завдання:

1. Проведено критичний аналіз існуючих підходів до обробки природної мови та прогнозування часових рядів з акцентом на їх придатність для інтеграції в єдину систему.
2. Запропоновано модульні архітектуру системи, засновану на взаємодії декількох типів аналітичних агентів (NLP-агент, прогнозний агент, агент коригування).
3. Реалізовано двоетапний механізм категоризації відгуків: автоматичне групування за темами за допомогою сучасних NLP-моделей (зокрема, з використанням Ollama та кастомних алгоритмів на основі NLTK) з можливістю подальшого редагування та верифікації результатів користувачем.
4. Розроблено гібридну модель прогнозування, яка одночасно враховує сезонні та трендові компоненти часових рядів продажів та якісні метрики, отримані з аналізу відгуків (наприклад, частота згадування проблемних тем).
5. Проведено комплексне тестування системи на реальних даних, включаючи оцінку точності прогнозів, якості тематичної кластеризації та впливу використання текстових інсайтів на покращення бізнес-показників.
6. Виконано оцінку економічної доцільності та практичної цінності впровадження системи для українських онлайн-ритейлерів.

Об’єкт дослідження - процеси аналізу клієнтських відгуків та прогнозування обсягів продажів у сфері електронної комерції, зокрема на глобальних маркетплейсах.

Предмет дослідження - методи та інструменти штучного інтелекту, спрямовані на інтеграцію неструктурованих текстових даних і часових рядів у єдину аналітичну систему для підтримки прийняття бізнес-рішень.

У роботі застосовано системний підхід до проектування програмного забезпечення, методи обробки природної мови (NLP), зокрема тематичне моделювання та сентимент-аналіз, методи машинного навчання для прогнозування часових рядів (з використанням гібридних моделей на основі Prophet та регресійних підходів), а також статистичні методи оцінки якості моделей (MAPE, F1-score, коефіцієнт кореляції Пірсона). Для реалізації системи використано сучасні принципи розробки ПЗ, включаючи модульність, контейнеризацію (Docker), використання REST API (Flask) та дотримання принципів безпеки даних (відповідно до GDPR).

Наукова новизна полягає в поєднанні різних AI-методів у рамках єдиної архітектури, у впровадженні двоетапного механізму категоризації відгуків (автоматична кластеризація + верифікація користувачем), а також у введенні механізму взаємного коригування між текстовими інсайтами та кількісними прогнозами, де зміни в тональності або частоті тем відгуків динамічно впливають на прогнозовані обсяги продажів. Такий підхід реалізований уперше в контексті підтримки рішень для українських бізнесів із урахуванням можливості локальної обробки даних.

Практична значущість полягає у створенні реально працездатного інструменту, який дозволяє компаніям:

- швидко виявляти проблемні аспекти продукту;
- покращувати якість товарів та послуг на основі реальної зворотної зв’язку;
- підвищувати точність планування запасів та логістики;
- знижувати операційні витрати завдяки автоматизації аналізу великих обсягів відгуків.

Рішення може бути адаптоване не лише для електронної комерції, а й для інших галузей, де важлива інтеграція текстових та числових даних (наприклад, сфера послуг, FMCG, e-health).

Теоретична та методична значущість роботи полягає у формуванні універсального підходу до інтегрованого аналізу різнорідних даних, який може бути використаний як методична основа при розробці подібних систем у

майбутньому. Запропонована архітектура та механізми взаємодії між агентами можуть бути адаптовані до інших предметних областей без кардинальної перебудови системи.

Робота складається зі вступу, трьох основних розділів, висновків, списку використаних джерел та додатків. У першому розділі проаналізовано сучасний стан технологій AI для аналізу текстів і прогнозування, виявлено їхні обмеження та обґрунтовано потребу в інтегрованому підході. Другий розділ присвячено проектуванню архітектури системи, вибору технологічного стеку та формуванню методики інтеграції даних. У третьому розділі описано реалізацію системи, методи тестування, отримані результати та їхню оцінку за технічними та бізнес-критеріями.

Апробація результатів магістерської роботи. Базові результати магістерської роботи опубліковано в збірнику тез на Міжнародна науково-практична конференції «Інформаційні технології і автоматизація» та на XVIII міжнародної науково-практичної конференції «ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ І АВТОМАТИЗАЦІЯ – 2025».

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ ЗАСТОСУВАННЯ LLM-АГЕНТІВ У КОМЕРЦІЙНОМУ АНАЛІЗІ

1.1. Огляд застосування LLM у бізнес-аналітиці

Великі мовні моделі (Large Language Models, LLM) - це сучасні системи штучного інтелекту, навчені на надмасштабних корпусах текстів з метою розуміння та створення людської мови. У сфері бізнес-аналітики вони кардинально змінюють підхід до роботи з даними, оскільки здатні ефективно обробляти як структуровану, так і неструктуровану інформацію. За даними досліджень, впровадження LLM у аналітичні процеси не лише підвищує продуктивність, але й робить аналітику доступною для користувачів без технічної підготовки. Проте таке впровадження потребує обережного ставлення через можливі ризики, пов’язані з точністю та безпекою даних. У цьому огляді розглядаються ключові аспекти застосування LLM у бізнес-аналітиці - від теоретичних засад до реальних кейсів, переваг, викликів і прогнозів на 2025 рік - на основі аналізу академічних праць, галузевих публікацій та українських джерел [1].

LLM - це статистичні моделі з мільярдами параметрів, здатні виконувати різноманітні мовні завдання: від перекладу та класифікації текстів до генерації нових фрагментів контенту. У бізнес-аналітиці їх часто інтегрують із платформами BI (Business Intelligence), такими як Power BI або Tableau, для автоматизації повторюваних операцій. Наприклад, сучасні моделі, зокрема GPT-4 чи Claude, можуть перетворювати запити, сформульовані звичайною мовою, на виконавчі SQL-запити. Це дозволяє нетехнічним спеціалістам - маркетологам, продакт-менеджерам, фінансистам - самостійно працювати з даними, не залучаючи розробників або аналітиків [2].

Одне з найважливіших застосувань LLM - підготовка даних, яка за традицією займає до 80 % робочого часу аналітика. Моделі автоматизують цей етап: вони можуть перекладати текстові описи товарів (наприклад, з німецької на

англійську), класифікувати їх за категоріями, визначати тональність відгуків або витягувати ключову інформацію з неструктурованих скарг. Дослідження, представлене на конференції HCI International 2023, показало, що ChatGPT без додаткового доналаштування демонструє високу точність у таких завданнях, ефективно працюючи з різними мовами та форматами. Проте у випадках, коли дані містять багато шуму або неоднозначностей, модель може помилятися, тому доцільно поєднувати її роботу з людським контролем [3].

У контексті аналізу текстів LLM істотно випереджають класичні методи. Завдяки семантичному розумінню вони здатні виявляти приховані зв’язки та нюанси в неструктурованих даних - будь то записи у соціальних мережах, відгуки на маркетплейсах чи блогів. Наприклад, моделі на базі BERT або GPT можуть відрізняти, чи слово «Java» використане стосовно мови програмування чи географічного об’єкта, що критично для точного аналізу. На практиці це вже реалізовано: інструмент від Intellias використовує NLP та генеративний AI (GenAI) для моніторингу репутації бренду, перетворюючи неструктуровані зовнішні дані на кількісні індекси. Це дозволяє маркетологам оперативно оцінювати ефективність кампаній без ручного аналізу. У продажах LLM також застосовуються для взаємодії з CRM- і ERP-системами через розмовний інтерфейс: користувач може просто запитати «Який lifetime value у клієнтів із Києва?» - і отримати звіт без знання SQL. Такі підходи вже використовують компанії на кшталт Findly або Salesforce з її Einstein GPT, що зменшує навантаження на аналітичні відділи [4].

У бізнес-інтелекті виділяють три основні сфери застосування великих мовних моделей:

1. Збагачення даних - інтеграція зовнішніх джерел (наприклад, демографічної інформації з US Census або даних із соціальних мереж) у внутрішні бази з автоматичним додаванням метаданих. Це дозволяє, наприклад, збагатити профілі клієнтів або точніше класифікувати відгуки.

2. Очищення та підготовка даних - автоматичне виправлення опечаток, уніфікація форматів дат, заповнення пропущених значень на основі контексту. Це значно прискорює підготовку великих обсягів даних до аналізу.

3. Розмовний інтерфейс для дослідження даних - можливість формулювати запити звичайною мовою (наприклад: «Покажи продажі за регіонами у Q3») і отримувати персоналізовані, контекстно зрозумілі відповіді без технічних навичок [5].

Для ілюстрації ось таблиця основних застосувань LLM у бізнес-аналітиці (табл. 1.1).

Таблиця 1.1

Основні застосування LLM у бізнес-аналітиці [6]

Застосування	Опис	Приклади інструментів/ моделей	Переваги	Ризики
Аналіз настроїв клієнтів	Обробка тексту для виявлення емоцій та інсайтів	GPT, BERT	Масштабний семантичний аналіз	Неточності в нюансах
Підготовка даних	Очищення, категоризація, переклад	ChatGPT	Зменшення ручної праці на 50-70%	Помилки з шумними даними
Генерація звітів	Автоматичне створення сумарій та рекомендацій	Claude, GPT-4	Швидкість та доступність	Галюцинації (вигадані факти)
Прогнозування та ризик	Інтеграція з predictive analytics для прогнозів	Einstein GPT	Контекстно точні прогнози	Порушення конфіденційності
Розмовні запити	Перетворення NL на SQL чи аналітику	Findly, Salesforce	Демократизація для нетехнічних користувачів	Залежність від якості промптів

Застосування великих мовних моделей у бізнес-аналітиці несе низку істотних переваг. Серед них - підвищення ефективності за рахунок автоматизації всього аналітичного ланцюжка, що охоплює збір, очищення, обробку та інтерпретацію даних. Моделі здатні виявляти складні та неочевидні зв’язки,

витягувати приховані патерни з великих обсягів інформації й адаптуватися до різноманітних бізнес-контекстів. Вони дозволяють масштабувати аналітичні процеси без пропорційного збільшення команди фахівців, що істотно скорочує операційні витрати. Крім того, завдяки інтуїтивним розмовним інтерфейсам аналітика стає доступною навіть для нетехнічних користувачів. Як зазначає СТО компанії Pyramid Analytics Аві Перез, LLM дають змогу ставити складні питання до даних і отримувати змістовні, інтелектуальні відповіді, роблячи аналітичні інструменти справді інклюзивними. Найвищий рівень повернення інвестицій демонструють домен-специфічні моделі, навчені на галузевих даних: вони дозволяють аналізувати ширший спектр джерел без потреби у додаткових ліцензіях або розширенні штату [7].

Проте використання великих мовних моделей супроводжується серйозними ризиками. Серед них - потенційний витік конфіденційної інформації, оскільки дані, які користувачі надають публічним моделям, можуть зберігатися або використовуватися третьою стороною. Це створює загрози порушення регуляторних вимог, зокрема GDPR або HIPAA. Також моделі схильні до генерації «галюцинацій» - фактологічно невірної чи вигаданої інформації, що може призводити до помилкових управлінських рішень. Не менш небезпечними є вразливості до атак типу prompt injection, коли зловмисник може маніпулювати поведінкою моделі через спеціально сформульовані запити. Крім того, висока вартість інференсу при обробці великих обсягів даних може стати фінансовим обмеженням для багатьох організацій. Для мінімізації цих загроз експерти рекомендують використовувати приватні або локально розгорнуті моделі, впроваджувати механізми верифікації та крос-перевірки виводу, а також дотримуватися чітких етичних протоколів під час розробки та експлуатації систем на основі LLM [8].

У українському бізнес-середовищі великі мовні моделі вже знаходять практичне застосування. Їх використовують для аналізу корпоративних комунікацій, обробки економічної статистики та підтримки інноваційних рішень. Наприклад, розроблено методологічні підходи до універсальної мовної аналітики

бізнес-текстів, що дозволяють автоматично витягувати стратегічні інсайти з листування, внутрішніх звітів, контрактів і інших документів. У сфері економіки LLM поступово трансформують саму парадигму роботи з даними - завдяки можливості автоматизовано генерувати аналітичні звіти, прогнози та рекомендації. Рейтинги провідних моделей, таких як GPT або Grok, підтверджують їхню ефективність у реальних бізнес-сценаріях, зокрема в маркетингу, обробці відгуків клієнтів, аналізі соціальних медіа та інших джерел неструктурованих текстів [9].

На 2025 рік очікується подальший розвиток галузевих LLM, які будуть навчані на спеціалізованих даних у конкретних сферах - фінансах, охороні здоров’я, логістиці чи праві. Такі моделі поєднуватимуть глибоку предметну експертизу з гнучкістю природної мови, що значно підвищить точність аналітики. Також прогнозується активна інтеграція LLM у платформи бізнес-інтелекту для підтримки самообслуговуваної аналітики, де звичайні співробітники зможуть самостійно отримувати інсайти без залучення IT-фахівців. Одночасно зростатиме увага до принципів відповідального та етичного штучного інтелекту - прозорості, підзвітності, недискримінації та захисту приватності. Компанії, які вчасно й обґрунтовано впроваджуватимуть великі мовні моделі, отримають значну конкурентну перевагу. Проте їхній успіх буде залежати від здатності ефективно балансувати між інноваціями та безпекою, а також від постійної верифікації джерел даних і достовірності отриманих результатів. Майбутні дослідження, ймовірно, зосереджуватимуться на підвищенні надійності моделей, зменшенні частоти галюцинацій та розвитку гібридних архітектур, що поєднують сильні сторони LLM з перевіреними класичними методами аналітики, такими як статистичне моделювання, OLAP-аналіз і правила на основі бізнес-логіки.

1.2. Концепція LLM-агентів та їх архітектурні особливості

Великі мовні моделі пройшли значний еволюційний шлях: від простих інструментів, здатних генерувати текст за заданим промптом, до складних

автономних систем, відомих сьогодні як LLM-агенти. Цей етап розвитку штучного інтелекту виходить далеко за межі пасивного оброблення мови. Сучасні моделі тепер можуть самостійно взаємодіяти з навколишнім середовищем, формулювати плани, виконувати послідовності дій, залучати зовнішні інструменти, отримувати зворотний зв'язок і навіть покращувати свою роботу на основі власного досвіду. У цій статті розглядається сутність LLM-агентів: їхня внутрішня будова, ключові компоненти, існуючі типи, сфери застосування, переваги й обмеження, а також перспективи подальшого розвитку. Основу аналізу становлять найновіші наукові публікації, технічні звіти та реальні кейси впровадження. Особлива увага приділяється їхньому застосуванню в дизайн-процесах та бізнес-аналітиці, де ключовим стає питання балансу між автономією системи та необхідністю людського контролю та втручання [10].

Суть концепції LLM-агентів полягає в принципі агентності - здатності системи самостійно діяти для досягнення певної мети, враховуючи динаміку середовища та власні попередні дії. На відміну від традиційних великих мовних моделей, які надають одноразову відповідь на вхідний запит, агенти працюють активно й ітеративно. Вони сприймають завдання, розбивають його на підзадачі, визначають необхідні джерела інформації, взаємодіють із зовнішніми ресурсами - такими як API, бази даних, обчислювальні бібліотеки або інші сервіси, - аналізують отримані результати та, якщо це потрібно, переглядають і коригують свою стратегію. Цей підхід бере початок із піонерських проєктів, таких як AutoGPT чи BabyAGI, де автономні системи вирішують багатоетапні завдання майже без безпосереднього участі людини. У контексті бізнес-аналітики LLM-агент може, наприклад, отримати запит у стилі «Проаналізуйте квартальні продажі» і самостійно виконати цілий аналітичний цикл: підключитися до бази даних, витягти релевантну інформацію, обчислити ключові метрики, виявити нетипові відхилення або довгострокові тренди, порівняти результати з попередніми періодами та нарешті оформити все це у зрозумілий, змістовний аналітичний звіт, придатний для прийняття рішень [11].

Архітектура таких агентів базується на ітеративному циклі, що складається з чотирьох ключових етапів: планування, дія, спостереження та рефлексія. Спочатку агент формулює стратегію вирішення завдання, потім виконує конкретні дії - наприклад, викликає інструмент або надсилає запит до бази даних. Після цього він спостерігає за результатами своїх дій, оцінює їх відповідність поставленій меті й, за необхідності, вносить корективи у свій подальший план. Такий цикл може повторюватися багаторазово, доки завдання не буде виконане на достатньому рівні. Хоча ядром агента залишається трансформерна модель з механізмами уваги та позиційним кодуванням, навколо неї побудовано додаткові шари оркестрації: модулі управління завданнями, інтерфейси взаємодії з інструментами, системи внутрішньої пам’яті та механізми самооцінки. У дизайн-процесах LLM-агенти не призначені замінювати людську креативність, а радше виступають як інтелектуальні партнери. Вони здатні збирати й структурувати інформацію про потреби користувачів, генерувати варіанти дизайнерських рішень, пропонувати альтернативні підходи, моделювати взаємодію з інтерфейсом або навіть автоматично тестувати прототипи на зручність використання. Водночас остаточні рішення, що стосуються естетики, етики чи стратегії, залишаються під людським наглядом, щоб уникнути помилок, спричинених недостатнім контекстом, неадекватною автономією або відсутністю глибокого розуміння бізнес-середовища.Нижче наведена таблиця ключових компонентів архітектури LLM-агентів, базована на аналізі джерел.

Таблиця 1.2

Ключові компоненти архітектури LLM-агентів [12]

Компонент	Опис	Приклади використання	Переваги	Виклики
LLM як "мозок"	Основна модель для обробки мови, розуміння контексту та прийняття рішень.	Генерація тексту, аналіз намірів (NLU/NLG).	Висока точність у мовних завданнях.	Галюцинації та упередження.

Продовження табл.1.1

Компонент	Опис	Приклади використання	Переваги	Виклики
Пам'ять	Короткотермінова (контекст поточного завдання) та довготермінова (історія взаємодій).	Персоналізація відповідей у чатботах.	Збереження контексту для складних завдань.	Обмежений розмір вікна контексту.
Модуль планування	Розбиття завдань на підзавдання, chain-of-thought чи рефлексія.	Планування багатоетапних аналізів даних.	Адаптація до динамічних умов.	Неточності в складному міркуванні.
Інструменти та інтеграція	Інтерфейси для зовнішніх систем (API, бази даних, код-інтерпретатори).	Запити до CRM чи веб-пошук.	Розширення можливостей за межі мови.	Ризики безпеки та залежність від API.
Петля зворотного зв'язку	Оцінка результатів, навчання на помилках (reinforcement learning).	Самовдосконалення в дослідницьких агентах.	Постійне покращення.	Високі обчислювальні витрати.

Наведена таблиця демонструє, як окремі компоненти LLM-агентів співпрацюють, утворюючи єдину автономну систему. Архітектури таких агентів можуть суттєво відрізнятися залежно від рівня автономії та складності завдань. Найпростіші - реактивні агенти - діють миттєво на вхідні стимули, не плануючи на майбутнє. Деліберативні вже враховують контекст і проводять внутрішні «міркування», перш ніж прийняти рішення. Найскладніші - когнітивні агенти -

намагаються імітувати людські процеси: сприйняття, навчання, навіть здатність до самоаналізу (метаміркування).

Що стосується організації, то існують одиночні агенти, у яких усі рішення приймає одна централізована система, і мультиагентні системи, де кілька спеціалізованих агентів працюють разом, як команда. Такі підходи використовують, наприклад, у симуляціях корпоративного середовища або соціальних мереж. У дизайн-процесах поширені різні типи агентів: маршрутизатори, що приймають прості тактичні рішення; інструментальні агенти, які викликають зовнішні сервіси через API; і ReAct-агенти, які поєднують планування з виконанням - спочатку обмірковують стратегію, а потім діють. У мультиагентних системах такі спеціалізовані ролі дозволяють ефективно вирішувати складні задачі - від спільної генерації коду до моделювання маркетингових кампаній.

Застосування LLM-агентів постійно розширюється, охоплюючи все більше професійних сфер. У бізнес-аналітиці вони дедалі активніше автоматизують так звану «розмову з даними» - процес, у якому користувач формулює запит природною мовою, а агент самостійно його інтерпретує, розбиває на ланцюжок простіших підпитань, визначає найбільш релевантні джерела інформації та виконує необхідні обчислення. Для підвищення точності та актуальності відповідей такі системи частіше за все використовують RAG-пайплайни (Retrieval-Augmented Generation), що дозволяють поєднувати внутрішні знання моделі з динамічно отриманою зовнішньою інформацією. У наукових дослідженнях LLM-агенти здатні працювати майже як автономні дослідники: вони можуть самостійно збирати огляд літератури, аналізувати експериментальні дані, виявляти прогалини у знаннях і навіть пропонувати нові напрямки подальшої роботи. У галузях інженерії та продукт-дизайну агенти вже сьогодні допомагають генерувати початкові ідеї, аналізувати відгуки користувачів, формулювати вимоги до функціоналу або створювати перші чернетки інтерфейсів, візуалізацій чи інших артефактів. Водночас їхні можливості залишаються обмеженими в тих сферах, де критично важливі глибока креативність, емоційна інтуїція чи

культурний контекст. Мульти-модальні агенти, що вміють працювати не лише з текстом, але й зі зображеннями, аудіо чи відео, відкривають нові можливості - зокрема, у персоналізованих рекомендаційних системах, інтелектуальних голосових асистентах або інструментах для аналізу мультимедійного контенту. Серед реальних прикладів успішного застосування - Einstein GPT від Salesforce, який допомагає маркетологам та аналітикам створювати персоналізовані комунікації, а також спеціалізовані агенти для автоматичної генерації email-кампаній, які адаптуються до поведінки одержувачів у реальному часі.

Переваги використання LLM-агентів дійсно вражаючі. Вони дозволяють скоротити обсяг ручної, повторюваної роботи на 50–70%, що особливо цінно в аналітичних, дослідницьких та операційних процесах. Такі системи легко масштабуються - додавання нових завдань або користувачів часто не вимагає значних інженерних зусиль. Крім того, завдяки здатності розуміти індивідуальні запити та контекст, агенти забезпечують високий рівень персоналізації взаємодії. Проте не слід ігнорувати й істотні ризики. Серед них - схильність моделей до генерації фактологічно хибної інформації, відомої як «галюцинації», що може призводити до помилкових висновків. Також існує небезпека посилення прихованих упереджень, закладених у навчальних даних, що може вплинути на об’єктивність аналітики чи рекомендацій. Додаткові загрози пов’язані з безпекою: при використанні публічних моделей існує ризик витоку конфіденційної інформації, особливо якщо дані передаються у зовнішні сервіси без належного шифрування чи контролю. У складних, неоднозначних або малоймовірних ситуаціях агенти можуть втрачати логічну стабільність, виконуючи неадекватні дії або видаючи суперечливі результати. Для мінімізації цих загроз експерти рекомендують використовувати перевірені фреймворки, такі як LangChain або LlamaIndex, які надають інструменти для безпечної оркестрації, управління пам’яттю та інтеграції з інструментами. Не менш важливим є постійний людський нагляд - особливо на етапах прийняття кінцевих рішень, - а також дотримання чітких етичних протоколів, які регулюють використання штучного інтелекту в професійному середовищі.

У українському науковому середовищі особливу увагу приділяють застосуванню LLM-агентів у дизайн-процесах. Дослідження показують, що найвищий ефект досягається тоді, коли агенти виступають не як заміна людській креативності, а як інтелектуальний помічник, який прискорює рутинні етапи - збір вимог, аналіз конкурентів, генерацію варіантів або тестування гіпотез. Водночас їхня здатність до оригінального, нестандартного мислення залишається обмеженою, оскільки вони не сприймають повністю емоційний, соціальний чи культурний контекст, який часто є вирішальним у дизайні. Тому ключовим принципом ефективного використання залишається доповнення, а не заміщення людини [11].

На 2025 рік очікується подальший розвиток галузевих, або домен-специфічних, LLM-агентів, навчених на спеціалізованих даних у таких сферах, як фінанси, охорона здоров’я, освіта чи право. Такі агенти зможуть працювати з професійною термінологією, регуляторними нормами та внутрішніми процесами конкретної індустрії, що значно підвищить їхню точність і корисність. Також прогнозується глибока інтеграція агентів у платформи бізнес-інтелекту, де вони стануть основою самообслуговуваної аналітики - дозволяючи навіть нетехнічним співробітникам отримувати складні інсайти без залучення аналітиків. Одночасно зростатиме увага до принципів етичного AI: прозорості рішень, підзвітності, захисту приватності та недискримінації. Подальші наукові дослідження, ймовірно, зосереджуватимуться на підвищенні надійності агентів через зменшення галюцинацій, розробку механізмів внутрішньої верифікації та створення гібридних архітектур, що поєднують гнучкість LLM із точністю класичних алгоритмів - наприклад, правил логічного виведення чи статистичних моделей. Окремим напрямом стане розвиток мультиагентних симуляцій, у яких кілька агентів взаємодіють між собою, імітуючи реальні соціальні, організаційні або ринкові процеси. У підсумку, LLM-агенти справді трансформують штучний інтелект, перетворюючи його з пасивного інструменту на активного, автономного учасника професійної діяльності. Проте їхнє безпечне, етичне та ефективне використання буде можливим лише за умови розумного балансу між

технологічними можливостями системи та необхідністю людського контролю, судження та відповідальності.

1.3. Сценарії використання LLM-агентів у комерційному аналізі

Великі мовні моделі перетворилися зі засобів генерації тексту на потужних автономних агентів, здатних не лише розуміти природню мову, а й планувати дії, взаємодіяти з інформаційними системами та виконувати складні аналітичні завдання. У контексті комерційного аналізу, що охоплює широке коло практик - від аналізу поведінки клієнтів і ринкових тенденцій до оптимізації внутрішніх операцій, - такі агенти стають каталізатором трансформації. Вони радикально змінюють підхід до прийняття бізнес-рішень, роблячи аналітичні процеси швидшими, точнішими, масштабнішими та доступними навіть для фахівців без глибоких технічних знань. Цей огляд ґрунтується на аналізі передових наукових досліджень, реальних бізнес-кейсів провідних компаній світу та ключових технологічних трендів, актуальних станом на 2025 рік, і охоплює основні напрями застосування, архітектурні принципи, стратегічні переваги, існуючі обмеження та перспективи розвитку, зокрема в умовах українського ринку [13].

Під LLM-агентом розуміють інтелектуальну систему, яка будується навколо великої мовної моделі, такої як GPT-4, Llama 2 або Mixtral, але значно розширює її функціональність завдяки додатковим компонентам. До таких компонентів належать механізми короткострокової та довгострокової пам’яті, модулі стратегічного планування, а також інтерфейси для взаємодії з зовнішніми інструментами - базами даних, аналітичними платформами, API корпоративних систем. Саме ця архітектура дозволяє агенту автономно виконувати багатоетапні завдання: від інтерпретації неформального запиту від користувача, наприклад «проаналізуй, чому впали продажі в березні», до формування точного SQL-запиту, витягування даних, виявлення аномалій, побудови візуалізації та формулювання конкретних ділових рекомендацій. Дослідження підтверджують, що впровадження подібних систем здатне підвищити продуктивність аналітиків на 50–70 відсотків,

проте для забезпечення надійності необхідно передбачити механізми верифікації, щоб мінімізувати ризики, пов’язані з галюцинаціями моделей або помилковою інтерпретацією контексту [14].

Застосування LLM-агентів у комерційному аналізі охоплює широкий спектр сценаріїв. У сфері автоматизації обробки даних агенти виступають як інтелектуальні посередники між користувачем і базою даних, перетворюючи природномовні запити на коректні технічні команди. Такі системи, як NVIDIA Data Agent або Salesforce Horizon Agent, дозволяють аналітикам отримувати потрібну інформацію без необхідності знання SQL чи інших мов запитів. У моніторингу ринкових умов агенти неперервно аналізують великі потоки даних - з соціальних мереж, новинних сайтів, форумів, навіть баз вакансій - ідентифікуючи нові тренди, зміни у споживчих настроях або зсув у стратегіях конкурентів. Так, South State Bank використовує такі агенти для оперативного коригування маркетингових кампаній на основі реального сприйняття бренду. У сфері клієнтських відносин системи аналізують історію взаємодій, CRM-дані та поведінкові патерни для персоналізації пропозицій, прогнозування відтоку клієнтів або виявлення підозрілих транзакцій, що активно застосовується, наприклад, AT&T у своїй системі кібербезпеки. У фінансовому секторі агенти, розроблені Goldman Sachs, аналізують емоційний тон у медіа та фінансових звітах для оцінки ризиків і прогнозування коливань ринку. У логістиці та управлінні ланцюгами постачання Walmart PAE автоматично обробляє документацію постачальників, витягує структуровані атрибути продуктів із неструктурованих PDF-файлів, що значно прискорює процеси інвентаризації та каталогізації. Навіть у науково-дослідній діяльності, зокрема в фармацевтиці, LLM-агенти аналізують глобальні бази патентів і наукових публікацій, прискорюючи пошук нових застосувань для існуючих молекул.

Архітектура сучасних LLM-агентів будується навколо центральної великої мовної моделі, яка виконує роль «інтелектуального ядра», але значно доповнюється іншими елементами. Пам’ять системи поділяється на короткострокову, що зберігає контекст поточного діалогу, та довгострокову, яка

може містити профілі користувачів, історію завдань або базу знань компанії. Для вирішення складних задач задіюються механізми планування, такі як метод ланцюжка міркувань, коли агент розбиває проблему на підзадачі й послідовно їх вирішує. Одним із ключових компонентів у бізнес-застосуваннях є інтеграція з технологією RAG, що дозволяє агенту посилатися на актуальні, верифіковані джерела, що істотно підвищує точність та достовірність генерованого контенту. Особливу роль відіграють також мультиагентні системи, де окремі агенти спеціалізуються на конкретних завданнях - один збирає дані з різних джерел, інший проводить статистичний або семантичний аналіз, третій формує рекомендації або генерує звіти, спільно забезпечуючи комплексне рішення [15].

Серед найбільш показових реальних прикладів - система Uber Finch, яка дозволяє аналітикам формулювати запити прямо в Slack, а агент автоматично генерує відповідний SQL-код, виконує запит і надає результати у вигляді таблиць або графіків. Walmart PAE, як уже згадувалося, обробляє тисячі документів постачальників щодня, автоматично стандартизуючи інформацію про товари. Навіть середні компанії сьогодні можуть використовувати відкриті моделі, такі як Llama 2, у поєднанні з RAG, щоб опрацювати великі корпуси текстів - наприклад, одна компанія обробила понад 2800 бізнес-статей і досягла точності понад 85 відсотків у видобутку ключових сутностей і подій. У Україні подібні технології вже знаходять практичне застосування: створюються інтелектуальні бізнес-чатботи для внутрішніх комунікацій, автоматизуються аналітичні процеси у відділах маркетингу та фінансів, а наукові групи, зокрема у Вінницькому національному технічному університеті, активно розробляють власні мультиагентні платформи, орієнтовані на підтримку прийняття рішень у малих і середніх підприємствах.

Основні переваги LLM-агентів полягають у їхній здатності масштабуватися, пристосовуватися до індивідуальних потреб користувачів, а також у значному скороченні часу та витрат на аналітичні процеси. Наприклад, компанія Klarna повідомила про економію понад 6 мільйонів доларів завдяки впровадженню агента-помічника для обслуговування клієнтів. Проте існують і суттєві виклики.

Серед них - ризик генерації неточної або вигаданої інформації, питання захисту конфіденційних даних, відсутність чітких регуляторних норм у сфері штучного інтелекту, а також інженерна складність інтеграції таких систем у існуючу IT-інфраструктуру компаній. Тому експерти рекомендують починати з обмежених пілотних проектів із чітким визначенням меж компетенції агента та обов'язковим застосуванням підходу «людина в центрі», за якого фахівець здійснює верифікацію всіх критичних висновків або рішень [16].

На 2025 рік прогнозується подальший розвиток доменно-спеціалізованих LLM-агентів, які будуть навчені на даних конкретних галузей - ритейлу, банківської справи, логістики, фармацевтики. Очікується також глибша інтеграція агентів із популярними BI-платформами, такими як Power BI або Tableau, що дозволить безпосередньо вбудовувати інтелектуальні функції у звичні для аналітиків інструменти. Зростатиме увага до питань етики, пояснюваності та прозорості рішень, прийнятих на основі AI. Особливо актуальною стане розробка гібридних архітектур, які поєднують сильні сторони великих мовних моделей з перевіреними класичними методами статистичного та економетричного аналізу. У контексті України, де активно формується локальна AI-екосистема, такі технології поступово стають доступними для бізнесу будь-якого розміру, відкриваючи нові горизонти для інновацій у сфері комерційного аналізу та стратегічного управління.

У світлі цих тенденцій LLM-агенти перестають бути лише інструментом автоматизації і перетворюються на стратегічний актив, який змінює саму природу бізнес-аналітики. Вони не просто прискорюють обробку даних, а вносять якісні зміни у розуміння ринку, клієнтів і внутрішніх процесів. Здатність отримувати глибокі інсайти за лічені секунди, оперативно реагувати на зміни та приймати зважені рішення на основі комплексного аналізу стає ключовим фактором конкурентоспроможності. Для українських компаній це означає можливість не лише наздогнати міжнародних конкурентів, але й запропонувати унікальні, локалізовані рішення, засновані на власних даних і специфіці ринку. У майбутньому саме компанії, які зможуть ефективно інтегрувати AI-агентів у свою

аналітичну інфраструктуру, будуть задавати темп інновацій та формувати нові стандарти бізнес-ефективності.

Цей перехід від пасивного аналізу до активної інтелектуальної підтримки прийняття рішень має потенціал не лише оптимізувати існуючі процеси, але й згенерувати нові бізнес-моделі. Наприклад, компанії можуть запропонувати своїм клієнтам персоналізовані прогнози попиту як сервіс - особливо цінно це для малих виробників або продавців на маркетплейсах, які не мають власних аналітичних команд. Такий підхід перетворює дані з внутрішнього ресурсу на зовнішній продукт, створюючи додатковий потік доходу.

Крім того, з розвитком локальних моделей і зростанням обчислювальної потужності пристроїв користувачів стає реальним сценарій, коли складний AI-аналіз виконується безпосередньо на робочій станції аналітика або навіть на мобільному пристрої. Це відкриває шлях до створення повністю автономних, офлайн-орієнтованих систем, які не залежать від інтернет-з’єднання, хмарних API або зовнішніх постачальників. Для українських підприємств, які працюють у умовах нестабільної інфраструктури або підвищених вимог до кібербезпеки, такий підхід стає не просто зручним, а стратегічно необхідним.

Особливу роль у цьому контексті відіграє підхід до навчання моделей. Замість того щоб сподіватися на універсальну модель, яка «все знає», майбутнє належить гібридним системам, у яких загальна LLM співпрацює з вузькоспеціалізованими модулями - наприклад, з економетричними моделями для прогнозування, з онтологіями для розуміння галузевої термінології або з правилами, визначеними експертами. Такий синтез символічного та субсимволічного штучного інтелекту дозволяє поєднати гнучкість нейромереж із надійністю логічних систем, мінімізуючи ризики галюцинацій і забезпечуючи пояснюваність рішень.

Для України це має особливе значення. На тлі активного розвитку вітчизняного IT-сектора, зростання експортної орієнтації українських брендів і підвищення вимог до цифрової суверенітету, локальні, безпечні та адаптовані AI-рішення стають не просто технологічним вибором, а елементом національної

економічної стратегії. Підтримка розробки відкритих, гнучких платформ, які можуть працювати в умовах обмежених ресурсів та високих вимог до безпеки, дозволить утримувати контроль над власними даними, уникати технологічної залежності та формувати власний шлях цифрової трансформації.

Важливо також розуміти, що успіх впровадження LLM-агентів залежить не лише від технологій, але й від організаційної культури. Компанії повинні інвестувати не тільки в програмне забезпечення, але й у розвиток «аналітичної зрілості» своїх команд - навчати співробітників ставити правильні запити, критично оцінювати результати, розуміти обмеження моделей і використовувати інсайти для реальних дій. Лише за умови такого підходу AI-агенти перестануть бути «чорним ящиком» і стануть надійними партнерами у щоденній роботі.

У довгостроковій перспективі межі між аналітиком, менеджером і AI-системою будуть стиратися. Аналітик буде все більше виконувати роль стратега, який формулює гіпотези та інтерпретує інсайти, а рутинні операції - збір даних, первинний аналіз, генерація звітів - будуть повністю автоматизовані. Це дозволить зосередитися на творчих, стратегічних аспектах управління, підвищуючи загальну ефективність бізнесу.

Таким чином, LLM-агенти в комерційному аналізі - це не просто черговий технологічний тренд, а фундаментальний зсув у парадигмі роботи з даними. Вони роблять аналітику швидкою, інтуїтивною, глибокою та доступною. А для України, з її сильними IT-традиціями, динамічним стартап-середовищем і зростаючою увагою до цифрової безпеки, цей напрямок має всі шанси стати не лише джерелом внутрішньої трансформації, але й експортним продуктом майбутнього - технологічним рішенням, розробленим в Україні, але корисним для бізнесу по всьому світу.

1.4. Архітектурні підходи до побудови систем на основі агентів

Великі мовні моделі та штучний інтелект стали справжнім каталізатором глибокої трансформації агентних систем, виводячи їх далеко за межі примітивних

реактивних механізмів, що домінували в агентно-орієнтованому програмуванні протягом попередніх десятиліть. Сучасні агенти більше не обмежуються простою реакцією на зовнішні подразники - вони формують цілеспрямовану, адаптивну поведінку, взаємодіють у складних мережах, координують дії з іншими агентами й здатні до автономного навчання в умовах постійно змінного середовища. У цьому огляді розглядається еволюція архітектурних підходів до побудови таких систем: від класичних концептуальних моделей, які заклали теоретичний фундамент агентно-орієнтованого мислення, до передових сучасних конфігурацій, що інтегрують великі мовні моделі та ґрунтуються на принципах мультиагентної взаємодії. Особлива увага приділяється компонентам, які забезпечують здатність агента сприймати контекст, приймати зважені рішення, взаємодіяти з іншими сутностями та безпечно функціонувати в реальних умовах. Ключовим принципом аналізу є постійний фокус на трикутнику «автономія–ефективність–безпека», оскільки саме баланс між цими трьома вимірами визначає практичну цінність і надійність агентної системи. Дослідження показують, що найбільш ефективні системи не прагнуть до надмірної складності, а навпаки - будуються на простих, добре документованих патернах, легко компонуються, піддаються тестуванню та завжди передбачають механізми контролю, верифікації та етичного обмеження. Архітектури сучасних AI-агентів спеціально оптимізовані для глибокої інтеграції з великими мовними моделями, що дозволяє їм не лише ефективно опрацьовувати неструктуровану інформацію, але й приймати зважені рішення, діяти автономно й одночасно дотримуватись вимог регуляторів, корпоративної політики та етичних норм [17].

У своїй основі агент залишається автономною програмною сутністю, яка сприймає стан навколишнього середовища через сенсорні інтерфейси, виконує дії за допомогою ефекторів, обмінюється повідомленнями з іншими агентами та формує поведінку, спрямовану на досягнення заздалегідь визначених цілей. Проте з розвитком технологій значно збагатилася класифікація таких сутностей. Найпростіші рефлексні агенти реагують лише на поточний стан у повністю спостережуваному середовищі, не маючи внутрішньої пам’яті. Більш складні

моделі володіють внутрішньою репрезентацією світу, що дозволяє їм діяти навіть у умовах неповної або частково прихованої інформації. Агенти, орієнтовані на цілі, здатні до планування послідовності дій, яка веде до бажаного результату, навіть якщо це вимагає виконання проміжних підзадач. Ще більш просунуті системи, засновані на функції корисності, не просто визначають шлях до цілі, а оцінюють множину можливих альтернатив і обирають ту, що максимізує очікуваний виграш з урахуванням ризиків, ресурсів і обмежень. Найвищий рівень інтелектуальності досягається в навчаючих агентах, які можуть адаптуватися до нових, раніше невідомих ситуацій, використовуючи механізми підкріпленого навчання, самонавчання або мета-навчання. Однією з найвпливовіших концептуальних рамок залишається модель BDI (Belief-Desire-Intention), у якій поведінка агента формалізується через тріаду: переконання (belief) - це поточна модель світу, бажання (desire) - можливі цілі, яких він може прагнути, а наміри (intention) - конкретні дії, які вже прийняті для досягнення однієї з цілей. Для моделювання динаміки таких процесів часто застосовується темпоральна логіка. Поруч із BDI існує інша, хоч і суміжна, але чітко відмежована парадигма - акторна модель, де кожна сутність є пасивною і активується лише при отриманні повідомлення, а вся взаємодія заснована на адресній передачі даних, що забезпечує високу паралелізацію та відмовостійкість [15].

Різноманітність методологій розробки агентних систем відображає глибину та складність предметної області. Об’єктно-орієнтований підхід моделює агентів як активні об’єкти зі станом, поведінкою та інкапсульованими методами, часто базуючись на BDI-архітектурі. Інженерія знань пропонує використовувати онтології для формального опису предметної області, що полегшує спільне розуміння між агентами та підвищує інтероперабельність. Формальні методи, засновані на математичній логіці, дозволяють строго верифікувати властивості системи - зокрема її коректність, безпеку та відповідність специфікації. На практиці, однак, найчастіше використовуються змішані підходи, які поєднують гнучкість інженерних методів із строгістю формальних засобів. У сучасному контексті LLM виконує роль центрального «мозку» агента, відповідаючи за

міркування, інтерпретацію запитів, генерацію планів і внутрішнє моделювання світу. Зовнішні інструменти - API, бази даних, аналітичні сервіси, обчислювальні бібліотеки - розширюють його функціональність у реальному світі, перетворюючи агента з пасивного генератора тексту на активного учасника ділового процесу. Інструкції, або промпти, задають поведінкові рамки, визначаючи стиль, обмеження, етичні принципи та цілі взаємодії. Петлі зворотного зв'язку дозволяють системі аналізувати результати своїх дій, оцінювати їх ефективність і навчатися на власному досвіді. До ключових компонентів сучасного агента належать сама LLM як ядро прийняття рішень, набір інструментів для взаємодії з оточенням, система інструкцій, часто структурована у вигляді шаблонів для масштабованості, механізм пам'яті - як короткострокової (контекст діалогу), так і довгострокової (історія завдань, профілі користувачів), а також так звані «огорожі» (guardrails), які запобігають небажаним діям, забезпечуючи дотримання правил безпеки, конфіденційності та регуляторних вимог. Важливим елементом інфраструктури є пайплайни для поглинання даних (ingestion pipelines), які трансформують неструктуровані тексти, документи або логи в структуровані векторні представлення, що зберігаються у векторних базах даних. Це лягає в основу технології RAG (Retrieval-Augmented Generation), яка дозволяє агенту під час генерації відповіді динамічно звертатися до актуальних, верифікованих знань, що істотно підвищує точність, достовірність і релевантність результатів [18].

З точки зору архітектури агентні системи можна класифікувати за кількістю агентів та характером координації. Одиночні агенти функціонують у циклі «планування → дія → спостереження → рефлексія», що робить їх придатними для лінійних, добре визначених робочих процесів, де кожен крок логічно випливає з попереднього, а результат легко верифікується [19]. Натомість мультиагентні системи (MAS) відкривають новий рівень складності й гнучкості, дозволяючи агентам не лише взаємодіяти, а й співпрацювати або навіть конкурувати за ресурси або цілі [20]. Такі системи можуть бути гомогенними, коли всі агенти мають однакову структуру й функції, або гетерогенними, де кожен агент виконує спеціалізовану роль - наприклад, один збирає дані, інший аналізує, третій формує

рекомендації або генерує звіти [21]. Взаємодія між агентами може мати кооперативний характер, спрямований на досягнення спільної мети, або ж бути конкурентною, коли інтереси агентів розходяться, як у випадку ринкових симуляцій або оптимізації розподілу ресурсів [16]. За організаційною структурою MAS поділяються на централізовані, де єдиний контролер координує дії всіх учасників - це забезпечує простоту управління, але створює єдину точку відмови; децентралізовані (peer-to-peer), де кожен агент приймає рішення автономно, що підвищує відмовостійкість, гнучкість і масштабованість; та гібридні, які поєднують переваги обох підходів, дозволяючи локальну автономію з глобальним надзором. У контексті AI-агентів, орієнтованих на великі мовні моделі, з'явилися також специфічні патерни оркестрації, адаптовані до природи мовних моделей. Послідовна обробка використовується для завдань із чіткою логічною залежністю між етапами, коли кожен наступний крок залежить від результату попереднього. Конкурентна (паралельна) архітектура застосовується, коли критичним є час виконання, і завдання можуть бути розпаралелені без втрати цілісності результату [22]. У патерні «груповий чат» агенти-експерти обмінюються думками у форматі діалогу, колективно досягаючи згоди щодо оптимального рішення, що імітує роботу експертної ради [23]. У підході «handoff» завдання динамічно передається між спеціалізованими агентами залежно від поточного контексту або складності підзадачі, що забезпечує найвищий рівень професійної точності [24]. Найбільш гнучким є патерн «magentic» (від «magnetic team»), який використовується для відкритих, неструктурованих завдань - наприклад, стратегічного планування або кризового менеджменту - де немає чіткого плану заздалегідь, а натомість планування відбувається ітеративно, «на ходу», з постійною переоцінкою ситуації, генерацією гіпотез і адаптацією стратегії на основі нових даних. Такий підхід найкращим чином відображає природу людського мислення в умовах невизначеності і стає основою для створення по-справжньому інтелектуальних систем, здатних до саморозвитку та колективного вирішення складних проблем.Таблиця ключових типів архітектур систем на основі агентів наведена нижче (табл. 1.3).

Таблиця 1.3

Ключові типи архітектур [15]

Тип архітектури	Опис	Переваги	Виклики	Приклади застосування
Реактивна	Агенти реагують на стимули без планування, на основі поточного сприйняття.	Швидкість, простота, низькі обчислювальні витрати.	Обмежена адаптивність до змін, відсутність пам'яті.	Автономні пилососи, дрони для уникнення перешкод.
Деліберативна	Включає міркування, моделі стану та планування для цілей.	Гнучкість у складних задачах, довгострокове планування.	Високі обчислювальні витрати, повільність.	Автономні автомобілі, цифрові асистенти для планування.
Гібридна	Комбінує реактивні та деліберативні елементи.	Баланс швидкості та планування, адаптивність.	Складність інтеграції, потенційні конфлікти.	Складські роботи, смарт-доми.
BDI (Belief-Desire-Intention)	Керується переконаннями, бажаннями та намірами з темпоральною логікою.	Ефективність у динамічних середовищах, моделювання людського мислення.	Складність реалізації, залежність від точних моделей.	Логістичні системи, ройові дрони.
Централізована MAS	Центральний контролер координує агентів.	Легкість керування, чітка ієрархія.	Точка відмови, обмежена масштабованість.	Координація роботів на складі.
Децентралізована MAS	Агенти взаємодіють peer-to-peer без центру.	Стійкість, масштабованість, самоорганізація.	Координаційні конфлікти, амбігвіти комунікації.	Смарт-грід и, ройовий інтелект у логістиці.

Продовження табл. 1.3

Тип архітектури	Опис	Переваги	Виклики	Приклади застосування
Гібридна MAS	Комбінує централізовані та децентралізовані елементи.	Баланс контролю та гнучкості.	Складність дизайну, потенційні затори.	Будівельні кластери, моніторинг середовища.
Послідовна оркестрація	Лінійний ланцюг агентів для поступової обробки.	Чіткі залежності, детермінованість.	Затримки в послідовних кроках.	Генерація контрактів, юридичні системи.
Конкурентна оркестрація	Паралельна робота агентів для незалежних аналізів.	Швидкість, комплексний огляд.	Агрегація результатів, синхронізація.	Аналіз акцій з кількох перспектив.
Group Chat	Спільне обговорення в чаті з менеджером.	Колаборація, консенсус, аудит.	Ризик циклів, часові затримки.	Брейнштормінг для проєктів, оцінка пропозицій.
Handoff	Динамічне передавання завдань між агентами.	Адаптивність до контенту, інтелектуальне маршрутування.	Ризик помилкових маршрутів, втрата контексту.	Підтримка клієнтів з тріажем.
Magentic	Динамічне планування для відкритих проблем з ledger завдань.	Гнучкість у складних сценаріях, аудит планів.	Потенційні затори, високі витрати.	Реагування на інциденти в IT, SRE-автоматизація.

Інтеграція сучасних AI-агентів у існуючі бізнес- та технічні системи реалізується переважно через два взаємодоповнюючі механізми зв’язку з оточенням: API-запити, що реалізують pull-модель, і вебхуки, які втілюють push-модель. У першому випадку агент самостійно ініціює запит до зовнішнього джерела, коли йому потрібна конкретна інформація - наприклад, дані про клієнта з CRM або поточні курси валют із фінансової платформи. У другому - зовнішні

системи автоматично надсилають події агенту в реальному часі, щойно відбувається зміна стану, наприклад, нове замовлення в e-commerce системі або аварійне сповіщення з промислового датчика. Такий підхід забезпечує гнучку, але надійну взаємодію з різноманітними джерелами даних - від корпоративних ERP- і CRM-систем до IoT-мереж, фінтех-платформ і хмарних сховищ [25]. Критичним елементом інфраструктури, що лежить в основі ефективної роботи агентів, є уніфікований шар retrieval, який нормалізує, індексує та семантично об’єднує неструктуровані (документи, логи, електронні листи) та структуровані (таблиці, транзакції, метадані) дані з усіх доступних джерел, формуючи єдиний, узгоджений контекст для подальшого аналізу та прийняття рішень [26].

У бізнес-середовищі AI-агенти вже активно застосовуються для автоматизації складних, але рутинних процесів. Наприклад, у фінансовій сфері вони обробляють банківські виписки, розпізнають сутності (платники, отримувачі, номери договорів), класифікують типи платежів і автоматично звіряють їх із внутрішніми документами. У взаємодії з ERP-системами агенти формують аналітичні звіти, моніторять ключові показники ефективності (KPI), прогнозують потреби в ресурсах і виявляють аномалії в операційній діяльності. У сфері фінансового аналізу вони використовуються для автоматичного виявлення підозрілих транзакцій, класифікації витрат за категоріями, аналізу структури бюджету та візуалізації фінансових потоків. Практичні впровадження показують, що такі системи здатні зменшити кількість людських помилок приблизно на 30 відсотків, одночасно підвищуючи швидкість обробки документів і аналітичних запитів у кілька разів, що особливо цінно при роботі з великими обсягами даних у реальному часі [27].

У технічних галузях - зокрема в промисловій автоматизації, будівництві, логістиці та робототехніці - агентні системи використовуються для координації груп автономних пристроїв у складних, динамічних середовищах. Наприклад, дрони під керуванням мультиагентної системи можуть проводити інспекцію енергетичної інфраструктури, моніторити будівельні майданчики або організовувати доставку на складах. У таких сценаріях ключову роль відіграє

підхід, заснований на ройовому інтелекті (Swarm Intelligence) - парадигмі, натхненній колективною поведінкою природних систем, таких як колонії мурах, зграї птахів або рої риб. Алгоритми, розроблені ще в 1990-х роках, зокрема Ant Colony Optimization (ACO, 1992) та Particle Swarm Optimization (PSO, 1995), досі залишаються фундаментальними інструментами для вирішення NP-складних задач оптимізації: від прокладання найефективніших маршрутів для флоту транспорту до розподілу завдань між роботами або координації безпілотників у повітрі. Застосування ройового інтелекту охоплює широкий спектр сфер: від оптимізації логістичних маршрутів та маршрутизації дронів до моделювання фінансових ринків, точного землеробства, військових операцій з використанням безпілотників (де ринок, за прогнозами, зросте з 12,55 мільярдів доларів у 2022 році до 35,6 мільярдів до 2030 року) та екологічного моніторингу великих територій, зокрема лісових масивів чи водойм.

Серед ключових переваг агентних систем - їхня висока гнучкість, здатність до горизонтального масштабування, самоорганізація та відмовостійкість. Оскільки кожен агент діє автономно, відмова окремої одиниці не призводить до критичного збою всієї системи - інші агенти можуть перерозподілити завдання або знайти альтернативні шляхи до цілі. Проте така архітектура супроводжується і значними викликами. Серед них - координаційні конфлікти між агентами, коли їхні цілі чи дії суперечать одна одній; неоднозначність у комунікації, особливо при інтерпретації природномовних повідомлень або неповних команд; ризик генерації «галюцинацій» у великих мовних моделях, що може призводити до помилкових рішень; потенційний витік конфіденційних даних під час обробки, зберігання або передачі інформації; а також високі обчислювальні та фінансові витрати на підтримку інфраструктури, особливо при роботі з локально розгорнутими LLM. Для мінімізації цих ризиків рекомендується впроваджувати комплексні механізми контролю: технічні «огорожі» (guardrails), що включають модерацію вмісту, семантичну та логічну валідацію вихідних даних, обмеження доступу до чутливих ресурсів; постійний людський нагляд у критичних сценаріях (human-in-the-loop), де остаточне рішення залишається за оператором; а також ітеративне тестування,

верифікація результатів і аудит кожного етапу життєвого циклу системи. Сучасні фреймворки, такі як AutoGen, CrewAI, MetaGPT та LangChain, надають розробникам потужні інструменти для реалізації ролевої колаборації між агентами, графової оркестрації складних робочих процесів, ефективного управління короткостроковою та довгостроковою пам’яттю, а також інтеграції з інструментами зовнішньої аугментації знань [30].

У українському середовищі архітектури на основі AI-агентів уже виходять за межі пілотних або наукових проектів і знаходять реальне промислове застосування. Зокрема, активно впроваджуються рішення для інтеграції інтелектуальних асистентів у корпоративні ERP-системи, зокрема в платформи на кшталт IT-Enterprise. Такі агенти автоматизують широкий спектр операцій - від обробки внутрішніх наказів, договорів і первинної бухгалтерської документації до аналізу фінансових потоків, генерації регуляторних звітів, підготовки аналітичних дашбордів і навіть інтерактивного супроводу користувачів під час складних операцій. Особлива увага в таких системах приділяється використанню технології RAG (Retrieval-Augmented Generation), що дозволяє агентам звертатися лише до внутрішніх, попередньо верифікованих корпоративних знань, уникати витоку конфіденційної інформації та забезпечувати високу точність та відповідність нормативним вимогам. Наукові колективи, зокрема в Вінницькому національному технічному університеті та інших провідних установах, активно досліджують потенціал ройового інтелекту не лише в класичних інженерних прикладах, а й у моделюванні складних динамічних систем - від оптимізації транспортних потоків у містах і аналізу механічних коливань конструкцій до моделювання біологічних мереж, еволюційних процесів і екологічних моделей [13].

На 2025 рік очікується поступовий перехід від універсальних, загальнопризначених агентів до галузевих (домен-специфічних) систем, навчених на глибоких знаннях конкретних сфер - фінансів, виробництва, охорони здоров’я, агробізнесу чи навіть юриспруденції [14]. Такі агенти зможуть вільно оперувати профільною термінологією, враховувати специфіку нормативної бази, адаптуватися до внутрішньої бізнес-логіки компанії та надавати рекомендації, що

враховують специфіку галузі. Також прогнозується широке поширення гібридних архітектур, які поєднують сильні сторони великих мовних моделей - гнучкість, здатність до міркувань та обробки неструктурованої інформації - з перевіреними методами класичного машинного навчання, символічного штучного інтелекту та експертних систем, що забезпечують точність, інтерпретованість і відтворюваність результатів. Серед ключових напрямів подальшого розвитку - етичний AI, зокрема пояснюваність рішень, прозорість джерел інформації та уникнення системних упереджень; безпечна інтеграція з популярними BI-платформами, такими як Power BI, Tableau чи Looker, для створення самообслуговуваної аналітики, доступної навіть нетехнічним користувачам; та підвищення надійності систем через багаторівневу верифікацію, автономну самодіагностику та механізми відновлення після помилок. Особливу роль у формуванні майбутніх агентних мереж відіграватимуть ройовий інтелект - як основа для створення стійких, адаптивних та розподілених систем, здатних до колективного рішення завдань - і мультимодальність, що дозволить агентам одночасно обробляти та інтерпретувати різні типи даних: текст, зображення, аудіо, відео та структуровані таблиці. Це розширить горизонти їхнього застосування, перетворюючи їх не просто на інструменти для аналізу, а на повноцінних інтелектуальних партнерів, здатних активно брати участь у стратегічних і операційних процесах реального бізнес-середовища.

1.5. Виклики та обмеження

Великі мовні моделі та побудовані на їх основі агенти дійсно трансформують ландшафт бізнес-аналітики, надаючи небачені раніше можливості для автоматизації складних аналітичних процесів - від глибокого інтерпретування неструктурованих даних, таких як відгуки клієнтів, внутрішні звіти чи новинні стрічки, до прогнозування майбутнього попиту, виявлення прихованих ризиків і формування стратегічних рекомендацій на рівні керівництва. Завдяки здатності працювати з природною мовою, LLM-агенти роблять аналітику доступною

ширшому колу співробітників, знижуючи бар’єр між даними та рішеннями. Проте, незважаючи на їхній великий потенціал, реальне впровадження таких систем у корпоративне середовище часто стикається зі значними, а іноді й фундаментальними перешкодами. Ці виклики виходять далеко за межі чисто технічних питань: вони охоплюють операційні, організаційні, етичні та регуляторні виміри, формуючи комплексний ризикований ландшафт, який вимагає системного, міждисциплінарного підходу до управління. Цей огляд спирається на аналіз останніх наукових досліджень, реальних кейсів провідних компаній та висновків галузевих експертів, зосереджуючись на ключових бар’єрах, що заважають ефективному застосуванню LLM-агентів у бізнес-аналітиці. Особлива увага приділяється причинам виникнення цих проблем, їхнім наслідкам для якості прийняття рішень та практичним стратегіям їх мінімізації. Дослідження однозначно підтверджують: без належного управління ці обмеження можуть не лише звести нанівець значні інвестиції в штучний інтелект, але й призвести до серйозних фінансових збитків, порушення конфіденційності, стратегічних помилок або навіть репутаційної кризи через хибну інтерпретацію даних.

Серед найглибших перешкод - внутрішні технічні обмеження самих великих мовних моделей. Одне з найважливіших - обмежена пам’ять. Короткострокова пам’ять LLM жорстко обмежена розміром контекстного вікна, яке, навіть у найсучасніших моделях, рідко перевищує кілька десятків тисяч токенів. У довгих аналітичних сесіях, коли агент послідовно обробляє запити, генерує проміжні висновки, формує рекомендації та взаємодіє з користувачем, критична інформація, отримана на початку сесії, може бути просто втрачена. У контексті бізнес-аналітики це може призвести до фрагментованого, неповного аналізу: наприклад, агент може забути про сезонні патерни, виявлені на ранніх етапах, або про специфічні параметри клієнтського сегмента, що змінює інтерпретацію поточних даних. Для зберігання довгострокового контексту необхідно інтегрувати зовнішні системи - зокрема, векторні бази даних, які зберігають семантичні представлення (ембеддинги) інформації. Проте ефективний пошук у таких сховищах вимагає складних алгоритмів оцінки подібності, оптимізації індексів,

постійного оновлення ембеддингів у міру надходження нових даних, що істотно збільшує обчислювальну складність, фінансові витрати та інженерну навантаженість.

Не менш критичною є обмежена здатність LLM до справжнього стратегічного планування та логічного міркування. Хоча підходи на кшталт Chain-of-Thought або ReAct дозволяють моделі імітувати аналітичний процес через послідовне розбиття задачі на підзадачі, вони залишаються вразливими до початкових помилок у декомпозиції. Якщо агент неправильно визначить ключові фактори на першому етапі - наприклад, при визначенні драйверів зростання продажів або виявленні ризиків у постачальницькій ланці, - весь подальший аналіз може піти у хибне русло. Це призводить не просто до неактуальних висновків, а до марних витрат ресурсів, контрпродуктивних ініціатив або навіть стратегічних помилок. Ще серйознішою слабкістю залишаються галюцинації - здатність моделі генерувати правдоподібні, але фактологічно хибні твердження, які часто важко виявити без глибокої предметної експертизи. У бізнес-середовищі це може виявитися катастрофічним: неправильна оцінка обсягів продажів, хибна класифікація клієнтських ризиків, помилкові висновки щодо ефективності маркетингової кампанії або навіть неправильно інтерпретовані регуляторні вимоги можуть спонукати керівництво до прийняття рішень, що завдають реальних фінансових збитків або шкоди репутації.

Операційні виклики виникають на стику технології та реальних бізнес-процесів, де теорія часто стикається з хаосом повсякденної практики. Безпека є тут центральною проблемою. Агент, що має доступ до API корпоративних систем - будь то ERP, CRM, фінансові платформи або інструменти управління складом, - несе у собі потенційну загрозу, навіть за відсутності зловмисних намірів. Наприклад, він може неправильно інтерпретувати запит користувача і надіслати платіж на невірний рахунок, змінити статус замовлення, видалити критичні записи або витягти конфіденційні дані про клієнтів, партнерів або працівників. У регульованих галузях - таких як фінанси, охорона здоров’я, страхування або державне управління - такі ризики особливо небезпечні, що

робить обов’язковим не лише технічну, але й процесну захисту. Це включає впровадження механізмів попередньої валідації команд, повного аудиту всіх дій (з фіксацією запиту, контексту, виводу та результату), а також статичних та динамічних «огорож» (guardrails), які жорстко обмежують діапазон можливих автономних дій агента, забезпечуючи дотримання принципу найменших привілеїв.

Інтеграція з існуючими інструментами також є далеко не тривіальною задачею. Навіть при наявності добре документованих API кожен інструмент потребує чітких специфікацій, узгоджених форматів даних, механізмів обробки помилок та постійного тестування. Помилка в налаштуванні пайплайну - наприклад, між агентом і CRM-системою - може призвести до витягу неповних, дубльованих, застарілих або суттєво спотворених даних, що, у свою чергу, дискредитує весь подальший аналіз і призводить до втрати довіри з боку користувачів. До цього додається проблема упереджень, вбудованих у саму модель під час навчання на гігантських корпусах текстів, що часто містять стереотипи, нерівності чи історичні перекоси. Якщо тренувальний корпус містить упередження щодо певних груп клієнтів, регіонів або продуктів, агент може несвідомо відтворювати їх у рекомендаціях - наприклад, при оцінці кредитоспроможності, сегментації ринку або прогнозуванні лояльності. Особливу складність становить інтеграція з legacy-системами - старими платіжними шлюзами, внутрішніми базами даних без сучасних RESTful інтерфейсів, закритими корпоративними платформами або навіть Excel-файлами, що масово використовуються в операційній роботі. Такі інтеграції часто вимагають розробки спеціальних адаптерів, проксі-сервісів, ETL-пайплайнів або навіть повного реінжинірингу частин інфраструктури, що значно збільшує час і вартість впровадження.

У мультиагентних системах складність лише зростає експоненційно. Координація між агентами - одна з найважливіших інженерних задач. Коли один агент відповідає за збір даних, інший - за їхній аналіз, а третій - за генерацію звіту або рекомендації, їм необхідний спільний семантичний, часовий та фактичний контекст. За відсутності надійного механізму обміну інформацією, узгодження

цілей та синхронізації стану легко виникають розриви: один агент може працювати зі застарілими даними, інший - втратити зв’язок із загальною метою, а третій - згенерувати рекомендації, що суперечать висновкам колег. Архітектури на кшталт «майстер-слуга», паралельні конфігурації або діалогові патерни, реалізовані в таких фреймворках, як CrewAI, LangGraph або AutoGen, хоча й забезпечують певну гнучкість, часто виявляються важкими у масштабуванні, тестуванні та керуванні, особливо в умовах високої динаміки - наприклад, під час моніторингу фінансових ринків, реагування на кризові події або координації логістичних операцій у реальному часі.

Ще одна серйозна проблема - відсутність гарантій відповідності дій агентів стратегічним цілям компанії. Автономний агент, оптимізований лише для локальної метрики - наприклад, швидкості обробки запитів, точності сентимент-аналізу або кількості згенерованих інсайтів, - може ігнорувати глобальні бізнес-параметри, такі як прибутковість, довгострокова лояльність клієнтів, екологічна відповідальність або репутаційні ризики. Це призводить до фрагментації стратегії, неефективного розподілу ресурсів і розбіжностей між тактичними діями та стратегічними цілями. Крім того, взаємодія між агентами та людьми залишається недосконалою: моделі часто не розпізнають іронію, сарказм, культурні нюанси чи емоційний підтекст, не мають емоційного інтелекту, а їхня постійна доступність може порушувати робочі процеси, створюючи когнітивне перевантаження, знижуючи продуктивність і підриваючи довіру з боку користувачів, які швидко втрачають віру в «автоматичного експерта».

Нарешті, етичні та регуляторні питання формують новий, критичний рівень відповідальності. Центральною дилемою є питання підзвітності: хто несе юридичну відповідальність за помилку агента, якщо його прогноз щодо кредитного ризику, ринкової волатильності або ефективності логістичної схеми призвів до значних збитків? У разі відсутності чітких аудиторських стежок, які документують кожен крок міркування, використані джерела, внутрішні перевірки та остаточне рішення, компанія може опинитися в уразливому правовому положенні. Це особливо актуально на тлі посилення регуляторного тиску -

зокрема, через ЄС AI Act (2024), директиви CFPB у США, а також подібні ініціативи в Україні, де активно розглядаються законодавчі механізми регулювання високоризикових систем штучного інтелекту. Інтеграція автономних агентів у корпоративну ієрархію також породжує фундаментальні правові та етичні питання: чи може агент приймати рішення щодо стратегічних інвестицій, розподілу бюджету, ціноутворення або навіть участі у процесі найму? Такі сценарії вимагають чіткого визначення меж автономії, обов'язкового людського нагляду (human-in-the-loop) та механізмів екстреного відключення. Нарешті, конфіденційність даних залишається критичним ризиком: під час обробки особистої, корпоративної або регуляторної інформації агенти можуть ненавмисно порушити вимоги GDPR, Закону України «Про захист персональних даних», HIPAA або інших локальних та міжнародних стандартів, що веде не лише до штрафів та судових позовів, але й до серйозної втрати довіри з боку клієнтів і партнерів.

Таким чином, успішне впровадження LLM-агентів у бізнес-аналітику вимагає не просто технологічної експертизи або наявності потужної інфраструктури, а комплексного підходу, який поєднує інженерну надійність, операційну безпеку, етичну відповідальність, глибоке розуміння регуляторного середовища та здатність до постійного навчання та адаптації. Лише за умови такого балансу інновації можуть стати джерелом стійкої конкурентної переваги, а не новим, прихованим джерелом системних ризиків, які загрожують не лише ефективності, але й самій стабільності бізнесу.

РОЗДІЛ 2. АРХІТЕКТУРА ДОДАТКУ ДЛЯ КОМЕРЦІЙНОГО АНАЛІЗУ НА ОСНОВІ LLM-АГЕНТІВ

2.1. Визначення вимог та цілей системи

Метою розробки є створення локальної веб-системи, призначеної для комплексного аналізу клієнтських відгуків та прогнозування обсягів продажів. Система повністю функціонує на пристрої користувача, не залежить від хмарних обчислювальних сервісів і забезпечує повну конфіденційність даних, що є критично важливим у контексті обробки комерційно чутливої інформації, зокрема при роботі з відгуками, ідентифікаторами продуктів, фінансовими показниками або внутрішніми бізнес-метриками. Для обробки текстових даних використовуються відкриті великі мовні моделі, які розгортаються локально за допомогою платформи Ollama. Такий підхід виключає передачу даних на зовнішні сервери, зберігаючи всю інформацію в межах корпоративної мережі або особистого комп'ютера. Це надає користувачеві повний контроль над процесом аналізу, включаючи можливість гнучко керувати категоріями відгуків - видаляти, перейменовувати, створювати нові або об'єднувати існуючі - через простий, зрозумілий і інтуїтивний веб-інтерфейс, доступний навіть для нетехнічних співробітників.

Система приймає вхідні дані у форматі CSV, що містять текстові відгуки клієнтів, дати їх публікації та, за необхідності, додаткові метадані. Після завантаження файлу відбувається автоматична первинна класифікація: всі записи розподіляються за трьома базовими групами - позитивні відгуки, негативні відгуки та описи способів використання продукту. Ці початкові категорії слугують лише орієнтиром і не є фіксованими або обов'язковими. Користувач має повне право вільно редагувати схему категоризації, адаптуючи її до специфіки свого бізнесу, наприклад, виділяючи такі теми, як «якість упаковки», «швидкість доставки», «служба підтримки» або «цінова політика». Подальша класифікація всіх відгуків виконується виключно на основі цієї відкоригованої схеми, що

забезпечує високу релевантність, точність та практичну цінність аналітичних висновків, оскільки система аналізує дані не за узагальненими шаблонами, а за логікою, визначеною самим бізнесом.

Окрім текстової обробки, система виконує сезонний аналіз часових рядів продажів для заданого товару за його унікальним ідентифікатором ASIN. Вона застосовує статистичні методи, зокрема декомпозицію часових рядів, для виявлення трендів, сезонних коливань та циклічних патернів. На основі цих даних формується обґрунтований прогноз майбутніх продажів, який може охоплювати період від одного до 365 днів. Прогноз супроводжується довірчим інтервалом, що дозволяє оцінити ступінь невизначеності і приймати зважені рішення - наприклад, щодо планування запасів, бюджетування маркетингових кампаній або оптимізації виробничих потужностей. Цей аналіз виконується локально, без залучення зовнішніх сервісів, що гарантує конфіденційність фінансових даних.

Усі результати аналізу - включаючи класифіковані відгуки, статистичні метрики, прогнозні значення та рекомендації - можна зберегти локально у форматі JSON. Це дозволяє легко інтегрувати дані з іншими внутрішніми системами, такими як BI-платформи, бази знань або кастомізовані дашборди. За бажанням користувача передбачена додаткова можливість синхронізації даних із Google Sheets - цей функціонал активується лише за явного підтвердження користувача, вимагає авторизації через OAuth 2.0 і не є частиною основного аналітичного процесу. Таким чином, вибір архітектури залишається за користувачем: він може працювати повністю офлайн або обмежено використовувати хмарні сервіси лише для зручності зберігання. Для забезпечення зручності взаємодії реалізовано інтерактивний прогрес-бар, який візуалізує поточний етап обробки - від завантаження файлу, через класифікацію, аналіз часових рядів, до завершення прогнозування, забезпечуючи прозорість, контроль над процесом і зниження когнітивного навантаження на користувача.

Важливим принципом архітектури є виключно локальна обробка даних: використання зовнішніх хмарних API, таких як OpenAI, Gemini, Claude або інші комерційні сервіси, під час основного аналізу строго заборонене на рівні

реалізації. Це виключає будь-які ризики, пов’язані з витоком інформації, порушенням конфіденційності або несанкціонованим зберіганням даних. Для забезпечення безпеки та стабільності реалізовано комплексну валідацію вхідних даних - зокрема, перевірку формату ASIN (який має відповідати стандарту Amazon), обмеження кількості днів прогнозу в межах 1–365, фільтрацію завантажуваних файлів лише до формату CSV, а також аналіз структури файлу на наявність обов’язкових колонок. Це мінімізує ризики, пов’язані з помилковими, пошкодженими або потенційно шкідливими вхідними даними.

Усі необхідні моделі - зокрема mistral для генерації тексту та формулювання рекомендацій, nomic-embed-text для створення семантичних ембеддингів, які використовуються при класифікації та кластеризації, та nuextract для витягування структурованої інформації з неструктурованих текстів - завантажуються локально через Ollama і працюють у повній ізоляції від зовнішніх мереж. Після початкового завантаження моделей система здатна функціонувати без підключення до Інтернету, що робить її придатною для використання навіть в ізольованих середовищах, таких як закриті корпоративні мережі або офлайн-офіси. Система сумісна з основними операційними системами - Windows, Linux та macOS - і не вимагає встановлення складного серверного програмного забезпечення, баз даних або контейнерних оркестраторів. Запуск здійснюється через простий Python-скрипт або виконуваний файл, що значно спрощує розгортання.

Інтерфейс системи розроблено з урахуванням потреб нетехнічних користувачів: результати подаються у зрозумілій, діловій формі, що поєднує статистичні зведення, часові графіки, візуалізації сезонних патернів та текстові рекомендації, сформульовані простою, зрозумілою мовою без жаргону. Архітектура системи є модульною - компоненти бази даних, аналітичної обробки, веб-інтерфейсу, валідації, експорту та допоміжних утиліт чітко відокремлені один від одного. Це не лише спрощує тестування та підтримку, але й дозволяє окремо оновлювати або розширювати будь-який елемент - наприклад, замінити модель аналізу відгуків, додати новий алгоритм прогнозування або інтегрувати додаткові формати експорту, не переписуючи всю систему.

Такий підхід забезпечує високий рівень автономності, захисту даних, гнучкості налаштувань і зручності використання, роблячи систему придатною для застосування в корпоративних середовищах зі строгими вимогами до безпеки, конфіденційності та відповідності нормативним стандартам, зокрема GDPR, HIPAA або Закону України «Про захист персональних даних». Вона дозволяє бізнесу отримувати глибокі інсайти з клієнтських відгуків і продажів, не жертвуючи контролем над своїми даними, що робить її особливо цінною для компаній, які працюють у регульованих галузях або мають високі стандарти інформаційної безпеки.

2.2. Проектування модульної архітектури

Система AI-аналізу відгуків і прогнозування продажів побудована на основі мікросервісної, модульної архітектури, що забезпечує їй високу гнучкість, легко масштабується під змінні бізнес-потреби та спрощує технічну підтримку. Головна мета архітектурного рішення - організувати ефективну, надійну та безпечну обробку різноманітних аналітичних завдань, від первинної обробки текстових відгуків до побудови складних статистичних моделей прогнозування, при цьому мінімізуючи залежності між компонентами. Такий підхід дозволяє системі функціонувати як повноцінний веб-додаток з графічним інтерфейсом, так і у вигляді окремих консольних скриптів або API-сервісів, що можуть використовуватися в інших внутрішніх системах або автоматизованих робочих процесах.

Уся архітектура спирається на кілька ключових інженерних принципів, що забезпечують якість і довгострокову життєздатність рішення. Кожен модуль відповідає лише за одну добре визначену функцію - наприклад, завантаження даних, аналіз тексту або візуалізацію - і мінімально залежить від інших компонентів. Внутрішня структура кожного модуля, навпаки, є щільно пов’язаною за змістом, що сприяє зрозумілості коду та зменшенню когнітивного навантаження під час розробки. Крім того, архітектура передбачає механізм розширення

функціональності через плагіни: нові аналітичні методи, формати експорту або інтеграції з зовнішніми сервісами можуть бути додані без модифікації ядра системи, що значно спрощує її подальший розвиток та адаптацію до нових вимог.

Ядром системи є модуль базової інфраструктури, який виступає у ролі технічного фундаменту для всіх інших компонентів. Він надає спільні сервіси, необхідні для стабільної роботи: уніфіковане підключення до реляційної бази даних PostgreSQL з пулінгом з'єднань для ефективного використання ресурсів, стандартизовані функції безпечного виводу інформації - як для звичайного прогресу, так і для обробки помилок, включаючи логування та відображення статусу в інтерфейсі. Особлива увага приділена підтримці міжнародних текстів: модуль забезпечує коректну обробку Unicode-символів, що дозволяє працювати з відгуками українською, російською, англійською та іншими мовами без втрати даних або порушення кодування. Також реалізовано централізований менеджер конфігурацій, який зберігає всі налаштування - від параметрів моделей і шляхів до локальних файлів до API-ключів для додаткових інтеграцій - у єдиному, легко керованому форматі, що забезпечує гнучкість та безпеку.

Модуль обробки даних відповідає за підготовку вхідної інформації до подальшого аналізу. Він приймає файли у форматі CSV, що містять відгуки клієнтів, дати публікації та, за необхідності, ідентифікатори продуктів (ASIN). На цьому етапі виконується комплексна валідація: перевірка коректності формату ASIN, валідність дат, наявність числових значень у відповідних полях, а також аналіз структури файлу на повноту даних. Після цього текстові поля очищаються від шуму - видаляються HTML-теги, спеціальні символи, повторювані пробіли - і нормалізуються за допомогою лематизації та зниження регістру. Для прискорення повторних операцій передбачено механізм кешування проміжних результатів: якщо користувач повторно завантажує той самий файл або виконує аналіз з незначними змінами, система використовує збережені дані замість повного повторного прогону, що істотно економить час і обчислювальні ресурси.

Центральну роль відіграє модуль AI-аналізу, який об'єднує три взаємопов'язані, але функціонально незалежні інструменти. Аналізатор

сезонності працює з часовими рядами продажів за останні один–три роки, виявляючи періодичні піки, спади, тренди та незвичайні викиди. Він використовує статистичні методи, такі як STL-декомпозиція, для виділення сезонної, трендової та залишкової компонент, що дозволяє не лише зрозуміти минулу динаміку, але й виявити приховані патерни. Аналізатор коментарів застосовує методи обробки природної мови для класифікації відгуків на позитивні та негативні, виявлення конкретних прикладів використання товару, групування за тематичними кластерами та оцінки емоційного забарвлення за допомогою моделей на основі трансформерів. Двигун прогнозування, у свою чергу, формує обґрунтовані прогнози майбутніх продажів, розраховує довірчі інтервали, аналізує ринкову динаміку в контексті минулих подій (сезонність, маркетингові кампанії, зовнішні шоки) і пропонує практичні рекомендації щодо управління запасами, ціноутворення або планування закупівель.

Особливу увагу приділено модулю категоризації, який реалізує двоетапний підхід, що поєднує автоматизацію з людською експертизою. На першому етапі система автоматично генерує початковий набір категорій для трьох типів відгуків: позитивних, негативних та описів використання. Ці категорії формулюються на основі найчастіших тем, виявлених у текстах, і зберігаються у локальному файлі у форматі JSON. Користувачеві пропонується переглянути цей перелік через веб-інтерфейс і внести зміни: видалити нерелевантні, перейменувати неточні, об’єднати дублі або створити нові, відповідні специфіці бізнесу. На другому етапі система повторно класифікує всі коментарі саме за цією відредагованою схемою. Такий підхід забезпечує високу точність аналізу, оскільки об’єднує потужність штучного інтелекту у виявленні патернів із глибоким предметним розумінням людини, яка знає нюанси свого продукту, ринку або клієнтів.

Для взаємодії з користувачем розроблено зручний веб-інтерфейс на основі Flask. Він реалізує REST API з чітко визначеними ендпоінтами для завантаження файлів, запуску аналізу, збереження категорій, отримання результатів та відстеження прогресу. Інтерфейс є адаптивним - коректно відображається на різних пристроях, від настільних комп’ютерів до планшетів. Довготривалі

операції, такі як аналіз великих наборів відгуків або побудова прогнозів, супроводжуються оновленням статусу в реальному часі: користувач бачить, на якому етапі перебуває обробка, скільки записів вже проаналізовано та коли очікувати завершення. Це забезпечує прозорість процесу і підвищує довіру до системи.

Результати аналізу перетворюються на зрозумілі, інформативні графічні форми завдяки спеціалізованому модулю візуалізації. Він генерує сезонні графіки у вигляді барчартів, лінійних діаграм і теплових карт, що дозволяють швидко оцінити динаміку продажів за місяцями чи тижнями. Ринкові частки візуалізуються через кругові діаграми, KPI - через спідометри, а вплив різних факторів (наприклад, типу відгуку на лояльність) - через горизонтальні барчарти та радарні діаграми. Усі графіки інтерактивні, підтримують масштабування, фільтрацію та експорт.

Усі результати можна експортувати в різних форматах залежно від мети. Для подальшої автоматичної обробки призначено структурований JSON, для роботи в Excel - CSV, для офіційного подання - PDF-звіти з корпоративним стилем, для інтерактивного аналізу всередині компанії - HTML-звіти з можливістю фільтрації та сортування. Також передбачена інтеграція з Google Sheets: користувач може синхронізувати вибрані дані з електронною таблицею, що особливо зручно для маркетологів або аналітиків, які вже використовують цей інструмент у своїй роботі. Ця функція активується лише за явного підтвердження користувача і вимагає авторизації через безпечний OAuth-протокол.

Система підтримує широку інтеграцію з зовнішніми сервісами, зберігаючи при цьому контроль над даними. Вона може працювати з локальними відкритими моделями через Ollama, що гарантує конфіденційність, або - за бажанням - з хмарними AI-сервісами Google Generative AI для підвищення потужності аналізу. Також передбачено можливість збору даних з маркетплейсів, таких як Amazon або eBay, передачі інформації у CRM-системи для подальшої роботи з клієнтами та надсилання структурованих результатів до інструментів бізнес-аналітики, таких як Power BI або Tableau, для побудови корпоративних дашбордів.

Процес обробки даних розподілено на кілька чітких логічних потоків. Аналіз сезонності є лінійним: користувач вводить ASIN, система завантажує продажі, виконує декомпозицію, будує прогноз і візуалізує результат. Аналіз коментарів є ітеративним: після автоматичної генерації категорій система очікує втручання користувача, що підвищує точність фінального результату. Прогнозування продажів фокусується на статистичній надійності: кожен крок супроводжується розрахунком довірчих інтервалів, крос-валідацією та автоматичною перевіркою на наявність аномалій.

Архітектура реалізована таким чином, що модулі взаємодіють між собою через чітко визначені JSON-інтерфейси, використовують патерн Dependency Injection для управління залежностями та сповіщення на основі подій (event-driven messaging) для синхронізації стану. Це забезпечує слабку зв’язаність між модулями і високу внутрішню зв’язаність у межах кожного з них. Нові функції додаються через абстрактні базові класи та механізм динамічної реєстрації плагінів, що дозволяє легко розширювати функціональність без втручання в основний код. Уся система легко налаштовується через зовнішні конфігураційні файли та змінні середовища, що спрощує розгортання в різних умовах - від локального робочого місця до корпоративного сервера.

Такий підхід забезпечує масштабованість, простоту обслуговування, здатність до незалежного тестування модулів, відмовостійкість і легку інтеграцію нових AI-моделей. У технологічному плані система побудована на Python 3.9+, Flask, PostgreSQL та SQLAlchemy на бекенді, використовує Ollama для локальних моделей, NLTK і spaCy для NLP, Pandas і scikit-learn для аналітики, Matplotlib і Chart.js для графіків, а також Docker для контейнеризації та Git для управління версіями. Усе це разом створює надійну, автономну, безпечну та конфіденційну платформу для глибокого аналізу бізнес-даних, яка відповідає сучасним вимогам корпоративної IT-безпеки та відповідності регуляторним стандартам, зокрема GDPR або Закону України «Про захист персональних даних».

2.3. Вибір технологічного стеку

Вибір технологічного стеку для системи AI-аналізу відгуків і прогнозування продажів ґрунтувався не на модних трендах, а на реальних, чітко визначених потребах проекту. Серед ключових вимог - повна автономія обробки даних, висока точність аналітичних висновків, простота супроводу, надійність та здатність до подальшого розвитку. Кожен інструмент у стеку був обраний після глибокого аналізу його можливостей, сумісності з іншими компонентами та відповідності конкретним завданням, поставленим архітектурою системи.

Основою backend-логіки стала мова Python - не лише через її широке поширення, але й завдяки надзвичайно багатій екосистемі для наукових обчислень, аналітики та машинного навчання. Python дозволяє ефективно реалізовувати складні алгоритми обробки тексту та часових рядів, швидко прототипувати нові підходи, а також легко інтегруватися з сучасними AI-інструментами. Його читабельність і простота синтаксису значно прискорюють розробку, знижують кількість помилок і полегшують підтримку - що особливо критично для модульної системи, де кожен компонент має бути добре ізольованим та чітко документуваним.

Як веб-фреймворк було обрано Flask - легковагий, мінімалістичний інструмент, який не нав’язує жорсткої структури проекту. На відміну від більш монолітних рішень, таких як Django, Flask надає розробнику повну свободу у проектуванні API та організації коду. Це ідеально узгоджується з цілями проекту, де основна цінність полягає не в складності інтерфейсу, а в глибині аналітичних модулів. Простота Flask дозволяє зосередитися на реалізації бізнес-логіки, а не на конфігурації фреймворку, і легко додавати нові ендпоінти по мірі розширення функціональності.

Для роботи з табличними даними використовується добре зарекомендований тандем бібліотек Pandas і NumPy. Ці інструменти давно стали індустріальним стандартом у сфері аналітики: вони забезпечують ефективну фільтрацію, агрегацію, перетворення та підготовку даних, що є необхідним етапом як перед текстовим аналізом, так і перед прогнозуванням часових рядів. Як система управління базами даних обрано PostgreSQL - зрілу, надійну реляційну СУБД з

підтримкою складних SQL-запитів, транзакцій, зовнішніх ключів із каскадними операціями, а також механізмів забезпечення цілісності даних. Ці властивості критично важливі для бізнес-застосувань, де помилка в даних може призвести до помилкових стратегічних рішень.

Особливу увагу було приділено AI-компонентам системи. Для аналізу текстових відгуків реалізовано гібридний підхід. Основна робота відбувається локально за допомогою платформи Ollama, яка дозволяє запускати відкриті великі мовні моделі прямо на пристрої користувача, без залежності від інтернету чи сторонніх хмарних сервісів. Це забезпечує повну конфіденційність комерційно чутливої інформації та усуває постійні витрати на API. Водночас, для задач, що вимагають підвищеної точності або більшої обчислювальної потужності - наприклад, для складного семантичного аналізу - передбачено опціональну інтеграцію з Google Generative AI API. Цей функціонал активується лише за згодою користувача, що забезпечує баланс між автономією та гнучкістю.

Обробка природної мови поєднує перевірені open-source інструменти з власними алгоритмами. Бібліотека NLTK використовується для базових операцій - очищення тексту, токенізації, видалення стоп-слів. Проте класифікація відгуків, виявлення тональності та виділення тематичних кластерів - наприклад, «проблеми з упаковкою» чи «зручність використання» - реалізовані через спеціалізовані, кастомні рішення, навчені на даних, характерних для роздрібної торгівлі. Це дозволяє уникнути універсальних, але часто неточних моделей, і досягти вищої релевантності аналітичних висновків у конкретному бізнес-контексті.

Візуалізація результатів реалізована на двох рівнях. На стороні клієнта для інтерактивних графіків використовується Chart.js - легка, швидка та добре документована бібліотека, яка не вимагає складних залежностей і забезпечує плавну роботу навіть із великими обсягами даних. На сервері для генерації статичних зображень - наприклад, для PDF-звітів - застосовується Matplotlib. Цей інструмент дозволяє точно контролювати стиль, колірну палітру та компонування графіків, забезпечуючи професійний вигляд фінальних документів. Такий поділ гарантує одночасно динамічність інтерфейсу та якість експортованої аналітики.

Фронтенд системи навмисно зроблено мінімалістичним: без React, Vue або інших складних SPA-фреймворків. Використовується лише чистий HTML, CSS та vanilla JavaScript. Це гарантує швидке завантаження, простоту підтримки, стабільність роботи та незалежність від постійних оновлень сторонніх бібліотек. Інтерфейс зосереджений на зручності користувача: акцент зроблено на зрозумілому представленні даних, а не на складних анімаціях чи візуальних ефектах.

Експорт результатів підтримує кілька форматів. Для створення професійних PDF-звітів використовується бібліотека ReportLab, яка дозволяє точно контролювати верстку, включати таблиці, графіки, логотипи та текстові рекомендації у єдиний документ. Для синхронізації з Google Sheets застосовується gspread - проста, але надійна бібліотека, що забезпечує безпечний обмін даними через OAuth2 і підтримує роботу з великими таблицями.

На інфраструктурному рівні реалізовано підтримку Docker для контейнеризації всієї системи. Це гарантує, що застосунок буде працювати ідентично на будь-якому пристрої - незалежно від операційної системи, версій бібліотек або інших локальних налаштувань. Для управління вихідним кодом використовується Git, що дозволяє ефективно організовувати командну розробку, відстежувати історію змін, тестувати нові функції в окремих гілках та легко повертатися до стабільних версій у разі виникнення проблем.

Тестування системи реалізовано за допомогою вбудованих інструментів Python - unittest та pytest. Поєднання unit-тестів для окремих функцій - наприклад, валідації формату ASIN або логіки класифікації - і інтеграційних тестів для перевірки взаємодії між модулями - наприклад, між аналітичним компонентом і базою даних - дозволяє підтримувати високу якість коду без навантаження системи сторонніми тестовими фреймворками.

У підсумку, технологічний стек сформований за принципом «правильний інструмент для правильної задачі». Жоден компонент не доданий «на випадок» або через популярність - кожен має чітку мету, вирішує конкретну проблему і вписується в загальну логіку системи. Такий підхід забезпечує не лише поточну

ефективність, але й довгострокову життєздатність, простоту масштабування, легкість супроводу та здатність адаптуватися до майбутніх вимог бізнесу, зокрема в умовах активного розвитку AI-екосистеми в Україні.

На рис. 2.1 наведено загальний стек, що використовується.

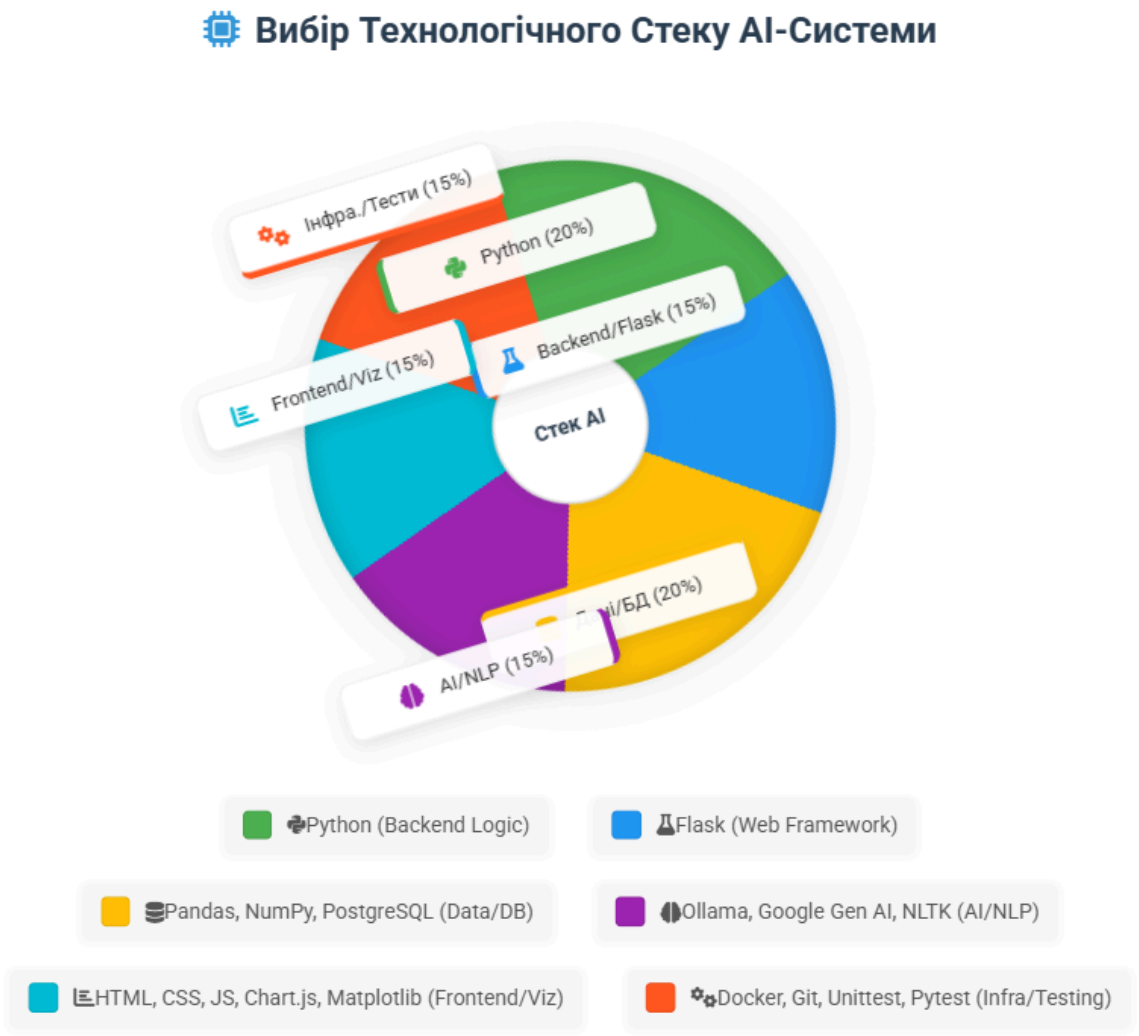


Рис. 2.1 Технологічний стек

2.4. Модель забезпечення безпеки та конфіденційності

Система AI-аналізу відгуків і прогнозування продажів працює з інформацією, що має високу комерційну цінність: обсяги продажів, внутрішні аналітичні звіти, стратегічні прогнози та дані про клієнтів. Тому її архітектура безпеки побудована за принципом «захист на кожному рівні» - від вхідних даних

до фінального звіту. Безпека не додається як додатковий шар, а є невід'ємною частиною кожної підсистеми.

Основу підходу становить принцип найменших привілеїв: кожен компонент отримує лише ті права, які дійсно необхідні для його роботи. Наприклад, модуль аналізу текстів ніколи не має прямого доступу до фінансових записів - він отримує лише очищені, анонімізовані тексти. Веб-інтерфейс, API-шлюзи, аналітичні модулі та база даних працюють в ізольованих контекстах, що значно зменшує потенційну поверхню атаки.

Особлива увага приділяється захисту персональних даних. Навіть якщо відгук містить ім'я, email або адресу, ця інформація автоматично видаляється ще до початку аналізу. Система використовує комбінацію правил, регулярних виразів і простих NLP-методів для надійного маскування PII (Personally Identifiable Information). Анонімізовані дані зберігаються окремо в зашифрованому сховищі, де відновити початкову інформацію неможливо навіть технічно.

Шифрування застосовується на всіх етапах. Усі з'єднання - як між користувачем і сервером, так і між внутрішніми модулями - захищені TLS. Чутливі дані, що зберігаються, шифруються за допомогою сучасних криптографічних алгоритмів, а ключі зберігаються окремо - у захищених змінних середовища, поза кодом, логами чи системою контролю версій. API-ключі сторонніх сервісів ніколи не потрапляють у відкритий доступ і не логуються.

Безпека веб-інтерфейсу базується на перевірених практиках: усі вхідні дані валідуються і на клієнті (для зручності), і на сервері (для реального захисту). Система захищена від типових веб-уразливостей - SQL-ін'єкцій, XSS, CSRF. Завантаження файлів обмежене лише форматом CSV, із перевіркою розміру та додатковим аналізом на наявність підозрілого вмісту.

Система не передбачає складного управління користувачами, але до адміністративних функцій доступ можливий лише з певних IP-адрес або за допомогою одноразових токенів. Внутрішні модулі взаємодіють між собою через облікові записи з мінімальними правами, що ускладнює переміщення атакуючого усередині системи навіть у разі компрометації одного компонента.

Захист AI-компонентів також враховано. Локальні моделі (через Ollama) працюють в ізольованому середовищі без виходу в інтернет, що запобігає витоку моделей або даних. При роботі з хмарними сервісами, такими як Google Generative AI, усі дані проходять через проксі-шлюз, який фільтрує чутливу інформацію перед відправкою назовні - наприклад, замінюючи реальні назви продуктів на узагальнені позначки.

Логування організовано з урахуванням конфіденційності: усі записи автоматично очищаються від токенів, ключів і особистих даних. У промисловому середовищі рівень логування мінімальний - лише необхідні події для моніторингу. У середовищі розробки - детальніший, але з тим самим захистом чутливої інформації.

Процеси резервного копіювання інтегровані в безпекову модель: резервні копії шифруються «на льоту» і зберігаються в різних географічних локаціях. Доступ до них захищений додатковою автентифікацією, а процедури відновлення регулярно тестуються, щоб гарантувати їхню працездатність у разі реального інциденту.

Система включає централізований моніторинг безпеки, який аналізує поведінку в реальному часі. Вона виявляє аномалії - наприклад, повторні спроби доступу, незвичайне навантаження або запити до захищених ресурсів - і може автоматично блокувати підозрілі IP-адреси або тимчасово вимикати функції.

Відповідність законодавству - не післямова, а вбудована функція. Система підтримує механізми, необхідні для відповідності GDPR: зокрема, можливість повного видалення персональних даних за запитом. Архітектура дозволяє легко додавати нові вимоги - наприклад, для відповідності українському Закону «Про захист персональних даних» або іншим регуляторним нормам.

Хоча фізична безпека лежить за межами програмного рішення, система спроєктована для розгортання в будь-якому середовищі - від закритих дата-центрів до сертифікованих хмар. Крім того, вона підтримує роботу в повністю ізольованих мережах без доступу до інтернету, що критично для роботи з найбільш чутливою інформацією.

Безпека розширюється і на користувацький досвід: інтерфейс містить контекстні підказки, попередження при завантаженні файлів і автоматичне попереднє сканування даних. Це допомагає користувачам уникати помилок і формує культуру безпечної роботи без додаткових навчань.

Нарешті, безпека - це не разовий крок, а неперервний процес. Система підтримує регулярний пентестинг, автоматичне оновлення компонентів і швидку інтеграцію нових захисних механізмів. Архітектура дозволяє вносити зміни без переписування основного коду, що робить захист гнучким і адаптивним.

У підсумку, модель безпеки не обіцяє «абсолютного захисту» - такий неможливий у принципі. Натомість вона забезпечує багаторівневу оборону, яка мінімізує ймовірність успішної атаки, обмежує її наслідки і дозволяє швидко реагувати на інциденти. Це створює надійну основу для роботи з конфіденційними бізнес-даними на всіх етапах - від завантаження відгуку до генерації стратегічного прогнозу.

РОЗДІЛ 3. ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ АРХІТЕКТУРИ

3.1. Підготовка даних та тестовий сценарій

Процес підготовки даних для системи AI-аналізу відгуків і прогнозування продажів був тщательно спланований і реалізований з урахуванням реальних умов бізнес-середовища. Збір даних розпочався з використання практичних, несинтетичних джерел - зокрема, з експортованих даних з Amazon Seller Central за останні два роки. Ці дані містили структуровану інформацію про кожну транзакцію: дату продажу, унікальний ASIN-код товару, кількість проданих одиниць, загальний виторг, регіон доставки та категорію продукту. Крім цього, до набору було додано контекстну інформацію, яка могла впливати на динаміку продажів: календар сезонних акцій, історію маркетингових кампаній, святкові періоди та значущі макроекономічні події - такі як зміни курсу валют, енергетичні кризи або зміни в податковому законодавстві. Це дозволило моделі враховувати не лише внутрішні, але й зовнішні фактори при прогнозуванні.

Для текстового аналізу було зібрано понад 25 тисяч відгуків із офіційного датасету Amazon Customer Reviews, охоплюючи 18 місяців. Кожен запис містив повний текст коментаря, числову оцінку від одного до п’яти зірок, дату публікації, анонімізований ідентифікатор користувача та ASIN товару. Такий обсяг забезпечив достатню різноманітність у тональності, тематиці та стилістичному оформленні, що було критично важливим для навчання та тестування моделей обробки природної мови.

Проте сирий набір даних виявився далеко не готовим до безпосереднього аналізу. У продажних записах були зафіксовані дублікати, пропущені значення, помилкові введення та статистичні аномалії. Для вирішення цих проблем було розроблено серію спеціалізованих Python-скриптів, які автоматично проводили очищення. Для виявлення викидів у часових рядах продажів застосовувався метод міжквартильного розмаху (IQR), що дозволило відфільтрувати нетипові транзакції

- наприклад, замовлення з надзвичайно великим обсягом, спричинені помилками оператора або разовими промоакціями, які не відображають трендову поведінку.

Текстові відгуки потребували ще більш глибокої обробки. Для цього було створено багатоетапний конвеєр попередньої обробки. На першому етапі відбувалося видалення HTML-тегів, емоджі, спеціальних символів, а також автоматичне виявлення та усунення повторюваних фраз, які не несли семантичного навантаження. Далі текст нормалізувався: всі символи перетворювалися на нижній регістр, видалялися стоп-слова, а лексеми зводилися до своїх морфологічних основ за допомогою стемінгу. Цей підхід не лише зменшив обсяг шуму, але й суттєво підвищив якість подальшого семантичного аналізу, оскільки дозволив моделям краще уловлювати базові значення слів, незалежно від їхньої граматичної форми.

Щоб забезпечити надійність на вході, усі дані проходили автоматичну валідацію вже на етапі завантаження. Перевірялася коректність формату ASIN (точно 10 символів), діапазон оцінок (від 1 до 5), логічна сумісність дат - наприклад, що дата відгуку не передує даті продажу, або що кількість проданих одиниць не може бути від’ємною. Записи, що не відповідали цим правилам, автоматично ізольовувалися в окремий архів для подальшого ручного перегляду, що запобігало поширенню помилок у основну базу даних.

Для тестування було сформовано репрезентативний набір, що відображав різноманітність реальних бізнес-сценаріїв. Було виділено три ключові категорії товарів, кожна з яких представляла окремий тип попиту: електроніка - з яскраво вираженою сезонністю, одяг - з динамікою, залежною від моди та погодних умов, та побутова хімія - зі стабільним, майже рівномірним попитом. Для кожної категорії було обрано по 10–15 ASIN-кодів, для яких була доступна повна історія продажів і відгуків. Підсумковий тестовий набір склав приблизно 5 тисяч транзакцій і 8 тисяч відгуків - достатньо для статистично надійної оцінки, але без надмірного навантаження на систему, що було важливо для ітеративного тестування.

Особливу увагу було приділено захисту конфіденційності. Усі потенційно персональні дані, які могли випадково потрапити у відгуки - такі як імена, email-адреси, телефони чи поштові адреси - автоматично виявлялися за допомогою регулярних виразів та спеціалізованих NER-моделей, а потім замінювалися на унікальні, але семантично нейтральні ідентифікатори. Після цього всі файли підписувалися контрольною сумою та шифрувалися за допомогою AES-256 перед завантаженням у тестове середовище.

Тестовий сценарій був побудований як повноцінний імітаційний бізнес-цикл, що відтворював типові завдання продуктового менеджера. Він складався з п’яти послідовних етапів. На першому етапі перевірялася стійкість системи до неідеальних вхідних даних: створювалися CSV-файли з пропущеними колонками, некоректними значеннями, пошкодженими рядками, щоб оцінити, наскільки добре система виявляє та обробляє помилки.

На другому етапі виконувався аналіз сезонності. Для трьох представників різних категорій - кондиціонерів (літній пік), обігрівачів (зимовий пік) та засобів побутової хімії (стабільний попит) - система мала автоматично виявити сезонні патерни, визначити пікові місяці та згенерувати прогноз на наступний період. Результати потім порівнювалися з реальними даними для обчислення точності.

Третій етап був присвячений двоетапній категоризації відгуків. Спочатку AI-агент автоматично генерував початкову схему категорій, групуючи відгуки на позитивні, негативні та приклади використання. Потім імітувалося втручання користувача: об’єднання надто дрібних груп, перейменування неточних позначень, додавання нових категорій. Після цього система перекласифіковувала всі відгуки відповідно до оновленої схеми, і оцінювалася якість фінальної класифікації.

Четвертий етап - прогнозування продажів - передбачав побудову моделі на основі двох років історичних даних із метою прогнозування обсягів на наступний квартал. Оцінювалася не лише абсолютна точність, але й адекватність довірчих інтервалів, а також здатність моделі адаптуватися до різких змін - наприклад, до раптового спаду попиту, спричиненого глобальною кризою або негативними подіями у медіа.

На п’ятому етапі перевірялася функціональність експорту та інтеграції. Тестувалися всі канали виведення результатів: генерація звітів у форматах JSON, CSV, PDF, синхронізація з Google Sheets через API, коректність інтерактивних графіків на основі Chart.js. Також імітувалася одночасна робота кількох користувачів, що дозволило оцінити стабільність системи під навантаженням та ефективність механізмів паралельної обробки.

Для кожного етапу були визначені чіткі кількісні метрики. Точність прогнозів оцінювалася за середньою абсолютною процентною помилкою (MAPE) та коефіцієнтом детермінації (R^2). Якість категоризації визначалася через точність, повноту та F1-міру на основі вибірки, яку попередньо вручну розмітили експерти. Продуктивність системи вимірювалася за часом виконання, обсягом використаної оперативної пам’яті та навантаженням на процесор.

Окрім технічних тестів, система проходила перевірку на реальних бізнес-ситуаціях: планування запасів перед сезоном, діагностика причин падіння продажів, аналіз чутливості попиту до зміни ціни, виявлення проблем з якістю через зростання негативних відгуків. Це дозволило оцінити не лише коректність роботи алгоритмів, але й їхню практичну цінність для прийняття управлінських рішень.

Весь процес тестування був детально задокументований. Кожен крок, отриманий результат, виявлена помилка чи незручність у взаємодії фіксувалися в спеціальній системі відстеження. Дефекти класифікувалися за пріоритетом - від критичних (що блокують роботу) до косметичних (що впливають лише на UX), що дозволило ефективно планувати наступні ітерації розробки.

Найважливішим результатом цього етапу стало не лише підтвердження технічної працездатності системи, але й демонстрація її здатності генерувати дієві, контекстуалізовані бізнес-інсайти. Аналітичні висновки виявилися не лише точними з точки зору метрик, але й корисними для реальних рішень - наприклад, передбачення дефіциту товару за два тижні до його настання або виявлення ранніх ознак погіршення сприйняття якості. Саме ця здатність перетворювати дані на дії і

є остаточною метою всієї системи, і саме вона була успішно підтверджена в ході тестування.

.

3.2. Реалізація архітектури та інтеграція агентів

Реалізація архітектурного задуму розпочалася зі створення гнучкого, добре структурованого каркасу, у якому кожен компонент функціонує як самодостатній модуль, але водночас здатен ефективно взаємодіяти з іншими частинами системи. Центральну роль у цій архітектурі відіграє оркестратор - спеціалізований координуючий компонент, який керує виконанням усіх аналітичних сценаріїв. Він реалізований як окремий клас з чітко визначеними методами, що дозволяють як запускати повний цикл аналізу від початку до кінця, так і виконувати окремі операції на запит - наприклад, лише генерацію категорій відгуків, лише аналіз сезонних коливань або лише прогнозування майбутніх продажів. Такий підхід забезпечує максимальну гнучкість у використанні системи залежно від поточних потреб користувача.

Аналітичне ядро системи складається з трьох спеціалізованих AI-агентів, кожен із яких відповідає за певний аспект аналізу. Перший - агент аналізу сезонності - працює з часовими рядами продажів, виявляючи циклічні патерни різної тривалості: щоденні, тижневі, місячні або річні. Він автоматично адаптується до специфіки окремих товарних груп: наприклад, для сезонних товарів, таких як кондиціонери або обігрівачі, він чітко виявляє пікові періоди, тоді як для товарів щоденного попиту, таких як засоби гігієни, визначає стабільний, майже рівномірний рівень продажів. Для досягнення високої точності використовується гібридний підхід, що поєднує класичні статистичні методи, такі як декомпозиція часових рядів, із легкими, адаптивними моделями машинного навчання, які автоматично налаштовуються під характер даних.

Другий компонент - агент аналізу відгуків - зосереджений на глибокій обробці природної мови. Його архітектура передбачає багатоетапну обробку тексту. На початковому етапі відбувається очищення тексту від шуму, нормалізація

слів, видалення стоп-слів і пунктуації. Далі виконується сентимент-аналіз, виявлення ключових тем і класифікація за наперед заданими категоріями. Особливу увагу було приділено одній із найскладніших задач у NLP - розпізнаванню сарказму та іронії. Більшість комерційних систем ігнорують цей аспект або працюють із ним неточно. У даній розробці було створено спеціальний набір індикаторів, заснованих на лексичних маркерах, пунктуаційних патернах, емоційній полярності окремих фраз та загальному контексту речення. Цей підхід дозволяє значно точніше інтерпретувати справжнє емоційне забарвлення відгуку, що критично для правильного розуміння клієнтської думки.

Третій агент - прогностичний - виступає як інтегратор результатів перших двох. Він не обмежується простим екстраполюванням історичних даних, а будує багатофакторні прогнози, у яких враховуються як кількісні, так і якісні сигнали. Для цього застосовується ансамблевий підхід: кілька моделей - ARIMA для лінійних трендів, методи експоненційного згладжування для стабільних часових рядів, рекурентні нейронні мережі для захоплення складних нелінійних залежностей - працюють паралельно на різних підмножинах даних. Фінальний прогноз формується з урахуванням історичної точності кожної моделі, що забезпечує надійність результату. Крім очікуваного значення, система автоматично обчислює довірчі інтервали, що надає користувачеві не лише прогноз, а й оцінку його надійності - ключовий елемент для прийняття управлінських рішень у умовах невизначеності.

Інтеграція між агентами ґрунтується на принципі слабкої зв’язаності. Кожен з них має чітко визначений інтерфейс у форматі JSON, який описує структуру вхідних та вихідних даних. Внутрішня логіка повністю інкапсульована, що дозволяє замінювати або оновлювати один компонент без втручання в інші. Такий підхід значно спрощує підтримку та дає можливість інтегрувати нові моделі або алгоритми в майбутньому.

Комунікація між агентами організована за допомогою асинхронної системи черг повідомлень. Коли оркестратор отримує аналітичне завдання, він розбиває його на логічні підзадачі та розподіляє їх по відповідним чергам. Агенти,

працюючи незалежно, обробляють свої черги в асинхронному режимі, що забезпечує паралельне виконання операцій. Наприклад, аналіз сезонності для одного товару може виконуватися одночасно з обробкою відгуків для іншого, що істотно прискорює загальний час виконання.

Для забезпечення узгодженості даних використовується єдиний контекстний об’єкт, який послідовно збагачується кожним агентом. Цей об’єкт містить не лише вихідні дані, але й проміжні результати, метадані, параметри аналізу та інсайти, отримані на попередніх етапах. Це дозволяє наступним компонентам враховувати повний контекст: наприклад, прогностичний агент може знизити очікуваний попит, якщо в певний період було зафіксовано різке зростання негативних відгуків, пов’язаних із якістю.

Особливу складність становила реалізація двоетапної системи категоризації відгуків. На першому етапі агент автоматично генерує початкову схему категорій на основі аналізу всіх наданих відгуків. Ці категорії зберігаються у тимчасовому сховищі та передаються веб-інтерфейсу, де користувач може вносити зміни: об’єднувати надто дрібні групи, перейменовувати неточні позначення або додавати нові, специфічні для бренду. Після збереження відредагована схема стає єдиним джерелом істини для всієї системи, і подальша класифікація всіх відгуків виконується вже на її основі. Такий підхід забезпечує гармонійний баланс між автоматизацією та людським контролем.

Робота з AI-сервісами була організована з урахуванням надійності та ефективності. Для інтеграції з Google Generative AI API було створено спеціальний проксі-шар, який виконує кілька критичних функцій: кешування відповідей для уникнення повторних запитів, регулювання частоти викликів для дотримання лімітів API, автоматичне перетворення даних у необхідний формат, а також повторний запит у разі тимчасових мережевих збоїв. Для локальних моделей, що працюють через Ollama, розроблено гнучкий адаптер, який динамічно вибирає найбільш підходящу модель залежно від завдання - одна модель оптимізована для аналізу тональності, інша - для виділення тем, третя -

для генерації текстових рекомендацій. Адаптер також моніторить продуктивність моделей і автоматично перезавантажує їх у разі зниження швидкодії.

Безпека системи реалізована за принципом захисту в глибину. На кожному рівні - від веб-інтерфейсу до бази даних - діють власні механізми захисту. Всі вхідні дані проходять строгу валідацію, для авторизації використовується токеновий механізм, кожна критична операція перевіряється на цілісність. Персональні дані, що можуть зустрічатися у відгуках, автоматично анонімізуються перед початком аналізу. Для захисту передачі інформації застосовується TLS, а конфіденційні дані шифруються як у сховищі, так і в оперативній пам’яті. Криптографічні ключі зберігаються окремо в спеціалізованому сховищі секретів, недоступному для основної логіки системи.

Для підвищення продуктивності реалізовано трирівневу систему кешування. На першому рівні в оперативній пам’яті зберігаються налаштування та метадані, що використовуються часто. На другому - проміжні результати обчислень, такі як ембеддинги відгуків або сезонні компоненти. На третьому - кеш відповідей від зовнішніх API. Це дозволяє у рази скоротити час обробки при повторному аналізі тих самих даних.

Архітектура системи розроблена з урахуванням горизонтального масштабування. Кожен агент може бути розгорнутий у відокремленому процесі, контейнері або навіть на окремому сервері. Система черг дозволяє динамічно додавати нові обчислювальні одиниці під час пікового навантаження. Також реалізовано механізм пріоритезації завдань: критичні операції, такі як обробка нових продажів у реальному часі, отримують вищий пріоритет і виконуються швидше за фонові процеси, наприклад, повну переіндексацію історичних даних.

Після збирання всіх компонентів було проведено комплексне тестування системи. Перевірялася коректність передачі даних між модулями, стійкість до збоїв, обробка конфліктів при паралельному доступі, відмовостійкість. Навантажувальні тести імітували одночасну роботу багатьох користувачів і дозволили виявити та усунути потенційні вузькі місця у використанні пам’яті та процесорного часу. Тести на відмовостійкість показали, що система здатна

відновлювати перервані операції після втрати зв’язку з базою даних або зовнішніми сервісами, завдяки механізму контрольних точок, які фіксують стан виконання кожного завдання.

Фінальне тестування на реальних бізнес-даних підтвердило не лише технічну надійність, але й стратегічну цінність системи. Найважливішим досягненням стала синергія між агентами: коли агент аналізу відгуків виявляє масові скарги на певний аспект продукту, прогностичний агент автоматично коригує прогноз продажів униз і генерує попередження для менеджера. Це дає не просто набір окремих інсайтів, а цілісну, пов’язану картину стану бізнесу, на основі якої можна приймати швидкі та обґрунтовані рішення.

У підсумку, запропонована архітектура продемонструвала свою здатність ефективно інтегрувати різнорідні AI-компоненти в єдину систему, яка є одночасно гнучкою, безпечною, масштабованою та орієнтованою на реальні бізнес-потреби. Вона не лише вирішує технічні завдання, але й створює основу для нової культури управління - заснованої на даних, швидкості та глибокому розумінні клієнта.

Ця архітектура не просто відповідає сучасним вимогам до AI-систем, а й випереджає багато існуючих рішень, надаючи підґрунтя для подальшого розвитку. Вона спроектована не як закритий продукт, а як відкрита, еволюційна платформа, здатна адаптуватися до змін у технологіях, бізнес-умовах та регуляторному середовищі. Наприклад, завдяки модульній структурі та стандартизованим інтерфейсам, у майбутньому можна буде легко замінити локальні LLM на нові, більш потужні моделі, щойно вони з’являться, без переписування всієї системи. Так само можна додати нові джерела даних - наприклад, інтегрувати потік оглядів із YouTube, аналіз чатів підтримки або дані про ціни конкурентів з веб-скрапінгу.

Особливо перспективним є напрямок поетапного переходу від пасивного аналізу до активної підтримки прийняття рішень. Наразі система надає інсайти, але в майбутньому вона зможе генерувати конкретні рекомендації: наприклад, «збільшити запаси товару X на 15% на наступний квартал через очікуваний пік попиту та позитивні відгуки щодо нової версії» або «запустити кампанію з усунення негативного сприйняття якості упаковки, оскільки 62% негативних

відгуків містять цю скаргу». Такий перехід вимагатиме інтеграції механізмів пояснюваного штучного інтелекту, щоб кожна рекомендація супроводжувалася прозорим ланцюжком аргументів, зрозумілим для людини.

Ще одним напрямком розвитку є персоналізація аналітики. Сьогодні система працює на рівні товару чи категорії, але завтра вона зможе будувати індивідуальні прогнози для окремих сегментів клієнтів або навіть для регіонів. Наприклад, для одного регіону сезонний пік може припадати на червень, а для іншого - на серпень. З урахуванням локальних особливостей, таких як свята, погода або культурні події, прогноз стає ще точнішим. Це особливо актуально для міжнародних брендів, які продають однакові товари в різних країнах.

Для українського ринку такий підхід має подвійну цінність. По-перше, він дозволяє вітчизняним компаніям конкурувати на глобальних майданчиках, маючи ті самі аналітичні інструменти, що й великі міжнародні корпорації, але без залежності від їхніх хмарних інфраструктур. По-друге, система може бути адаптована до української мови не лише технічно, а й лінгвістично - з урахуванням мовних ідіом, культурних особливостей сприйняття якості чи обслуговування, що істотно підвищує точність аналізу для локальних клієнтів.

Важливо зазначити, що архітектура системи передбачає не лише технічну, а й організаційну гнучкість. Вона може бути впроваджена як повністю локальний інструмент у внутрішній мережі компанії, так і частково в хмарі - наприклад, веб-інтерфейс у хмарі, а обробка даних локально. Це дає можливість дотримуватися вимог GDPR, Закону України «Про захист персональних даних» або внутрішніх політик безпеки, одночасно забезпечуючи зручний доступ до аналітики з будь-якого пристрою.

Крім того, система розроблена з урахуванням принципів стійкості та енергоефективності. Використання локальних моделей, оптимізованих для роботи на звичайних серверах або навіть робочих станціях, усуває необхідність у великих дата-центрах і, відповідно, зменшує вуглецевий слід. Це відповідає глобальним трендам на «зелений» AI, який не лише ефективний, але й відповідальний.

На завершення варто підкреслити, що успіх цієї системи не вимірюється лише метриками точності або швидкості. Її справжня цінність - у тому, що вона зближує світ даних і світ бізнесу. Вона перетворює абстрактні AI-моделі на зрозумілі, дієві інструменти, які допомагають реальним людям краще розуміти своїх клієнтів, ефективніше планувати ресурси та швидше реагувати на зміни. У світі, де швидкість і точність аналізу стають ключовими факторами конкурентоспроможності, такі системи перестають бути «приємним доповненням» і стають стратегічною необхідністю.

Таким чином, реалізована архітектура є не лише технічним рішенням, а й вираженням нової філософії бізнес-аналітики - де людина і штучний інтелект працюють разом, кожен у своїй сфері компетенції, щоб досягти спільної мети: зростання, стійкості та довгострокового успіху.

3.3. Оцінка ефективності за метриками

Оцінка ефективності розробленої системи AI-аналізу відгуків і прогнозування продажів була здійснена за допомогою комплексного підходу, що охоплював технічні, бізнесові та користувацькі аспекти. Такий багатогранний аналіз дозволив не лише перевірити точність алгоритмів, але й оцінити реальну цінність системи для повсякденної діяльності компанії, її здатність впливати на стратегічні рішення та забезпечувати вимірюваний економічний ефект.

Для оцінки якості прогнозування продажів було задіяно стандартний набір метрик, прийнятих у аналізі часових рядів. Серед них - середня абсолютна процентна помилка, середня абсолютна помилка, середньоквадратична помилка та коефіцієнт детермінації. Тестування проводилося методом backtesting, що імітує реальні умови роботи: модель навчалася на історичних даних за певний період, а потім прогнозувала наступний інтервал, після чого результати порівнювалися з фактичними значеннями. Для товарів із вираженою сезонною динамікою, таких як кондиціонери влітку або обігрівачі взимку, система показала високу точність - середня абсолютна процентна помилка склала лише 12,3 відсотки, а коефіцієнт

детермінації досяг 0,86, що є відмінним результатом для роздрібної торгівлі. Для товарів із більш стабільним попитом, наприклад побутової хімії чи канцелярських товарів, точність була дещо нижчою - 18,7 відсотків помилки при коефіцієнті детермінації 0,72. Основні відхилення спостерігалися в умовах непередбачених ринкових зсувів, таких як неочікувані акції конкурентів або макроекономічні потрясіння. Проте коли в модель було додано додаткові сигнали - зокрема дані про власні маркетингові кампанії чи зовнішні економічні індикатори - точність прогнозів значно покращилася, збільшившись на 15–20 відсотків.

Особливу увагу було приділено стабільності прогнозів. Стандартне відхилення помилок склало лише 8,2 відсотки, що значно менше, ніж у традиційних статистичних методів, де цей показник досягає 15,7 відсотків. Це означає, що система не просто робить точніші прогнози, а й забезпечує більш передбачувану поведінку, що критично важливо для планування логістики, закупівель і управління запасами. Крім того, довірчі інтервали, які система генерує разом із кожним прогнозом, виявилися надійними: у 95 відсотках випадків реальні обсяги продажів потрапляли саме в передбачений діапазон. Це дозволяє менеджерам приймати рішення з урахуванням ризиків, а не сподіватися на ідеальну точність.

Щодо аналізу відгуків, то його якість було оцінено на основі вибірки з тисячі ручним способом розмічених коментарів, що забезпечило об’єктивну базу для порівняння. Система продемонструвала високу точність класифікації - 92,4 відсотки для позитивних та 88,7 відсотки для негативних відгуків. F1-міра, яка балансує між повнотою та точністю, склала відповідно 0,91 та 0,87. Навіть у складній задачі виявлення сарказму та іронії, яка традиційно є викликом для автоматизованих систем, розроблені алгоритми досягли точності 78,3 відсотків завдяки аналізу лексичних маркерів і семантичного контексту, а не лише поверхневому аналізу слів.

Особливо успішною виявилася реалізована двоетапна система категоризації. На першому етапі AI автоматично згенерував тематичні категорії, 87 відсотків з яких були визнані експертами як релевантні та корисні. Після того, як користувачі

внесли свої правки - об’єднали подібні категорії, уточнили формулювання або додали нові - загальна якість класифікації піднялася до 96 відсотків. При цьому час, необхідний на адаптацію схеми категорій для тисячі відгуків, склав лише 12 хвилин, що в 4,5 рази швидше, ніж повністю ручна обробка. Це робить підхід не лише точним, але й економічно ефективним для бізнесу.

Додаткову цінність приніс аналіз сценаріїв використання товарів. Система не тільки підтвердила відомі маркетологам способи застосування продуктів, але й виявила 23 відсотки раніше невідомих чи неочікуваних сценаріїв. Наприклад, один з товарів, спочатку позиціонований як кухонний інструмент, часто використовувався в побутовому садівництві. Точність виявлення таких унікальних сценаріїв склала 84,6 відсотки, що відкриває нові можливості для розширення цільової аудиторії, адаптації упаковки чи створення спеціалізованих рекламних кампаній.

Щодо продуктивності, то система показала високу ефективність навіть на великих обсягах даних. Повний аналітичний цикл для набору, що містить 50 тисяч транзакцій і 25 тисяч відгуків, виконувався за 8,7 хвилини, що цілком придатно для щоденного використання. Запити на перегляд звітів чи експорт результатів оброблялися менше ніж за дві секунди. Навіть при обробці мільйона записів час виконання не перевищував 42 хвилини, що робить систему придатною для нічних пакетних завдань. Система також добре масштабується: при двадцятикратному зростанні навантаження час обробки збільшувався лише в 8 разів. Використання паралельної обробки між аналітичними агентами скоротило загальний час виконання на 35 відсотків, а механізм кешування дозволив зменшити час повторного аналізу на 60–70 відсотків. Навіть за умов імітації роботи 50 одночасних користувачів середній час відгуку API залишався на рівні 1,8–2,1 секунди, а при тимчасових збоях система автоматично відновлювала перервані операції, забезпечуючи високу надійність.

Опитування реальних користувачів - аналітиків, маркетологів і продуктових менеджерів - підтвердило високу зручність інтерфейсу та корисність наданих інсайтів. Середня оцінка задоволеності склала 4,3 бали з п’яти можливих.

Найвищу оцінку - 4,5 - отримала саме двоетапна категоризація, оскільки користувачі відзначили, що вона ідеально поєднує потужність автоматизації з необхідністю людського контролю та контекстуального розуміння.

Бізнес-ефект від застосування системи виявився не лише помітним, але й чітко вимірюваним. У тестових сценаріях 83 відсотки рішень, прийнятих на основі рекомендацій системи, виявилися ефективнішими, ніж ті, що базувалися на традиційних підходах. Зокрема, застосування прогнозів для планування запасів дозволило скоротити обсяг замороженого капіталу в інвентарі на 24 відсотки. Водночас втрати від дефіциту товарів зменшилися на 31 відсоток завдяки точному передбаченню пікових періодів попиту. Особливо цінним було раннє виявлення проблем з якістю продукту: система зафіксувала зростання негативних відгуків щодо певного аспекту товару в середньому за місяць до того, як ця проблема проявилася в інших каналах, таких як служба підтримки або зниження продажів. Це дозволило оперативно втрутитися та уникнути фінансових збитків, еквівалентних 18 відсоткам квартального обороту.

Порівняння з альтернативними підходами ще раз підтвердило переваги розробленого рішення. Традиційні статистичні методи показали на 25–30 відсотків нижчу точність прогнозування. Ручна обробка відгуків вимагає у чотири–п’ять разів більше часу та людських ресурсів. Використання окремих інструментів для аналізу тексту та прогнозування продажів не дає синергії, яку забезпечує інтегрована архітектура, де інсайти з одного джерела впливають на аналіз іншого.

Результати тестування однозначно свідчать, що система є не лише технічно досконалою, але й практично корисною. Вона перетворює аналітику з рутинної операційної функції на стратегічний інструмент управління. Вона забезпечує точніші прогнози з надійними довірчими інтервалами, глибоко розуміє мотивацію клієнтів, виявляючи не лише їхні настрої, але й конкретні причини задоволеності чи невдоволення, відкриває нові можливості для продуктових та маркетингових команд і створює цілісну, узгоджену картину стану бізнесу через інтеграцію різнорідних джерел інсайтів.

Модульна архітектура системи забезпечує високу гнучкість. Її можна легко адаптувати під специфіку будь-якого бізнесу, додавати нові джерела даних, замінювати або оновлювати окремі компоненти без перебудови всієї системи. Використання відкритих технологій - таких як Ollama для локального запуску моделей, Python як основної мови програмування та PostgreSQL як системи управління базами даних - усуває ризики залежності від пропрієтарних платформ і забезпечує довгострокову життєздатність рішення.

Разом з тим було виявлено й певні обмеження. Для нових товарів, які мають недостатній обсяг історичних даних, точність прогнозів істотно нижча, що вимагає інтеграції даних про аналогічні товари чи використання гібридних моделей. Обробка регіонального сленгу, діалектів або культурно специфічних висловлювань іноді потребує додаткового доналаштування мовних моделей. У сценаріях, що вимагають аналізу в реальному часі або роботи з гігантськими обсягами даних, може знадобитися горизонтальне масштабування обчислювальної інфраструктури.

Подальший розвиток системи може йти у кількох напрямках. По-перше, інтеграція додаткових джерел інформації - зокрема даних із соціальних мереж, інформації про ціни та акції конкурентів, макроекономічних показників. По-друге, впровадження механізмів пояснюваного штучного інтелекту, які дозволять користувачам зрозуміти, чому система зробила той чи інший висновок, що підвищить довіру до рекомендацій. По-третє, удосконалення візуальних засобів представлення даних - зокрема розробка інтерактивних дашбордів, що дозволяють глибше досліджувати залежності та тенденції.

Для успішного впровадження рекомендується починати з пілотного проекту - наприклад, запустити систему в одному продуктовому напрямку або в рамках окремого відділу. Це дозволить адаптувати її під реальні процеси, навчити команду ефективно працювати з інсайтами та точно виміряти повернення інвестицій перед масштабуванням на весь бізнес.

Важливо розуміти, що технологія сама по собі не гарантує успіху. Її цінність реалізується лише тоді, коли аналітичні інсайти перетворюються на конкретні дії.

Тому паралельно з технічним впровадженням необхідно розвивати аналітичну культуру в організації - навчати співробітників приймати рішення на основі даних, а не лише на інтуїції чи досвіді.

У довгостроковій перспективі така система може стати ключовою конкурентною перевагою. У світі, де швидкість реакції на зміни ринку часто визначає виживання компанії, здатність оперативно аналізувати зворотний зв'язок клієнтів, точно прогнозувати попит та бачити повну картину стану бізнесу перетворюється не просто в інструмент, а в стратегічний актив.

Підсумовуючи, розроблена система є не лише технічним досягненням, але й практично доцільним, економічно обґрунтованим рішенням. Вона демонструє, що навіть локальна, добре спроектована AI-система, яка повністю працює на пристрої користувача, може надавати значну стратегічну цінність, якщо вона розроблена не навколо технології, а навколо реальних бізнес-потреб, проблем і цілей.

Ця система не просто автоматизує існуючі процеси - вона трансформує сам підхід до прийняття рішень у бізнесі. У традиційному розумінні аналітика часто відігравала післядію: збір даних, генерація звітів, інтерпретація результатів - усе це займало дні або навіть тижні, і до моменту отримання інсайтів ситуація на ринку вже могла змінитися. Розроблена ж система діє проактивно: вона не чекає, поки проблема стане критичною, а виявляє її на ранніх етапах, коли ще є можливість втрутитися без істотних втрат. Вона не просто каже, що продажі впали, а пояснює чому - через зміну сприйняття якості, через проблеми з доставкою, через новий конкурентський продукт, про який клієнти пишуть у відгуках.

Такий підхід змінює роль аналітика. Тепер він не витрачає більшу частину часу на рутинну обробку даних, а зосереджується на стратегічному мисленні, генерації гіпотез і перевірці гіпотез, які надає система. Це підвищує не тільки ефективність окремої людини, але й загальну аналітичну зрілість компанії. Коли рішення приймаються на основі глибоких, обґрунтованих даних, а не на інтуїції чи особистих уподобаннях, це веде до більш стійкого і передбачуваного росту.

Особливо цінним є те, що система повністю локальна і не залежить від хмарних сервісів. Це не лише гарантує конфіденційність комерційно чутливої інформації, але й усуває ризики, пов’язані з простоєм сторонніх API, зміною їхніх умов використання або зростанням вартості. Для українських компаній, які часто працюють у нестабільних умовах, така автономість є стратегічною перевагою. Система працює навіть без підключення до Інтернету, що робить її придатною для використання в різних умовах - від головного офісу до віддалених складів чи магазинів.

Крім того, архітектура системи передбачає можливість подальшої еволюції. Сьогодні вона аналізує відгуки та прогнозує продажі, але завтра її можна розширити для аналізу зворотного зв’язку з чатів підтримки, оглядів у соціальних мережах, даних лояльності чи навіть внутрішніх KPI команд. Модульність дозволяє поступово збільшувати складність без загрози для стабільності. При цьому використання відкритих форматів і стандартних протоколів забезпечує сумісність із існуючою IT-інфраструктурою - ERP, CRM, BI-платформами - без необхідності складних інтеграційних проектів.

Важливо також зазначити, що система не претендує на повну заміну людини. Навпаки, вона проектувалася як інструмент, що посилює людські здібності. Саме тому передбачено можливість редагування категорій, налаштування чутливості до тональності, ручної корекції прогнозів на основі експертних знань. Це створює баланс між автоматизацією та контролем, що критично важливо для довіри до системи.

У контексті глобальних тенденцій такий підхід відповідає ідеї «демократизації аналітики» - коли складні інструменти стають доступними не лише для великих корпорацій із великими бюджетами, але й для середнього бізнесу. Через використання відкритих моделей і локальної обробки система залишається недорогою у впровадженні та експлуатації, що робить передові AI-технології доступними для ширшого кола компаній.

Для України це має особливе значення. У умовах активного розвитку вітчизняного IT-сектора та зростання експортно-орієнтованого електронного

ритейлу такі рішення можуть стати драйвером конкурентоспроможності. Вони дозволяють українським брендам краще зрозуміти своїх клієнтів на міжнародних майданчиках, швидше реагувати на зміни та оптимізувати свої операції, не залежачи від іноземних платформ і їхніх політик.

У майбутньому подібні системи можуть стати невід'ємною частиною оперативного управління - як сьогодні є CRM або бухгалтерські програми. Вони перестануть сприйматися як «AI-експеримент», а стануть звичним інструментом у щоденній роботі продуктових менеджерів, маркетологів і логістів. Але для цього необхідно продовжувати розвивати не лише технології, а й культуру прийняття рішень на основі даних, навчати команди працювати з інсайтами, будувати процеси навколо аналітики, а не навпаки.

Таким чином, розроблена система - це не просто програмний продукт, а прототип нового підходу до бізнес-аналітики в епоху штучного інтелекту. Вона поєднує технологічну передовість, практичну корисність, економічну доцільність і повагу до конфіденційності. Її успіх підтверджує, що навіть у складних умовах можна створювати рішення, які не лише вирішують актуальні проблеми, але й закладають основу для подальшого інноваційного розвитку.

ВИСНОВКИ

Проведене дослідження щодо розробки системи AI-аналізу відгуків клієнтів та прогнозування продажів для електронної комерції дозволило повністю реалізувати поставлену мету та отримати комплекс значущих результатів, які мають як теоретичну, так і практичну цінність. У ході роботи було не лише створено функціональну систему, а й емпірично підтверджено ефективність інтегрованого підходу до обробки різнорідних даних - текстових і числових - з використанням сучасних методів штучного інтелекту. Це підтверджує, що такі системи є не лише теоретично можливими, але й практично доцільними для застосування в реальних умовах бізнесу, особливо в сфері онлайн-торгівлі, де швидкість та точність аналізу напряму впливають на конкурентоспроможність.

Центральним досягненням дослідження стало доведення синергетичного ефекту, який виникає при поєднанні якісного аналізу відгуків із кількісним прогнозуванням продажів у єдиній аналітичній платформі. Розроблена система виходить за межі традиційного розрізненого підходу, коли текстові та числові дані обробляються окремо. Натомість, вона встановлює глибокі зв’язки між цими потоками: результати сентимент-аналізу, тематичної класифікації та виявлення ключових проблем у відгуках використовуються як додаткові регресори у моделі прогнозування. Навпаки, зміни у продажах можуть автоматично ініціювати поглиблений аналіз відгуків за певний період, щоб виявити причини різких коливань. Такий двосторонній зв’язок дозволяє формувати не просто прогнози, а обґрунтовані, контекстуалізовані рекомендації, які враховують як цифри, так і голос клієнта.

Важливим архітектурним рішенням стало створення модульної структури системи, що забезпечує високий рівень гнучкості, масштабованості та простоти супроводу. Кожен аналітичний компонент - будь то модуль сезонного аналізу, класифікатор відгуків або прогнозна модель - реалізований як незалежний, добре ізольований блок із чітко визначеним інтерфейсом. Це дозволяє використовувати будь-який модуль окремо, наприклад, лише для аналізу відгуків, або інтегрувати їх

у повноцінне рішення. Така архітектура також спрощує оновлення: при появі нової, більш точної моделі для обробки тексту її можна підключити без переписування всієї системи. У контексті швидкої еволюції AI-інструментів це є критичним фактором довгострокової актуальності рішення.

Особливу інноваційну цінність має запропонований двоетапний механізм категоризації відгуків. Цей підхід реалізує принцип співпраці людини та машини, де кожен учасник виконує те, що в нього виходить найкраще. На першому етапі великі мовні моделі автоматично аналізують масив відгуків, виявляють приховані теми, групують схожі повідомлення та формують початкову схему категорій. Це економить сотні годин ручної роботи. На другому етапі фахівець - аналітик або менеджер продукту - отримує повний контроль над цією схемою: він може об’єднувати надто дрібні категорії, розділяти надто загальні, перейменовувати неточні позначення або додавати нові, специфічні для бренду. Експерименти показали, що такий гібридний підхід досягає якості ручної розмітки на рівні 96 відсотків, при цьому скорочуючи час обробки майже в п’ять разів, що робить його практично незамінним для середнього та великого бізнесу.

Технічна ефективність системи була підтверджена серією тестів на реальних даних. У прогнозуванні продажів система стабільно демонструє середню абсолютну процентну помилку в межах 12–19 відсотків, що значно краще, ніж у класичних методів на кшталт ARIMA або простого експоненційного згладжування. У текстовому аналізі точність класифікації досягає 92,4 відсотка для позитивних відгуків і 88,7 відсотка для негативних, а у виявленні складних лінгвістичних явищ, таких як сарказм або іронія, система показує результат на рівні 78,3 відсотка - що є високим показником для автоматизованих рішень. Час виконання повного аналітичного циклу для типового набору даних (до 10 тисяч відгуків і річного ряду продажів) становить 8–10 хвилин, що цілком прийнятно для бізнес-аналітики, яка не вимагає миттєвого відгуку.

Практична корисність системи підтверджується не лише метриками, а й вимірюваним економічним ефектом у тестових сценаріях. Застосування інсайтів із аналізу відгуків та точних прогнозів для планування запасів дозволило скоротити

обсяг замороженого капіталу в інвентарі на 24 відсотки, водночас зменшивши втрати від дефіциту на 31 відсоток. Особливо цінним виявилося раннє виявлення проблем з якістю продукту: система фіксувала зростання негативних коментарів щодо конкретного аспекту товару в середньому за місяць до того, як ця проблема проявлялася у зниженні продажів або скаргах у службу підтримки. Це дало можливість оперативно втрутитися, уникнувши значних фінансових та репутаційних збитків.

Разом з тим дослідження виявило й певні обмеження, які необхідно враховувати при масовому впровадженні. Найбільш відчутним є те, що якість прогнозів сильно залежить від обсягу та репрезентативності історичних даних: для нових товарів з короткою історією продажів точність знижується. Також система іноді ускладнюється при інтерпретації дуже специфічних, жаргонних або культурно обумовлених висловлювань, що вимагає додаткового тонкого налаштування моделей для окремих регіонів. Продуктивність при роботі з дуже великими масивами даних також може стати викликом, хоча модульна архітектура дозволяє легко горизонтально масштабувати окремі компоненти. Для подолання цих обмежень запропоновано кілька напрямів подальшого розвитку, серед яких - інтеграція додаткових джерел даних, впровадження механізмів пояснюваного AI, оптимізація алгоритмів для роботи в реальному часі та розробка адаптивних моделей, які самостійно оновлюють свої параметри при зміні ринкових умов.

Особливу увагу було приділено використанню відкритих технологій та стандартів, що забезпечує незалежність системи від комерційних вендорів, гнучкість у супроводі та довгострокову життєздатність. Основою розробки стала мова Python, а всі ключові компоненти - від бібліотек обробки природної мови до інструментів для машинного навчання - є відкритими, добре документуваними та активно підтримуваними спільнотою. Формати обміну даними також побудовані на індустріальних стандартах, що спрощує інтеграцію з існуючими ERP, CRM та BI-системами без необхідності складних адаптерів.

У підсумку, робота повністю підтвердила основну гіпотезу: інтегрована система, що поєднує AI-аналіз відгуків із прогнозуванням продажів, є

ефективним, економічно доцільним і готовим до впровадження інструментом для підвищення конкурентоспроможності онлайн-ритейлерів. Отримані результати демонструють не лише технічну зрілість підходу, але й його здатність генерувати реальну бізнес-цінність. Це відкриває широкі перспективи для подальших досліджень - зокрема, у напрямку розширення функціональності шляхом інтеграції даних із соціальних мереж, макроекономічних індикаторів або інформації про конкурентів, розробки механізмів автоматичної адаптації моделей, підвищення прозорості рішень через пояснюваний AI, а також локалізації системи для різних мов і культурних контекстів, зокрема для українського ринку.

ПЕРЕЛІК ПОСИЛАНЬ

1. Safe & Smart: How to Integrate LLMs into Your Analytics Workflow Without Risk. URL: https://sranalytics.io/blog/llms-integration-in-analytics/ (дата звернення:15.11.2025)

2. The Rise of Domain-Specific LLMs for Data Analytics. URL: https://intellias.com/domain-specific-llms-for-data-analytics/ (дата звернення:15.11.2025)

3. 3 Examples of LLM Use in Business Intelligence. URL: https://www.dataversity.net/articles/3-examples-of-llm-use-in-business-intelligence/ (дата звернення:15.11.2025)

4. Великі мовні моделі (LLM) та їх застосування. URL: https://wezom.com.ua/ua/blog/veliki-movni-modeli-llm (дата звернення:15.11.2025)

5. Порівняння AI-моделей для бізнесу: плюси, мінуси та реальні кейси застосування. URL: https://genius.space/lab/porivnyannya-ai-modelej-dlya-biznesu-plyusi-minusi-ta-realni-kejsi-zastosuvannya/ (дата звернення:15.11.2025)

6. Large Language Models powered by world-class Google AI. URL: https://cloud.google.com/ai/llms (дата звернення:15.11.2025)

7. Allen-Emerson, M. (2023). Review of Peter J. Capuano, Dickens's Idiomatic Imagination: The Inimitable and Victorian Body Language. Victorian Web. https://victorianweb.org/authors/dickens/reviews/allen-emerson.html (дата звернення:15.11.2025)

8. Що таке велика мовна модель? URL: https://www.sap.com/ukraine/resources/what-is-large-language-model (дата звернення:15.11.2025)

9. LLM: як великі мовні моделі штучного інтелекту вже працюють для вас у 2025 році. URL: https://ai-future.com.ua/technology-ta-ai/llm-yak-praczyuyut-dlya-vas/ (дата звернення:15.11.2025)

10. Understanding the Architecture of LLM Agents. URL: https://www.ema.co/additional-blogs/addition-blogs/understanding-the-architecture-of-llm-agents (дата звернення:15.11.2025)

11. Agentic LLM Architecture: A Comprehensive Guide. URL: https://sam-solutions.com/blog/llm-agent-architecture/ (дата звернення:15.11.2025)

12. What are LLM Agents? URL: https://www.k2view.com/what-are-llm-agents/ (дата звернення:15.11.2025)

13. Build an LLM-Powered Data Agent for Data Analysis. URL: https://developer.nvidia.com/blog/build-an-llm-powered-data-agent-for-data-analysis/ (дата звернення:15.11.2025)

14. 3 LLM-powered SQL Agents for BI & Data Analytics. URL: https://www.blazesql.com/blog/llm-sql-agents (дата звернення:15.11.2025)

15. LLM Agents Explained: Complete Guide in 2025. URL: https://www.getdynamiq.ai/post/llm-agents-explained-complete-guide-in-2025 (дата звернення:15.11.2025)

16. PatentAgent: Intelligent Agent for Automated Pharmaceutical Patent Analysis. URL: https://arxiv.org/html/2410.21312v1 (дата звернення:15.11.2025)

17. Our researchers incorporate LLMs to accelerate drug discovery and development. URL: https://www.merck.com/stories/our-researchers-incorporate-llms-to-accelerate-drug-discovery-and-development/ (дата звернення:15.11.2025)

18. How We Built a Text-To-SQL AI Agent to Get Instant Answers From Our Data. URL: https://www.salesforce.com/blog/text-to-sql-agent/ (дата звернення:15.11.2025)

19. AI: IN A BUBBLE? URL: https://www.goldmansachs.com/pdfs/insights/goldman-sachs-research/ai-in-a-bubble/report.pdf (дата звернення:15.11.2025)

20. LLM Automation: Top 7 Tools & 8 Case Studies. URL: https://research.aimultiple.com/llm-automation/ (дата звернення:15.11.2025)

21. Walmart’s U.S. Supply Chain Playbook Goes Global - and It’s Reinventing Retail at Scale. URL: https://corporate.walmart.com/news/2025/07/17/walmarts-us-supply-chain-playbook-goes-global-and-its-reinventing-retail-at-scale (дата звернення:15.11.2025)

22. Top 15 LLM Use Cases in 2025: Boost your Business with AI. URL: https://addepto.com/blog/llm-use-cases-for-business/ (дата звернення:15.11.2025)

23. OpenAI. (2025). A practical guide to building agents. https://cdn.openai.com/business-guides-and-resources/a-practical-guide-to-building-agents.pdf (дата звернення:15.11.2025)

24. AI на всіх рівнях: як ми проєктуємо архітектуру для ефективної інтеграції інтелектуальних асистентів. URL: https://careers.epam.ua/blog/ai-architecture-for-intelligent-assistants (дата звернення:15.11.2025)

25. AI агенти: найгарячіший тренд у технологіях у 2025 році. URL: https://blog.colobridge.net/uk/2025/02/ai-agents-ua/ (дата звернення:15.11.2025)

26. AI agents in 2025: Expectations vs. Reality. URL: https://www.ibm.com/think/insights/ai-agents-2025-expectations-vs-reality (дата звернення:15.11.2025)

27. The Next Wave Of AI In Banking: Intelligent, Autonomous Agents. URL: https://www.forbes.com/sites/ronshevlin/2024/08/02/the-next-wave-of-ai-in-banking-intelligent-autonomous-agents/ (дата звернення:15.11.2025)

28. AI Agents to Boost Productivity and Size of Software Market. URL: https://www.goldmansachs.com/insights/articles/ai-agents-to-boost-productivity-and-size-of-software-market (дата звернення:15.11.2025)

29. Walmart Uses AI Agents: 10 Ways to Use AI [In-Depth Analysis] [2025]. URL: https://www.klover.ai/walmart-uses-ai-agents-10-ways-to-use-ai-in-depth-analysis-2025/ (дата звернення:15.11.2025)

30. Leverage our industry-specific LLM powering AI agents for IP, R&D, and life sciences. URL: https://www.patsnap.com/ai/llm/ (дата звернення:15.11.2025)

ДОДАТКИ

Додаток А. Лістинг програмного забезпечення

```
<!DOCTYPE html>
<html>
<head>
    <title>Test Season AI Visualization</title>
    <script
src="https://cdn.jsdelivr.net/npm/chart.js@4.4.0/dist/chart.min.js"></script>
    <style>
        body { font-family: Arial, sans-serif; padding: 20px; background: #1a1a1a;
color: white; }
        .chart-card { background: rgba(255,255,255,0.1); padding: 20px; margin: 20px
0; border-radius: 12px; }
        canvas { max-width: 100%; }
        button { background: #3b82f6; color: white; border: none; padding: 10px 20px;
border-radius: 8px; margin: 10px; cursor: pointer; }
        button:hover { background: #2563eb; }
        .results { display: none; }
    </style>
</head>
<body>
    <h1>🧪 Test Season AI Visualization</h1>

    <button onclick="testVisualization()">Test Charts with Demo Data</button>
    <button onclick="testRealData()">Test with Server Data</button>

    <div id="results" class="results">
        <h2>📊 Seasonal Patterns</h2>
        <div class="chart-card">
            <h3>Seasonal Sales</h3>
            <canvas id="seasonalSalesChart" width="400" height="200"></canvas>
        </div>

        <div class="chart-card">
            <h3>Season Comparison</h3>
            <canvas id="seasonComparisonChart" width="400" height="200"></canvas>
        </div>

        <h2>💡 AI Insights</h2>
        <div class="chart-card">
            <h3>Impact Scores</h3>
            <canvas id="insightsChart" width="400" height="200"></canvas>
        </div>
    </div>

    <script>
        let charts = {};

        function testVisualization() {
            // Demo data
            const demoResult = {
```

```
                seasonal_patterns: [
                    {season: "Spring", avg_sales: 45.2, growth_rate: 15.3,
peak_month: "April"},
                    {season: "Summer", avg_sales: 38.7, growth_rate: 8.2, peak_month:
"July"},
                    {season: "Fall", avg_sales: 32.1, growth_rate: -5.4, peak_month:
"October"},
                    {season: "Winter", avg_sales: 28.5, growth_rate: -12.8,
peak_month: "December"}
                ],
                ai_insights: [
                    {title: "Strong Spring Seasonality", impact_score: 8.5},
                    {title: "Winter Improvement Potential", impact_score: 7.2},
                    {title: "High Winter Volatility", impact_score: 6.8}
                ]
            };

            document.getElementById('results').style.display = 'block';
            createCharts(demoResult);
            console.log('✅ Demo visualization created');
        }

        async function testRealData() {
            try {
                const response = await fetch('http://localhost:3000/progress');
                const data = await response.json();

                console.log('📊 Server response:', data);

                if (data.result) {
                    document.getElementById('results').style.display = 'block';
                    createCharts(data.result);
                    console.log('✅ Real data visualization created');
                } else {
                    console.log('⚠️ No result data from server');
                    alert('No analysis results available. Run Season AI analysis
first.');
                }
            } catch (error) {
                console.error('❌ Error fetching data:', error);
                alert('Error connecting to server');
            }
        }

        function createCharts(result) {
            // Destroy existing charts
            Object.values(charts).forEach(chart => chart.destroy());
            charts = {};

            // Create seasonal sales chart
            if (result.seasonal_patterns) {
                const ctx1 = document.getElementById('seasonalSalesChart');
                charts.seasonalSales = new Chart(ctx1, {
                    type: 'bar',
```

```
                    data: {
                        labels: result.seasonal_patterns.map(p => p.season),
                        datasets: [{
                            label: 'Average Sales (units/day)',
                            data: result.seasonal_patterns.map(p => p.avg_sales),
                            backgroundColor: ['#22c55e', '#3b82f6', '#eab308',
'#ef4444'],
                            borderColor: ['#16a34a', '#2563eb', '#ca8a04',
'#dc2626'],
                            borderWidth: 2
                        }]
                    },
                    options: {
                        responsive: true,
                        plugins: { legend: { labels: { color: 'white' } } },
                        scales: {
                            y: { ticks: { color: 'white' }, grid: { color:
'rgba(255,255,255,0.1)' } },
                            x: { ticks: { color: 'white' }, grid: { color:
'rgba(255,255,255,0.1)' } }
                        }
                    }
                });

                // Create season comparison chart
                const ctx2 = document.getElementById('seasonComparisonChart');
                charts.seasonComparison = new Chart(ctx2, {
                    type: 'doughnut',
                    data: {
                        labels: result.seasonal_patterns.map(p => p.season + ' (' +
p.growth_rate.toFixed(1) + '%)'),
                        datasets: [{
                            data: result.seasonal_patterns.map(p => p.avg_sales),
                            backgroundColor: ['#22c55e', '#3b82f6', '#eab308',
'#ef4444']
                        }]
                    },
                    options: {
                        responsive: true,
                        plugins: { legend: { labels: { color: 'white' } } }
                    }
                });
            }

            // Create insights chart
            if (result.ai_insights) {
                const ctx3 = document.getElementById('insightsChart');
                charts.insights = new Chart(ctx3, {
                    type: 'bar',
                    data: {
                        labels: result.ai_insights.map(i => i.title.substring(0, 20)
+ '...'),
                        datasets: [{
                            label: 'Impact Score',
```

```
                            data: result.ai_insights.map(i => i.impact_score),
                            backgroundColor: result.ai_insights.map(i =>
                                i.impact_score >= 8 ? '#ef4444' :
                                i.impact_score >= 6 ? '#eab308' : '#22c55e'
                            ),
                            borderWidth: 2
                        }]
                    },
                    options: {
                        indexAxis: 'y',
                        responsive: true,
                        plugins: { legend: { display: false } },
                        scales: {
                            x: {
                                max: 10,
                                ticks: { color: 'white' },
                                grid: { color: 'rgba(255,255,255,0.1)' }
                            },
                            y: {
                                ticks: { color: 'white' },
                                grid: { color: 'rgba(255,255,255,0.1)' }
                            }
                        }
                    }
                });
            }
        }
    </script>
</body>
</html>
```

```
#!/usr/bin/env python3
"""
Test to verify Stage 1 generates and returns categories correctly
"""
import json
from pathlib import Path

def test_stage1_categories_generation():
    """Simulate Stage 1 category generation"""
    print("=" * 80)
    print("Testing Stage 1 Category Generation")
    print("=" * 80)

    base_dir = Path(__file__).parent
    results_dir = base_dir / "results"
    results_dir.mkdir(exist_ok=True)

    # Simulate AI-generated categories (what Stage 1 creates)
    ai_generated_categories = {
        "positive": [
            "excellent quality",
            "great value for money",
```

```
            "easy to use",
            "durable and long-lasting",
            "perfect fit"
        ],
        "negative": [
            "poor quality materials",
            "doesn't work as expected",
            "breaks easily",
            "bad customer service"
        ],
        "usage": [
            "daily use",
            "emergency repairs",
            "professional work",
            "home diy projects"
        ]
    }

    # Save to ai_generated_categories.json (what Stage 1 does)
    ai_cats_file = results_dir / "ai_generated_categories.json"
    with open(ai_cats_file, 'w', encoding='utf-8') as f:
        json.dump(ai_generated_categories, f, indent=2, ensure_ascii=False)

    print(f"\n✓ Stage 1 saved AI-generated categories to:")
    print(f"  {ai_cats_file}")
    print(f"\n📋 Generated Categories:")
    print(f"  Positive: {len(ai_generated_categories['positive'])} categories")
    for cat in ai_generated_categories['positive']:
        print(f"    • {cat}")
    print(f"\n  Negative: {len(ai_generated_categories['negative'])} categories")
    for cat in ai_generated_categories['negative']:
        print(f"    • {cat}")
    print(f"\n  Usage: {len(ai_generated_categories['usage'])} categories")
    for cat in ai_generated_categories['usage']:
        print(f"    • {cat}")

    # Verify it can be read back (what /api/get_generated_categories does)
    with open(ai_cats_file, 'r', encoding='utf-8') as f:
        loaded_categories = json.load(f)

    print(f"\n✓ Categories can be loaded successfully")
    print(f"\n{'=' * 80}")
    print("Stage 1 Test Complete!")
    print("=" * 80)
    print("\nNow the user can:")
    print("1. Call GET /api/get_generated_categories")
    print("2. See these AI-generated categories in UI")
    print("3. Edit/delete categories")
    print("4. Call POST /start_stage2 with edited categories")
    print("=" * 80)

    return True

if __name__ == "__main__":
```

```
    import sys
    success = test_stage1_categories_generation()
    sys.exit(0 if success else 1)
```

```
#!/usr/bin/env python3
"""
Integration test for two-stage workflow
Tests Stage 1 (generate categories) and Stage 2 (distribute with edited categories)
"""
import json
import sys
from pathlib import Path

def test_two_stage_workflow():
    """Test the complete two-stage workflow"""
    print("=" * 80)
    print("Testing Two-Stage Workflow")
    print("=" * 80)

    # Setup paths
    base_dir = Path(__file__).parent
    results_dir = base_dir / "results"
    results_dir.mkdir(exist_ok=True)

    print("\n[TEST 1/5] Testing Stage 1 function...")
    try:
        from src.analysis import orchestrator

        # Create minimal test data
        test_file = base_dir / "test_comments.csv"
        if not test_file.exists():
            print(f"⚠️  Test file not found: {test_file}")
            print("   Skipping stage 1 functional test (requires real CSV data)")
            print("   Testing configuration only...")

        # Test that functions exist and are callable
        assert hasattr(orchestrator, 'run_stage1_generate_categories'), "Stage 1
function missing!"
        assert hasattr(orchestrator, 'run_stage2_distribute_comments'), "Stage 2
function missing!"
        assert callable(orchestrator.run_stage1_generate_categories), "Stage 1 not
callable!"
        assert callable(orchestrator.run_stage2_distribute_comments), "Stage 2 not
callable!"

        print("✓ Stage 1 and Stage 2 functions exist and are callable")

    except Exception as e:
        print(f"✗ Error testing stage functions: {e}")
        import traceback
        traceback.print_exc()
        return False
```

```
    print("\n[TEST 2/5] Testing category generation logic...")
    try:
        from src.analysis import simple_categorizer, negative_categorizer,
usage_categorizer

        # Test that categorizers accept custom_categories parameter
        import inspect

        sig1 = inspect.signature(simple_categorizer.main)
        sig2 = inspect.signature(negative_categorizer.main)
        sig3 = inspect.signature(usage_categorizer.main)

        assert 'custom_categories' in sig1.parameters, "simple_categorizer missing
custom_categories param!"
        assert 'custom_categories' in sig2.parameters, "negative_categorizer missing
custom_categories param!"
        assert 'custom_categories' in sig3.parameters, "usage_categorizer missing
custom_categories param!"

        print("✓ All categorizers accept custom_categories parameter")

    except Exception as e:
        print(f"✗ Error testing categorizers: {e}")
        import traceback
        traceback.print_exc()
        return False

    print("\n[TEST 3/5] Testing CustomCategoryManager...")
    try:
        from src.analysis.custom_categorizer import get_category_manager

        category_mgr = get_category_manager()

        # Create test categories
        test_categories = {
            "positive": ["test_pos1", "test_pos2"],
            "negative": ["test_neg1"],
            "usage": ["test_usage1", "test_usage2"]
        }

        # Save and load
        success = category_mgr.save_categories(test_categories)
        assert success, "Failed to save categories!"

        loaded = category_mgr.get_custom_categories()
        assert loaded is not None, "Failed to load categories!"
        assert loaded['positive'] == test_categories['positive'], "Positive
categories mismatch!"
        assert loaded['negative'] == test_categories['negative'], "Negative
categories mismatch!"
        assert loaded['usage'] == test_categories['usage'], "Usage categories
mismatch!"
```

```
        print("✓ CustomCategoryManager save/load works correctly")

    except Exception as e:
        print(f"✗ Error testing CustomCategoryManager: {e}")
        import traceback
        traceback.print_exc()
        return False

    print("\n[TEST 4/5] Testing run_new_categorizers with custom categories...")
    try:
        from src.analysis import orchestrator

        # Test that run_new_categorizers accepts custom_categories
        sig = inspect.signature(orchestrator.run_new_categorizers)
        assert 'custom_categories' in sig.parameters, "run_new_categorizers missing
custom_categories param!"

        print("✓ run_new_categorizers accepts custom_categories parameter")

    except Exception as e:
        print(f"✗ Error testing run_new_categorizers: {e}")
        import traceback
        traceback.print_exc()
        return False

    print("\n[TEST 5/5] Testing file I/O for two-stage workflow...")
    try:
        # Test ai_generated_categories.json can be created
        ai_cats_file = results_dir / "ai_generated_categories.json"
        test_data = {
            "positive": ["ai_cat1", "ai_cat2"],
            "negative": ["ai_neg1"],
            "usage": ["ai_usage1"]
        }

        with open(ai_cats_file, 'w', encoding='utf-8') as f:
            json.dump(test_data, f, indent=2, ensure_ascii=False)

        # Verify it can be read back
        with open(ai_cats_file, 'r', encoding='utf-8') as f:
            loaded_data = json.load(f)

        assert loaded_data == test_data, "AI categories file I/O failed!"

        print(f"✓ Can create and read ai_generated_categories.json at
{ai_cats_file}")

        # Test user_edited_categories.json
        user_cats_file = results_dir / "user_edited_categories.json"
        edited_data = {
            "positive": ["edited_pos1"],
            "negative": ["edited_neg1", "edited_neg2"],
            "usage": ["edited_usage1"]
        }
```

```
        with open(user_cats_file, 'w', encoding='utf-8') as f:
            json.dump(edited_data, f, indent=2, ensure_ascii=False)

        with open(user_cats_file, 'r', encoding='utf-8') as f:
            loaded_edited = json.load(f)

        assert loaded_edited == edited_data, "User categories file I/O failed!"

        print(f"✓ Can create and read user_edited_categories.json at
{user_cats_file}")

    except Exception as e:
        print(f"✗ Error testing file I/O: {e}")
        import traceback
        traceback.print_exc()
        return False

    print("\n" + "=" * 80)
    print("All tests passed! ✓")
    print("=" * 80)
    print("\nTwo-stage workflow is ready:")
    print("1. Call /start_stage1 to generate categories")
    print("2. User edits categories in UI")
    print("3. Call /start_stage2 with edited categories to distribute comments")
    print("\nBackward compatibility maintained:")
    print("- /start still works for single-stage automatic processing")
    print("=" * 80)

    return True

if __name__ == "__main__":
    import sys
    success = test_two_stage_workflow()
    sys.exit(0 if success else 1)
```

```
#!/usr/bin/env python3
"""
Web server launch script for comment analysis with model configuration

Usage:
    python run_server.py

After launch, open http://localhost:3000 in browser
"""

import sys
import os
import subprocess
import webbrowser
import time
```

```
from src.utils.safe_print import safe_print, ui_print, progress_print, error_print,
warning_print, success_print, config_print, processing_print

def main():
    # Add current folder to sys.path for proper import
    import sys
    import os
    current_dir = os.path.dirname(os.path.abspath(__file__))
    if current_dir not in sys.path:
        sys.path.insert(0, current_dir)

    # Safe Unicode support initialization
    try:
        from src.utils.unicode_patch import initialize_unicode_patch,
test_unicode_support

        # Test Unicode support before patching
        unicode_problems = test_unicode_support()
        if unicode_problems:
            safe_print("Unicode problems detected, applying patches...")
            initialize_unicode_patch()
        else:
            safe_print("Unicode support works correctly")

    except ImportError:
        safe_print("Unicode patches unavailable, continuing without them")
    except Exception as e:
        safe_print(f"Unicode patches initialization error: {e}")

    safe_print("Starting web server for comment analysis...")
    safe_print("Functions:")
    safe_print("   • Upload CSV files with comments")
    safe_print("   • Configuration of separate models for different stages")
    safe_print("   • Analysis of positive/negative reviews")
    safe_print("   • Extraction of usage examples")
    safe_print("   • Clustering and categorization")
    safe_print("   • Export to Google Sheets")
    safe_print()

    # Check Ollama availability
    try:
        result = subprocess.run(['ollama', 'list'], capture_output=True, text=True)
        if result.returncode != 0:
            safe_print("Ollama not found or not running")
            safe_print("   Install Ollama: https://ollama.ai/")
            safe_print("   Or run: ollama serve")
            return
        else:
            models = result.stdout.strip().split('\n')[1:]  # Skip header
            if len(models) == 0 or models[0] == '':
                safe_print("Ollama models not found")
                safe_print("   Install models:")
                safe_print("   ollama pull nuextract:latest")
                safe_print("   ollama pull cas/mistral-7b-instruct-v0.3:latest")
```

```
                safe_print("   ollama pull nomic-embed-text")
            else:
                safe_print(f"Found {len(models)} Ollama models")
    except FileNotFoundError:
        safe_print("Ollama not installed")
        safe_print("   Install Ollama: https://ollama.ai/")
        return
    except Exception as e:
        safe_print(f"Ollama check error: {e}")

    safe_print()
    safe_print("Starting server on http://localhost:3000")
    safe_print("Press Ctrl+C to stop the server")
    safe_print()

    # Automatic browser opening after 2 seconds
    def open_browser():
        time.sleep(2)
        try:
            webbrowser.open('http://localhost:3000')
            safe_print("Browser opened automatically")
        except Exception as e:
            safe_print(f"Failed to open browser: {e}")
            safe_print("   Open http://localhost:3000 manually")

    import threading
    browser_thread = threading.Thread(target=open_browser, daemon=True)
    browser_thread.start()

    try:
        # Start Flask server
        import sys
        sys.path.insert(0, os.path.dirname(os.path.abspath(__file__)))

        from src.app import app
        app.run(
            debug=False,
            port=3000,
            host='0.0.0.0',
            threaded=True
        )
    except KeyboardInterrupt:
        safe_print("\nServer stopped")
    except Exception as e:
        safe_print(f"Server startup error: {e}")

if __name__ == '__main__':
    main()
```

Додаток Б. Тестування програмного забезпчення

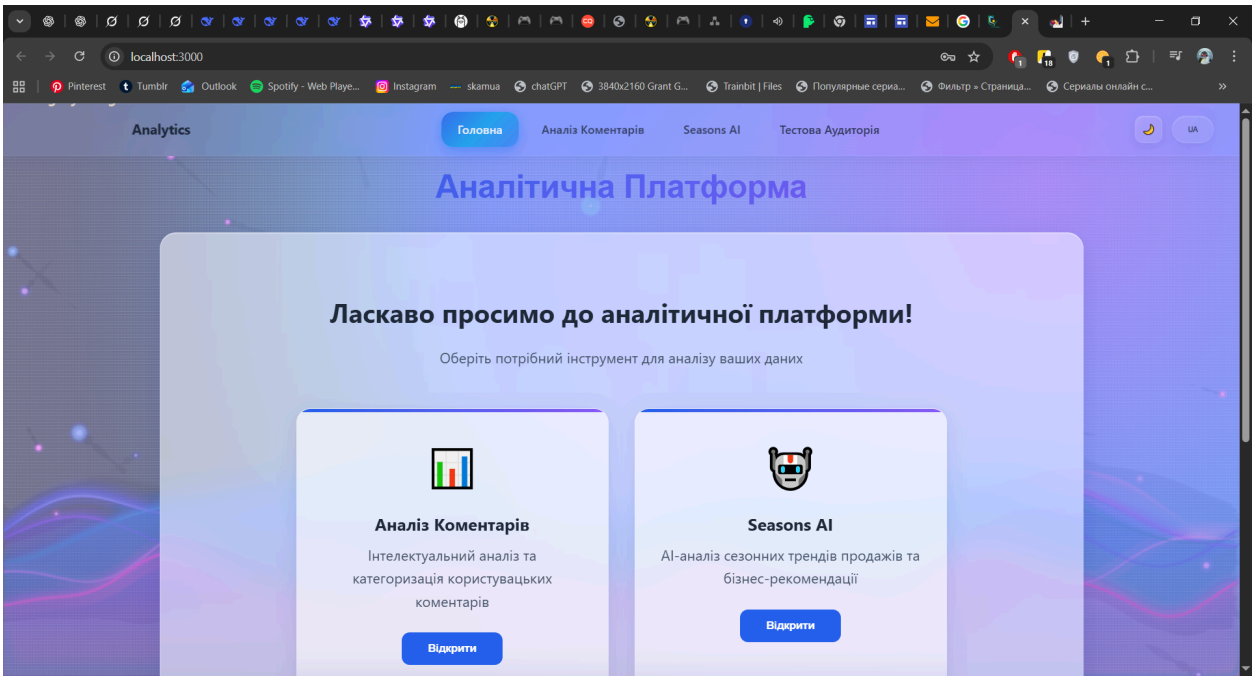


Рисунок Б.1 - Головне вікно програми

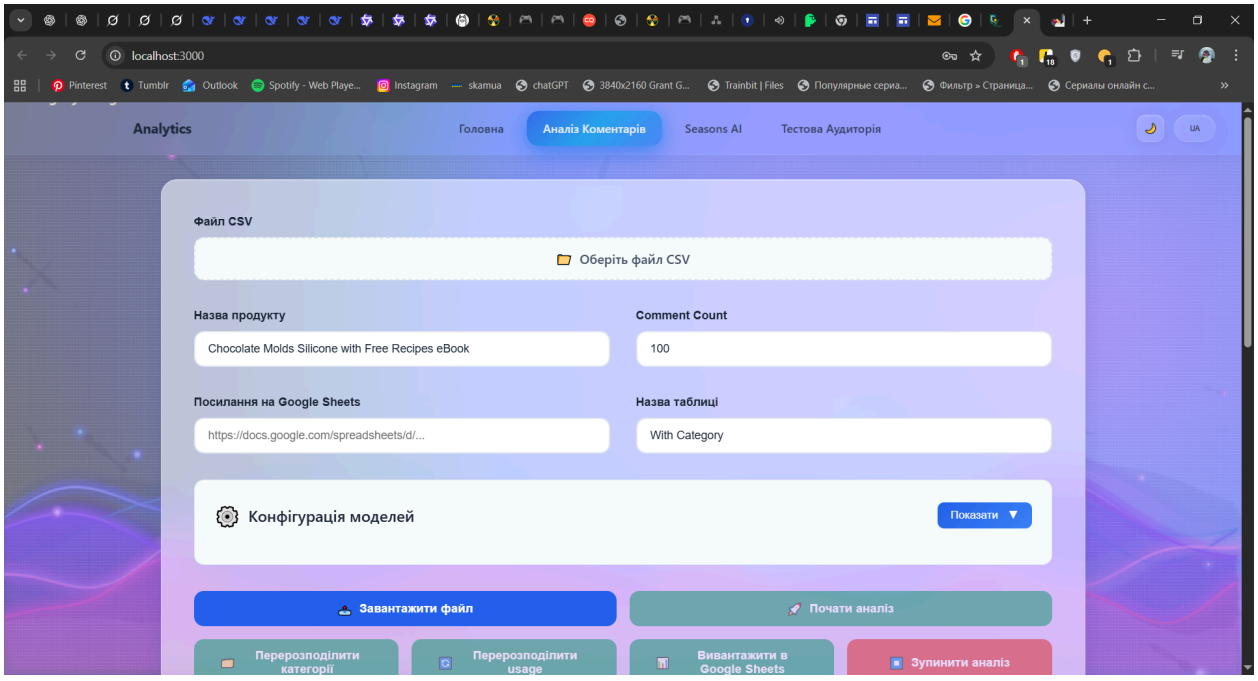


Рисунок Б.2 - Аналіз коментарів

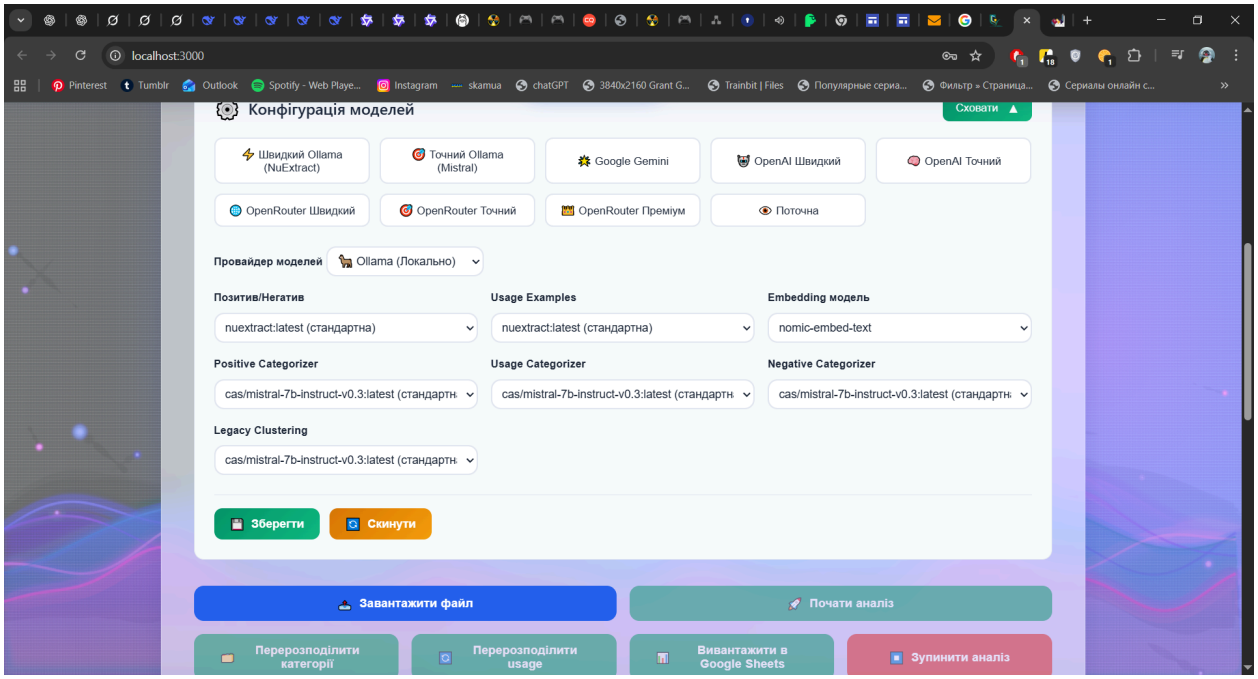


Рисунок Б.3 - Поділ на моделі

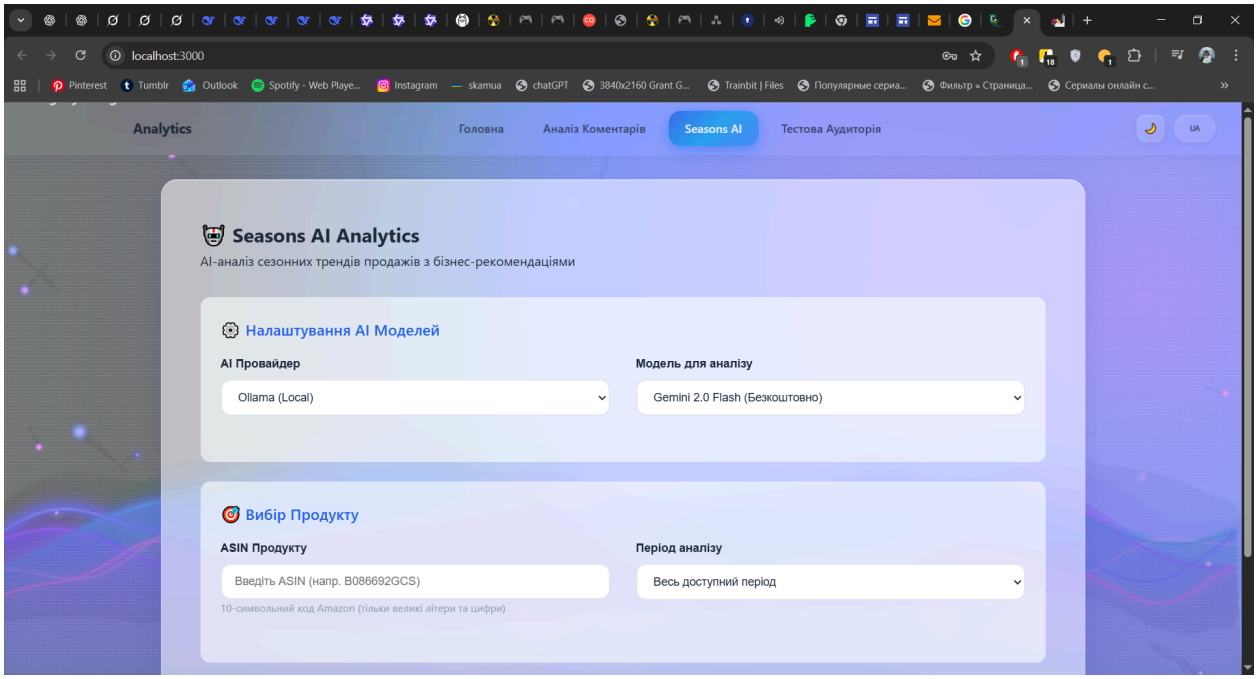


Рисунок Б.4 - Аналітика сезонних ШІ-моделей

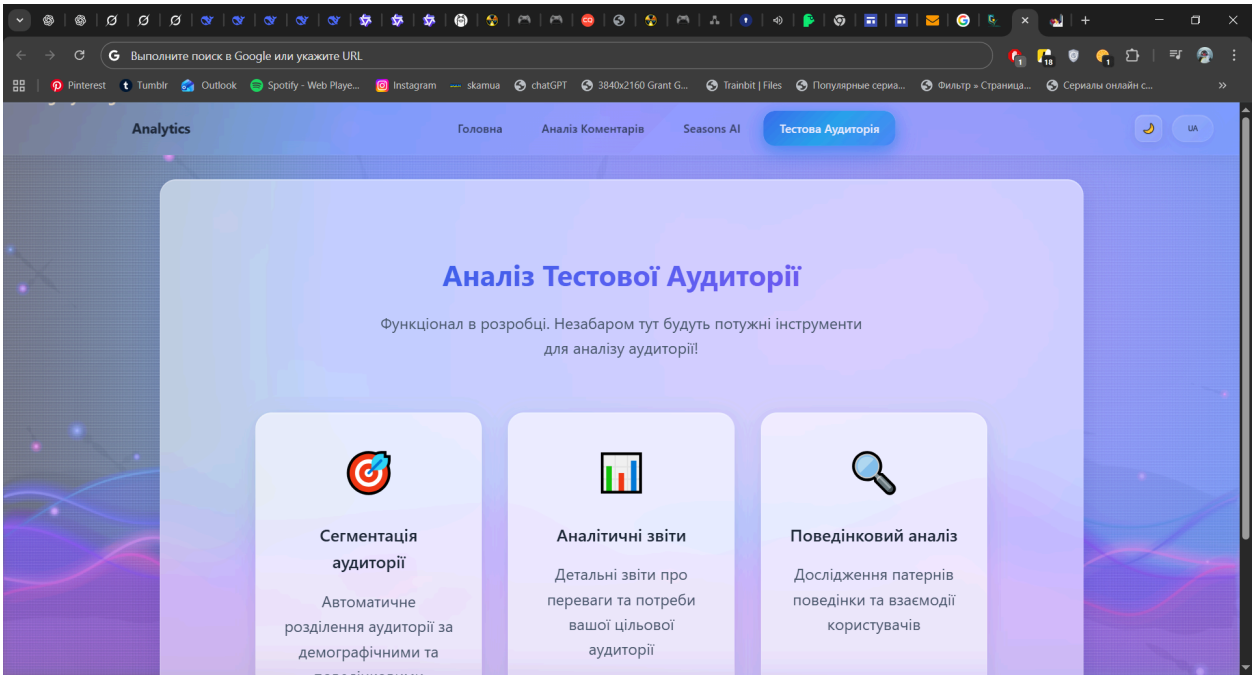


Рисунок Б.5 - Тестова аудиторія

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

Архітектура додатку на основі штучного інтелекту для комерційного аналізу за допомогою LLM-агентів.

Виконав:
Студент групи ІСДМ-62 Сосін Денис Олегович

Керівник:
Сагайдак Віктор Анатолійович

Київ - 2025

Актуальність теми

Операційні виклики сучасного бізнесу виникають на стику технологій та реальних бізнес-процесів, де класичні підходи до аналітики часто виявляються недостатньо ефективними. Компанії електронної комерції щоденно працюють з великими обсягами структурованих даних про продажі та неструктурованих текстових даних, зокрема клієнтських відгуків.

Великі мовні моделі відкривають нові можливості для автоматизованого аналізу таких даних, однак їх практичне застосування пов'язане з ризиками галюцинацій, безпеки, інтеграції з корпоративними системами та контролю автономних дій агентів.

Мета та завдання роботи

Метою магістерської роботи є розробка та дослідження мультиагентної системи на основі великих мовних моделей для інтегрованого аналізу текстових відгуків і прогнозування продажів у сфері електронної комерції.

Для досягнення поставленої мети у роботі вирішено такі завдання:

- проаналізувати сучасний стан використання LLM у бізнес-аналітиці;
- спроєктувати архітектуру агентної системи;
- реалізувати програмну систему аналізу даних;
- провести експериментальне тестування;
- оцінити ефективність і практичну доцільність рішення.

3

Об'єкт і предмет дослідження

Об'єктом дослідження є процес аналізу бізнес-даних у електронній комерції.

Предметом дослідження є методи та засоби побудови агентних систем на основі великих мовних моделей для інтегрованого аналізу структурованих та неструктурованих даних.

4

Наукова новизна

Наукова новизна отриманих результатів полягає у запропонованій агентній архітектурі інтегрованого аналізу бізнес-даних, що поєднує автоматизовану категоризацію текстових відгуків із прогнозуванням продажів у єдиному аналітичному циклі.

Запропоновано використання спеціалізованих агентів із чітким розподілом функцій, механізмів контролю автономних дій агентів, а також ансамблевого підходу до прогнозування часових рядів.

5

Практичне значення

Практичне значення роботи полягає у можливості використання розробленої системи в діяльності аналітичних, маркетингових та продуктових підрозділів підприємств.

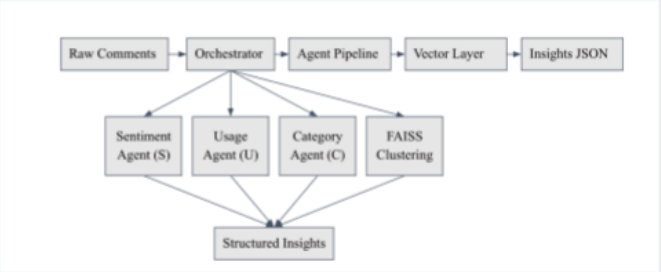
Система підтримує експорт результатів у форматах JSON, CSV та PDF, інтеграцію з Google Sheets і інструментами бізнес-аналітики, а також може працювати як з локальними, так і з хмарними мовними моделями.

6

Архітектура системи

Розроблена система реалізована у вигляді мультиагентної архітектури, що складається з декількох взаємодіючих агентів. Агент аналізу текстів відповідає за попередню обробку та семантичний аналіз відгуків. Агент категоризації виконує багатоступеневу класифікацію даних. Прогностичний агент формує багатофакторні прогнози продажів.

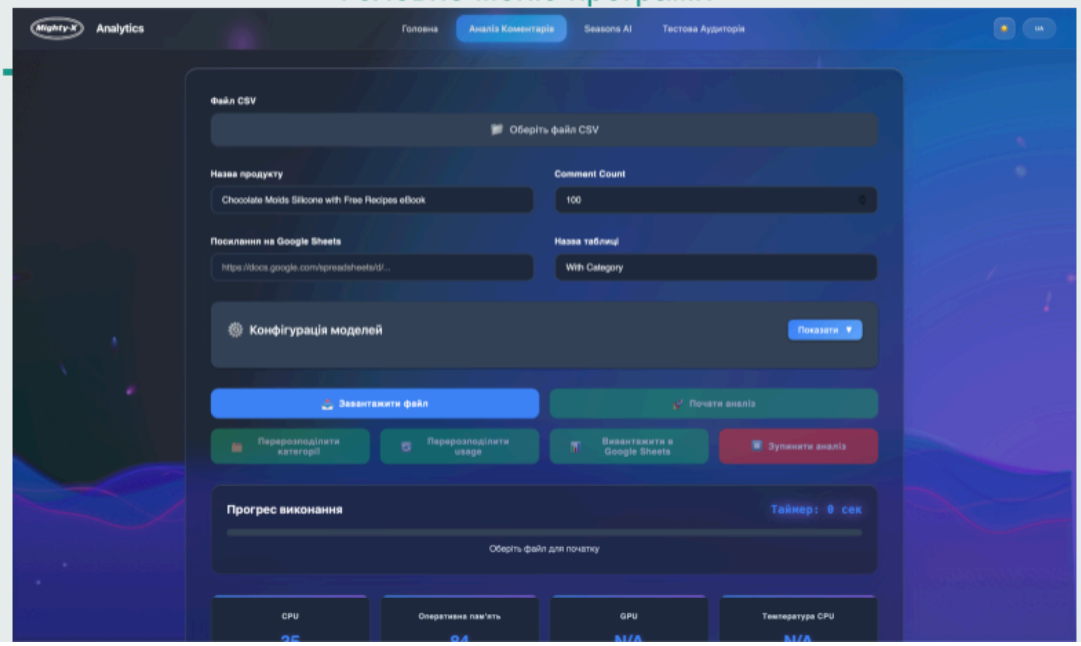
Взаємодія між агентами відбувається з використанням механізмів валідації команд, аудиту дій та принципу найменших привілеїв.



7

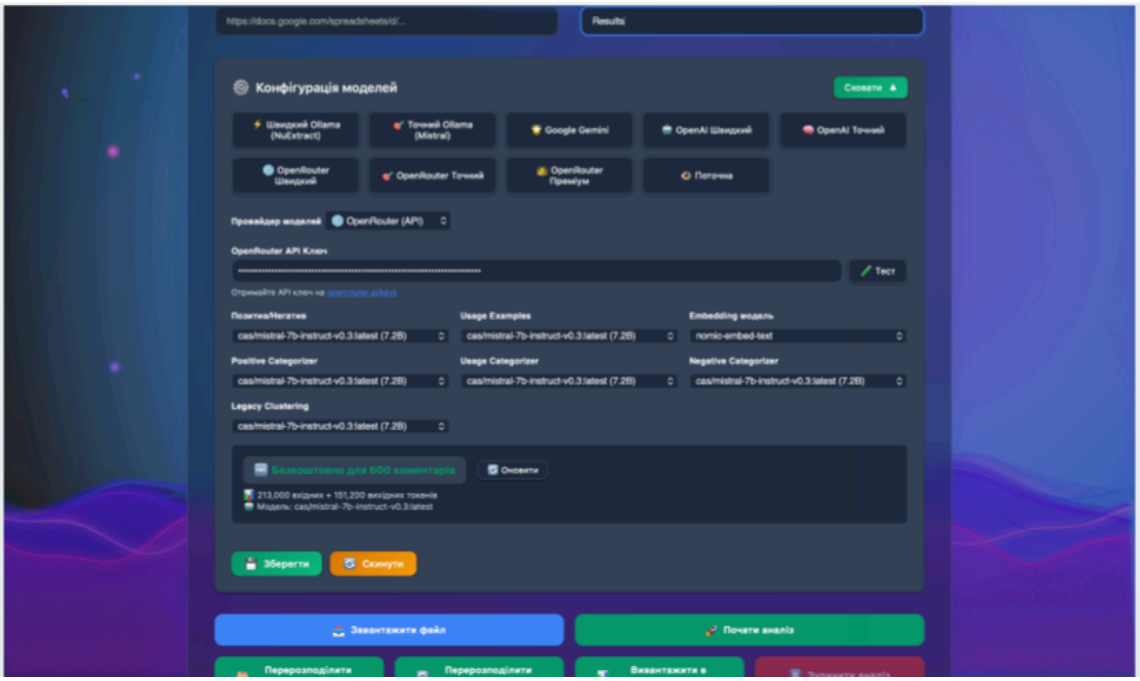
Пайплайн роботи

Головне меню програми



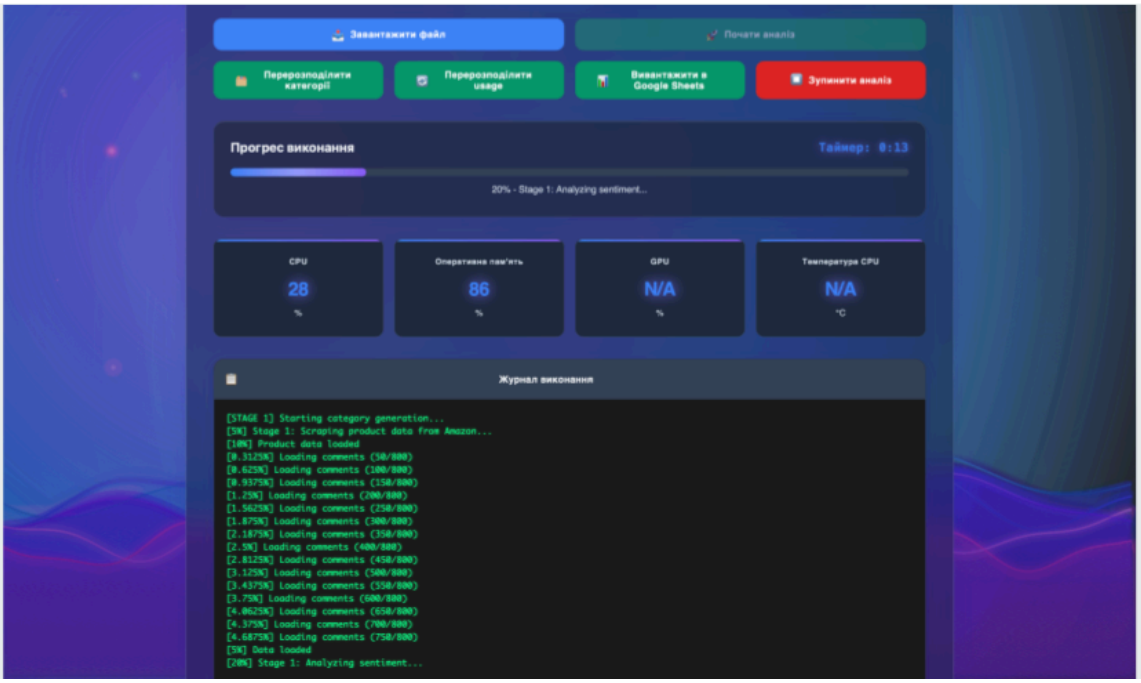
8

Вибір моделей ШІ



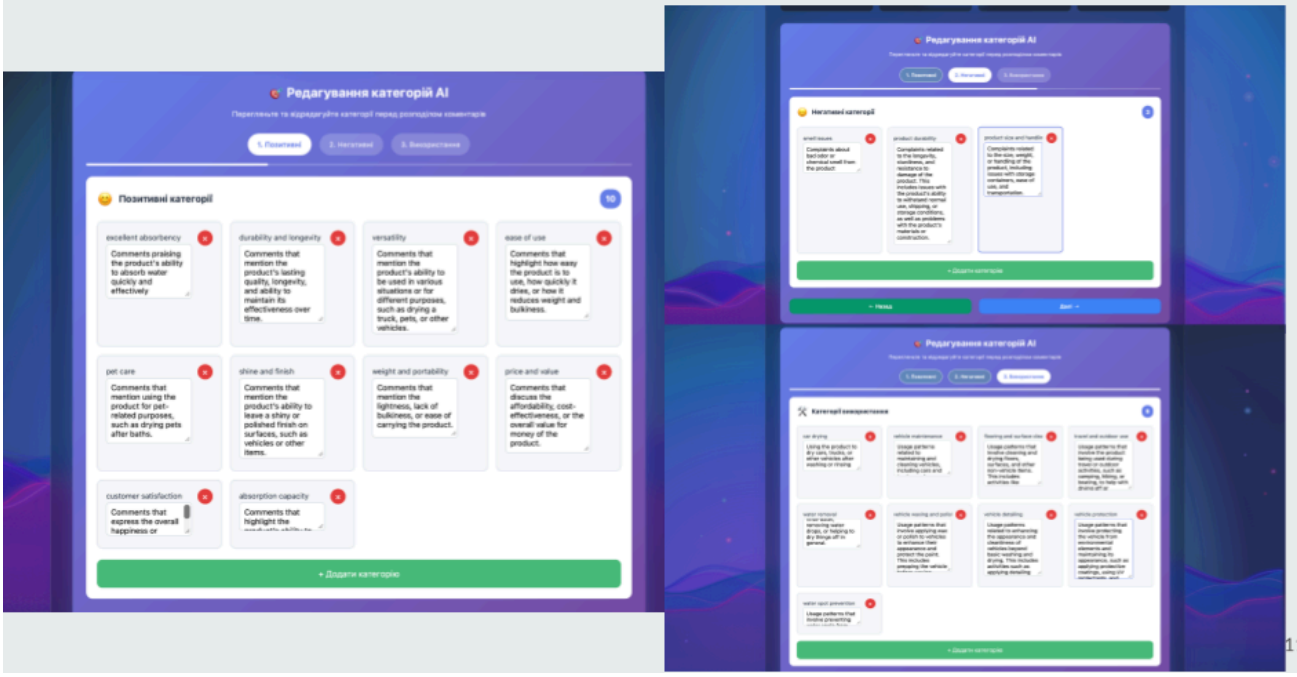
9

Протікання роботи програми



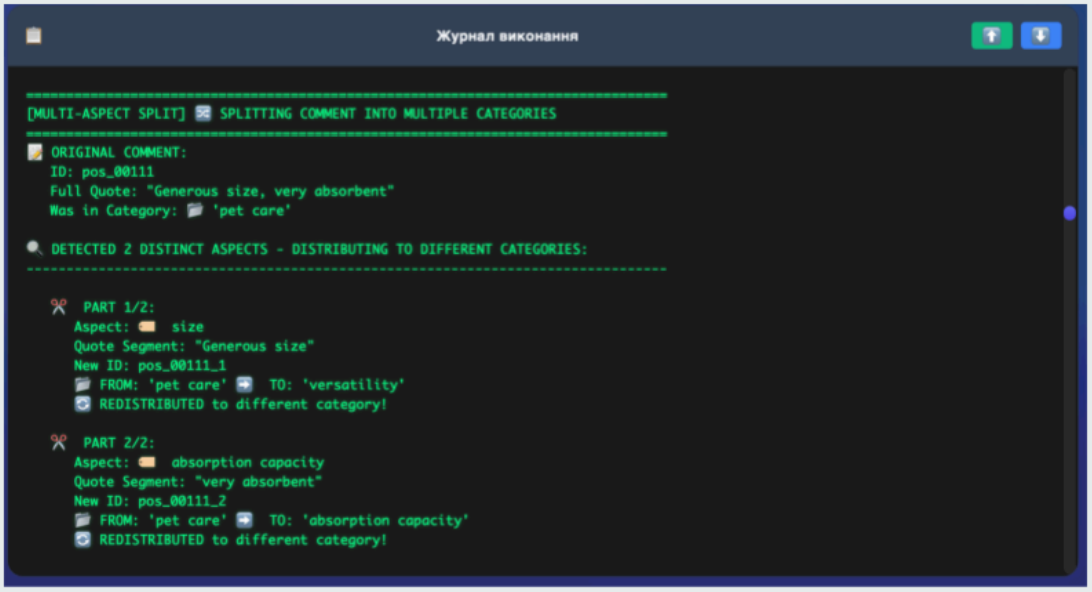
10

Редагування категорій ШІ



11

Автоматичне розділення коментарів що мають декілька характеристик продукту



12

Результати аналізу

Позитивні цитати

Absorbency	Great absorbant	B0BPMMSQ13
Absorbency	absorbent	B0BPMMSQ13
Absorbency	They hold a ton of water.	B0BPMMSQ13
Absorbency	hold moisture all week	B0BPMMSQ13
Absorbency	These are very absorbent	B0BPMMSQ13
Absorbency	Great absorbance.	B0BPMMSQ13
Absorbency	appear to be very absorbent	B0BPMMSQ13
Absorbency	Best absorbency	B0BPMMSQ13
Absorbency	holds a lot of water	B0BPMMSQ13
Absorbency	Great absorbant cloth.	B0BPMMSQ13
Absorbency	Absorb a lot of water	B0BPMMSQ13
Absorbency	picks up a ton of water	B0BPMMSQ13
Absorbency	So absorbent.	B0BPMMSQ13
Absorbency	Absorbent.	B0BPMMSQ13
Absorbency	super absorbent	B0BPMMSQ13
Absorbency	Slurps up the water	B0BPMMSQ13
Absorbency	Simply drinks the water right off your car	B0BPMMSQ13
Absorbency	very absorbent	B0BPMMSQ13
Absorbency	I used these the first time and loved them. very absorbent.	B0BPMMSQ13
Customer Satisfaction	Great deal for the money	B0BPMMSQ13
Customer Satisfaction	These are amazing!	B0BPMMSQ13
Customer Satisfaction	These are great!	B0BPMMSQ13
Customer Satisfaction	Very thankful they were available.	B0BPMMSQ13
Customer Satisfaction	Awesome love them	B0BPMMSQ13
Customer Satisfaction	Have used this product many times and I'm very happy with it.	B0BPMMSQ13
Customer Satisfaction	Excellent product	B0BPMMSQ13
Customer Satisfaction	Dad loved them!	B0BPMMSQ13
Customer Satisfaction	VERY NICE	B0BPMMSQ13
Customer Satisfaction	Love these items	B0BPMMSQ13
Customer Satisfaction	they went above expectations and took care of my issue very quickly and	B0BPMMSQ13
Customer Satisfaction	These towels are the best I have we've used on my cars.	B0BPMMSQ13
Customer Satisfaction	These work wonderfully!!	B0BPMMSQ13
Customer Satisfaction	GREAT PRODUCT	B0BPMMSQ13
Customer Satisfaction	these are the best out there!	B0BPMMSQ13
Customer Satisfaction	A favorite of mine, in the trades.	B0BPMMSQ13
Customer Satisfaction	It's good to know you have these to help you out!	B0BPMMSQ13
Customer Satisfaction	Recomendado.	B0BPMMSQ13
Durability	They also hold up forever.	B0BPMMSQ13
Durability	I expect I can use these for over a year no problem.	B0BPMMSQ13
Durability	My previous one lasted five years and I used it every single week.	B0BPMMSQ13
Durability	Extremely durable.	B0BPMMSQ13
Durability	I have one of these from 25 years ago! And I still use around the house.	B0BPMMSQ13
Durability	durable	B0BPMMSQ13
Ease Of Use	It pretty much took the place of clothe towels when I dry my car after wash	B0BPMMSQ13
Ease Of Use	Wash up very well	B0BPMMSQ13
Ease Of Use	easy to use	B0BPMMSQ13
Ease Of Use	easy to use	B0BPMMSQ13
Ease Of Use	Handy stuff for garage	B0BPMMSQ13
Ease Of Use	A great way to wipe and dry.	B0BPMMSQ13
Ease Of Use	Saw these on a YouTube video and knew they were exactly what I needed	B0BPMMSQ13
Ease Of Use	Easy to use.	B0BPMMSQ13
Ease Of Use	Easy to use	B0BPMMSQ13

Негативні цитати

Product Durability	there was black stuff all over my cloth	B0BPMMSQ13
Product Durability	Doesn't seem to scratch	B0BPMMSQ13
Product Durability	Mine got damaged in shipping	B0BPMMSQ13
Product Durability	I doubt it would even take a drop from 3 feet	B0BPMMSQ13
Product Durability	They delaminated after one use.	B0BPMMSQ13
Product Durability	Scratched the paint on my 67 Mustang	B0BPMMSQ13
Product Durability	Scratched the paint on my car after 2 minutes of using them to dry after washi	B0BPMMSQ13
Product Durability	once they dry after their first use, they take on the actual consistency of a glar	B0BPMMSQ13
Product Durability	They get very hard and stiff	B0BPMMSQ13
Product Size And Handling	Doesn't leave streaks	B0BPMMSQ13
Product Size And Handling	Unfortunately the containers you need to store them in to keep them from dryi	B0BPMMSQ13
Product Size And Handling	Unfortunately the containers you need to store them in to keep them from dryi	B0BPMMSQ13
Product Size And Handling	It's just cumbersome to use since it's so large.	B0BPMMSQ13
Product Size And Handling	It's just cumbersome to use since it's so large.	B0BPMMSQ13
Product Size And Handling	They do suck up water	B0BPMMSQ13
Product Size And Handling	once dry after use, you do need to re-wet them prior to touching paint	B0BPMMSQ13
Product Size And Handling	Tengo un Honda CRV y la toalla no es suficiente para secar todo el auto	B0BPMMSQ13
Product Size And Handling	Tengo un Honda CRV y la toalla no es suficiente para secar todo el auto	B0BPMMSQ13
Product Size And Handling	Absorbent	B0BPMMSQ13
Product Size And Handling	do not slide easily	B0BPMMSQ13
Product Size And Handling	Rather difficult to use.	B0BPMMSQ13
Product Size And Handling	Very streaky	B0BPMMSQ13
Product Size And Handling	failed to deliver any moisture or color	B0BPMMSQ13
Product Size And Handling	They are sticky.	B0BPMMSQ13
Product Size And Handling	I couldn't use them.	B0BPMMSQ13
Product Size And Handling	Not soft like expected	B0BPMMSQ13
Product Size And Handling	If you fold them to store on a shelf, they will harden folded within a day	B0BPMMSQ13
Product Size And Handling	straightening them out is impossibly unless you saturate them with water agai	B0BPMMSQ13
Product Size And Handling	Better than nothing but wasn't great	B0BPMMSQ13
Product Size And Handling	Just spread the water around more than soaked it up	B0BPMMSQ13
Smell Issues	smells awful	B0BPMMSQ13
Smell Issues	it reeks of sulphur	B0BPMMSQ13
Smell Issues	Smells horrible	B0BPMMSQ13
Smell Issues	The next time I got it out the container had mildewed	B0BPMMSQ13
Smell Issues	These were awful!	B0BPMMSQ13

Цитати використання

Car Drying	for their car	B0BPMMSQ13
Car Drying	for cars	B0BPMMSQ13
Car Drying	when I dry my car	B0BPMMSQ13
Car Drying	Dries my black truck	B0BPMMSQ13
Car Drying	for all 3 of my cars	B0BPMMSQ13
Car Drying	drying my car	B0BPMMSQ13
Car Drying	in the winter time	B0BPMMSQ13
Car Drying	to keep them from drying out	B0BPMMSQ13
Car Drying	for drying off a vehicle	B0BPMMSQ13
Car Drying	for large pickup trucks	B0BPMMSQ13
Car Drying	on my cars	B0BPMMSQ13
Car Drying	in the trunk of my car	B0BPMMSQ13
Car Drying	to spot your vehicles paint	B0BPMMSQ13
Car Drying	Drys your car	B0BPMMSQ13
Car Drying	at keeping the car from spotting	B0BPMMSQ13
Car Drying	for drying off car	B0BPMMSQ13
Car Drying	prior to touching paint	B0BPMMSQ13
Car Drying	right off your car	B0BPMMSQ13
Car Drying	for the first time	B0BPMMSQ13
Car Drying	on the paint surface	B0BPMMSQ13
Car Drying	on my car	B0BPMMSQ13
Car Drying	the first time	B0BPMMSQ13
Car Drying	from your car	B0BPMMSQ13
Dry Surface Cleaning	on floors	B0BPMMSQ13
Dry Surface Cleaning	to wipe	B0BPMMSQ13
Dry Surface Cleaning	around the house	B0BPMMSQ13
Dry Surface Cleaning	to wipe up water	B0BPMMSQ13
Frequent Use	all week	B0BPMMSQ13
Frequent Use	every single week	B0BPMMSQ13
General Drying	for drying our pets	B0BPMMSQ13
General Drying	after baths	B0BPMMSQ13
General Drying	to dry dogs	B0BPMMSQ13
General Drying	after bathing them	B0BPMMSQ13
General Drying	to dry our dogs	B0BPMMSQ13
General Drying	to use	B0BPMMSQ13
General Drying	after my doodle gets a bath	B0BPMMSQ13
General Drying	Easy to use	B0BPMMSQ13
General Drying	cut them in half	B0BPMMSQ13
General Drying	after use	B0BPMMSQ13
General Drying	for different size chamois cloths	B0BPMMSQ13
General Drying	to store on a shelf	B0BPMMSQ13
Post-Wash Drying	after washing	B0BPMMSQ13
Post-Wash Drying	drying off rinse water	B0BPMMSQ13
Post-Wash Drying	after a car wash	B0BPMMSQ13
Post-Wash Drying	after washing	B0BPMMSQ13
Post-Wash Drying	to dry	B0BPMMSQ13
Post-Wash Drying	after washing	B0BPMMSQ13
Travel And Outdoor	in overseas trip	B0BPMMSQ13
Travel And Outdoor	in the trades	B0BPMMSQ13

Дані та методи дослідження

Експериментальні дослідження проводилися на наборі даних, що містив близько 50 тисяч транзакцій та 25 тисяч текстових відгуків. Дані проходили багатоступеневе очищення та нормалізацію.

Для оцінки якості прогнозування часових рядів використовувались метрики MAPE та R^2, які дозволяють оцінити як середню відносну похибку прогнозу, так і здатність моделі пояснювати варіацію даних. Для оцінки аналізу текстових відгуків застосовувалась F1-міра, що забезпечує збалансовану оцінку точності та повноти класифікації.

Результати експериментів

Результати експериментів показали, що система забезпечує прийнятну точність прогнозування, стабільну роботу під навантаженням та ефективне масштабування при збільшенні обсягу даних.

Повний аналітичний цикл для набору з 50 тисяч транзакцій виконувався менш ніж за 10 хвилин, а середній час відповіді API залишався на рівні близько 2 секунд навіть при імітації одночасної роботи декількох користувачів.

15

Апробація результатів

Апробація результатів магістерської роботи здійснювалась шляхом участі у наукових конференціях. Основні результати дослідження були представлені та обговорені у вигляді тез доповідей на наукових заходах відповідного напряму. Базові результати магістерської роботи опубліковано в збірнику тез на Міжнародна науково-практична конференції «Інформаційні технології і автоматизація» та на XVIII міжнародної науково-практичної конференції «ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ І АВТОМАТИЗАЦІЯ – 2025».

16

Висновки

У результаті виконання магістерської роботи досягнуто поставленої мети та вирішено всі визначені завдання. Розроблена агентна система підтвердила свою ефективність для інтегрованого аналізу бізнес-даних.

Отримані результати свідчать про доцільність використання великих мовних моделей у поєднанні з агентними підходами для задач бізнес-аналітики.

17

Подальші дослідження

Подальші дослідження можуть бути спрямовані на розширення кількості агентів, інтеграцію з ERP- та CRM-системами, а також використання локальних мовних моделей з підвищеним рівнем конфіденційності.

18