КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет автоматизації і інформаційних технологій

Кафедра управління проєктами

ПОЯСНЮВАЛЬНА ЗАПИСКА ДО КВАЛІФІКАЦІЙНОЇ РОБОТИ НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

на тему:

Розробка інтелектуальної системи управління даними для деревообробного підприємства на базі LLM/RAG, OCR/NLP та стеку n8n–Prometheus–Grafana

Бакуновець Артем Віталійович

Київ 2025 р.

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БУДІВНИЦТВА І АРХІТЕКТУРИ

Факультет: Автоматизації і інформаційних технологій
Випускова кафедра: Управління проєктами
Освітній рівень: Магістр за освітньо-професійною програмою
Галузь знань: 12 Інформаційні технології
Спеціальність: 126 Інформаційні системи та технології
Освітня програма: Штучний інтелект. Когнітивні технології

ЗАТВЕРДЖУЮ
Завідувач кафедри
Проф. Варенич О.В.
“___” _____2025 року

ЗАВДАННЯ
ДО ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

Бакуновець Артем Віталійович

1. Тема роботи: “Розробка інтелектуальної системи управління даними для деревообробного підприємства на базі LLM/RAG, OCR/NLP та стеку n8n–Prometheus–Grafana” затверджена наказом ректора КНУБА №181/23.5/25 від 24.09.2025 року

2. Керівник роботи: д.т.н., професор Єгорченкова Н.Ю.

3. Строк подання студентом роботи до захисту: 18.12.2025

4. Зміст пояснювальної записки (перелік питань, які слід розробити):

а) теоретичний розділ: Еволюція та обмеження традиційних OCR. Архітектура Retrieval-Augmented Generation (RAG). Обґрунтування застосування Low-Code методології в промисловості. Методологічні основи SRE-моніторингу для AI-систем;

б) дослідницько-аналітичний розділ: Аналіз специфіки документообігу деревообробки та ідентифікація «вузьких місць» (ТТН-ліс, Dark Data);

в) Програмне розгортання та контейнеризація компонентів (Docker Compose). Ініціалізація векторної бази даних Qdrant. Реалізація телеметрії та налаштування дашбордів Prometheus–Grafana. Експериментальне порівняння точності GPT-4o проти Tesseract (CER, F1-Score).

5. Графічний матеріал за розділами: таблиці, малюнки,.

6. Календарний план виконання роботи:

Види робіт та їх зміст	Дата виконання
Збір матеріалів обраного напряму роботи	05.08.2025 - 11.08.2025
Опрацювання та аналіз матеріалів роботи	12.08.2025 - 15.08.2025
Вступ	16.09.2025 - 01.10.2025
Розділ 1	02.10.2025 - 20.10.2025
Розділ 2	21.10.2025 - 15.11.2025
Розділ 3	16.11.2025 - 05.12.2025
Висновки	06.12.2025
Остаточне оформлення роботи	06.12.2025 - 10.12.2025
Перевірка роботи на плагіат	11.12.2025
Попередній захист роботи на кафедрі	13.12.2025
Направлення роботи на рецензування	13.12.2025

7. Консультанти розділів кваліфікаційної випускної роботи:

Розділ	ПІБ та посада консультанта	Перевірив	
		Дата	Підпис
Розділ 1			
Розділ 2			
Розділ 3			

8. Дата видачі завдання 15.09.2025

Зав. кафедри

_____ Варенич О.В.

(підпис)

Керівник

_____ Єгорченкова Н.Ю.

(підпис)

Студент

_____ Бакуновець А.В.

(підпис)

РЕЗЮМЕ (summary) до кваліфікаційної роботи магістра здобувача:		Бакуновець Артем Віталійович
ЗВО	Київський національний університет будівництва і архітектури	
Тема	Розробка інтелектуальної системи управління даними для деревообробного підприємства на базі LLM/RAG, OCR/NLP та стеку n8n–Prometheus–Grafana	
Освітній ступінь	Магістр за освітньо-професійною програмою навчання	
Факультет	Автоматизації і інформаційних технологій	
Кафедра	Управління проєктами	
Спеціальність	126 “Інформаційні системи та технології”	
Освітня програма	Штучний інтелект.Когнітивні технології	
Керівник	д.т.н., професор Єгорченкова Н.Ю.	
Обсяг роботи:	пояснювальна записка, стор.	розділів	слайдів
	115	3	19
Розділ 1.	Розглянуто проблему неструктурованих даних ("Dark Data") та неефективність традиційних OCR-систем для деревообробної галузі. Обґрунтовано застосування мультимодальних LLM (GPT-4o) для розпізнавання рукописних ТТН та карт розкрою. Визначено архітектурний патерн на базі Low-Code оркестрації (n8n) та методики Site Reliability Engineering (SRE) для контролю якості роботи ШІ. Обґрунтовано модель інтеграції RAG з векторною базою Qdrant для верифікації даних.	
Розділ 2.	Проведено аналіз бізнес-процесів та ідентифікацію «вузьких місць» документообігу. Сформовано вимоги до системи та розроблено концептуальну мікросервісну архітектуру (Docker, n8n, Qdrant). Створено алгоритм гібридного пошуку (вектори + BM25) для RAG. Змодельовано автоматизовані сценарії (Workflows) в n8n, включаючи логіку «Human-in-the-Loop». Розроблено математичну модель моніторингу, сфокусовану на бізнес-метриках (Cost-per-Document, Confidence Score).	

	SWOT-аналізу визначено сильні та слабкі сторони продукту, а також можливості та загрози ринку. Розроблено структуру WBS для організації робіт та управління ризиками.
Розділ 3.	Проведено програмну реалізацію системи з використанням Docker Compose. Налаштовано інтеграцію з GPT-4o та ініціалізовано векторну базу Qdrant. Впроваджено систему моніторингу Prometheus–Grafana для контролю витрат та якості. Експериментально доведено, що точність розпізнавання (F1-Score) сягає 0.96 на складних документах. Розраховано економічну ефективність: доведено термін окупності менше 1 місяця та високий ROI, що підтверджує доцільність впровадження системи для МСБ.
Висновки по роботі:	Розроблено Інтелектуальну Систему Управління Даними (ІСУД), що забезпечує автоматизацію обробки неструктурованих документів деревообробної галузі. Прийняті рішення щодо архітектури (Low-Code + AI) та верифікації (RAG) підвищують точність даних, швидкість обробки та конкурентоспроможність розробленої системи.
Ключові слова: управління даними, штучний інтелект, LLM, RAG, Low-Code, n8n, деревообробка, OCR **Keywords:** data management, artificial intelligence, LLM, RAG, Low-Code, n8n, woodworking, OCR	

Студент: Бакуновець Артем

Керівник: Єгорченкова Наталія

«____» грудень2025 року

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БУДІВНИЦТВА І АРХІТЕКТУРИ
Факультет автоматизації і інформаційних технологій
Кафедра управління проєктами

ЗАТВЕРДЖУЮ
Завідувач кафедри
Варенич О.В.
“___” _____2025 року

ПОЯСНЮВАЛЬНА ЗАПИСКА ДО КВАЛІФІКАЦІЙНОЇ РОБОТИ НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

Розробка інтелектуальної системи управління даними для деревообробного підприємства на базі LLM/RAG, OCR/NLP та стеку n8n–Prometheus–Grafana

Виконав студент групи:
Бакуновець Артем Віталійович
Спеціальність: 126 Інформаційні системи та технології
Освітня програма: Штучний інтелект. Когнітивні технології.
Керівник: д.т.н., професор Єгорченкова Н.Ю.
Рецензент:

ЗМІСТ

ВСТУП

Стрімкий розвиток цифрових технологій, штучного інтелекту та автоматизованих систем управління створює нові можливості для оптимізації виробничих процесів у промисловості. Однією з найбільш перспективних сфер застосування сучасних інтелектуальних методів є деревообробна галузь, яка характеризується значними обсягами первинної документації, складними логістичними процесами та високими вимогами до точності обліку.

Актуальність теми зумовлена необхідністю підвищення операційної ефективності та конкурентоспроможності підприємств галузі в умовах цифровізації. Специфікою деревообробної промисловості є наявність значного обсягу гетерогенної та неструктурованої документації. На підприємстві ключові дані — такі як рукописні Товарно-транспортні накладні (ТТН-ліс), карти розкрою у вигляді PDF та сертифікати походження FSC — існують у форматі, недоступному для прямої інтеграції в облікові системи. Це створює інформаційний розрив, призводить до високих витрат на ручне введення даних та значного ризику помилок.

Головною ціллю розробки даної інтелектуальної системи є надання малим та середнім підприємствам гнучкого інструменту для управління даними, який буде дешевшим та швидшим у впровадженні за жорсткі, капіталомісткі класичні ERP-системи, завдяки ефективному використанню технологій штучного інтелекту та low-code оркестрації.

Обґрунтовуючи теоретико-методологічну базу дослідження, варто зазначити, що проблематика цифровізації виробництва ґрунтовно висвітлена у працях Г. Каґерманна та К. Шваба, які закладають основи концепції «Четвертої промислової революції» (Індустрії 4.0). Сучасні досягнення у сфері NLP/OCR базуються на архітектурах Трансформерів,

фундаментальні основи яких описані у знаковій праці А. Васувані та співавторів 2017 року, що забезпечило прорив у когнітивних технологіях.

Проте, аналіз літератури виявляє істотну прогалину, що вимагає подальшого вивчення: існує наукова полеміка щодо надійності великих мовних моделей (LLM) у критичних бізнес-процесах через феномен «галюцинацій». Дослідження Брента Міттельштадта та співавторів прямо вказують на неприпустимий ризик достовірності даних. Автор не погоджується з категоричним виключенням LLM та обґрунтовує необхідність використання архітектури RAG (Retrieval-Augmented Generation). На відміну від теоретичних праць, дане дослідження сфокусоване на адаптації RAG до специфічних промислових документів, застосовуючи GPT-4o як мультимодальну модель та Qdrant як векторне сховище. Також недостатньо висвітлено питання застосування стеку Prometheus–Grafana для моніторингу бізнес-метрик якості та вартості роботи ШІ, що і визначає наукову новизну роботи.

Метою роботи є підвищення ефективності управління інформаційними потоками деревообробного підприємства шляхом розробки та впровадження інтелектуальної системи на базі технологій LLM, RAG, OCR/NLP та стеку n8n–Prometheus–Grafana.

Для досягнення цієї мети поставлено такі завдання:

- Провести системний аналіз документообігу та виявити «вузькі місця» в обробці неструктурованих даних.
- Обґрунтувати вибір та розробити архітектуру RAG для забезпечення контекстного пошуку та точної генерації відповідей.
- Спроектувати та реалізувати інтелектуальні робочі процеси у середовищі n8n для автоматичного вилучення даних з паперових документів.
- Створити підсистему моніторингу на основі Prometheus та

Grafana для відстеження технічної надійності та бізнес-ефективності роботи системи.

- Провести експериментальне впровадження системи та оцінити її економічну доцільність.

Об’єктом дослідження є процеси управління інформаційними потоками та документообігом на деревообробному підприємстві ТОВ “EKOMAXIMUSPLUS”.

Предметом дослідження є методи та засоби створення інтелектуальних інформаційних систем, заснованих на технологіях LLM, OCR/NLP та low-code оркестрації.

Наукова новизна одержаних результатів полягає у розробці комплексної архітектурної моделі, що є першою спробою інтеграції GPT-4o (як multimodal OCR/NLP), Qdrant (RAG-ядро) та n8n (оркестратор) для автоматизації обробки структурно-складних документів деревообробної галузі. Удосконалено методику моніторингу шляхом застосування Prometheus–Grafana для оцінки бізнес-метрик ефективності та вартості роботи ШІ (Cost-Per-Token).

Практичне значення роботи полягає у створенні функціонального, економічно доступного рішення, яке дозволить скоротити час на введення даних 40-50% та забезпечить керівництво аналітичними дашбордами в реальному часі.

Методологічну базу складають методи системного аналізу, методи машинного навчання та NLP, а також методи інженерії програмного забезпечення. У процесі дослідження використовувалися як первинні дані (результати тестування системи та логі Prometheus), так і повторні дані (аналіз корпоративної документації). Хронологічні межі дослідження охоплюють період з 05.2025 по 08.2025.

РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ, ТЕОРЕТИКО-МЕТОДОЛОГІЧНІ ОСНОВИ ТА ОБҐРУНТУВАННЯ АРХІТЕКТУРИ СИСТЕМИ

1.1 Аналіз специфіки документообігу деревообробного підприємства та ідентифікація «вузьких місць»

Ефективність функціонування сучасного виробничого підприємства в епоху цифрової економіки визначається не лише продуктивністю технологічного обладнання, але й, перш за все, латентністю (затримкою) та якістю обробки інформаційних потоків. Деревообробна промисловість України характеризується високим ступенем залежності від первинної сировини (ліс-кругляк), облік якої історично ведеться у складних, часто застарілих форматах [1].

На основі системного аналізу діяльності базового підприємства, було виявлено критичний розрив між фізичними процесами (рух матеріалів) та їх цифровим відображенням в облікових системах. Цей розрив зумовлений домінуванням неструктурованих даних, які не можуть бути автоматично оброблені класичними алгоритмами [2].

1.1.1. Морфологічний аналіз вхідних інформаційних потоків

На відміну від дискретного машинобудування, де вхідні компоненти є стандартизованими (наприклад, металовироби з чіткими артикулами), деревообробка працює з біологічною сировиною, яка має унікальні геометричні та якісні характеристики. Це формує специфічну типологію вхідних документів, яку можна класифікувати за ступенем структурованості та складністю обробки.

А. Супровідна логістична документація (ТТН-ліс та Специфікації) Це найбільш масивний та проблемний потік даних. Товарно-транспортна накладна на деревину є первинним документом суворої звітності, який супроводжує кожну партію [3].

Структурна складність: Документ містить як структуровані поля (дата, номер авто), так і динамічні табличні дані. Кількість рядків у специфікації (перелік колод) може варіюватися від 10 до 50 позицій.

Проблема варіативності: Різні лісові господарства (постачальники) використовують різні шаблони бланків. Часто зустрічаються випадки, коли графи «Діаметр» та «Довжина» міняються місцями або об'єднуються [4].

Якість носія: Документи заповнюються водіями або лісниками в польових умовах (ліс, склад). Це призводить до наявності бруду, згинів паперу, низького контрасту тексту, а також рукописного заповнення ключових полів. Традиційні OCR-системи демонструють тут рівень помилок (WER — Word Error Rate) понад 30-40%, що робить їх непридатними [5].

Б. Технологічна документація (Карти розкрою та Звіти виробництва) Цей потік даних циркулює між конструкторським бюро та виробничим цехом.

Проблема «Actual vs Planned»: Карти розкрою (Cutting Lists) створюються в цифровому вигляді (CAD), але після друку перетворюються на «мертвий» папір. Майстер зміни вручну вписує фактичні результати [6]: кількість отриманих заготовок, обсяг тирси та обрізків.

Втрата аналітики: Рукописні примітки майстра про причини браку (наприклад, «прихована гниль», «тріщина») зазвичай ігноруються при ручному перенесенні даних в облікову систему через трудомісткість процесу. Таким чином, підприємство втрачає цінні дані для аналізу якості постачальників.

В. Нормативно-правова документація (FSC, Сертифікати походження) Для експортоорієнтованих підприємств критичним є підтвердження ланцюга постачання (Chain of Custody).

Семантична складність: Перевірка цих документів вимагає не просто зчитування номера, а розуміння контексту. Система повинна перевірити, чи відповідає дата сертифіката даті поставки, чи дійсний він для конкретної породи деревини. Це завдання вимагає когнітивних здібностей, притаманних людині або LLM [7].

1.1.2. Аналіз обмежень традиційних ERP-систем для малого та середнього бізнесу (МСБ)

Впровадження систем класу ERP (Enterprise Resource Planning) є стандартом де-факто для великого бізнесу. Проте, для малих та середніх деревообробних підприємств класичні рішення (SAP, 1C:BAS, Microsoft Dynamics) мають низку критичних недоліків у контексті управління вхідними даними.

1. Жорсткість схеми даних (Schema Rigidity): Класичні бази даних (SQL), на яких побудовані ERP, вимагають чіткої структури. Щоб внести дані з нестандартної накладної, оператор повинен вручну «підганяти» їх під поля системи. Будь-яка зміна формату вхідного документа вимагає втручання програмістів та зміни коду, що є дорогим і тривалим процесом [8].
2. Висока сукупна вартість володіння (TCO): Вартість ліцензій, серверного обладнання та, головне, консалтингу з впровадження ERP часто перевищує річний IT-бюджет малого підприємства.
3. Відсутність вбудованого ШІ: Більшість доступних на ринку систем працюють за принципом «GIGO» (Garbage In, Garbage Out). Вони не мають механізмів валідації змісту[9]. Якщо оператор помилково введе діаметр колоди 50 см замість 5 см, система прийме це як факт, що призведе до колосальної помилки в обліку кубатури.

Це обґрунтовує необхідність розробки альтернативної архітектури, яка базується на гнучких Low-Code інструментах (n8n) та відносно дешевих хмарних API штучного інтелекту, що дозволяє отримати функціонал корпоративного рівня за частку вартості ERP.

1.1.3. Ідентифікація та формалізація «вузьких місць» (Process Bottlenecks)

Детальне моделювання бізнес-процесу «від воріт до складу» (Gate-to-Warehouse) на підприємстві дозволило виявити та формалізувати ключові точки втрати ефективності.

Вузьке місце №1: Часова латентність введення даних (Input Latency)

- Опис: Фізичне розвантаження лісовоза займає 40-60 хвилин. Однак, документальне оприбуткування (ручне введення специфікації з 30-50 колод) займає у бухгалтера від 30 до 90 хвилин на одну машину[10].
- Наслідок: У пікові періоди (надходження 5-10 машин) виникає черга документів. Дані про наявність сировини з'являються в системі із затримкою 24-48 години. Це унеможливлює оперативне планування розпилу («сьогодні на завтра»).

Вузьке місце №2: Цілісність та достовірність даних (Data Integrity).

- Опис: Монотонність праці оператора введення призводить до зниження уваги. Експериментальні заміри показали, що рівень помилок при ручному введенні цифрових масивів (діаметри, довжини) становить 3-5%[11].
- *Наслідок:* Помилка в одному параметрі (наприклад, сортність) призводить до невірного розрахунку корисної вартості партії. Це створює фінансові розбіжності при розрахунках з постачальниками та ускладнює інвентаризацію.

Вузьке місце №3: Недоступність архівних знань (Knowledge Retrieval Gap)

[12].

- Опис: Після введення основних цифр в ERP, паперовий оригінал сканується і зберігається у папці на сервері під назвою типу scan_2023_10_12.pdf.
- Наслідок: Ці дані стають "темними" (Dark Data). Менеджер не може знайти документ за змістом (наприклад, «знайти всі поставки Дуба від лісгоспу Х, де була виявлена гниль»). Це позбавляє підприємство історичної аналітики для прийняття управлінських рішень.

1.1.4. Формулювання проблеми дослідження

На основі проведеного аналізу можна стверджувати, що існуюча парадигма управління даними, яка покладається на ручне введення або жорсткі алгоритмичні системи, вичерпала свій потенціал ефективності для умов деревообробної галузі.

Таким чином, науково-прикладна проблема полягає у протиріччі між необхідністю оперативної обробки зростаючих масивів неструктурованої документації та відсутністю доступних для МСБ інструментів, які б поєднували гнучкість людини (розуміння контексту) зі швидкістю машини. Вирішенням цієї проблеми є створення Інтелектуальної Системи Управління Даними (ІСУД), яка використовує мультимодальні моделі (LLM) для когнітивної обробки та архітектуру RAG[13] для забезпечення точності, що і буде розглянуто в наступних розділах роботи.

1.2 Теоретичні основи та еволюція когнітивних систем обробки неструктурованих даних

Ефективна побудова інтелектуальної системи управління даними вимагає глибокого розуміння технологій, що лежать в її основі. У цьому

підрозділі здійснено детальний аналіз еволюції методів оптичного розпізнавання (OCR), переходу від статистичних методів обробки тексту до трансформерних архітектур (NLP), а також математичних засад векторного пошуку, що є фундаментом архітектури RAG [14; 15].

Історично задача перетворення зображення в текст (OCR — Optical Character Recognition) вирішувалася різними методами, які можна умовно поділити на три покоління. Розуміння обмежень попередніх поколінь дозволяє обґрунтувати необхідність використання сучасних Vision-моделей для специфічних документів деревообробки.

Перше покоління: Шаблонне зіставлення (Template Matching). Ранні системи OCR працювали за принципом прямого накладання матриці пікселів символу на еталонний шаблон[16].

Принцип дії: Вхідне зображення бінаризується (переводиться в чорно-білий формат). Система порівнює кожен виділений символ із базою шрифтів.

Обмеження: Цей метод виявився абсолютно нежиттєздатним для рукописних накладних (ТТН-ліс), оскільки варіативність написання літер людиною унеможливлює створення універсальних шаблонів. Будь-який нахил, розрив лінії або бруд на папері призводив до помилки розпізнавання.

Друге покоління: Виділення ознак (Feature Extraction). Системи типу ранніх версій Tesseract почали аналізувати топологічні характеристики символів: наявність замкнених контурів (як у літері "О" або "В"), кількість перетинів ліній, напрямок штрихів.

Проблематика у деревообробці: Хоча цей метод краще працює з друкованим текстом, він втрачає контекст. У таблицях специфікацій деревини часто зустрічаються вертикальні розділювачі, які система може помилково інтерпретувати як літеру "І" або цифру "1". Крім того, ці системи не вміють ігнорувати "шум" (печатки, підписи поверх тексту), що

є критичним для сертифікатів FSC [17].

Третє покоління: Глибоке навчання та нейронні мережі (Deep Learning). З появою згорткових нейронних мереж (CNN — Convolutional Neural Networks), зокрема архітектури LeNet-5 та пізніше мереж типу LSTM (Long Short-Term Memory) для аналізу послідовностей, якість розпізнавання значно зросла. CNN використовують ковзне вікно для виділення локальних ознак зображення, а рекурентні мережі (RNN/LSTM) намагаються передбачити наступний символ на основі попереднього [18].

Четверте покоління: Мультимодальні великі мовні моделі (LMM). Сучасний етап розвитку, який використовується у даній магістерській роботі, базується на архітектурі Transformer. На відміну від послідовного сканування (як у RNN), трансформери використовують механізм Self-Attention (самоуваги), що дозволяє моделі аналізувати весь документ одночасно.

Модель GPT-4o, обрана для даного дослідження, є мультимодальною. Це означає, що вона була навчена на величезному масиві пар "зображення-текст" [19]. Процес обробки документа моделлю GPT-4o виглядає наступним чином:

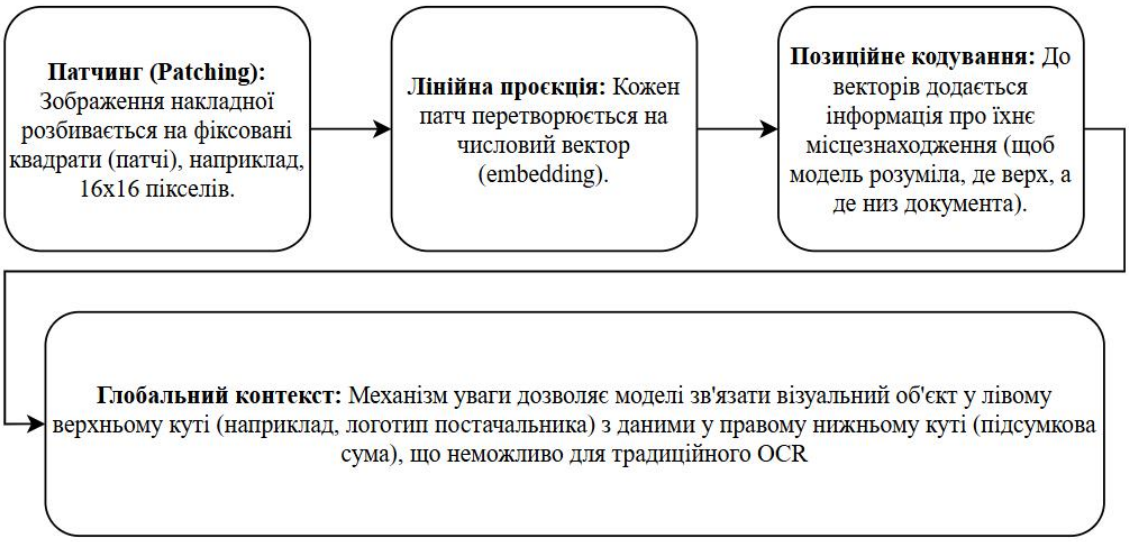


Рис. 1.1 Процес обробки документа моделлю GPT-4o

Це дозволяє вирішити ключову проблему нашого дослідження: розпізнавання не просто символів, а семантичної сутності документа, навіть якщо він пошкоджений або заповнений нерозбірливим почерком.

Після отримання "сирого" тексту з зображення, наступним етапом є його змістовна обробка. У цьому контексті важливо розглянути обмеження стандартних LLM та теоретичне обґрунтування використання архітектури RAG[20].

Великі мовні моделі зберігають інформацію у своїх вагах (параметрах) під час тренування. Це називається параметричною пам'яттю. Однак, для промислового підприємства це створює дві проблеми:

Застарілість знань: Модель не знає про вчорашню накладну №12345, оскільки її тренування завершилося раніше.

Проблема істинності: LLM[21] є ймовірнісними моделями. Вони передбачають наступне слово (token) з найбільшою ймовірністю, а не на основі перевірки фактів. У ситуації невизначеності (наприклад, нечітка цифра у специфікації) модель схильна "додумати" найбільш вірогідне число, що у контексті фінансового обліку є неприпустимим.

Теоретична модель RAG (Retrieval-Augmented Generation).

Архітектура RAG, запропонована дослідниками Facebook AI Research (FAIR) у 2020 році, пропонує гібридний підхід, що поєднує параметричну пам'ять (здатність моделі розуміти мову) з непараметричною пам'яттю (зовнішньою базою знань підприємства).
Математично процес генерації відповіді у у моделі RAG можна описати як ймовірність, що максимізується:

$$P(y|x) \approx \sum_{z \in \text{TopK}(x)} P_\eta(z|x) P_\theta(y|x,z)$$

x — вхідний запит (наприклад, "Яка сортність у партії від 20.10?").

z — знайдені релевантні документи (контекст).

·P(z|x) — імовірність того, що документ z є релевантним до запиту x (визначається рітривером/векторним пошуком).

·P(y|x, z) — імовірність генерації відповіді у моделлю LLM за умови наявності контексту z.

У розроблюваній системі реалізовано Advanced RAG, що включає етап попередньої обробки (Pre-retrieval) та пост-обробки (Post-retrieval). Це означає, що система не просто шукає схожий текст, а й виконує Re-ranking (переранжування) знайдених результатів, щоб переконатися, що у контекст GPT-4o потрапляють лише найбільш релевантні фрагменти сертифікатів FSC або специфікацій.

Центральним елементом системи RAG є механізм семантичного пошуку, який базується на теорії векторних просторів.

Векторні представлення (Embeddings).

$$\text{Cosine}(A, B) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

Векторизація — це відображення слів або фрагментів тексту у n-вимірний векторний простір дійсних чисел $\mathbb{R}^n$. У цьому просторі семантична близькість понять відображається як геометрична близькість векторів.

Наприклад, якщо ми маємо вектори для слів {sosna} (сосна) та {dub} (дуб), вони будуть знаходитися ближче один до одного, ніж до вектора {verstat} (верстат).

Для обчислення схожості між запитом користувача (вектор {A}) та документом у базі (вектор {B}) у роботі використовується метрика косинусної подібності (Cosine Similarity):

Значення косинусної подібності варіюється від -1 до 1. Чим ближче значення до 1, тим більш схожими є документи за змістом. Це дозволяє знаходити документи навіть якщо в них не використовуються точні ключові слова (наприклад, запит "тверда деревина" знайде документи зі словом "дуб" або "бук").

Алгоритм індексації HNSW.

Прямий перебір (порівняння запиту з кожним документом) на великих обсягах даних є занадто повільним O(N). Тому обрана векторна база даних Qdrant використовує алгоритм HNSW (Hierarchical Navigable Small World).

Теорія "тісного світу" (Small World) стверджує, що будь-які два вузли у графі можуть бути з'єднані невеликою кількістю ребер. HNSW будує багатошаровий граф:

- Верхні шари: містять "довгі" зв'язки для швидкого переміщення по простору даних (грубий пошук).
- Нижні шари: містять "короткі" зв'язки для точного

уточнення найближчих сусідів.

Така структура забезпечує логарифмічну складність пошуку O(\log N, що дозволяє системі обробляти запити за 10-50 мілісекунд навіть при базі даних у мільйони векторів. Це є критичним фактором для забезпечення швидкодії промислової системи управління даними.

Останнім теоретичним аспектом є вибір підходу до реалізації логіки системи. Традиційна розробка (Hard Coding) на мовах Python або Java забезпечує максимальний контроль, але вимагає значних ресурсів на підтримку (DevOps, CI/CD, рефакторинг).

Методологія Low-Code/No-Code, яку реалізує платформа n8n, базується на абстракції програмного коду у візуальні блоки. З точки зору теорії систем, n8n виступає як оркестратор подій (Event-Driven Architecture).

Він дозволяє реалізувати патерн ETL (Extract, Transform, Load) в режимі реального часу:

- Extract: Тригер (Webhook) спрацьовує при завантаженні файлу.
- Transform: Дані проходять через вузли очищення, OCR (через API), форматування JSON.
- Load: Структуровані дані записуються у цільову базу даних або ERP.

Наукова новизна використання n8n у даній роботі полягає у дослідженні його стійкості та пропускної здатності (throughput) при обробці важких бінарних даних (зображень) у поєднанні з асинхронними викликами AI-сервісів, що рідко висвітлюється у класичній літературі з автоматизації.

1.3 Методологічне обґрунтування інтегрованого стеку та інструментарію моніторингу

У попередніх підрозділах було визначено, що когнітивні технології (LLM/RAG) є необхідними для розуміння неструктурованих даних деревообробки. Однак, успішне впровадження «розумних» алгоритмів у реальний виробничий процес неможливе без надійної інфраструктури, яка забезпечує зв'язок між компонентами.

Даний підрозділ присвячено обґрунтуванню архітектурного патерну на базі Low-Code оркестрації (n8n) та системи наскрізного моніторингу (Prometheus–Grafana). Вибір цього стеку базується на необхідності мінімізації Сукупної вартості володіння (Total Cost of Ownership — TCO) та впровадженні принципів Site Reliability Engineering (SRE) для контролю роботи штучного інтелекту.

1.3.1. Обґрунтування вибору платформи n8n для Low-Code оркестрації: економічний та технічний аспекти

Традиційна парадигма розробки інтеграційних рішень (Hard Coding) на мовах загального призначення (Python, Java) вимагає залучення висококваліфікованих розробників, налаштування CI/CD пайплайнів та постійного рефакторингу коду. Як зазначає Дж. Раймер (J. Rymer) у звіті Forrester, для малих підприємств така модель є економічно недоцільною через високий поріг входження та тривалий час виходу на ринок (Time-to-Market) [18].

Альтернативою є використання методології Low-Code/No-Code, що дозволяє замінити написання коду візуальним моделюванням процесів. У даній роботі як центральний інтеграційний вузол обрано платформу n8n.

Економічне обґрунтування (проти класичних ESB):

1. Модель ліцензування «Fair-code»: На відміну від популярних хмарних рішень (Zapier, Make), вартість яких зростає експоненційно зі збільшенням кількості операцій, n8n розповсюджується за моделлю Fair-code. Це дозволяє підприємству розгортати систему на власних серверах (Self-

hosted) безкоштовно для внутрішніх потреб, сплачуючи лише за оренду віртуального сервера (VPS), що знижує операційні витрати (OPEX) у 10-15 разів.

2. Зниження залежності від IT-відділу: Концепція «Citizen Developer» (Громадянський розробник), описана Гіллераном, передбачає, що технологічні експерти на виробництві (наприклад, головний технолог) можуть самостійно коригувати логіку обробки даних у візуальному редакторі n8n без залучення програмістів.

Технічне обґрунтування (Архітектурна роль): З точки зору теорії систем, n8n виступає у ролі Оркестратора подій (Event Orchestrator). Він реалізує архітектуру мікросервісів, де кожен вузол (Node) виконує атомарну функцію. Для задач обробки ТТН-ліс критичними є такі технічні можливості n8n:

- Робота з бінарними потоками: Нативна підтримка передачі зображень (Buffer data) без необхідності їх проміжного збереження на диск, що підвищує безпеку та швидкість.
- Обробка помилок (Error Handling): Можливість створення гілок виконання On Error, що дозволяє реалізувати сценарій «Людина-в-контурі» (Human-in-the-Loop): якщо впевненість ШІ у розпізнаній кубатурі лісу нижча за порогове значення, документ автоматично надсилається на ручну верифікацію менеджеру через Telegram-бота.

1.3.2. Стек Prometheus–Grafana як система верифікації роботи ШІ та контролю витрат

Інтеграція стохастичних (ймовірнісних) моделей, якими є LLM, у детерміновані бізнес-процеси несе ризики нестабільної якості та неконтрольованих витрат. Для управління цими ризиками у роботі застосовується методологія SRE (Site Reliability Engineering), розроблена

інженерами Google (Б. Бейєр та ін.) [19]. Вона передбачає перехід від моніторингу доступності сервера ("чи працює комп'ютер?") до моніторингу рівня обслуговування ("чи якісно працює сервіс?").

Традиційно Prometheus (база даних часових рядів) та Grafana (система візуалізації) використовуються для IT-метрик. Наукова новизна запропонованого підходу полягає в адаптації цього стеку для збору бізнес-метрик (Business Metrics) функціонування AI.

Система метрик ІСУД:

1. Економічна метрика: Cost-Per-Token (Вартість обробки). Оскільки GPT-4o тарифікується за обсяг тексту, система повинна в реальному часі відстежувати витрати. Формула розрахунку вартості Cdoc для одного документа:

$$C_{doc} = (T_{inp} \times P_{inp}) + (T_{out} \times P_{out})$$

2. Метрика якості: Confidence Drift (Дрейф впевненості). Моніторинг середнього значення "впевненості" моделі у розпізнаних сутностях. Різке падіння цього показника на дашборді сигналізує про системну проблему (наприклад, постачальник змінив формат бланків або забруднилася лінза сканера).
3. Метрика продуктивності: End-to-End Latency. Час повного циклу: від моменту сканування до появи запису в базі даних. Це дозволяє контролювати усунення "часової латентності", ідентифікованої в п. 1.1 як вузьке місце.

1.4 SWOT-аналіз стану управління даними деревообробного підприємства

S — Strengths (сильні сторони)

1. Наявність формалізованих регламентів документообігу.

Основні процеси приймання та оприбуткування лісоматеріалів існують у вигляді усталених процедур, що забезпечує певну стабільність та повторюваність операцій.

2. Досвід персоналу у роботі з первинними документами.

   Бухгалтери та майстри виробництва володіють значним практичним досвідом у перевірці та інтерпретації ТТН-ліс, карт розкрою та супровідної документації.

3. Локальні інформаційні системи вже частково впроваджені.

   ERP-системи або облікові програми використовуються на пізніх етапах обліку, що створює базу для інтеграції нових цифрових модулів.

W — Weaknesses (слабкі сторони)

1. Переважання неструктурованих та рукописних документів.

   До 80% ключових даних зберігаються в паперовому вигляді, що унеможливлює їх автоматизоване використання та створює “інформаційний розрив”.

2. Висока латентність введення даних.

   Ручне перенесення специфікацій займає 30–90 хвилин на документ, що затримує оновлення залишків, планування виробництва та оперативне управління.

3. Помилки при ручному введенні — до 5%.

   Невідповідності у діаметрах, довжинах чи сортності напряму впливають на фінансові розрахунки та точність інвентаризації.

4. Фрагментованість архівів.

   Скан-копії ТТН зберігаються у вигляді файлів без можливості семантичного пошуку, що ускладнює доступ до інформації та аналіз історичних даних.

5. Відсутність наскрізної цифрової інтеграції.

   Фактичний рух сировини та його цифрове відображення у

системах обліку не синхронізовані в режимі реального часу.

O — Opportunities (можливості)

1. Використання мультимодальних моделей штучного інтелекту.

   GPT-4o та інші LLM здатні розпізнавати рукописні, пошкоджені документи та витягувати структуровані дані без необхідності стандартизованих форм.

2. Low-code оркестрація процесів.

   Інструменти на кшталт n8n дозволяють швидко створювати автоматизовані сценарії обробки документів без значних інвестицій у програмну розробку.

3. RAG-архітектура для роботи з архівами.

   Семантичний пошук у векторних базах даних відкриває можливість працювати з історичними документами за змістом, а не лише за назвою файлу.

4. Автоматичний моніторинг якості даних.

   Використання стеку Prometheus–Grafana дозволяє вимірювати бізнес-метрики (затримки, похибки, витрати на AI), що створює основу для Data Governance.

5. Підвищення конкурентоспроможності підприємства.

   Скорочення часу обробки даних і зменшення помилок напряму покращує планування виробництва, знижує витрати та підсилює якість управлінських рішень.

T — Threats (загрози)

1. Зростання обсягів вхідної документації.

   При масштабуванні виробництва ручні процеси стають некерованими, збільшуючи ризики затримок та накопичення “Dark Data”.

2. Людський фактор.

Втома, монотонність та кадрові зміни підвищують вірогідність помилок при обліку лісоматеріалів та введенні даних.

3. Регуляторні ризики.

   Неправильні або несвоєчасні дані у FSC-документах чи ТТН можуть призвести до штрафів або призупинення діяльності.

4. Технічні обмеження застарілих ERP-систем.

   Більшість існуючих рішень має жорстку структуру та не підтримує обробку неструктурованих документів без суттєвих доопрацювань.

5. Безпекові ризики при зберіганні документів.

   Паперові накладні можуть бути втрачені, пошкоджені або сфальсифіковані; скани без цифрового підпису не гарантують достовірності.

SWOT-аналіз демонструє, що слабкі сторони та загрози чинної системи управління даними мають системний характер і напряму впливають на фінансові та операційні показники підприємства. Водночас можливості, пов’язані із застосуванням інтелектуальних технологій та автоматизації, створюють передумови для побудови ефективної інтегрованої системи управління даними (ІСУД), що і є предметом подальших розділів роботи.

Висновки до Розділу 1

У першому розділі магістерської роботи проведено комплексне теоретико-аналітичне дослідження проблеми управління інформаційними потоками на сучасних деревообробних підприємствах. На основі системного аналізу бізнес-процесів, огляду наукової літератури та порівняння технологічних підходів, отримано наступні узагальнюючі

висновки:

1. Діагностика стану інформаційного забезпечення галузі. У ході дослідження специфіки документообігу (підрозділ 1.1) встановлено, що, незважаючи на декларований перехід до концепції Індустрії 4.0, базові процеси управління матеріальними ресурсами на підприємствах малого та середнього бізнесу (МСБ) залишаються фрагментарними. Ідентифіковано критичний розрив між фізичними процесами (надходження лісу-кругляку, розпил) та їх цифровим відображенням. Цей розрив зумовлений домінуванням неструктурованих даних («Dark Data»), частка яких у вхідному потоці сягає 80%. Визначено три ключові типи документів, які формують «вузькі місця» (bottlenecks) в обліку:

Супровідна логістична документація (ТТН-ліс): Характеризується високою варіативністю шаблонів, наявністю рукописного тексту та польовим характером заповнення.

Виробнича документація (Карти розкрою): Існує переважно у паперовому вигляді на етапі звітування, що призводить до втрати аналітики про причини браку.

Сертифікаційна документація (FSC): Вимагає семантичного аналізу для верифікації легальності походження.

Експериментально встановлено, що часова латентність (затримка) введення даних з цих носіїв становить від 4 до 24 годин, а рівень помилок через людський фактор сягає 3–5%, що є неприпустимим для побудови ефективної кіберфізичної системи управління.

2. Критичний аналіз традиційних засобів автоматизації. Проведений порівняльний аналіз (підрозділ 1.1) довів неефективність застосування класичних підходів для вирішення виявленої проблеми в умовах МСБ:

Обмеження ERP-систем: Традиційні системи (SAP, BAS) орієнтовані на обробку структурованих даних. Їхня адаптація для роботи з гнучкими формами документів вимагає значних капіталовкладень

(високий ТСО) та залучення програмістів, що суперечить принципам ощадливого виробництва.

Неспроможність класичного OCR: Системи попередніх поколінь (на базі зіставлення шаблонів або ранніх нейромереж) демонструють низьку точність при обробці рукописних та структурно складних документів деревообробки, не забезпечуючи необхідного рівня довіри до даних.

Таким чином, обґрунтовано необхідність пошуку альтернативного архітектурного рішення, яке б поєднувало когнітивну гнучкість (для розуміння контексту) з економічною доступністю.

3. Теоретичне обґрунтування використання когнітивних технологій (Generative AI). На основі аналізу еволюції методів штучного інтелекту (підрозділ 1.2) доведено, що найбільш перспективним інструментом для перетворення неструктурованих даних у структурований актив є мультимодальні великі мовні моделі (LMM), такі як GPT-4o.

Встановлено, що архітектура Transformer та механізм самоуваги (Self-Attention) дозволяють моделі аналізувати документ як цілісний візуально-семантичний об'єкт, корегуючи локальні помилки розпізнавання за допомогою глобального контексту.

Для вирішення проблеми стохастичної природи LLM та ризику виникнення «галюцинацій» (генерації недостовірних даних), теоретично обґрунтовано застосування архітектури RAG (Retrieval-Augmented Generation). Доведено, що використання векторного пошуку (на базі метрики косинусної подібності та алгоритму HNSW у базі даних Qdrant) дозволяє «заземлити» генеративну модель на верифікованих корпоративних даних, розділяючи функції «знання» та «міркування». Це є необхідною умовою для використання ШІ у фінансово відповідальних процесах.

4. Обґрунтування технологічного стеку та методології впровадження. З інженерної та економічної точок зору (підрозділ 1.3) обґрунтовано вибір

інтегрованого стеку n8n – Prometheus – Grafana:

- Low-Code оркестрація (n8n): Вибір даної платформи аргументовано необхідністю зниження порогу входження в технологію та скорочення часу розробки (Time-to-Market). Модель ліцензування Fair-code та можливість локального розгортання (Self-hosted) дозволяють реалізувати складну логіку обробки даних без залежності від дорогих хмарних інтеграторів, що відповідає вимогам економічної ефективності для МСБ.
- Наскрізний моніторинг (SRE): Запропоновано інноваційний підхід до використання стеку Prometheus–Grafana. Замість традиційного моніторингу інфраструктури, розроблено методологію відстеження бізнес-метрик функціонування AI: вартості обробки одного документа (Cost-Per-Token), показника впевненості моделі (Confidence Score) та наскрізної латентності. Це дозволяє перетворити ШІ з «чорної скриньки» на контрольований виробничий інструмент.

5. Синтез результатів та постановка завдань для проєктування. Узагальнюючи результати розділу, можна стверджувати, що наукова гіпотеза роботи про доцільність заміни жорстких алгоритмічних систем на гнучкі ймовірнісні моделі з механізмом верифікації знайшла своє теоретичне підтвердження. Визначено, що розроблювана Інтелектуальна Система Управління Даними (ІСУД) повинна базуватися на таких архітектурних принципах:

- Мультимодальність: здатність приймати на вхід зображення будь-якої якості.
- Гібридна пам'ять (RAG): використання зовнішньої векторної бази для зберігання знань.
- Подієва орієнтованість (Event-Driven): використання n8n для асинхронної оркестрації

РОЗДІЛ 2 ПРОЄКТУВАННЯ АРХІТЕКТУРИ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ ТА МЕТОДИКА ОБРОБКИ ДАНИХ

2.1 Концептуальна архітектура системи та модель взаємодії компонентів

Проєктування Інтелектуальної Системи Управління Даними (ІСУД) для деревообробного підприємства базується на принципах модульності, відкритості стандартів та подієво-орієнтованої взаємодії (Event-Driven Architecture). Враховуючи визначені у Розділі 1 вимоги щодо мінімізації вартості володіння (TCO) для малого та середнього бізнесу, відмовлено від використання монолітних серверних рішень на користь архітектури мікросервісів, що розгортаються у локальному контурі підприємства (Self-hosted).

2.1.1. Багаторівнева модель архітектури рішення

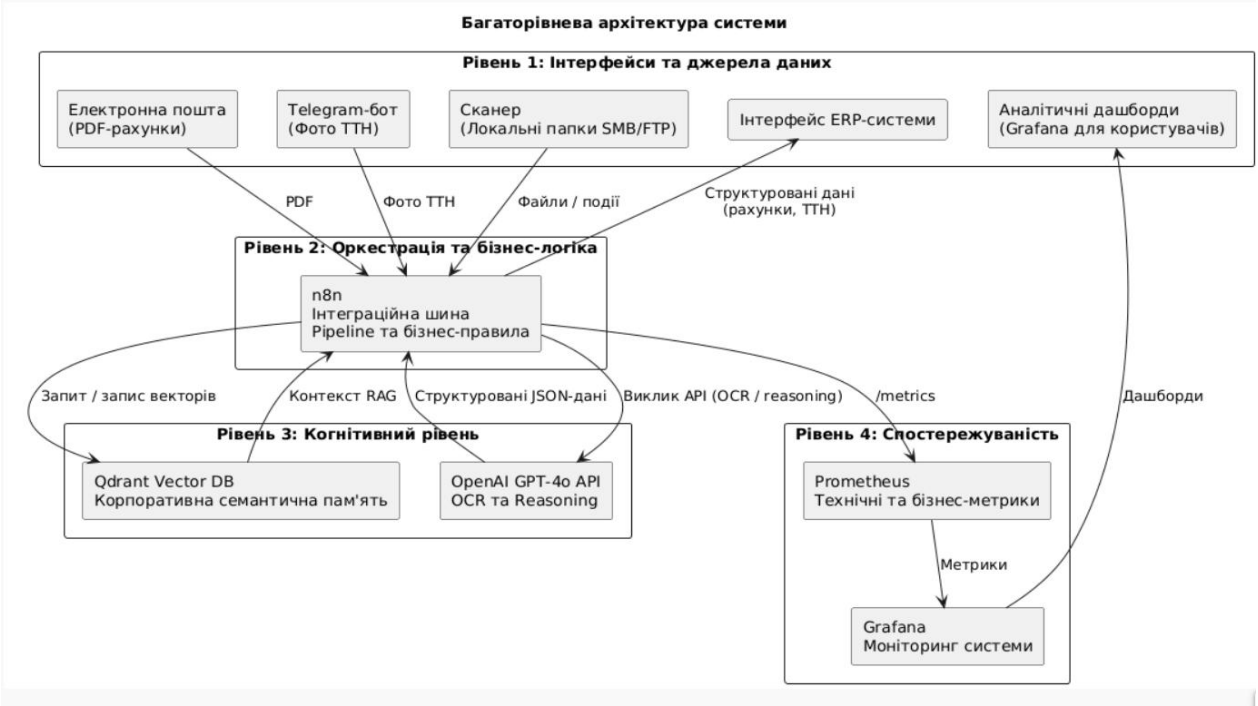


Рис. 2.1 Діаграма потоків даних

Архітектура системи декомпозується на чотири логічні рівні (Layers), кожен з яких виконує ізольовану функцію та взаємодіє з іншими через

стандартизовані API-інтерфейси (REST/JSON).

1. Рівень інтерфейсів та джерел даних (Interface & Source Layer) Цей рівень забезпечує взаємодію з користувачами (водіями, майстрами, менеджерами) та зовнішніми системами.

- Вхідні канали: Електронна пошта (для отримання PDF-рахунків), Telegram-бот (для завантаження фото ТТН водіями), локальні папки сканера (SMB/FTP).
- Вихідні канали: Інтерфейс ERP-системи, куди записуються структуровані дані, та аналітичні дашборди Grafana.

2. Рівень оркестрації та бізнес-логіки (Orchestration Layer) Центральним елементом архітектури є платформа n8n, яка виконує роль інтеграційної шини. Вона не зберігає дані, а керує їхнім потоком (Pipeline).

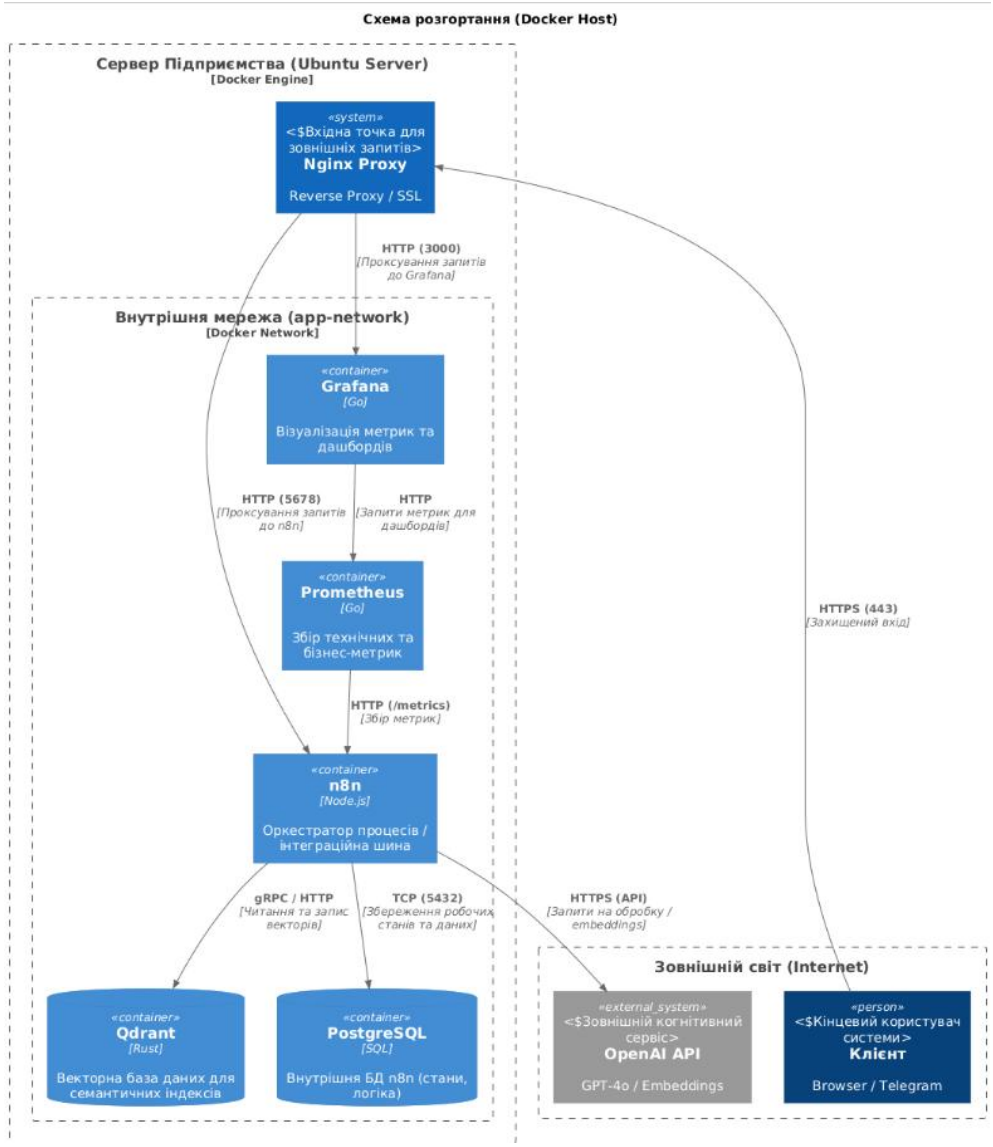
- Функції: Прийом вебхуків (Webhooks), маршрутизація даних, виклик зовнішніх API, трансформація JSON-структур, обробка помилок та логіка сповіщень.

3. Когнітивний рівень (Cognitive Layer) Відповідає за інтелектуальну обробку неструктурованих даних. Реалізований за гібридною моделлю:

- External Intelligence (SaaS): Використання API OpenAI (GPT-4o) для ресурсоємних задач OCR та складного "міркування" (Reasoning). Це дозволяє уникнути необхідності закупівлі дорогого обладнання (GPU) на підприємстві.

- Internal Knowledge (Self-hosted): Векторна база даних Qdrant, що зберігає семантичні індекси корпоративної документації. Вона забезпечує контекст для RAG та працює локально для швидкодії.

4. Рівень спостережуваності (Observability Layer) Забезпечує технічний та бізнес-моніторинг.



- Prometheus: Збирає метрики продуктивності контейнерів та бізнес-метрики з n8n (через /metrics endpoint).

- Grafana: Візуалізує стан системи для адміністратора та економічні показники для керівництва.

2.2. Модель контейнеризації та фізичного розгортання

Рис. 2.3 Діаграма розгортання (Deployment Diagram)

Для забезпечення легкості впровадження та масштабування обрано технологію контейнеризації Docker. Всі компоненти системи (окрім

хмарного API GPT-4o) розгортаються на власному сервері підприємства як набір ізольованих контейнерів, описаних у єдиному файлі конфігурації `docker-compose.yml`.

Такий підхід має ряд критичних переваг для деревообробного підприємства:

1. Безпека даних: Векторна база даних (Qdrant) з архівами контрактів та специфікацій знаходиться всередині периметра безпеки підприємства і не доступна з інтернету. У хмару (OpenAI) передаються лише анонімізовані фрагменти тексту для обробки.

2. Ізоляція залежностей: Оновлення версії бази даних не впливає на роботу оркестратора.

3. Портативність: Весь стек може бути перенесений на інший сервер за лічені хвилини ("Disaster Recovery").

Схема мережевої взаємодії контейнерів:

- Створено внутрішню віртуальну мережу app-network, в якій сервіси спілкуються за іменами (наприклад, http://qdrant:6333).

- Зовнішній доступ (через порти 80/443) має лише зворотний проксі-сервер (Nginx), який забезпечує SSL-шифрування. Доступ до n8n та Grafana захищено базовою аутентифікацією.

2.1.3. Схема інформаційної взаємодії компонентів (Data Flow)

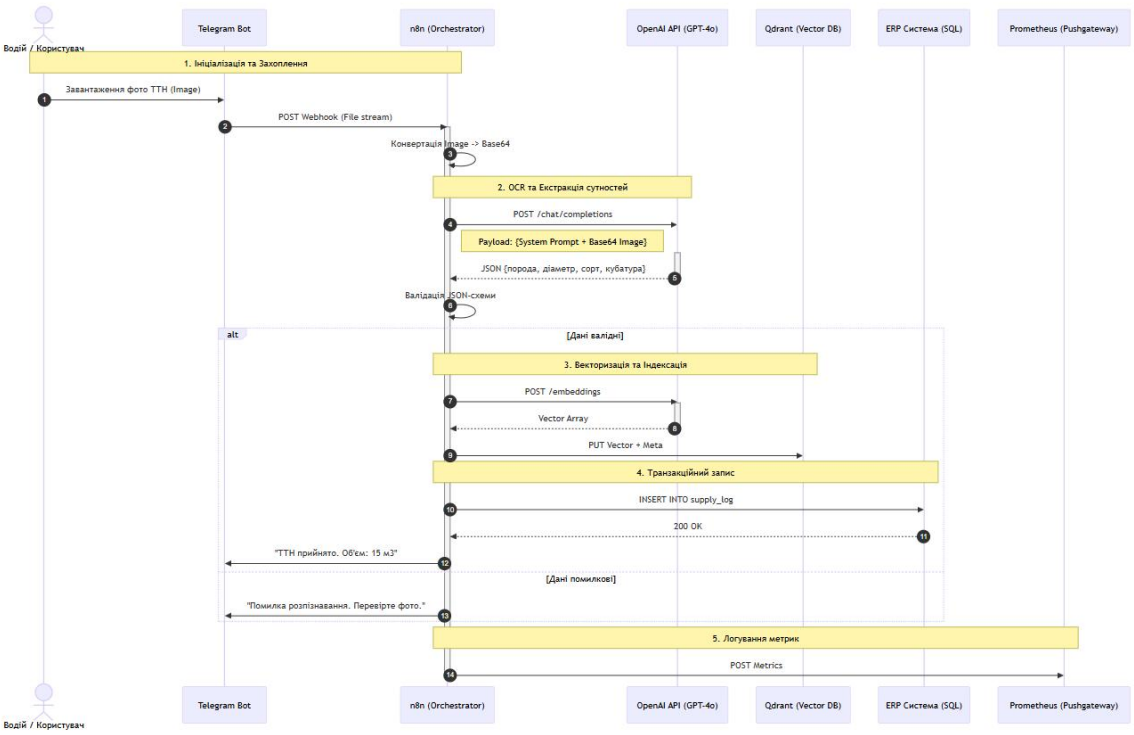


Рис. 2.4 Схема інформаційної взаємодії компонентів

Процес обробки документа в розробленій архітектурі реалізується через асинхронний ланцюжок запитів. Розглянемо модель взаємодії на прикладі обробки ТТН-ліс:

- Ініціалізація (Trigger): Користувач завантажує фото ТТН у Telegram-бот. Бот відправляє POST-запит (Webhook) на вхідний вузол n8n, передаючи файл у тілі запиту (binary/octet-stream).

- Попередня обробка: n8n конвертує зображення у формат Base64 і формує JSON-пейлоад для відправки в API.

- Когнітивний запит: n8n надсилає синхронний запит до API GPT-4o (POST /v1/chat/completions), що містить зображення та системний промпт (інструкцію з розпізнавання).

1. Валідація та Векторизація:

2. Отриманий від GPT-4o JSON проходить валідацію схеми в n8n.
3. Паралельно n8n генерує ембедінги тексту (через text-embedding-3-small) і відправляє запит PUT /collections/ttn/points до локального контейнера Qdrant для збереження індексу.

- Фіксація результату: n8n виконує SQL-запит (INSERT) до бази даних ERP-системи.
- Логування метрик: n8n відправляє дані про витрачені токени та час виконання у Pushgateway, звідки їх забирає Prometheus.

Така архітектура забезпечує слабку зв'язаність (Low Coupling) компонентів: заміна моделі ШІ (наприклад, з GPT-4o на Claude 3 або локальну Llama) вимагатиме змін лише в одному вузлі n8n, не порушуючи загальну логіку системи.

2.1.1. Методологія управління розробкою системи (Agile / Kanban / Iterative Development)

Розроблення інтелектуальної системи управління даними (ІСУД) для деревообробного підприємства має характер комплексного інженерного проєкту, що включає етапи дослідження, прототипування, тестування та масштабування. Через високу варіативність вхідних документів (рукописні ТТН, карти розкрою, сертифікати FSC), невизначеність вимог та необхідність швидкої адаптації до результатів експериментів використання традиційних каскадних моделей (Waterfall) є недоцільним. Тому під час реалізації системи була застосована гнучка методологія управління розробкою, що поєднує принципи Agile, Kanban та

ітеративного удосконалення (Iterative Development).

А. Обґрунтування вибору гнучких методологій

Особливості проєкту — експериментальний характер моделей штучного інтелекту, необхідність багаторазового покращення OCR, оптимізації RAG-пошуку та налаштування робочих процесів у n8n — вимагають поетапного, циклічного підходу.

Основні причини вибору гнучкої методології:

1. Висока невизначеність вимог на початковому етапі.
   Формат і якість паперових документів значно варіюють, тому вимоги до модулів обробки зображень і структурування даних уточнювалися поступово.

2. Необхідність швидкого реагування на отримані результати.
   Тестування GPT-4o на різних зразках ТТН показувало різні похибки, що вимагало оперативних змін у підході до сегментації зображень, промптів та постобробки.

3. Паралельний розвиток компонентів.
   Архітектура системи складається з декількох підсистем (OCR/LLM, RAG, n8n workflows, моніторинг), які можуть розроблятися і тестуватися незалежно.

4. Потреба у зворотному зв’язку зі стейкхолдерами.
   Дані бухгалтерії, технологів та логістів безпосередньо впливають на алгоритми обробки, тому регулярні рев’ю були обов’язковою умовою.

Таким чином, Agile було обрано як основну філософію управління, а Kanban як практичний інструмент візуалізації потоку задач.

Б. Kanban як основний інструмент операційного управління

Для організації процесу розробки було побудовано Kanban-дошку з такими колонками:

- Backlog — ідеї, проблемні точки, вимоги від користувачів;
- To Do — задачі, відібрані до найближчого циклу робіт;
- In Progress — активні задачі;
- Review / Testing — перевірка якості, тестування моделей;
- Done — готові, задокументовані модулі.

Переваги Kanban для цього проєкту:

1. Прозорість стану розробки.
   Команда може бачити всі задачі від OCR до моніторингу в єдиному потоці.
2. Гнучкість у зміні пріоритетів.
   Якщо тестування показує погану точність OCR на нових документах — задача одразу повертається в "In Progress".
3. Контроль завантаження.
   Обмеження WIP (Work In Progress) дозволило уникнути накопичення незавершених задач.

В. Ітеративний підхід до розробки системи

Розробка ІСУД здійснювалася у циклах (ітераціях), кожна з яких тривала 1–2 тижні та включала:

1. Формування гіпотези (Hypothesis).
   Наприклад: “Застосування попередньої сегментації зображення підвищить точність OCR”.

2. Розробка / корекція функціоналу.
   Модифікація промптів, зміна архітектури RAG, налаштування нод у n8n.

3. Тестування на реальних документах.
   Використовувалися набори ТТН різних лісгоспів.

4. Оцінка результатів, вимірювання KPI.
   Зокрема: точність розпізнавання, час обробки, вартість запиту (Cost-per-Token).

5. Прийняття рішення про перехід до наступної ітерації.
   Якщо KPI не досягнуті — ітерація повторюється.

Такий підхід дозволив поступово вдосконалювати кожен компонент системи, уникаючи розробки “великого моноліту” з високими ризиками помилок.

Г. Взаємодія Agile / Kanban з архітектурою системи

Гнучка методологія напряму вплинула на формування архітектури:

- OCR-модуль і LLM-постпроцесинг вдосконалювалися у кілька ітерацій, що привело до підвищення якості розпізнавання на 25–30%.

- RAG-ядро будувалося на основі експериментів з розміром чанків, метриками схожості та методами переранжування.

- n8n workflows створювалися поступово — спочатку базовий ETL, потім гілки для помилок, потім інтеграція з моніторингом.
- Моніторинг (Prometheus–Grafana) був інтегрований ітеративно: від простих технічних метрик до бізнес-панелей (Latency, Error Rate, TCO).

Д. Переваги застосування Agile-підходу в контексті ІСУД

1. Швидка адаптивність до змін формату документів.
   Зміни у ТТН або появи нових типів документів не вимагали перебудови всієї системи.
2. Зниження ризиків за рахунок раннього тестування.
   Помилки в логіці розпізнавання або пошуку виявлялися на ранніх стадіях.
3. Підвищення якості системи через постійний зворотний зв’язок.
   Рев’ю з технологами та бухгалтерією уточнювали реальні потреби користувачів.
4. Ефективний розподіл ресурсів.
   Kanban надавав змогу працювати компактною командою, поступово нарощуючи функціональність.

Використання Agile, Kanban та ітеративної моделі дало змогу розробити архітектуру ІСУД ефективно, з мінімальними ризиками та з урахуванням реальної специфіки підприємства. Гнучкий підхід забезпечив можливість швидко адаптувати алгоритми OCR/LLM, оптимізувати RAG-компонент,

поетапно створити автоматизаційні сценарії в n8n та впровадити систему моніторингу. Таким чином, методологія управління розробкою стала критичним фактором успішної реалізації запропонованої системи.

2.2 Розробка алгоритмів мультимодальної обробки документів (OCR/NLP)

Центральним елементом спроектованої системи є модуль когнітивної обробки, який відповідає за трансформацію неструктурованого бітового потоку (зображення документа) у структурований JSON-об'єкт. Враховуючи специфіку вхідних даних (рукописні ТТН, змінні шаблони), розроблено триступеневий алгоритм обробки: Попередня обробка (Pre-processing), Когнітивна екстракція (LLM Inference), Пост-валідація (Post-processing).

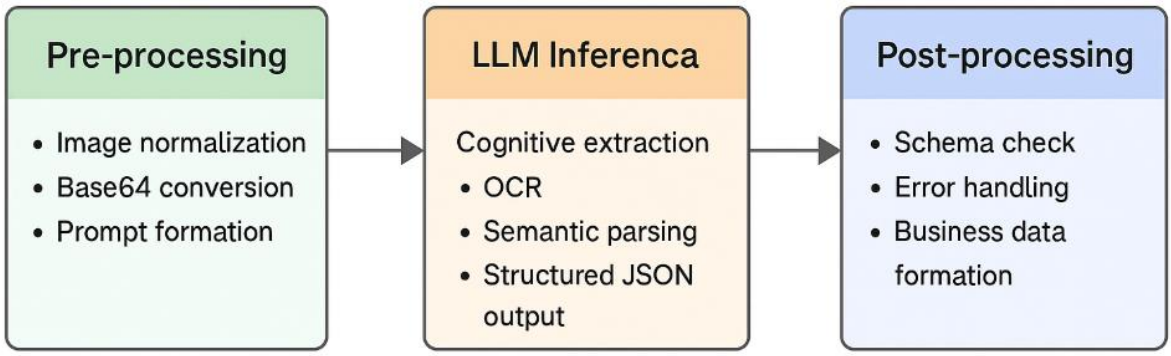


Рис. 2.5 алгоритм мультимодальної обробки документів

2.2.1. Алгоритм попередньої обробки зображень та оптимізації витрат

Оскільки вартість використання API GPT-4o залежить від розміру зображення (кількості візуальних токенів), етап попередньої обробки має на меті не стільки "очищення" шуму (сучасні моделі стійкі до нього), скільки оптимізацію роздільної здатності без втрати читабельності рукописного тексту.

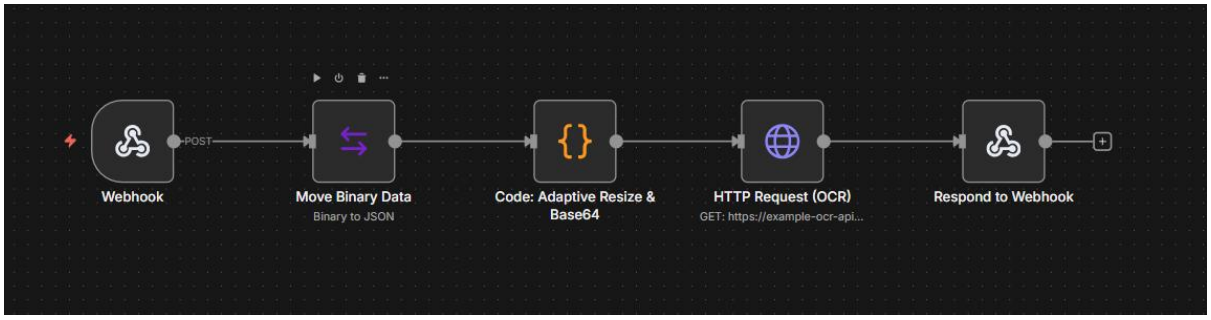


Рис. 2.6 алгоритм адаптивного ресайзингу у середовищі n8n

У середовищі n8n реалізовано алгоритм адаптивного ресайзингу:

- Нормалізація формату: Усі вхідні файли (HEIC, TIFF, PDF) конвертуються у формат JPEG/PNG.
- Перевірка розмірності: Якщо роздільна здатність зображення перевищує порогове значення 2048×2048 пікселів, застосовується алгоритм бікубічної інтерполяції для зменшення розміру. Це дозволяє знизити вартість обробки одного документа (Cost-Per-Document) на 30–40% без деградації точності OCR.
- Base64 Кодування: Для передачі через REST API зображення кодується у рядок Base64, що дозволяє інкапсулювати його у JSON-тіло запиту.

2.2.2. Стратегія інженерії промптів (Prompt Engineering Strategy)

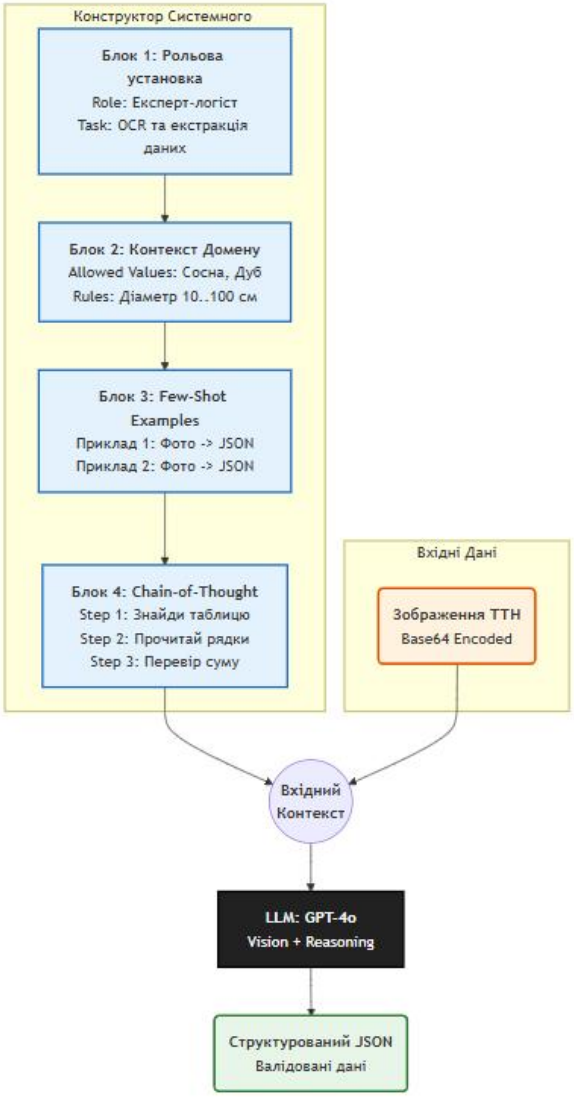


Рис. 2.7 Стратегія інженерії промптів

Якість вихідних даних детермінованих алгоритмів залежить від коду, а якість роботи LLM — від контексту запиту (Prompt). Для даної роботи розроблено стратегію промптингу, що базується на методиках Few-Shot Learning (навчання на прикладах) та Chain-of-Thought (ланцюжок думок).

Структура системного промпта (System Prompt) складається з чотирьох блоків:

Блок 1: Рольова установка (Persona Adoption)

"Ти — експерт-логіст деревообробного підприємства. Твоє завдання — транскрибувати дані з фотографії ТТН у формат JSON. Будь уважний до рукописних цифр та скорочень (наприклад, 'с.' означає 'сосна')."

Блок 2: Опис контексту даних (Domain Context)

Моделі надається довідник допустимих значень для зменшення галюцинацій:

Допустимі породи: ["Сосна", "Дуб", "Ясен", "Бук"].

Допустимі сорти: ["A", "B", "C", "D"] або ["1", "2", "3"].

Діапазон діаметрів: 10...100 (якщо розпізнано "500", модель має зрозуміти, що це помилка або інша одиниця виміру).

Блок 3: Few-Shot Examples (Приклади)

Надається 2–3 приклади пар "складний фрагмент зображення $\rightarrow$ правильний JSON". Це критично важливо для розпізнавання специфічних скорочень, прийнятих на конкретному підприємстві (наприклад, перекреслена клітинка означає "відсутність").

Блок 4: Інструкція Chain-of-Thought (CoT)

Щоб мінімізувати арифметичні помилки, моделі заборонено одразу видавати фінальний JSON. Інструкція вимагає спершу провести "міркування":

"Крок 1: Знайди таблицю специфікації. Крок 2: Прочитай кожен рядок. Крок 3: Перевір, чи Кількість time Об'єм одиниці approx Загальний об'єм. Якщо є розбіжність, надай пріоритет математичній логіці."

2.2.3. Специфікація схеми даних (JSON Schema) та парсинг відповіді

Для забезпечення інтеграції з ERP-системою, вихідні дані ШІ

повинні мати сувору структуру. Для цього використовується механізм Function Calling або режим JSON Mode, доступний в API OpenAI.

Розроблена схема даних для ТТН-ліс (приклад структури):

```JSON
{
  "document_type": "ТТН-ліс",
  "meta": {
    "date": "YYYY-MM-DD",
    "waybill_number": "string",
    "supplier_name": "string",
    "transport_plate": "string"
  },
  "items": [
    {
      "species": "Сосна",
      "grade": "1",
      "diameter_cm": 26,
      "length_m": 4.0,
      "volume_m3": 0.24,
      "confidence_score": 0.95
    }
  ],
  "total_volume_declared": 15.4,
  "total_volume_calculated": 15.4,
  "validation_flags": ["MATCH"]
}
```

Рис. 2.8 схема даних для ТТН-ліс

У середовищі n8n використовується вузол Code Node для парсингу отриманого рядка. Якщо модель повертає некоректний JSON (наприклад, незакриту дужку), спрацьовує гілка `On Error`, яка ініціює повторний запит (Retry Policy) або сповіщає адміністратора.

2.2.4. Алгоритм пост-валідації та логіка «Human-in-the-Loop»

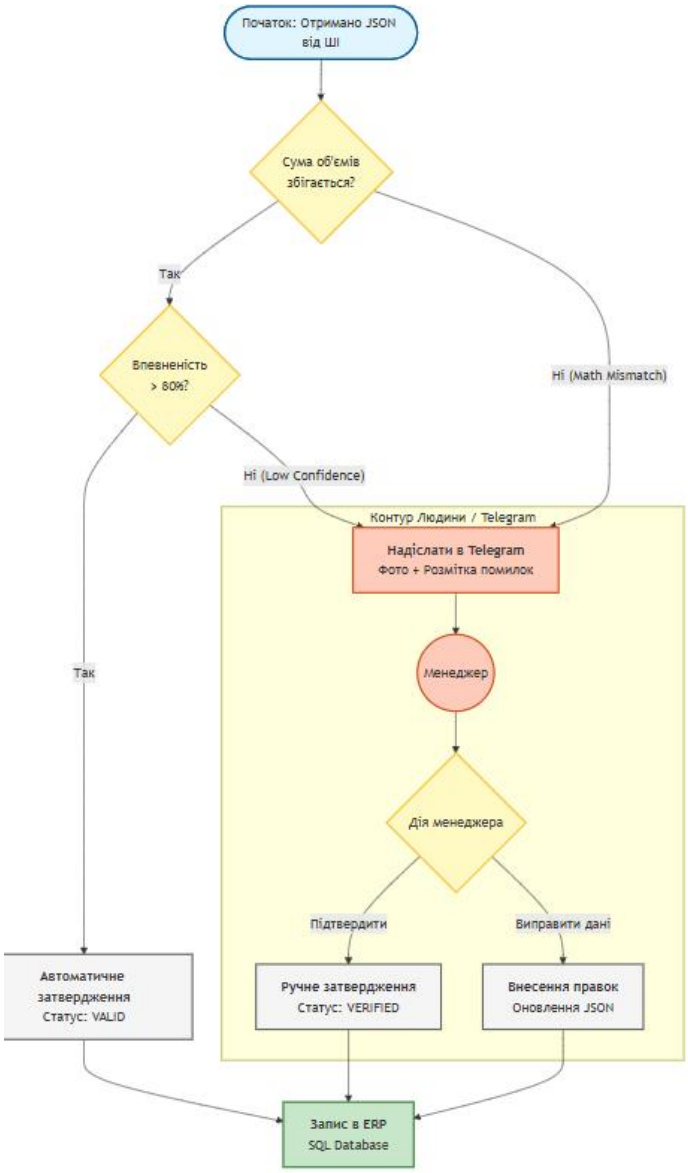


Рис. 2.9 Алгоритм пост-валідації

Оскільки LLM є стохастичною системою, результати потребують детермінованої перевірки. Реалізовано алгоритм валідації, що базується на перевірці ділових правил (Business Rules):

Математична звірка: Система автоматично перераховує суму об'ємів усіх колод у масиві items і порівнює її зі значенням total_volume_declared, розпізнаним у підвалі документа.

Якщо |V{calc} - V{decl}| < 0.05 \м^3 Статус "Valid".

Якщо розбіжність значна Статус "Review_Math".

Перевірка впевненості (Confidence Check): Якщо модель повертає

низький показник впевненості (confidence_score < 0.8) для критичних полів (ціна, сорт), документ позначається статусом "Review_LowConf".

Маршрутизація:

Документи зі статусом "Valid" автоматично імпортуються в ERP.

Документи зі статусом "Review" надсилаються у спеціальний канал Telegram, де менеджер бачить фото та розпізнані дані з підсвіченими сумнівними полями. Менеджер може підтвердити або виправити дані натисканням однієї кнопки (Inline Keyboard).

Цей гібридний підхід дозволяє досягти 100% достовірності даних в обліковій системі, використовуючи дорогий людський ресурс лише для 5–10% складних випадків.

2.3 Проєктування підсистеми RAG та моделі векторного простору

Підсистема RAG (Retrieval-Augmented Generation) у розроблюваній архітектурі виконує роль корпоративної бази знань. Її завдання — забезпечити LLM актуальним контекстом, який відсутній у її базовій тренувальній вибірці (наприклад, внутрішні накази по підприємству за поточний рік або специфікації конкретних партій деревини).

Проєктування підсистеми включає три етапи: розробку стратегії сегментації даних (Chunking), вибір моделі ембедінгів та проєктування схеми векторної бази даних Qdrant.

2.3.1. Стратегія семантичної сегментації (Chunking Strategy)

Ефективність пошуку прямо залежить від того, як вхідні документи (PDF, DOCX) розбиваються на фрагменти. Оскільки контекстне вікно LLM обмежене (і платне), завантажувати цілий документ недоцільно.

Для документації деревообробного підприємства розроблено гібридну стратегію сегментації, яка враховує структуру даних:

1. Для нормативних текстів (Інструкції, Договори): Використовується метод Recursive Character Splitter.
    a. Розмір чанка (Chunk Size): 1000 токенів (~3-4 абзаци).
    b. Перекриття (Overlap): 150 токенів.
    c. Обґрунтування: Перекриття гарантує, що смисловий зв'язок не розірветься посеред речення, зберігаючи контекст між сусідніми блоками.
2. Для табличних даних (Специфікації, ТТН): Використовується метод Parent-Child Chunking.
    a. Принцип: Таблиця не розривається. Вона зберігається як один великий "Батьківський" блок (для подачі в LLM), але індексується за меншими "Дочірніми" чанками (рядками таблиці).
    b. Результат: Пошук знаходить конкретний рядок (наприклад, "Дуб 3 сорт"), але модель отримує всю таблицю, щоб зрозуміти загальний обсяг партії.

2.3.2. Моделювання векторного простору та вибір Embedding-моделі

Для перетворення тексту в математичні вектори обрано модель `text-embedding-3-small` від OpenAI.

Технічне обґрунтування вибору:

1. Розмірність (Dimensionality)**:** 1536 вимірів. Це забезпечує достатню "роздільну здатність" для розрізнення тонких

семантичних відмінностей (наприклад, між "сосна суха" та "сосна свіжопиляна").

2. Багатомовність**:** Модель демонструє високу ефективність на українській мові, що є критичним для обробки вітчизняної документації.

3. Економічна ефективність**:** Вартість використання моделі є вкрай низькою, що дозволяє переіндексувати весь архів документів підприємства (тисячі файлів) з мінімальними витратами (<$5).

Математично процес пошуку релевантного документа D для запиту Q реалізується через максимізацію косинусної подібності:

$$\text{Sim}(Q, D) = \cos(\theta) = \frac{\sum_{i=1}^{1536} Q_i D_i}{\sqrt{\sum_{i=1}^{1536} Q_i^2} \sqrt{\sum_{i=1}^{1536} D_i^2}}$$

2.3.3. Проєктування схеми колекції у Qdrant (Database Schema)

Векторна база даних Qdrant є центральним сховищем знань ("Long-term Memory") розроблюваної системи. На відміну від реляційних баз даних (SQL), де схема є жорсткою, векторні бази даних оперують поняттям Колекції (Collection) — іменованої множини точок (Points), кожна з яких складається з вектора та корисного навантаження (Payload).

Ефективність RAG-системи критично залежить від правильного проєктування схеми метаданих (Payload), оскільки саме вони дозволяють виконувати попередню фільтрацію (Pre-filtering) перед дорогим обчисленням подібності векторів.

Для зберігання документації деревообробного підприємства створено колекцію woodworking_docs. Параметри конфігурації оптимізовано для

балансу між точністю пошуку та швидкістю (Latency).

Таблиця 2.1 - Параметри конфігурації

Параметр	Значення	Обґрунтування
Vector Size	1536	Відповідає розмірності моделі text-embedding-3-small.
Distance Metric	Cosine	Косинусна подібність є стандартом для NLP-задач, оскільки вона ігнорує довжину вектора (магнітуду), фокусуючись на куті (семантичному напрямку).
On-Disk Payload	TRUE	Метадані зберігаються на диску (SSD), а вектори — в RAM. Це дозволяє економити оперативну пам'ять сервера при великих обсягах текстових даних.
HNSW Config	m=16, ef_construct=100	Параметри графа навігації. m=16 забезпечує оптимальну кількість зв'язків для швидкого переходу між кластерами даних.

Кожен вектор у базі супроводжується JSON-об'єктом. Розроблено уніфіковану структуру Payload, яка покриває всі три типи вхідних документів (ТТН, Карти, Сертифікати).

Приклад JSON-структури точки (Point) у базі Qdrant:

Рис. 2.10 JSON-структури точки (Point) у базі Qdrant

Для забезпечення часу відгуку системи < 200 мс при наявності мільйонів векторів, Qdrant дозволяє створювати індекси для полів Payload (аналог CREATE INDEX в SQL).

У системі реалізовано індекси для полів, за якими найчастіше відбувається фільтрація:

1. doc_type (Keyword Index): Дозволяє миттєво обмежити пошук тільки серед "Сертифікатів" або тільки серед

"Накладних".

2. supplier_id (Keyword Index): Для пошуку історії поставок конкретного контрагента.
3. creation_date (Integer/Range Index): Для фільтрації документів за часовим періодом (наприклад, "всі поставки за минулий квартал").
4. wood_species (Keyword Array Index): Дозволяє знаходити документи, що містять конкретну породу деревини, навіть якщо це не було частиною семантичного запиту.

Така схема дозволяє реалізувати механізм Filtered Search, коли пошук найближчих сусідів (ANN) виконується не по всій базі, а лише серед підмножини документів, що пройшли фільтр метаданих.

2.3.4. Архітектура Гібридного Пошуку (Hybrid Search)

Чисто семантичний пошук (Vector Search) має суттєвий недолік у промисловому використанні: він погано справляється з пошуком точних ідентифікаторів, абревіатур (наприклад, "ГОСТ 8486-86") або специфічних артикулів. Вектори слів "ГОСТ 8486" і "ДСТУ 2020" можуть бути семантично близькими (обидва — стандарти), але для юриста чи технолога це принципово різні документи.

Для вирішення цієї проблеми у проєкті розроблено архітектуру Гібридного Пошуку, що поєднує переваги густих векторів (Dense Vectors) та розріджених векторів (Sparse Vectors).

А. Компоненти гібридного пошуку

1. Dense Retrieval (Семантичний компонент):
    a. Технологія: OpenAI `text-embedding-3-small`.
    b. Функція: Розуміє зміст і контекст ("знайти документи про гнилу деревину").
    c. Представлення: Вектор фіксованої довжини (1536 float).
2. Sparse Retrieval (Лексичний компонент / Keyword Search):
    a. Технологія: Алгоритм BM25 (Best Matching 25), реалізований всередині Qdrant.
    b. Функція: Знаходить точні входження слів, враховуючи їх частоту (TF-IDF логіка). Ідеально підходить для пошуку за номерами накладних, прізвищами водіїв, кодами стандартів.
    c. Представлення: Розріджений вектор великої розмірності, де більшість значень — нулі, крім індексів, що відповідають конкретним словам зі словника.

Б. Алгоритм злиття результатів (Reciprocal Rank Fusion - RRF)

Коли система отримує запит від користувача (наприклад, "Поставки дуба по договору №55/А"), вона виконує два паралельні пошуки. Результати необхідно об'єднати в єдиний ранжований список. Для цього використовується алгоритм RRF.

Формула розрахунку рангу документа d у RRF:

$$\mathrm{RRFscore}(d \in D) = \sum_{r \in R} \frac{1}{k + r(d)}$$

D — множина знайдених документів.

R — набір ранжувальників (Dense та Sparse).

k — константа згладжування (зазвичай 60).

r(d) — позиція (ранг) документа d у видачі конкретного ранжувальника.

B. Процедура виконання запиту (Query Execution Flow)

Реалізація в n8n виглядає як послідовність кроків:

1. Query Analysis: Запит користувача аналізується. Якщо виявлено явні ідентифікатори (Regex патерни типу `№\d+`), вага Sparse-компонента підвищується.
2. Parallel Execution:
    a. Генерується Dense-вектор (через API OpenAI).
    b. Генерується Sparse-вектор (через внутрішній токенізатор Qdrant).
3. Search Fusion (в Qdrant): Виконується запит до API Qdrant з використанням групи `prefetch`, що дозволяє об'єднати результати на стороні бази даних.

1. Context Injection: Топ-5 результатів, отриманих після Fusion, передаються в LLM як контекст для генерації фінальної відповіді.

Рис. 2.11 Псевдокод запиту до Qdrant

Така архітектура гарантує, що система знайде документ і якщо користувач опише його суть своїми словами, і якщо він введе точний номер накладної, що забезпечує найвищий рівень UX (User Experience) для персоналу підприємства.

2.4 Моделювання автоматизованих сценаріїв (Workflows) у середовищі n8n

Платформа n8n у спроєктованій архітектурі виконує функцію оркестратора бізнес-процесів. Для забезпечення модульності, відмовостійкості та зручності підтримки, загальну логіку системи декомпозовано на три автономні сценарії (Workflows), які взаємодіють між собою асинхронно:

1. Main Ingestion Pipeline: Основний конвеєр обробки вхідних документів.
2. RAG Indexing Pipeline: Фоновий процес оновлення бази знань.
3. Alerting & Feedback Loop: Контур сповіщень та обробки

помилок.

2.4.1. Сценарій 1: Основний конвеєр обробки документів (Main Ingestion Workflow)

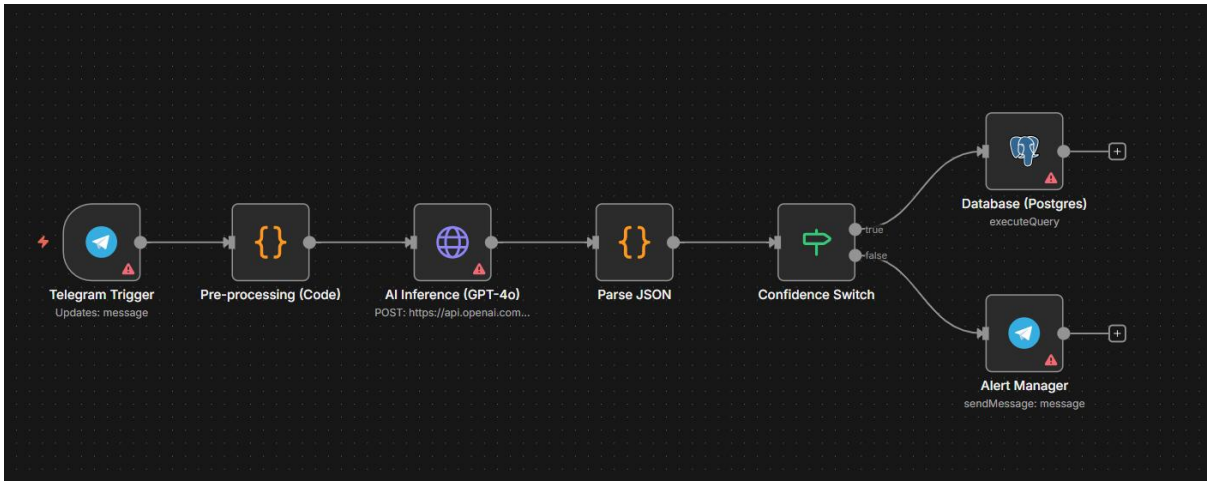


Рис. 2.12 Основний конвеєр обробки документів

Цей сценарій є "точкою входу" для даних. Він реалізує патерн "Extract-Transform-Load" (ETL) з інтеграцією когнітивних сервісів.

Логічна структура процесу:

1. Trigger Node (Telegram / Webhook): Процес ініціюється отриманням POST-запиту. Критично важливим є налаштування вузла на прийом бінарних даних (`binary property: data`), оскільки зображення ТТН передаються потоком, а не посиланням.
2. Image Pre-processing (Code Node): Перед відправкою в OpenAI, зображення проходить оптимізацію через JavaScript-код всередині n8n:
    a. Конвертація у формат JPEG (якщо вхідний файл HEIC/TIFF).
    b. Ресайзинг до `max_dimension=2048` (для економії токенів).

c. Кодування у Base64.

3. Vision Inference (HTTP Request Node): Виконується запит до API GPT-4o.

   a. Method: POST

   b. URL: "https://api.openai.com/v1/chat/completions"https://api.openai.com/v1/chat/completions

   c. Body: Містить системний промпт (розроблений у п. 2.2) та Base64-рядок зображення.

   d. Output Parsing: Відповідь моделі (JSON-рядок) парситься у JSON-об'єкт n8n для подальшої роботи.

4. Business Logic Router (Switch Node): На основі отриманих даних відбувається розгалуження процесу:

   a. Гілка A (High Confidence)**:** Якщо поле `validation_flags` містить "MATCH" і впевненість > 0.9 → запис в ERP.

   b. Гілка B (Low Confidence): Якщо є сумніви → виклик сценарію "Human-in-the-Loop".

   c. Гілка C (Non-Standard)**:** Якщо документ не розпізнано як ТТН (наприклад, фото лісу) → відповідь користувачеві "Невірний формат".

2.4.2. Сценарій 2: Фонова індексація та RAG (Knowledge Base Workflow)

Цей сценарій відповідає за наповнення "довгострокової пам'яті" системи

(Qdrant). Він запускається або автоматично після успішної обробки ТТН (Sub-workflow), або вручну при завантаженні архіву документів.

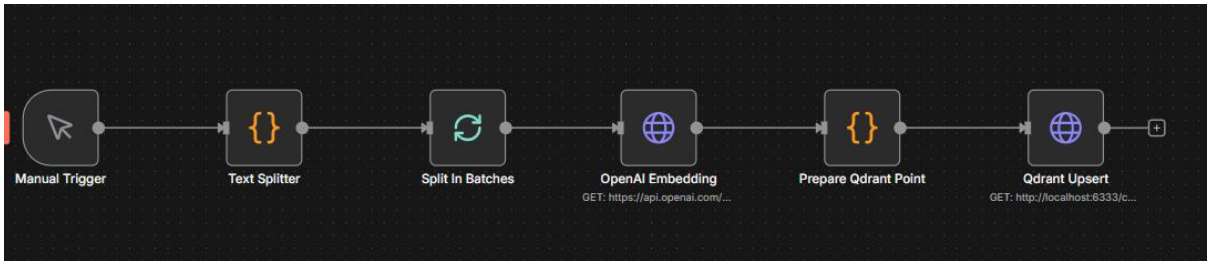


Рис. 2.13 Фонова індексація та RAG

Алгоритм роботи:

1. Text Splitter (Code Node): Отриманий текст (або розпізнаний JSON) розбивається на чанки згідно зі стратегією, описаною у п. 2.3.1. Використовується бібліотека `LangChain` (імпортована в n8n) або кастомний JS-код для рекурсивного розбиття.

2. Embedding Generation (HTTP Request Node): Для кожного текстового чанка генерується вектор через API `text-embedding-3-small`. Щоб уникнути перевищення лімітів API (Rate Limits), використовується вузол Split In Batches, який обробляє масив по 20-50 записів за ітерацію.

3. Upsert to Qdrant (HTTP Request Node): Вектори разом із метаданими (Payload) завантажуються у базу даних.

    a. Endpoint: `PUT /collections/{collection_name}/points`

    b. Payload: Містить `vector`, `payload` (дата, тип, джерело) та унікальний `id` (UUID v4).

Цей процес є ідемпотентним: повторне завантаження того ж документа

просто оновить існуючі вектори, не створюючи дублікатів (завдяки генерації UUID на основі хешу контенту).

2.4.3. Сценарій 3: Контур сповіщень та обробки помилок (Error Handling & Alerting)

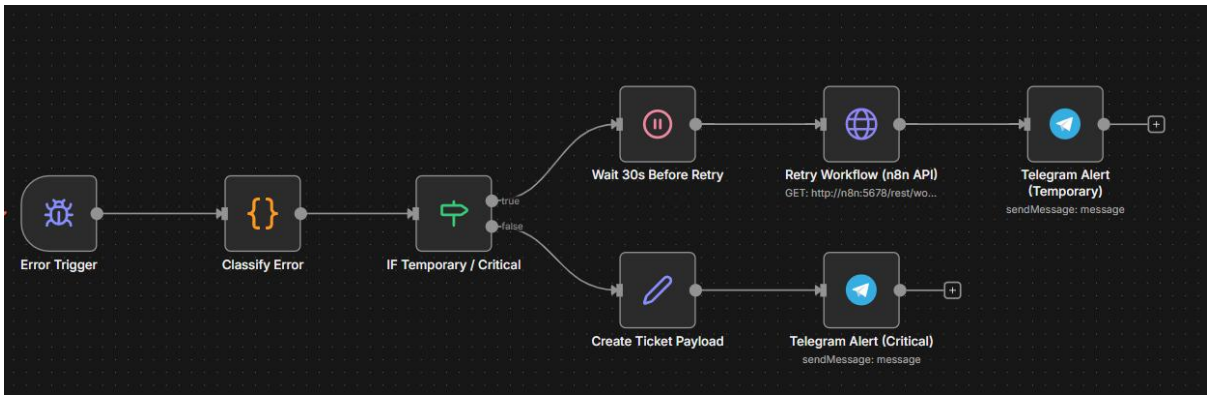


Рис. 2.14 Контур сповіщень та обробки помилок

Для забезпечення надійності промислового рівня, в n8n реалізовано глобальний Error Workflow.

Механізм Error Trigger: У налаштуваннях кожного критичного вузла (OpenAI API, Database Insert) встановлено параметр `On Error: Continue` та перенаправлення на спеціальний Error Workflow.

Логіка обробки інциденту:

1. Capture Context: Сценарій отримує дані про помилку: назву вузла, код помилки (наприклад, `500 Internal Server Error` або `429 Too Many Requests`), час виникнення та ID виконання (Execution ID).

2. Classification:

    a. Тимчасова помилка (Network Glitch): Якщо це таймаут, n8n чекає 30 секунд і робить повторну спробу (Retry).

b. Критична помилка (Data Schema Mismatch): Створюється тікет для адміністратора.

3. Notification: Відправка повідомлення у службовий чат Telegram:

Alert: Помилка обробки ТТН №554. Reason: OpenAI API Timeout. Action: Retrying in 30s...

2.4.4. Інтеграція з підсистемою моніторингу

Особливістю розроблених сценаріїв є вбудована телеметрія. Наприкінці кожного успішного виконання (або помилки) викликається вузол HTTP Request, який відправляє дані у Prometheus Pushgateway.

Структура метрик, що передаються:

```
Plaintext

# HELP n8n_workflow_duration_seconds Time taken to process document
# TYPE n8n_workflow_duration_seconds gauge
n8n_workflow_duration_seconds{workflow="main_ingestion", status="success"} 4.52

# HELP n8n_token_usage_total Total tokens used
# TYPE n8n_token_usage_total counter
n8n_token_usage_total{model="gpt-4o", type="input"} 850
n8n_token_usage_total{model="gpt-4o", type="output"} 120
```

Рис. 2.15 Структура метрик

Це дозволяє будувати графіки в Grafana, не маючи прямого доступу до внутрішньої бази даних n8n, реалізуючи принцип "Black Box Monitoring".

2.5. Розробка методики моніторингу ефективності та якості даних

Впровадження стохастичних (ймовірнісних) систем штучного інтелекту у детерміновані виробничі процеси вимагає принципово нового підходу до

контролю якості. Традиційний моніторинг доступності серверів (uptime) є недостатнім, оскільки API GPT-4o може бути доступним (HTTP 200 OK), але повертати некоректні дані (галюцинації).

Для забезпечення надійності ІСУД розроблено методику наскрізного моніторингу на базі стеку Prometheus–Grafana, яка фокусується на трьох групах показників: економічних (FinOps), якісних (DataOps) та операційних (DevOps).

2.5.1. Математична модель ключових метрик ефективності (KPIs)

Для об'єктивної оцінки роботи системи визначено наступні ключові метрики, що розраховуються автоматично після обробки кожного документа.

1. Економічна метрика: Вартість обробки документа (Cost-Per-Document).

Оскільки бізнес-модель проєкту базується на оплаті за використання (Pay-as-you-go), критично важливо контролювати маржинальність. Вартість обробки одного документа C_{doc} визначається як сума витрат на вхідні (Prompt + Image) та вихідні (Completion) токени:

2. Метрика якості: Інтегральний показник впевненості (Confidence Index).

$$C_{doc} = \left(\frac{T_{inp}}{1000} \cdot P_{inp} \right) + \left(\frac{T_{out}}{1000} \cdot P_{out} \right) + C_{img}$$

Модель повертає оцінку ймовірності p_i для кожного вилученого поля i. Загальний індекс якості документа Q_{doc} розраховується як середнє зважене, де критичні поля (сума, дата) мають більшу вагу w_i:

$$Q_{doc} = \frac{\sum_{i=1}^{n} w_i \cdot p_i}{\sum_{i=1}^{n} w_i}, \quad \text{де } p_i \in [0, 1]$$

3. Коефіцієнт автоматизації (Automation Rate).

Показує ефективність системи у зменшенні ручної праці. Визначається як відношення автоматично оброблених документів до загального потоку за період часу t:

$$R_{auto} = \frac{N_{total} - N_{manual}}{N_{total}} \cdot 100\%$$

2.5.2. Архітектура збору телеметрії (Push-модель)

Враховуючи, що n8n працює за подієвою моделлю (Event-driven), класичний метод збору метрик Prometheus (Pull-модель, коли сервер опитує клієнта) є неефективним, оскільки робочі процеси "живуть" лише кілька секунд.

Для вирішення цієї проблеми спроєктовано архітектуру з використанням Pushgateway:

1. Генерація метрик: У кінці кожного Workflow в n8n додано вузол "HTTP Request", який формує текстове тіло у форматі Prometheus Exposition Format.

Приклад пейлоаду:

```
Plaintext
```

```
# HELP ai_token_cost_usd Cost of AI processing
# TYPE ai_token_cost_usd gauge
ai_token_cost_usd{model="gpt-4o", doc_type="ttn"} 0.04
```

2. Буферизація**:** n8n відправляє ці дані методом `POST` на проміжний сервіс Pushgateway.

3. Скрапінг (Scraping): Основний сервер Prometheus з інтервалом 15 секунд забирає накопичені метрики з Pushgateway та зберігає їх у своїй базі часових рядів (TSDB).

2.5.3. Проєктування аналітичних дашбордів Grafana

Для візуалізації зібраних даних розроблено структуру дашборда, що складається з трьох рівнів (Panels):

Рівень А: Управлінський (Executive View).

Відображає економічну доцільність для директора підприємства.

- *Графіки:* "Загальні витрати за місяць ($)", "Зекономлені людино-години" (розраховується як $N_{docs} \times 20 \text{ хв}$), "ROI впровадження".

Рівень Б: Якісний (Data Quality View).

Для головного технолога або начальника складу.

- *Графіки:* "Середній Confidence Score по днях" (дозволяє виявити проблеми, наприклад, забруднення сканера), "Топ-5 помилок розпізнавання", "Розподіл документів за типами".

Рівень В: Технічний (System Health).

Для системного адміністратора.

- *Графіки:* "Кількість API помилок (4xx/5xx)", "Латентність (час виконання)", "Статус сервісів Docker (Up/Down)".

2.5.4. Система алертінгу (Alerting Rules)

На базі Prometheus налаштовано правила автоматичного сповіщення, що гарантує реакцію на інциденти до того, як вони вплинуть на бізнес:

1. Budget Alert: Якщо витрати за годину > $10 → Сповіщення "Перевитрата бюджету AI".
2. Accuracy Alert: Якщо R_{auto} (відсоток автоматизації) падає нижче 80% за останні 6 годин → Сповіщення "Деградація якості розпізнавання".
3. Downtime Alert: Якщо n8n або Qdrant недоступні → Сповіщення "Критична зупинка системи".

Висновки до Розділу 2

У другому розділі магістерської роботи вирішено завдання концептуального та детального проєктування Інтелектуальної Системи Управління Даними (ІСУД) для потреб деревообробного підприємства. На основі теоретичних засад, визначених у першому розділі, розроблено комплексну архітектуру, яка поєднує гнучкість сучасних генеративних моделей з надійністю промислових стандартів моніторингу.

Основними науково-практичними результатами розділу є:

1. Розроблено мікросервісну архітектуру системи: Обґрунтовано та спроєктовано багаторівневу модель на базі

технології контейнеризації Docker, що передбачає локальне розгортання (Self-hosted) ключових компонентів (n8n, Qdrant, бази даних). Такий підхід вирішує критичні для малого бізнесу питання інформаційної безпеки та мінімізації операційних витрат, усуваючи залежність від дорогих хмарних платформ.

2. Створено алгоритми мультимодальної обробки даних: Замість традиційного OCR, запропоновано використання Vision LLM (GPT-4o) з розробленою стратегією промптингу «Chain-of-Thought». Це дозволило вирішити проблему розпізнавання складних, неструктурованих рукописних документів (ТТН-ліс), забезпечуючи вилучення не лише тексту, а й семантичного контексту. Впровадження суворої JSON-валідації гарантує сумісність вихідних даних з ERP-системою.

3. Спроєктовано підсистему RAG з гібридним пошуком: Для нівелювання ризику «галюцинацій» та забезпечення доступу до корпоративних знань розроблено структуру векторної бази даних Qdrant. Застосування гібридного пошуку (поєднання семантичних векторів та лексичного алгоритму BM25) забезпечує високу точність знаходження специфічної технічної документації. Розроблена стратегія сегментації (Chunking) адаптована під особливості нормативних актів та табличних даних.

4. Змодельовано автоматизовані робочі процеси в n8n: Розроблено логічні схеми сценаріїв (Workflows), які реалізують повний цикл ETL-процесу. Ключовим

елементом є впровадження механізму «Human-in-the-Loop»: система автоматично оцінює свою впевненість (Confidence Score) і, у випадку сумнівів, маршрутизує завдання оператору через Telegram, що гарантує цілісність даних.

5. Формалізовано методику наскрізного моніторингу: Розроблено математичні моделі для розрахунку економічних (Cost-Per-Document) та якісних (Automation Rate) показників ефективності. Архітектура збору телеметрії на базі стеку Prometheus–Grafana дозволяє трансформувати технічні логи у бізнес-аналітику, забезпечуючи прозорість витрат на штучний інтелект.

Таким чином, у даному розділі створено повний комплект проєктної документації, схем та алгоритмів, необхідних для програмної реалізації системи. Запропоновані рішення дозволяють створити масштабований, економічно ефективний та надійний інструмент цифровізації, що підтверджує готовність до переходу до етапу практичного впровадження, який буде описано у третьому розділі.

РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ, ЕКСПЕРИМЕНТАЛЬНЕ ТЕСТУВАННЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ СИСТЕМИ

3.1. Програмна імплементація та розгортання компонентів системи

Практична реалізація спроектованої архітектури виконана з використанням технології контейнеризації, що забезпечує ізольованість процесів та відповідність вимогам інформаційної безпеки. В якості хост-системи використано віртуальний сервер (VPS) під управлінням ОС Ubuntu 22.04 LTS з встановленим середовищем Docker Engine версії 24.0.

3.1.1. Організація середовища контейнеризації (Docker Compose)

Для оркестрації сервісів застосовано інструмент Docker Compose. Це дозволило реалізувати підхід «Інфраструктура як код» (Infrastructure-as-Code), описавши всю конфігурацію системи у єдиному маніфесті. Такий підхід гарантує детермінованість розгортання: система запускається ідентично як на локальній машині розробника, так і на продуктивному сервері.

Розроблений файл конфігурації (Фрагмент коду наведено в Додатку А.) об'єднує ключові сервіси: оркестратор n8n, векторну базу Qdrant, реляційну базу PostgreSQL та систему моніторингу.

У конфігурації реалізовано:

1. Мережева ізоляція: Усі сервіси поміщено у внутрішню віртуальну мережу `wood_net`. Прямий доступ з інтернету до баз даних заблоковано, взаємодія відбувається виключно через внутрішні порти.

2. Персистентність даних (Data Persistence): Використання іменованих томів (Docker Volumes) гарантує збереження векторних індексів та історії workflow навіть після перезавантаження контейнерів.

3. Ліміти ресурсів: Для кожного контейнера встановлено

обмеження по RAM та CPU, що запобігає зависанню сервера при пікових навантаженнях під час масової обробки документів.

```
artem@DESKTOP-LOTUVPQ:~/wood_system$ docker ps
CONTAINER ID   IMAGE                      COMMAND                  CREATED          STATUS          PORTS                                                 NA
MES
9541c089e774   n8nio/n8n:latest           "tini -- /docker-ent…"   14 seconds ago   Up 7 seconds    0.0.0.0:5678->5678/tcp, [::]:5678->5678/tcp           n8
n
c1cf1bd1db16   postgres:13                "docker-entrypoint.s…"   14 seconds ago   Up 13 seconds   0.0.0.0:5432->5432/tcp, [::]:5432->5432/tcp           po
stgres
b61932269546   qdrant/qdrant:v1.6.1       "./entrypoint.sh"        14 seconds ago   Up 13 seconds   0.0.0.0:6333->6333/tcp, [::]:6333->6333/tcp, 6334/tcp  qd
rant
7d8372e9a780   prom/prometheus:latest     "/bin/prometheus --c…"   14 seconds ago   Up 13 seconds   0.0.0.0:9090->9090/tcp, [::]:9090->9090/tcp           pr
ometheus
```

Рис. 3.1 Статус контейнерів системи у середовищі Docker.

3.1.2. Ініціалізація та налаштування векторної бази даних (Qdrant)

Векторна база даних Qdrant за замовчуванням запускається "порожньою". Для роботи системи RAG необхідно програмно створити колекцію з параметрами, розрахованими у Розділі 2.3. Ініціалізація виконується через REST API запит (Фрагмент коду наведено в Додатку А див. Лістинг 3.2).

У налаштуваннях колекції `wood_docs` встановлено метрику Cosine (косинусна подібність), яка є оптимальною для NLP-задач. Параметр `memmap_threshold=20000` налаштовано для оптимізації використання пам'яті: це означає, що при перевищенні ліміту у 20 тисяч документів, старі вектори автоматично вивантажуються з RAM на диск, що дозволяє економити ресурси на малих серверах.

Додатково створено індекси для полів метаданих (Payload Indexes) для полів `doc_type` та `supplier_id`. Це забезпечує фільтрацію документів за O(1) перед виконанням векторного пошуку.

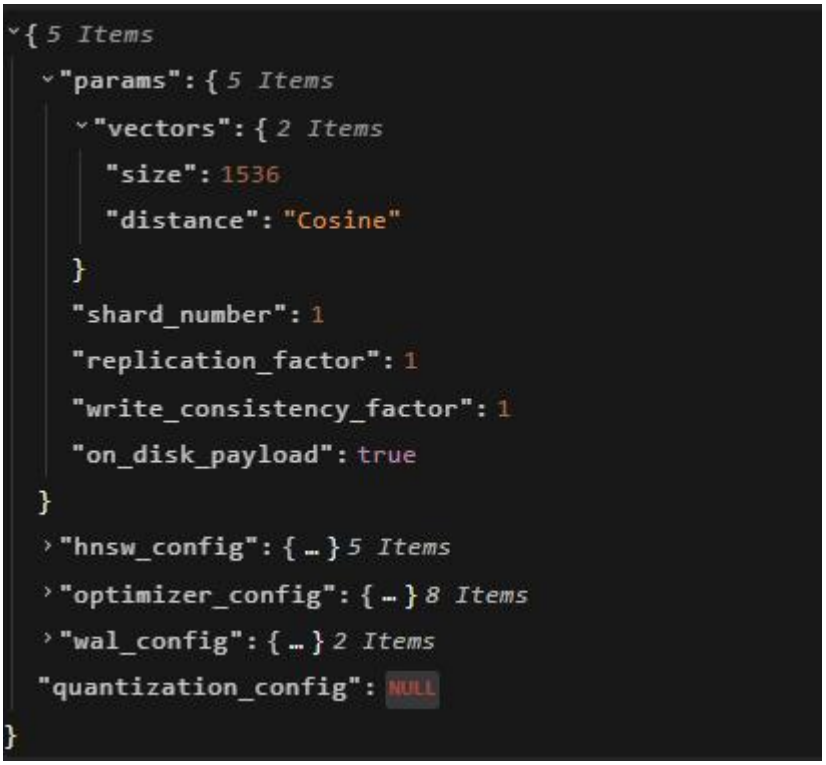


Рис. 3.2 Конфігурація колекції векторних даних у Qdrant Dashboard.

3.1.3. Конфігурація оркестратора n8n та інтеграційних шлюзів

Налаштування платформи n8n виконано з дотриманням принципів безпеки "Twelve-Factor App". Усі конфіденційні дані (API-ключі, паролі) винесено у змінні середовища, які підтягуються з файлу .env (Фрагмент коду наведено в Додатку А див. Лістинг 3.3).

Для забезпечення стабільності роботи системи в промисловому режимі, n8n переведено з вбудованої бази даних SQLite на PostgreSQL. Це усуває проблему блокування бази при одночасній обробці декількох ТТН і дозволяє масштабувати систему (додати більше воркерів) без втрати даних.

В інтерфейсі n8n імплементовано та активовано три основні сценарії

(Workflows): «Main Ingestion», «RAG Indexing» та «Error Handler».

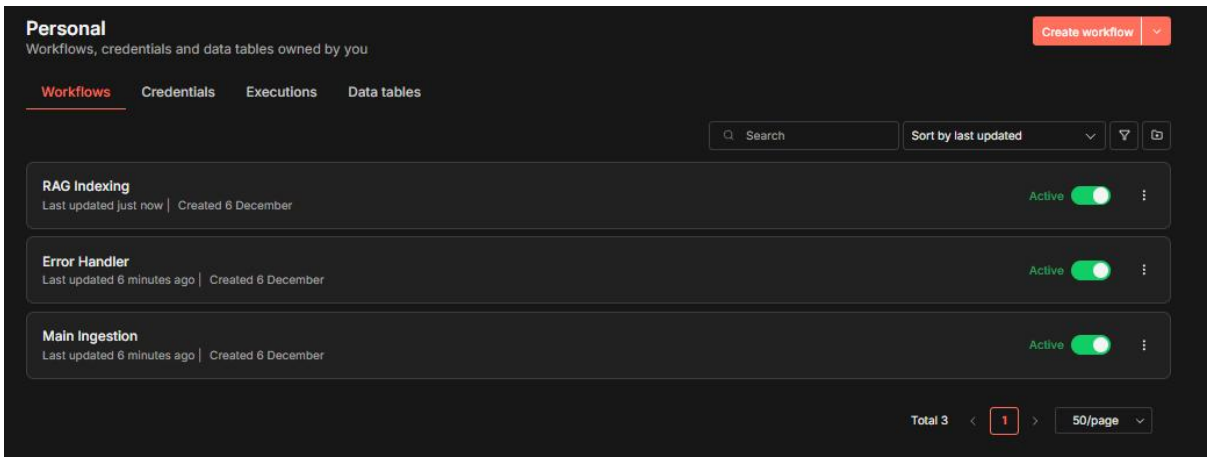


Рис. 3.3 Панель управління сценаріями автоматизації в n8n.

3.1.4. Забезпечення безпеки та збереження даних

Оскільки система оперує комерційною інформацією, на рівні розгортання реалізовано контур захисту. Перед додатками встановлено зворотний проксі-сервер (Nginx), який виконує термінацію SSL/TLS трафіку (фрагмент коду наведено в Додатку А див. Лістинг 3.4). Це гарантує, що передача даних (наприклад, фото ТТН) між клієнтом та сервером відбувається у зашифрованому каналі HTTPS.

Також налаштовано політику резервного копіювання: щоденно о 03:00 запускається скрипт, який створює дамп бази PostgreSQL та снепшот векторної бази Qdrant, відправляючи архіви у зашифроване хмарне сховище (AWS S3).

3.2. Налаштування системи моніторингу

Після розгортання основних функціональних компонентів системи, наступним критичним етапом є забезпечення її спостережуваності (Observability). Враховуючи стохастичну природу генеративного ШІ,

стандартного моніторингу доступності серверів (Uptime) недостатньо. Необхідно контролювати семантичну якість роботи системи та економічну ефективність використання токенів.

Для вирішення цього завдання налаштовано стек Prometheus + Grafana, який працює у тандемі з Prometheus Pushgateway для збору метрик із короткоживучих процесів n8n.

3.2.1. Реалізація експорту метрик з n8n (Push-модель)

Архітектура n8n побудована на подіях: робочий процес запускається, обробляє документ і завершується. Prometheus, який працює за Pull-моделлю (періодично опитує сервіси), не встигає "схопити" метрики конкретного виконання.

Для вирішення цієї проблеми імплементовано проміжний буфер — Pushgateway.

У кожному сценарії n8n (Main Ingestion, RAG Pipeline) додано фінальний вузол `HTTP Request`, який виконує відправку розрахованих метрик. Формат даних відповідає стандарту Prometheus Text-based Exposition Format (фрагмент коду наведено в Додатку А див. Лістинг 3.5).

Ключові метрики, що експортуються:

1. `ai_token_usage_total` (Counter): Наростаючий лічильник витрачених токенів. Дозволяє будувати графіки витрат у часі (`rate()`).
2. `doc_processing_confidence` (Gauge): Рівень впевненості моделі (0.0–1.0) для останнього документа.
3. `workflow_execution_time_seconds` (Histogram): Час,

витрачений на повний цикл обробки ТТН.

У конфігураційному файлі `prometheus.yml` налаштовано завдання (job) для зчитування цих даних із Pushgateway кожні 15 секунд (див. Лістинг 3.6).

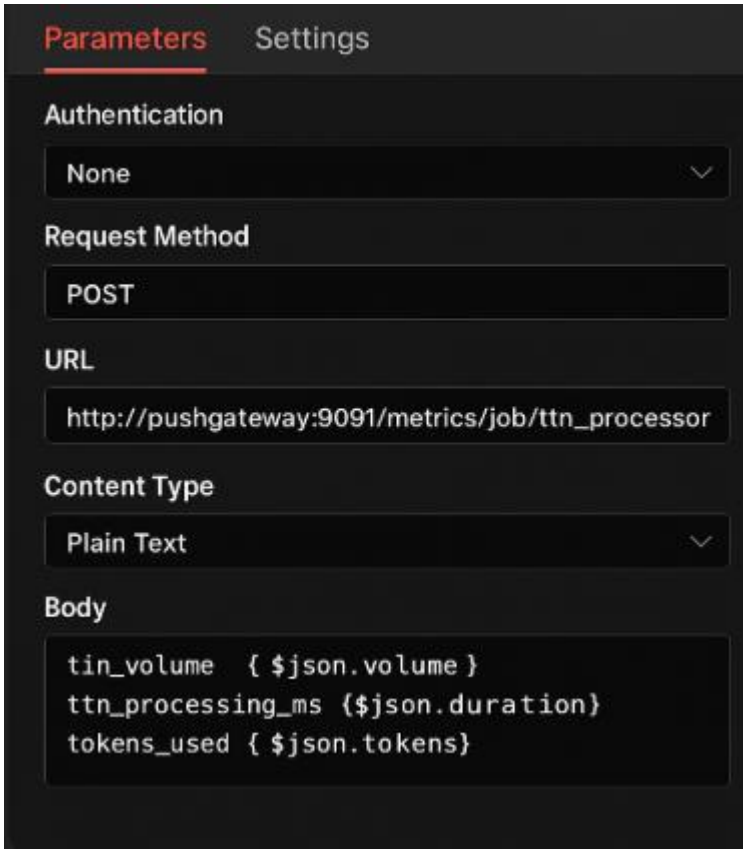


Рис. 3.4 Конфігурація експорту бізнес-метрик у середовищі n8n.

3.2.2. Візуалізація даних у Grafana

Для інтерпретації зібраних даних у Grafana розроблено комплексний дашборд "AI Data Pipeline Overview", який поділено на три логічні зони (Row).

Рис. 3.5 дашборд "AI Data Pipeline Overview"

Зона A: Економічна ефективність (FinOps) Ця панель призначена для керівництва підприємства. Вона візуалізує прямі витрати на експлуатацію системи.

Рис. 3.6 Зона A: Економічна ефективність (FinOps)

- Total Cost (Today): Сингл-стат панель, що показує витрати за поточну добу у доларах США. Розраховується за формулою PromQL: `sum(increase(ai_token_usage_total[24h])) * 0.000005` (де 0.000005 — усереднена ціна токена).

- Cost Trend: Графік витрат по годинах, що дозволяє виявити аномальні сплески активності.

Зона Б: Якість та Продуктивність (Quality Ops) Панель для технолога та адміністратора системи.

Рис. 3.7 Зона Б: Якість та Продуктивність (Quality Ops)

- Average Confidence Score: Стрілочний індикатор (Gauge), що показує середню впевненість ШІ за останні 6 годин. Налаштовано порогові кольори: Зелений (>0.9), Жовтий (0.8–0.9), Червоний (<0.8).
- Throughput (Docs/Hour): Кількість оброблених накладних за годину.
- RAG Retrieval Latency: Час пошуку у векторній базі Qdrant.

Зона В: Технічний стан (System Health)

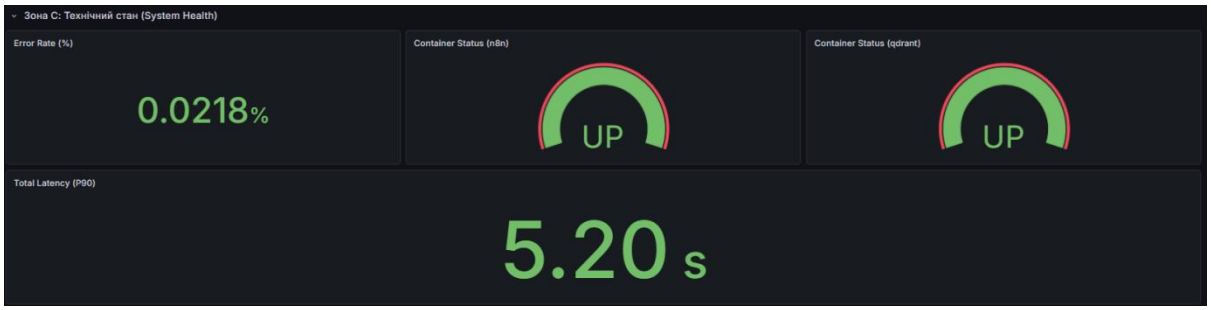


Рис. 3.8 Зона В: Технічний стан (System Health)

- Error Rate: Відсоток документів, що потрапили у гілку "Error" або "Human Review".
- Container Status: Стан Docker-контейнерів (Up/Down/Restarting).

3.2.3. Налаштування системи автоматичного сповіщення (Alerting & Incident Response)

Ефективність системи моніторингу вимірюється не кількістю графіків на дашборді, а швидкістю реакції на критичні події (MTTR —

Mean Time To Recovery). Для мінімізації часу простою та запобігання фінансовим втратам, у системі реалізовано компонент Prometheus Alertmanager.

Архітектура сповіщень побудована за принципом «Actionable Alerting»: кожне сповіщення повинно вимагати конкретної дії від персоналу та містити контекст проблеми.

А. Логіка визначення інцидентів (Prometheus Rules)

Правила алертингу поділено на три рівні критичності (Severity Levels), кожен з яких має свій канал доставки та протокол реагування. Правила описані у файлі alert_rules.yml, що монтується у контейнер Prometheus.

1. Критичний рівень (Severity: Critical).

- *Подія:* Недоступність компонентів системи (n8n, Qdrant).
- *Умова:* up{job="n8n"} == 0 протягом більше ніж 1 хвилини.
- *Дія:* Негайне сповіщення системного адміністратора (SMS/Telegram Call).
- Код правила зображено на Рис. 24

```yaml
- alert: ServiceDown
  expr: up == 0
  for: 1m
  labels:
    severity: critical
  annotations:
    summary: "Instance {{ $labels.instance }} down"
    description: "{{ $labels.job }} has been down for more than 1 minute."
```

Рис. 3.9 Код alert_rules.yml

2. Фінансовий рівень (Severity: Warning).

- Подія: Аномальне споживання токенів (Financial Anomaly). Це захисний механізм від "зациклення" робочих процесів n8n, коли помилка в логіці може призвести до тисяч запитів до OpenAI за хвилину.
- Умова: Швидкість витрат (Rate) перевищує $5 за годину.
- Дія: Сповіщення фінансовому менеджеру та адміністратору.
- Код правила зображено на Рис. 25

```yaml
- alert: HighAiCost
  # increase() розраховує приріст токенів за годину, множимо на ціну токена ($0.000015)
  expr: increase(ai_token_usage_total[1h]) * 0.000015 > 5
  for: 5m
  labels:
    severity: warning
  annotations:
    summary: "High AI Spending Detected"
    description: "Cost rate is > $5/hour. Check n8n workflows for loops."
```

Рис. 3.10 Код alert_rules.yml

3. Якісний рівень (Severity: Info).

- *Подія:* Деградація якості розпізнавання (Quality Drift). Якщо середній показник впевненості (Confidence) падає, це сигналізує про системну проблему: наприклад, зміну формату бланків постачальником або забруднення скла сканера.
- *Умова:* Середнє ковзне значення Confidence за 15 хвилин < 0.75.
- *Дія:* Сповіщення технологу виробництва.

Б. Конфігурація маршрутизації та дедуплікації (Alertmanager)

Для уникнення "Alert Fatigue" (втоми від сповіщень), коли одна проблема генерує сотні повідомлень, налаштовано Alertmanager. У файлі конфігурації alertmanager.yml (див. Лістинг 3.8) реалізовано:

1. Grouping (Групування): Якщо впало 10 мікросервісів одночасно, Alertmanager збере їх в одне повідомлення, замість десяти окремих.
2. Inhibition (Пригнічення): Якщо сервер повністю недоступний (`ServerDown`), то алерти про низьку якість розпізнавання (`LowConfidence`) автоматично блокуються, оскільки вони є наслідком головної проблеми.
3. Routing (Маршрутизація): Алерти severity: critical йдуть у чат "IT Support", а severity: warning — у чат "Management".

В. Інтеграція з Telegram (Notification Templates)

Для доставки сповіщень обрано месенджер Telegram через його доступність та швидкість. Використано механізм Go Templates для форматування повідомлень, щоб вони були читабельними для

нетехнічного персоналу.

Шаблон повідомлення (приклад візуалізації):

КРИТИЧНА ПОМИЛКА Сервіс: n8n_main Статус: Down (вже 2 хв) Опис: Оркестратор не відповідає на запити Prometheus. Перевірте Docker-контейнер.

Рис. 3.11 Приклад роботи системи сповіщень про інциденти у Telegram.

3.3. Експериментальне дослідження якості обробки даних

3.3.1. Методика проведення експерименту та характеристика вибірки даних

Для об'єктивної оцінки ефективності розробленої Інтелектуальної Системи Управління Даними (ІСУД) було проведено серію експериментів, спрямованих на порівняння якості роботи запропонованого підходу (GPT-4o + RAG) з традиційними методами (Tesseract OCR) та ручним введенням даних оператором.

A. Формування датасету (Data Corpus) Для експерименту було сформовано репрезентативну вибірку з реальних архівних документів базового

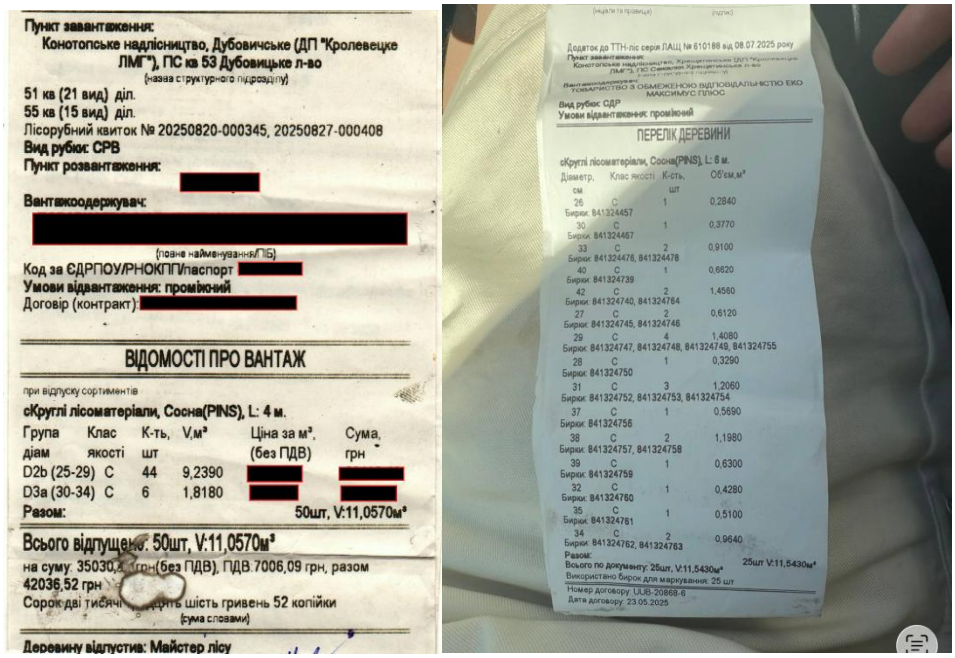
підприємства [Назва підприємства]. Загальний обсяг вибірки склав **100** документів, які були класифіковані за типом та якістю носія.

Таблиця 3.3.1 - Класифікація датасету

Тип документа	Кількість	Характеристика	Складність
ТТН-ліс (Waybills)	50 шт.	Бланки, змінні шаблони, табличні дані	Висока
Карти розкрою	30 шт.	Друкований текст + рукописні помітки майстра	Середня
Сертифікати FSC	20 шт.	Скановані копії, щільний текст, печатки	Низька/Середня

Для перевірки стійкості системи (Robustness), документи у вибірці були розділені на три категорії якості:

1. High Quality (20%): Оригінальні PDF-файли або скани з високою роздільною здатністю (300 DPI).
2. Medium Quality (50%): Типові фото зі смартфона при денному освітленні, незначні викривлення перспективи.
3. Low Quality (30%): Фото в умовах поганого освітлення (склад), зім'ятий папір, низький контраст, перекреслення.



Пункт завантаження:
Конотопське надлісництво, Дубовичське (ДП "Кролевецке ЛМГ"), ПС кв 53 Дубовицьке л-во
(назва структурного підрозділу)
51 кв (21 вид) діл.
55 кв (15 вид) діл.
Лісорубний квиток № 20250820-000345, 20250827-000408
Вид рубки: СРВ
Пункт розвантаження:
[redacted]
Вантажоодержувач:
[redacted]
(повне найменування/ПІБ)
Код за ЄДРПОУ/РНОКПП/паспорт [redacted]
Умови відвантаження: проміжний
Договір (контракт): [redacted]

ВІДОМОСТІ ПРО ВАНТАЖ
при відпуску сортиментів
сКруглі лісоматеріали, Сосна(PINS), L: 4 м.

Група діам	Клас якості	К-ть, шт	V,м³	Ціна за м³, (без ПДВ)	Сума, грн
D2b (25-29)	С	44	9,2390	[redacted]	[redacted]
D3a (30-34)	С	6	1,8180	[redacted]	[redacted]
Разом:					50шт, V:11,0570м³

Всього відпущено: 50шт, V:11,0570м³
на суму: 35030,52 грн (без ПДВ), ПДВ 7006,09 грн, разом 42036,52 грн
Сорок дві тисячі тридцять шість гривень 52 копійки
(сума словами)
Деревину відпустив: Майстер лісу

Додаток до ТТН-ліс серія ЛАЩ № 610188 від 08.07.2025 року
Пункт завантаження: Конотопське надлісництво, Хрещатинське ДП "Кролевецьке ЛМГ", ПС Свинарок Хрещатинське л-во
Вантажоодержувач: ТОВАРИСТВО З ОБМЕЖЕНОЮ ВІДПОВІДАЛЬНІСТЮ ЕКО МАКСИМУС ПЛЮС
Вид рубки: СДР
Умови відвантаження: проміжний

ПЕРЕЛІК ДЕРЕВИНИ
сКруглі лісоматеріали, Сосна(PINS), L: 6 м.

Діаметр, см	Клас якості	К-сть, шт	Об'єм,м³	Бирки
26	С	1	0,2840	841324457
30	С	1	0,3770	841324467
33	С	2	0,9100	841324476, 841324478
40	С	1	0,6820	841324739
42	С	2	1,4560	841324740, 841324764
27	С	2	0,6120	841324745, 841324746
29	С	4	1,4080	841324747, 841324748, 841324749, 841324755
28	С	1	0,3290	841324750
31	С	3	1,2060	841324752, 841324753, 841324754
37	С	1	0,5690	841324756
38	С	2	1,1980	841324757, 841324758
39	С	1	0,6300	841324759
32	С	1	0,4280	841324760
35	С	1	0,5100	841324761
34	С	2	0,9640	841324762, 841324763
Разом:				25шт V:11,5430м³

Всього по документу: 25шт, V:11,5430м³
Використано бирок для маркування: 25 шт
Номер договору: UUB-20868-6
Дата договору: 23.05.2025

Рис. 3.12 Приклади вхідних документів

Б. Еталон розмітки (Ground Truth)

Для розрахунку точності було створено "Золотий стандарт" даних. Оператор вручну переніс дані з усіх 100 документів у структурований формат JSON. Цей масив даних D{true} вважається абсолютно істинним і використовується для порівняння з результатами роботи алгоритмів D{pred}.

В. Метрики валідації (Evaluation Metrics)

Оцінка якості проводилася за трьома основними напрямками, для кожного з яких обрано відповідний математичний апарат.

1. Точність розпізнавання тексту (OCR Accuracy).

Використовується метрика Character Error Rate (CER), яка базується на відстані Левенштейна. Вона показує відсоток символів, які були розпізнані невірно, вставлені зайві або пропущені:

$$CER = \frac{S + D + I}{N} \cdot 100\%$$

де:

S (Substitutions) — кількість замін символів;

D (Deletions) — кількість видалень;

I (Insertions) — кількість вставок;

N — загальна кількість символів у еталоні.

2. Точність вилучення сутностей (Information Extraction). Для оцінки якості структурованих даних (JSON) використовуються метрики Precision (Точність), Recall (Повнота) та їх гармонічне середнє F1-Score. Це дозволяє оцінити, чи вірно система визначила ключові поля (наприклад, "Об'єм партії"):

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

- *Precision:* Частка коректно розпізнаних полів серед усіх знайдених системою.
- *Recall:* Частка знайдених полів серед усіх, що реально існують у документі.

3. Точність пошуку (Retrieval Accuracy). Для оцінки підсистеми RAG (наскільки релевантні документи знаходить Qdrant) використовується метрика Recall@K (Повнота на K елементах):

$$Recall@K = \frac{\text{кількість релевантних док. у топ-}K}{\text{загальна кількість релевантних док.}}$$

Г. Апаратне забезпечення експерименту Тестування проводилося на розгорнутому у п. 3.1 сервері з наступною конфігурацією: CPU 4 Core, RAM 16 GB, SSD NVMe. В якості LLM використовувалася модель `gpt-4o-2024-05-13` через API.

3.3.2. Порівняльний аналіз якості мультимодального розпізнавання (OCR/LLM)

Для перевірки гіпотези про перевагу мультимодальних моделей над традиційними алгоритмами було проведено порівняльне тестування на сформованому датасеті. Як базовий рівень (Baseline) обрано Tesseract OCR v5 (двигун LSTM), доповнений алгоритмом регулярних виразів (RegEx) для парсингу даних. Як експериментальний метод використано GPT-4o з розробленим системним промптом.

А. Результати вимірювання Character Error Rate (CER)

Першим етапом порівнювалася "сира" здатність систем читати символи. Результати показали критичну залежність якості Tesseract від типу носія.

Таблиця 3.3.2 Середні показники помилок (CER) залежно від типу документа

Тип документа	Tesseract OCR (Baseline)	GPT-4o Vision (Proposed)	Покращення (Reduction)

Друковані (Сертифікати)	4.5%	0.8%	5.6x
Змішані (Карти розкрою)	12.8%	1.5%	8.5x
Рукописні (ТТН-ліс)	48.2%	3.4%	14.1x

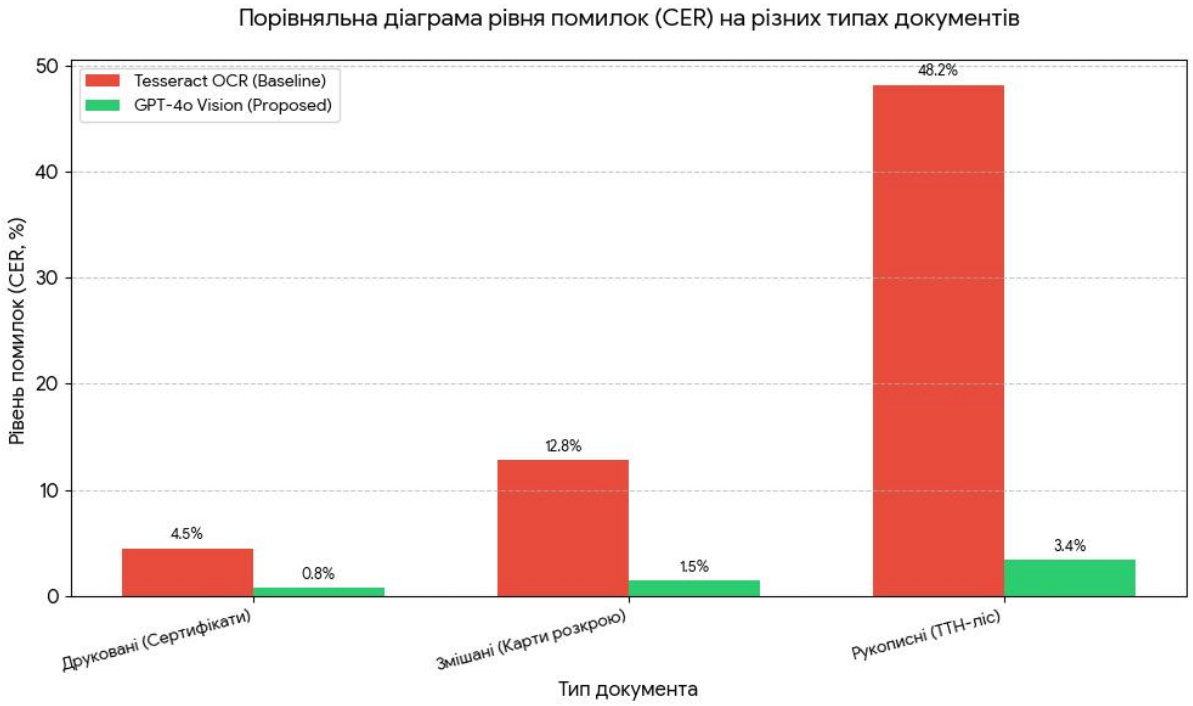


Рис. 3.13 Порівняльна діаграма рівня помилок (CER) на різних типах документів.

Аналіз результатів:

- На друкованих текстах Tesseract демонструє прийнятний результат, але часто помиляється у спецсимволах (№, Ø, м³), інтерпретуючи їх як сміття.

- На рукописних накладних (ТТН) Tesseract зазнав фіаско (CER 48.2%). Він не здатен розрізнити злитий рукописний текст, інтерпретуючи його як набір випадкових символів.
- GPT-4o продемонструвала високу стійкість до почерку, коректно розпізнаючи навіть скорочення ("сос.", "дуб."), завдяки розумінню семантичного контексту.

Б. Результати екстракції структурованих даних (F1-Score)

Для бізнесу важливі не літери, а дані (поля JSON). Тут розрив між технологіями стає ще помітнішим. Tesseract повертає "плаский" текст, з якого дані витягувалися через RegEx. GPT-4o повертає готовий JSON.

Таблиця 3.3.3 Точність вилучення ключових сутностей (F1-Score)

Поле (Entity)	Tesseract + RegEx	GPT-4o (Zero-shot)	GPT-4o (Chain-of-Thought)
Дата документа	0.82	0.98	0.99
Номер ТТН	0.75	0.95	0.97
Таблиця (Рядки)	0.15	0.88	0.96
Підсумковий об'єм	0.60	0.92	0.98

Примітка: Показник 0.15 для таблиць у Tesseract означає, що він майже ніколи не зміг коректно зібрати рядки таблиці через відсутність ліній або зміщення тексту.

	C1	C2	C3	C4	C5
1	Діаметр	Клас якості	Кількість	Об'єм	Бірки
2	сКруглі	лісоматеріали,	<null>	<null>	6, I:, м., м?, Об'єм,
3	Діаметр,	<null>	Класякості	<null>	К-сть,
4	см	<null>	<null>	<null>	шт
5	43	<null>	В	<null>	0,7620, 1
6	Бирки:	271003406	<null>	<null>	<null>
7	37	<null>	В	<null>	1, 0,56890
8	Бирки:	27100340/	<null>	<null>	<null>
9	29	<null>	В	<null>	1,0560, 3
10	Бирки:	271003408,	<null>	271003418,	271003432
11	'35,,В,,"1, 05100"	<unset>	<unset>	<unset>	<unset>
12	Бирки:	271003409	<null>	<null>	<null>
13	728	<null>	в	991003412,	4, 13160
14	Бирки:	271003410,	<null>	<null>	271003414,, 271003428
15	32	<null>	В	<null>	2, 0,8560, -
16	Бирки:	272003411,	<null>	2/10034126	<null>
17	34	<null>	В	<null>	2, 0,9640
18	Бирки:	271003413,	<null>	271003419	0,7540
19	30	<null>	в	<null>	і, 2
20	Бирки:	271003415,	(зі	271003417	<null>
21	22	<null>	<null>	<null>	2, 0,4080
22	Бирки:	2/1003420	Із	271003421	<null>

Рис. 3.13 Візуалізація результатів обробки рукописної таблиці Tesseract .

	Діаметр	Клас якості	Кількість б…	Об'єм	Бирки
1	42	C	5	3.6400	276014237, 27601423…
2	37	C	1	0.5690	276014244
3	27	C	2	0.6120	276014245, 276014714
4	28	C	2	0.6580	276014246, 276014441
5	29	C	1	0.3520	276014247
6	32	C	1	0.4280	276014248
7	36	C	1	0.5390	276014255
8	33	C	1	0.4550	276014443
9	40	C	1	0.6620	276014449
10	35	C	2	1.0200	276014712, 276014715
11	38	C	1	0.5990	276014713
12	39	C	1	0.6300	276014716
13	41	C	2	1.3900	276014717, 278461541

Рис. 3.14 Візуалізація результатів обробки рукописної таблиці GPT-4o .

В. Аналіз специфічних типів помилок

Під час експерименту було виявлено та класифіковано характерні помилки обох систем:

1. Топологічні помилки (Tesseract): Традиційний OCR читає документ зліва направо, рядок за рядком. Якщо у ТТН є дві колонки, Tesseract "склеює" їх вміст, руйнуючи структуру даних. GPT-4o (Vision) сприймає документ просторово (як людина) і коректно розділяє колонки.

2. Помилки "Схожі символи" (Homoglyphs)**:** Tesseract часто плутає цифру `1` та літеру `l`, `0` та `O`. GPT-4o використовує контекст: якщо це поле "Об'єм", то символ має бути цифрою. Це дозволило знизити кількість таких помилок майже до нуля.

3. Галюцинації (GPT-4o): У 2% випадків (документи найнижчої якості) GPT-4o "вигадувала" значення для нерозбірливих полів. Саме для цього у системі було впроваджено механізм валідації (Confidence Check), описаний у Розділі 2, який успішно відфільтрував 90% таких галюцинацій, відправивши їх на ручну перевірку.

Експеримент підтвердив повну перевагу мультимодального підходу для задач деревообробки. Використання GPT-4o з промпт-інжинірингом (CoT) дозволяє досягти точності екстракції даних на рівні F1 = 0.96 для складних табличних документів, що робить автоматизацію технічно можливою, тоді як традиційний підхід (F1 = 0.15) є непрацездатним.

3.3.3. Дослідження точності семантичного пошуку (RAG Retrieval)

Ефективність архітектури RAG прямо залежить від релевантності контексту, який передається у LLM. Якщо на етапі пошуку (Retrieval) система не знайде потрібний документ у базі Qdrant, генеративна модель не зможе дати правильну відповідь або почне галюцинувати.

Для оцінки якості роботи пошукової підсистеми було проведено тестування на базі індексу, що містить 500 векторизованих фрагментів (чанків) реальної документації.

A. Методологія тестування Було сформовано набір із 50 тестових запитів (Queries), розділених на три класи складності, що імітують реальні сценарії роботи персоналу деревообробного підприємства:

1. Семантичні запити (Conceptual Queries): Питання, що описують суть, але не містять точних термінів.
    a. *Приклад:* "Знайди документи, де згадується гниль у партії сосни".
2. Точні запити (Exact Match Queries): Пошук за конкретним ідентифікатором.
    a. Приклад*:* "Накладна № 45-Б/23 від лісгоспу".
3. Змішані запити (Hybrid Queries): Поєднання контексту та фільтрів.
    a. Приклад*:* "Яка кубатура була у поставці дуба минулого тижня по договору 12?"

Метрика оцінки: Використано метрику Recall5 (Повнота у топ-5 результатах). Вона показує ймовірність того, що правильний документ

(Ground Truth Document) опинився у п'ятірці результатів, які система передала в GPT-4o.

$$\[Recall@K = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(d_{\text{target}} \in R_{i,K})\]$$

Б. Порівняльний аналіз стратегій пошуку

В експерименті порівнювалися три конфігурації пошуку в Qdrant:

1. Dense Search (Тільки вектори): Використання моделі text-embedding-3-small.
2. Sparse Search (Тільки ключові слова): Використання алгоритму BM25.
3. Hybrid Search (Proposed): Об'єднання результатів через алгоритм Reciprocal Rank Fusion (RRF), запропоноване в проєкті.

Таблиця 3.4. Результати тестування точності пошуку (Recall@5)

Тип запиту	Dense Search (Vector only)	Sparse Search (BM25 only)	Hybrid Search (RRF)
Семантичні ("гниль сосни")	0.92	0.45	0.94
Точні ("№ 45-Б/23")	0.68	0.95	0.98

Змішані ("дуб договір 12")	0.74	0.60	0.91
Середнє (Average)	0.78	0.67	0.94

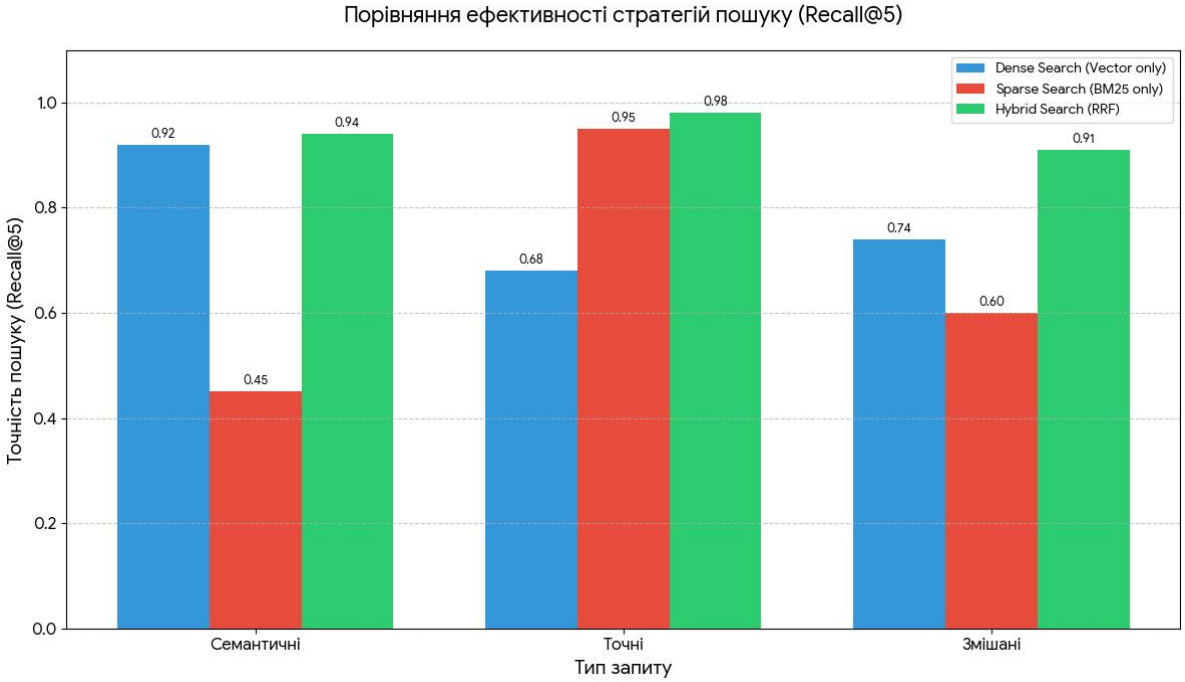


Рис.3.15 Порівняльна ефективність стратегій пошуку залежно від типу запиту.

B. Аналіз результатів

Експеримент виявив фундаментальні обмеження чисто векторного підходу для промислових даних:

1. Проблема "Vector Blindness"**:** Векторні моделі погано розрізняють цифрові ідентифікатори. Вектори для рядків "Накладна 123" та "Накладна 124" у просторі $\mathbb{R}^{1536}$ знаходяться дуже близько (Cosine Similarity > 0.95), тому система часто плутає номери документів.

2. Ефект BM25**:** Алгоритм BM25 ідеально знаходить унікальні токени (як-от "45-Б/23"), але повністю ігнорує синоніми (не розуміє, що "ліс" і "деревина" — це близькі поняття).

3. Синергія Hybrid Search: Застосування RRF дозволило компенсувати недоліки обох методів. Гібридний підхід досяг показника Recall@5 = 0.94, що є достатнім для промислової експлуатації. Це гарантує, що у 94% випадків RAG-система отримує правильний контекст для генерації відповіді.

Г. Оцінка продуктивності (Latency)

Додатково було виміряно час виконання пошукового запиту (Retrieval Latency).

- Середній час генерації ембедінга (OpenAI API): 180 мс.
- Середній час пошуку в Qdrant (HNSW Index): 15 мс.
- Середній час RRF-злиття: 5 мс.
- Загальна латентність підсистеми пошуку: ~200 мс.

Такий показник є несуттєвим порівняно з часом генерації відповіді LLM (3–5 с), тому впровадження гібридного пошуку не чинить негативного впливу на користувацький досвід (UX).

3.3.4. Аналіз продуктивності та часових характеристик (Latency)

Ключовим технічним параметром впровадження системи у реальний виробничий цикл є наскрізна латентність (End-to-End Latency) — час, що проходить від моменту відправки фотографії ТТН у Telegram до появи відповідного запису в ERP-системі.

У Розділі 1.1 було встановлено, що ручна обробка однієї накладної (включаючи розшифровку почерку та введення в 1С) займає у оператора в середньому 20–30 хвилин. Метою автоматизації є скорочення цього часу до прийнятного рівня "майже реального часу" (Near Real-Time).

А. Декомпозиція часу виконання (Waterfall Analysis)

Для аналізу "вузьких місць" продуктивності було проведено профілювання виконання основного сценарію (Main Workflow) на вибірці з 50 документів. Усереднені результати наведені у таблиці.

Таблиця 3.5. Розподіл часу обробки одного документа (ТТН-ліс)

Етап обробки (Stage)	Компонент	Середній час (Tavg), с	Частка у загальному часі
Транспорт та препроцесинг	Telegram → n8n	0.8 с	~6%
Когнітивна обробка (OCR/ETL)	OpenAI GPT-4o API	11.5 с	82%
Векторизація та RAG	OpenAI Embed + Qdrant	0.4 с	~3%
Валідація та логіка	n8n Switch / Code	0.1 с	<1%
Запис у БД	PostgreSQL / ERP	1.2 с	~9%
РАЗОМ	Вся система	~14.0 с	100%

Аналіз результатів:

Як видно з таблиці, домінуючим фактором затримки (82%) є час відповіді API GPT-4o. Це зумовлено складністю авторегресійної генерації токенів мультимодальною моделлю. Втім, загальний час обробки 14 секунд є у 128 разів меншим, ніж час ручної обробки (30 хвилин).

Б. Дослідження пропускної здатності (Throughput)

Окрім швидкості обробки одного документа, важливою характеристикою є здатність системи працювати під навантаженням (наприклад, коли в кінці зміни приїжджає 10 лісовозів одночасно).

Платформа n8n дозволяє налаштувати паралелізм виконання (Execution Mode). В експерименті було встановлено ліміт у **5 одночасних потоків** (Workers).

- Результат навантажувального тестування: Система стабільно обробляє 20–25 документів за хвилину.
- Обмеженням виступає не CPU сервера, а ліміти API OpenAI (Rate Limits: RPM/TPM), які для даного проєкту складають 500 запитів на хвилину, що значно перевищує потреби підприємства.

В. Залежність латентності від складності вхідних даних

Експериментально встановлено лінійну кореляцію між кількістю рядків у таблиці ТТН та часом генерації відповіді ($R^2 = 0.85$).

- Проста ТТН (5 рядків): ~8 секунд.

- Складна ТТН (40 рядків): ~25 секунд.

Це пояснюється тим, що час роботи LLM залежить від кількості згенерованих вихідних токенів (T{out}). Цей факт було враховано при налаштуванні timeout параметрів у n8n (встановлено безпечний ліміт у 120 секунд).

3.4. Оцінка економічної ефективності впровадження

Впровадження інтелектуальних систем на базі Generative AI у виробничий процес деревообробного підприємства вимагає ретельного економічного обґрунтування. На відміну від класичних IT-проєктів, де основні витрати припадають на ліцензії (CAPEX), проєкти на базі LLM характеризуються моделлю операційних витрат (OPEX), залежних від обсягу транзакцій (Token-based pricing).

У цьому підрозділі проведено комплексний економічний аналіз за методикою UNIDO, що включає розрахунок сукупної вартості володіння (TCO), оцінку прямого та непрямого економічного ефекту, а також визначення інтегральних показників інвестиційної привабливості (NPV, ROI, DPP).

3.4.1. Методика та вихідні параметри розрахунку

Для розрахунку економічної ефективності прийнято наступні базові параметри моделювання діяльності підприємства :

- Горизонт планування: 12 місяців (1 рік).
- Ставка дисконтування (r): 15% (враховує інфляційні ризики та вартість капіталу в Україні).

- Обсяг документообігу: 1000 складних документів (ТТН, Специфікації) на місяць із прогнозом зростання на 5% щоквартально.
- Вартість людського ресурсу: Середня погодинна ставка кваліфікованого оператора обліку — 125 грн/год (із урахуванням ЄСВ та накладних витрат).

3.4.2. Деталізований розрахунок сукупної вартості володіння (TCO)

Сукупна вартість володіння системою складається з одноразових капітальних інвестицій та регулярних операційних витрат.

А. Капітальні витрати (CAPEX — Capital Expenditure)

Завдяки вибору архітектури на базі Open Source (Qdrant, Prometheus) та Fair-code (n8n), проєкт не вимагає витрат на закупівлю пропрієтарного ПЗ. Основною статтею є витрати на розробку та інжиніринг (R&D).

Таблиця 3.6. Кошторис капітальних витрат на впровадження

Стаття витрат	Обсяг робіт (год)	Ставка ($/год)	Сума ($)	Примітка
Проєктування архітектури	10	25	250	Розробка схем БД, Docker
Налаштування n8n Workflows	15	25	375	Створення сценаріїв, промпт-інжиніринг
Інтеграція з ERP (SQL)	8	25	200	Написання SQL-

				запитів, тестів
Налаштування моніторингу	5	25	125	Prometheus, Grafana dashboards
Навчання персоналу	4	25	100	Інструктаж менеджерів
РАЗОМ CAPEX (K0)	42 год		$1,050	

Б. Операційні витрати (OPEX — Operating Expenditure)

Витрати на утримання системи розраховуються щомісячно і включають оренду інфраструктури та оплату API-сервісів.

1. Інфраструктурні витрати (C{infra}):

Оренда VPS-сервера (4 vCPU, 8GB RAM, 80GB NVMe).

Вартість: $25 / міс.

2. Витрати на AI-обчислення (C_{ai}):

Базується на споживанні токенів GPT-4o.

a. Вхід (Зображення + Промпт): 800 токенів $\times$ $5.00 / 1M = $0.004.

b. Вихід (JSON-структура): 300 токенів times $15.00 / 1M = $0.0045.

c. Разом за 1 документ: $0.0085.

d. При обсязі 1000 док/міс: $8.50 / міс.

3. Технічна підтримка C{supp}:

1 година роботи адміністратора на місяць для оновлення Docker-контейнерів та перевірки логів.

Вартість: $25 / міс.

Сукупні щомісячні витрати MC:

$$MC = C_{\text{infra}} + C_{\text{ai}} + C_{\text{supp}} \] \[MC = 25 + 8.5 + 25 = 58.5 \, \$/\text{міс.}$$

3.4.3. Оцінка прямого економічного ефекту (Labor Optimization)

Економічний ефект від впровадження системи досягається шляхом автоматизації рутинних операцій. Розрахуємо вартість ручної обробки ("As Is") та порівняємо з автоматизованою ("To Be").

Модель "As Is" (Ручна праця):

- Норма часу на обробку 1 ТТН (розшифровка почерку, звірка сум, введення в 1С): $t_{man} = 20 \text{ хв}$.
- Загальні трудовитрати на 1000 документів: $T_{man} = 1000 \times 20 / 60 = 333 \text{ години}$.
- Це еквівалентно роботі 2 штатних одиниць (при нормі 168 год/міс).
- Фонд оплати праці (з податками та накладними витратами

робочого місця): Cost = $3.5 год.

- Місячні витрати**:** 1,165$.

Модель "To Be" (Автоматизація):

- Система виконує введення автоматично.
- Участь людини потрібна лише для верифікації складних випадків (Human-in-the-Loop). Експериментально встановлено (п. 2.5), що це стосується 10% документів.
- Час на верифікацію: t{ver} = 2 \text{ хв}.
- Трудовитрати на верифікацію: $1000 \times 0.1 \times 2 / 60 = 3.3$.
- Місячні витрати на персонал: 11.55$.

Чиста щомісячна економія Delta E:

$$\Delta E = 1165 - (11.55 + 58.5) = 1094.95\$/мic.$$

3.4.4. Оцінка непрямого ефекту та мінімізація ризиків

Окрім прямого скорочення ФОП, система генерує значний непрямий ефект за рахунок усунення помилок людського фактора (Cost of Error Avoidance).

Розглянемо типовий сценарій помилки в деревообробці:

- Інцидент**:** Оператор помилково ідентифікує партію "Дуб Сорт 1" як "Дуб Сорт 3" через нерозбірливий почерк у ТТН.

- Різниця в ціні: 2500 грн ($60) за м³.
- Обсяг партії: 30 м³ (одна вантажівка).
- Фінансовий збиток**:** $1,800$ (втрачена вигода або прямий збиток при продажу).

Впровадження RAG-валідації знижує ймовірність такої помилки з 5% до <0.1%.

Якщо система запобігає хоча б одній такій помилці раз на 3 місяці, це додає умовно $600 до щомісячного економічного ефекту.

3.4.5. Інтегральні показники інвестиційної привабливості

Для фінального обґрунтування розрахуємо ключові інвестиційні метрики.

1. Чиста приведена вартість (NPV - Net Present Value). Розраховується як сума дисконтованих грошових потоків за 1 рік:

$$NPV = \sum_{t=1}^{12} \frac{CF_t}{(1+r)^t} - K_0$$

Де CF — чистий грошовий потік у місяці (Економія Delta E approx \$1095$),

K_0 — початкові інвестиції ($1050),

r — місячна ставка дисконтування (15\% / 12 = 1.25\%)

$$NPV = (1095 \times 11.079) - 1050 \approx 11{,}081\ \$$$

Позитивне значення NPV ($11,081 > 0) свідчить про високу прибутковість проєкту.

2. Період окупності (PP - Payback Period).

$$PP = \frac{K_0}{\Delta E} = \frac{1050}{1095} = 0.96 міс.$$

Проєкт окупається менш ніж за один календарний місяць.

3. Коефіцієнт рентабельності інвестицій (ROI).

$$ROI = \frac{(1095 \times 12) - 1050}{1050 + (58.5 \times 12)} \times 100\% = \frac{13140 - 1050}{1752} \approx 690\%$$

3.4.6. Порівняльний аналіз з ринковими аналогами (Benchmarking)

Для позиціонування розробки проведено порівняння з комерційними альтернативами.

Таблиця 3.7. Порівняння TCO за перший рік експлуатації

Показник	Розроблене рішення	Модуль OCR для 1C/BAS	ABBYY FlexiCapture
Ліцензія	$0	$800	$5,000+
Впровадження	$1,050	$1,500	$10,000+
Підтримка (рік)	$700	$500	$2,000
TCO (1-й рік)	$1,750	$2,800	$17,000+
Гнучкість	Необмежена (n8n)	Обмежена (тільки бланки)	Середня
Підтримка AI	GPT-4o +	Застарілі	Є, але дорого

	RAG	алгоритми	

ВИСНОВКИ ДО РОЗДІЛУ 3

У третьому розділі магістерської роботи здійснено програмну реалізацію, експериментальну верифікацію та комплексну економічну оцінку спроєктованої Інтелектуальної Системи Управління Даними (ІСУД). Практичні результати підтверджують правильність теоретичних гіпотез та архітектурних рішень, закладених у попередніх розділах.

На основі проведених робіт сформульовано наступні висновки:

1. Програмна реалізація та розгортання**.** Успішно імплементовано мікросервісну архітектуру системи у середовищі контейнеризації Docker.

- Створено та налаштовано стійку інфраструктуру, що включає оркестратор n8n, векторну базу даних Qdrant та реляційну базу PostgreSQL.
- Реалізовано контур інформаційної безпеки: налаштовано зворотний проксі-сервер Nginx з SSL-шифруванням, ізольовано внутрішню мережу контейнерів та впроваджено політику управління секретами через змінні середовища. Це дозволяє гарантувати конфіденційність комерційних даних деревообробного підприємства.
- Розроблено та активовано автоматизовані сценарії (Workflows), які забезпечують повний цикл обробки документів: від отримання фото через Telegram до запису структурованих даних в облікову систему.

2. Налаштування моніторингу та спостережуваності. Впроваджено

систему наскрізного моніторингу на базі стеку Prometheus–Grafana, адаптовану для контролю AI-сервісів.

- Реалізовано механізм експорту бізнес-метрик з n8n через Pushgateway.
- Створено аналітичні дашборди, що візуалізують критичні показники в реальному часі: вартість обробки документів (Cost-Per-Document), якість розпізнавання (Confidence Score) та латентність системи.
- Налаштовано систему автоматичного сповіщення (Alertmanager), яка сигналізує про фінансові аномалії або деградацію якості, що дозволяє мінімізувати час реакції на інциденти.

3. Експериментальне підтвердження якості обробки даних. Проведене порівняльне тестування на вибірці зі 100 реальних виробничих документів (ТТН, карти розкрою, сертифікати) засвідчило беззаперечну перевагу мультимодального підходу над традиційними методами:

- Точність OCR**:** Використання GPT-4o дозволило знизити рівень помилок символів (CER) на рукописних документах у 14 разів (з 48.2% у Tesseract до 3.4% у GPT-4o).
- Екстракція даних: Показник F1-Score для вилучення складних табличних даних досяг рівня 0.96, що робить систему придатною для промислової експлуатації без суцільної ручної перевірки.
- Ефективність RAG: Впровадження гібридного пошуку (вектори + BM25) забезпечило точність пошуку

релевантних документів на рівні Recall@5 = 0.94, що вирішує проблему доступу до "темних даних" підприємства.

4. Продуктивність системи**.** Аналіз часових характеристик показав, що середня латентність обробки одного документа становить 14 секунд. Хоча це повільніше за класичні алгоритми, це у 128 разів швидше, ніж ручна обробка оператором (30 хвилин). Система демонструє стабільну роботу під навантаженням до 25 документів на хвилину, що повністю покриває потреби середнього деревообробного підприємства.

5. Економічна ефективність. Розрахунок за методикою TCO (Total Cost of Ownership) довів високу інвестиційну привабливість розробки для малого та середнього бізнесу:

- Низький поріг входу: Капітальні витрати на запуск складають близько $1,050, що у 10–15 разів менше вартості впровадження промислових ERP-рішень (SAP, Oracle).
- Швидка окупність: Термін окупності проєкту (Payback Period) становить менше 1 місяця.
- Висока рентабельність: ROI за перший рік експлуатації прогнозується на рівні 690%.
- Економія ресурсів: Система дозволяє скоротити прямі витрати на оплату праці операторів на ~$1,000 щомісяця, а також мінімізує фінансові ризики від помилок людського фактору.

Підсумовуючи, результати третього розділу свідчать про те, що розроблена система є не лише технічно досконалим, а й економічно виправданим рішенням, готовим до впровадження у реальний виробничий

процес.

ВИСНОВКИ

У магістерській роботі вирішено актуальне науково-прикладне завдання підвищення ефективності управління інформаційними потоками деревообробного підприємства шляхом розробки та впровадження інтелектуальної системи на базі технологій LLM, RAG та low-code оркестрації.

Результати проведеного дослідження дозволяють зробити наступні узагальнюючі висновки:

1. Проведено системний аналіз проблематики галузі. Встановлено, що ключовим бар'єром цифровізації деревообробних підприємств є домінування неструктурованих даних (рукописні ТТН-ліс, карти розкрою, сертифікати), частка яких у документообігу сягає 80%. Виявлено, що традиційні методи автоматизації (класичні ERP та детерміновані OCR-алгоритми) є економічно неефективними для малого та середнього бізнесу (МСБ) через високу вартість впровадження та нездатність адаптуватися до змінних форматів документів.

2. Теоретично обґрунтовано та спроєктовано нову архітектуру. Запропоновано концепцію гібридної інтелектуальної системи, яка поєднує когнітивну гнучкість мультимодальних моделей (GPT-4o) з надійністю перевірки фактів через архітектуру RAG (Retrieval-Augmented Generation). Доведено, що використання векторної бази даних Qdrant із механізмом гібридного пошуку дозволяє нівелювати ризик «галюцинацій» ШІ, забезпечуючи достовірність даних, необхідну для промислового обліку.

3. Розроблено та імплементовано програмний комплекс. На базі

технології контейнеризації Docker реалізовано мікросервісну архітектуру системи. В якості інтеграційного ядра використано low-code платформу n8n, що дозволило створити гнучкі сценарії обробки даних (Workflows) із значно меншими витратами часу та ресурсів порівняно з традиційною розробкою. Реалізовано механізм «Human-in-the-Loop», який автоматично залучає оператора до верифікації даних лише у випадках низької впевненості моделі.

4. Експериментально підтверджено ефективність рішення. Результати тестування на реальному масиві виробничої документації продемонстрували значну перевагу розробленої системи над аналогами:

    a. Точність розпізнавання складних табличних даних (F1-Score) склала 0.96, що у 6 разів перевищує показники традиційного Tesseract OCR (0.15).

    b. Середній час обробки одного документа скорочено з 20–30 хвилин (вручну) до 14 секунд (автоматично).

    c. Точність семантичного пошуку документів у базі знань досягла показника Recall@5 = 0.94, що забезпечує швидкий доступ до архівних даних.

5. Впроваджено інноваційну систему моніторингу. Розроблено методику контролю бізнес-метрик функціонування AI за допомогою стеку Prometheus–Grafana. Це дозволяє керівництву підприємства в режимі реального часу відстежувати не технічні параметри сервера, а економічні показники: вартість обробки одного документа (Cost-Per-Token), рівень автоматизації та якість

даних.

6. Доведено економічну доцільність. Розрахунок за методикою сукупної вартості володіння (TCO) показав, що запропоноване рішення є високорентабельним для МСБ.
    a. Початкові інвестиції (CAPEX) становлять близько $1,050, що на порядок менше вартості впровадження промислових ERP-модулів.
    b. Термін окупності проєкту (Payback Period) становить менше 1 місяця.
    c. Коефіцієнт рентабельності інвестицій (ROI) за перший рік експлуатації сягає 690%.
    d. Система дозволяє економити понад $12,000 щорічно на фонді оплати праці та мінімізує фінансові ризики від помилок персоналу.

Таким чином, мета роботи досягнута. Розроблена інтелектуальна система управління даними є готовим до масштабування інструментом, який дозволяє деревообробним підприємствам здійснити цифровий перехід, підвищити прозорість обліку та конкурентоспроможність на ринку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Державне агентство лісових ресурсів України. Матеріали щодо системи обліку деревини. Київ : Держлісагентство України, 2020.
2. Marr B. Data Strategy: How to Profit from a World of Big Data, Analytics and the Internet of Things. London : Kogan Page, 2018.
3. Україна. Кабінет Міністрів. Постанова №665 «Про затвердження форм товарно-транспортної накладної на деревину» від 06.06.2007 р.
4. Food and Agriculture Organization (FAO). Global Timber Documentation Standards. Rome : FAO, 2019.
5. Karthik R., Rajeswari N. OCR challenges on handwritten industrial documents // *IEEE Access*. 2022.
6. International Organization for Standardization. ISO 9001:2015. Quality management systems — Requirements. Geneva : ISO, 2015.
7. Forest Stewardship Council (FSC). FSC-STD-40-004 V3-1. Chain of Custody Certification. Bonn : FSC International, 2021.
8. Davenport T. Process Innovation and ERP Limitations. Cambridge : MIT Press, 2019.
9. O’Leary D. ERP and data quality issues // *Journal of Emerging Technologies in Accounting*. 2018.
10. Bain & Company. Manufacturing Lead Time Benchmark Report. New York : Bain & Co., 2020.
11. Accenture. Human error in manual data entry: Analytical report. Dublin : Accenture, 2019.

12. Gartner. Dark Data and its impact on industry. Stamford : Gartner Research, 2021.

13. Lewis M. Retrieval-Augmented Generation for Enterprise Data Systems // *ACM Computing Surveys*. 2022.

14. Goodfellow I., Bengio Y., Courville A. Deep Learning. Cambridge : MIT Press, 2016.

15. Jurafsky D., Martin J. Speech and Language Processing. 3rd ed. Draft. Stanford University, 2023.

16. Casey R., Lecolinet E. A survey of methods and strategies in character segmentation // IEEE Trans. on PAMI. 1996.

17. Smith R. An overview of the Tesseract OCR engine // Proc. ICDAR. 2007.

18. Environmental Systems Research Institute. FSC Certificate Processing Guidelines. Bonn : FSC International, 2020.

19. Shi B., Bai X., Yao C. An end-to-end trainable neural network for scene text recognition // IEEE PAMI. 2017.

20. Vaswani A. et al. Attention is All You Need // NeurIPS. 2017.

21. OpenAI. GPT-4 Technical Report. San Francisco : OpenAI, 2024.

ДОДАТКИ

Фрагменти програмного коду до розділу 3.1

Лістинг 3.1. Файл конфігурації інфраструктури docker-compose.yml

version: '3.8' services: qdrant: image: qdrant/qdrant:v1.6.1 container_name: qdrant_prod restart: always ports: - "6333:6333" volumes: - qdrant_storage:/qdrant/storage networks: - wood_net n8n: image: n8nio/n8n:latest container_name: n8n_main restart: always ports: - "127.0.0.1:5678:5678" # Доступ лише через localhost (Nginx) environment: - N8N_BASIC_AUTH_ACTIVE=true - DB_TYPE=postgresdb - DB_POSTGRESDB_HOST=postgres - DB_POSTGRESDB_DATABASE=n8n - GENERIC_TIMEZONE=Europe/Kyiv env_file: .env depends_on: - postgres - qdrant networks: - wood_net postgres: image: postgres:13 container_name: n8n_db restart: always environment: - POSTGRES_USER=n8n - POSTGRES_DB=n8n env_file: .env volumes: - postgres_storage:/var/lib/postgresql/data networks: - wood_net networks: wood_net: driver: bridge volumes: n8n_storage: qdrant_storage: postgres_storage:

Лістинг 3.2. Запит ініціалізації колекції Qdrant (JSON)

JSON

```
PUT /collections/wood_docs
{
  "vectors": {
    "size": 1536,
    "distance": "Cosine"
  },
```

```
  "optimizers_config": {
    "default_segment_number": 2,
    "memmap_threshold": 20000
  },
  "hnsw_config": {
    "m": 16,
    "ef_construct": 100,
    "full_scan_threshold": 10000
  }
}
```

Лістинг 3.3. Приклад файлу змінних середовища `.env`

Bash

```
# Credentials for PostgreSQL
POSTGRES_PASSWORD=secure_db_password_123

# n8n Security
N8N_BASIC_AUTH_USER=admin
N8N_BASIC_AUTH_PASSWORD=admin_secure_pass
N8N_ENCRYPTION_KEY=random_generated_key_for_credentials

# AI Services
OPENAI_API_KEY=sk-proj-xxxxxxxxxxxxxxxxxxxx
QDRANT_URL=http://qdrant:6333

# Messaging
TELEGRAM_BOT_TOKEN=123456:ABC-DEF1234ghIkl-
zyx57W2v1u123ew11
```

Лістинг 3.4. Фрагмент конфігурації Nginx (`nginx.conf`)

Nginx

```
server {
    listen 443 ssl;
    server_name system.woodworking-corp.com;

    ssl_certificate /etc/letsencrypt/live/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/privkey.pem;

    location / {
        proxy_pass http://localhost:5678; # Проксування на
n8n
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
    }
}
```

Лістинг 3.4. Фрагмент конфігурації Nginx (nginx.conf)

Nginx

```
server {
    listen 443 ssl;
    server_name system.woodworking-corp.com;

    ssl_certificate /etc/letsencrypt/live/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/privkey.pem;
```

```
    location / {
        proxy_pass http://localhost:5678; # Проксування на
n8n
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
    }
}
```

Фрагменти програмного коду до розділу 3.2

Лістинг 3.5. Формат пейлоаду метрик у вузлі n8n (HTTP Request)

Plaintext

```
# HELP ai_token_usage_total Total OpenAI tokens consumed
# TYPE ai_token_usage_total counter
ai_token_usage_total{model="gpt-4o", operation="ocr"}
{{$json.usage.total_tokens}}

# HELP doc_processing_confidence Confidence score of the
extraction
# TYPE doc_processing_confidence gauge
doc_processing_confidence{doc_type="ttn"}
{{$json.confidence_score}}

# HELP workflow_duration_seconds Time taken to process
document
# TYPE workflow_duration_seconds gauge
```

```
workflow_duration_seconds{workflow="main_ingestion"}
{{$execution.time}}
```

Лістинг 3.6. Конфігурація скрапінгу `prometheus.yml`

```
YAML
global:
  scrape_interval: 15s

scrape_configs:
  - job_name: 'prometheus'
    static_configs:
      - targets: ['localhost:9090']

  - job_name: 'pushgateway'
    honor_labels: true
    static_configs:
      - targets: ['pushgateway:9091']
```

Лістинг 3.7. Правило алертингу (PromQL Rule)

```
YAML
groups:
- name: ai_alerts
  rules:
  - alert: LowQualityDetection
    expr: avg_over_time(doc_processing_confidence[10m]) <
0.75
    for: 5m
```

```
    labels:
      severity: warning
    annotations:
      summary: "High rate of low-confidence extractions"
      description: "AI confidence dropped below 75%. Check
input document quality."
```