

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 123 «Комп'ютернам інженерія»

Освітньо-професійна програма: «Комп'ютерна графіка та Web-дизайн»

Група: 4КГ-08

Дипломний проект

здобувача освіти денної форми навчання
КГ.08.27.000.ДП

**УСАТЮКА
ОЛЕКСАНДРА
ГЕННАДІЙОВИЧА**

м. Одеса

2025 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 123 «Комп'ютернам інженерія»

Освітньо-професійна програма: «Комп'ютерна графіка та Web-дизайн»

Група: 4КГ-08

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту на тему:

Розробка застосунку для аналізу поведінки тварин на базі LLM

Проектний матеріал складається з пояснювальної записки на 76 сторінках та графічного (перзентаційного) матеріалу на 12 аркушах (слайдах).

Дипломник Усатюк (Усатюк О.Г.)

Керівник Ломовцев (Ломовцев П.Б.)

Консультанти:

з економічного розділу А (Канський М. Ю.)

з розділу охорони праці та техніки безпеки Сол (Чорновол Н. І.)

з нормоконтролю Петрашова (Петрашова В. І.)

старший консультант Кривченко (Кривченко Ю. В.)

До захисту допущений

Голова циклової комісії Кривченко (Кривченко Ю. В.)

Завідувач відділення Краснокутська (Краснокутська К. Г.)

Захист «28» сервіс 2025 р.

Протокол ЕК № 7

Оцінка ЕК 4(добре) / 85%

Секретар ЕК Секретар

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Відділення Ком'ютерних систем Комісія КТ та ПІ
Спеціальність 123 «Комп'ютерна інженерія»
Освітня програма «Комп'ютерна графіка та Web-дизайн»

ЗАТВЕРДЖУЮ

Заст. дир. з НВР Беркань І. В.

« 19 » 08 2025 року

ЗАВДАННЯ

на дипломний проєкт (роботу)

Усатюка Олександра Геннадійовича

1. Тема проєкту (роботи) Розробка застосунку для аналізу поведінки тварин на базі LLM

Затверджена наказом по коледжу від « 14 » 11 2025 р., наказ № 246

2 Термін здачі закінченого проєкту (роботи) _____

3. Вихідні дані до проєкту (роботи)

Розробка застосунку для аналізу поведінки тварин на базі LLM; Великі мовні моделі (LLM);
Мова програмування Python; Фреймворк Streamlit та REST AP; Backend фреймворк API
сервер FastAPI; Фреймворк для AI-оркестрації LangChain; Конфігурація dotenv .env безпечно
зберігання параметрів.


4. Зміст розрахунково-пояснювальної записки (перелік питань, які необхідно розробити)

Аналіз існуючих програмних рішень; Розробка клієнт-серверної архітектури системи з
використанням Python, Streamlit та REST AP; Проектування структури застосунку;
Розробка застосунку; Тестування створеного програмного продукту; Економічний
розрахунок; Охорона праці та техніка безпеки; Економічний розрахунок; Висновки.

5. Перелік графічного (презентаційного) матеріалу (з точним зазначенням обов'язкових креслень, кількості слайдів)

Презентація Power Point - 13 слайдів Титульний слайд ДП; Еволюція великих мовних
моделей; Діаграма параметрів трансформерних мовних моделей; Структура системи;
Скріншот головної сторінки системи; Скріншот виділення кадрів із відео; Скріншот кадру
з обробки відео; Скріншот обробки запиту; Скріншот обробки запиту з кадрів
по підрахунку курей; Скріншот обробки запиту; Висновки.

6. Консультанти по проекту, із зазначенням розділів проекту, що їх стосується

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Основний розділ	Ломовцев П.Б.		
Економічний розділ	Канський М. Ю.		
Розділ охорони праці	Чорновол Н. І.		
Нормоконтроль	Петрашова В. І.		
Старший консультант	Кривченко Ю. В.		

7. Дата видачі завдання 29.04.25

Керівник

Ломовцев П.Б.



(підпис)

Завдання прийняв до виконання

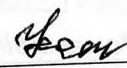
Усатюк О.Г.

(підпис)

КАЛЕНДАРНИЙ ПЛАН

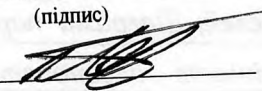
№ з/р	Назва етапів дипломного проекту (роботи)	Термін виконання етапів дипломного проекту (роботи)	Відмітка про виконання
1	Вступ. Постановка мети та задач проектування	29.04.25	в виконанні
2	Огляд основних компонентів серверної архітектури	01.05.25	в виконанні
3	Обґрунтування вибору технологій розробки	03.05.25	виконано
4	Розробка структури проекту	05.05.25	виконано
5	Налаштування структури проекту	07.05.25	виконано
6	Інтеграція бази даних	09.05.25	виконано
7	Тестування створеного програмного продукту	15.05.25	виконано
8	Економічний розрахунок	17.05.25	виконано
9	Опис охорони праці та техніки безпеки	19.05.25	виконано
10	Аналіз результатів проектування.	21.05.25	виконано
11	Оформлення пояснювальної записки	23.05.25	виконано
12	Оформлення графічної (презентаційної)	30.05.25	виконано
13	частини	05.06.25	виконано
14	Підготовка доповіді для захисту	15.06.25	в виконанні
15	Малий захист дипломного проекту	16.06.25	в виконанні

Дипломник



(підпис)

Керівник



(підпис)

ЗМІСТ

Вступ.....	7
1 Основний розділ.....	8
1.1 Аналіз можливостей штучного інтелекту	8
1.2 Автоматизація процесів за допомогою ШІ і LLM.....	9
1.3 Інтеграція штучного інтелекту та великих мовних моделей (LLM) у різні сфери діяльності.....	10
1.4 Огляд та вибір технологій для реалізації	13
1.4.1 Структура та копоненти серверної архітектури.....	13
1.4.2 Використання Python FastAPI та фронтенд-фреймворку Streamlit	15
1.4.3 Swagger для документації API.....	17
1.4.4 Можливості OpenAI API.....	18
1.5 Проектування схеми серверної архітектури для системи.....	19
1.6 Впровадження штучного інтелекту через API.....	22
1.6.1 Використання OpenAI для інтеграції LLM моделі.....	25
1.7 Реалізація серверної частини.....	27
1.7.1 Планування проекту.....	27
1.7.2 Апаратні вимоги.....	28
1.7.3 Програмні вимоги.....	30
1.7.4 Вимоги до безпеки.....	31
1.7.5 Технологічний стек для системи роботи з LLM.....	31
1.7.6 Розробка сервера та інтеграція.....	33
1.7.7 Реалізація сервера.....	34
1.8 Програмна реалізація застосунку.....	35
1.8.1 Загальна структура програми.....	36
1.8.2 Взаємодія компонентів системи.....	38
1.9 Приклад роботи застосунку.....	40
1.10 Підключення та використання OpenAI API.....	50

					КГ 08. 27 000. 00 ДП ПЗ	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		5

1.10.1	Налаштування доступу до OpenAI API.....	50
1.10.2	Файли .env.example-шаблону зміни середовища.....	51
2	Економічна розділ.....	56
2.1	Резюме.....	56
2.2	Вивчення трудомісткості розробки програмного забезпечення.....	56
2.3	Розрахунок ціни програмного продукту.....	59
3	Розділ охорони праці та техніки безпеки.....	61
3.1	Аналіз та безпека умов праці працівника на робочому місці.....	61
3.2	Розробка заходів з охорони праці.....	62
3.3	Організація робочого місця користувача ПК.....	63
3.4	Пожежна безпека.....	64
	Висновки.....	66
	Перелік використаних інформаційних джерел.....	67
	Додаток А. Програмний код «Опис ендпоїнтів».....	68
	Додаток Б. Слайди мультимедійної презентації.....	69

ВСТУП

Сучасні технології штучного інтелекту (ШІ) та великі мовні моделі (LLM, Large Language Models) знаходять усе ширше застосування в аграрному секторі, сприяючи автоматизації процесів моніторингу, підвищенню продуктивності та забезпеченню сталого розвитку сільського господарства. У даному дослідженні розглядається використання LLM у птахівництві, тваринництві та інших напрямках аграрної галузі, а також їх інтеграція з технологіями комп'ютерного зору та обробки відеоданих.

Запропонована система ґрунтується на застосуванні мультимодальних LLM на прикладі для аналізу поведінкової активності тварин на основі потокового відео. Це дає змогу оперативно виявляти аномалії у їх поведінці, прогнозувати потенційні загрози для здоров'я та автоматично генерувати рекомендації для фермерів щодо коригування умов утримання.

Результати проведеного дослідження підтверджують, що впровадження LLM у сільське господарство забезпечує низку ключових переваг, зокрема: автоматизований моніторинг стану тварин і сільськогосподарських культур; підвищення точності прогнозування врожайності та характеристик ґрунтів; зниження залежності від людського фактора у процесах моніторингу та прийняття рішень; підтримку оперативного ухвалення рішень на основі даних у режимі реального часу. Порівняльний аналіз сучасних моделей LLM, зокрема GPT-4, Flamingo та LLaVA, продемонстрував їхній значний потенціал у сфері обробки відеоданих і мультимодального аналізу. У процесі аналізу були ідентифіковані основні технічні та експлуатаційні чинники, що визначають ефективність впровадження зазначених технологій в аграрний сектор.

Отримані результати свідчать про те, що поєднання великих мовних моделей із технологіями машинного навчання та комп'ютерного зору відкриває нові перспективи для розвитку точного землеробства та автоматизованого контролю за здоров'ям тварин. Це, у свою чергу, сприяє підвищенню технологічного рівня аграрного виробництва, його економічної ефективності та екологічної сталості.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		7

1 ОСНОВНИЙ РОЗДІЛ

1.1 Аналіз можливостей штучного інтелекту

Станом на 2023 рік у сфері штучного інтелекту виокремлюється кілька провідних напрямів розвитку.

Артифіційний суперінтелект (ASI) - це теоретичний тип ШІ, здатний не лише імітувати, а й значно перевершувати людські інтелектуальні та когнітивні можливості. Очікується, що ASI матиме здатність до самостійного навчання, прийняття рішень у складних та нестандартних ситуаціях, а також зможе впливати на людські емоції та ментальні процеси. На сьогодні це радше концепція, яка перебуває на стадії теоретичних досліджень, однак багато експертів вважають, що саме цей напрямок стане одним із ключових викликів для науки у майбутньому.

Штучний загальний інтелект (AGI) - гіпотетичний вид ШІ, який повинен володіти універсальними інтелектуальними здібностями, що дозволяють виконувати широкий спектр завдань на рівні людини або наближеному до нього. Це включає не тільки технічні або математичні завдання, а й креативні, соціальні, емоційно забарвлені задачі. AGI передбачає наявність здатності до навчання на основі нових даних, самокорекції та генерації нових знань без явного програмування.

Слабкий або вузький інтелект (ANI) - найбільш розвинений на сьогодні тип ШІ, що спеціалізується на виконанні окремих завдань або функцій. Системи ANI створюються для конкретних застосувань -наприклад, розпізнавання мови, аналізу медичних зображень, управління автономними транспортними засобами або персоналізованих рекомендацій у маркетингу. Вони демонструють високу ефективність у своїй вузькій області, однак не здатні переносити свої знання або навички в інші контексти.

Результати підтверджують, що поєднання LLM із технологіями машинного навчання та комп'ютерного зору відкриває нові перспективи для розвитку точного землеробства та автоматизованого контролю за здоров'ям тварин, роблячи агропідприємства більш технологічним, економічно ефективним та екологічно орієнтованим.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		8

1.2 Автоматизація процесів за допомогою ШІ і LLM

У розвинених країнах світу вже давно доведено ефективність застосування інформаційних технологій (ІТ) у сільському господарстві. У США, Японії, Китаї та низці європейських держав (зокрема, у Німеччині, Великій Британії, Нідерландах, Данії, Ізраїлі) впровадження сучасних технологій у цю сферу розпочалося ще у 80-х роках минулого століття.

Сьогодні технології штучного інтелекту (ШІ) та великі мовні моделі (LLM, Large Language Models) стрімко знаходять практичне застосування в різних галузях економіки, у тому числі й в аграрному секторі. В умовах глобальних викликів - таких як зміна клімату, зростання чисельності населення, дефіцит трудових ресурсів та необхідність посилення продовольчої безпеки - впровадження передових цифрових рішень стає ключовим чинником сталого розвитку сільського господарства.

Великі мовні моделі мають значний потенціал у сфері обробки великих обсягів даних, які надходять від сільськогосподарських систем. Вони дозволяють аналізувати врожайність, оцінювати стан ґрунтів, прогнозувати потреби в добривах та зрошенні. LLM також можна інтегрувати із сенсорними системами для інтерпретації показників вологості, температури та рівня рН ґрунту.

В агропромисловому комплексі накопичено значні масиви технічної та наукової документації, нормативно-правових актів і дослідницьких звітів. Застосування LLM дає змогу автоматизувати обробку цієї інформації, що забезпечує швидке вилучення ключових даних, аналіз текстів і генерацію звітів. Це значно зменшує навантаження на фахівців і прискорює процеси ухвалення рішень.

Поєднання великих мовних моделей із технологіями комп'ютерного зору (зокрема, OpenCV, YOLO) дає змогу здійснювати аналіз зображень сільськогосподарських культур і тварин, виявляти ознаки захворювань і формувати рекомендації щодо їх усунення. Наприклад, у птахівництві LLM використовуються для аналізу поведінкових даних курей, що дає можливість своєчасно виявляти відхилення, які можуть свідчити про проблеми зі здоров'ям.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		9

Оскільки аграрний сектор часто пов'язаний з роботою з документацією різними мовами (особливо в умовах міжнародної торгівлі та наукового співробітництва), великі мовні моделі можуть автоматично перекладати та адаптувати складні агрономічні тексти. Це сприяє спрощенню міжнародної співпраці та впровадженню передових технологій у виробничі процеси.

Застосування мовних моделей також відкриває можливості для аналізу економічних і метеорологічних звітів, прогнозування змін попиту та пропозиції на сільськогосподарську продукцію, а також оцінювання впливу різноманітних факторів на ціноутворення. Це дозволяє аграрним компаніям ефективніше планувати стратегії виробництва та логістики.

Отже, впровадження великих мовних моделей у сільське господарство сприяє значному підвищенню ефективності виробництва, мінімізації ризиків і автоматизації рутинних процесів. Використання цих технологій дозволяє підвищити якість управління аграрними підприємствами, покращити підтримку фермерів, забезпечити більш ефективний аналіз даних і моніторинг здоров'я тварин та рослин.

1.3 Інтеграція штучного інтелекту та великих мовних моделей (LLM) у різні сфери діяльності

Сучасні технології штучного інтелекту (ШІ), зокрема машинне навчання, комп'ютерний зір і робототехніка, сприяли впровадженню концепції точного землеробства -підходу, заснованого на використанні даних, що забезпечує більш ефективний розподіл ресурсів та підтримує ухвалення обґрунтованих аграрних рішень. Це перетворення відповідає загальній тенденції [8,9], що передбачає інтеграцію цифрових інструментів, аналітики даних та автоматизації для підвищення продуктивності та сталого розвитку аграрного сектору.

У сільському господарстві застосування таких методів забезпечує точний моніторинг за рахунок підвищення чіткості зображень і точності отриманих даних, що безпосередньо відповідає завданням систем моніторингу птахів на базі ШІ.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		10

Оптимізуючи якість зображень і точність аналітичних даних, ці технології ефективно підтримують ухвалення рішень у сільському господарстві, особливо в умовах високої щільності утримання тварин, де точність виявлення має вирішальне значення.

ІІІ-системи також трансформували управління врожайністю шляхом інтеграції даних із різноманітних джерел:

- сенсорів Інтернету речей (ІоТ) для моніторингу ґрунтових та кліматичних параметрів;
- безпілотних літальних апаратів (дронів) для аерофотозйомки та оцінки стану сільськогосподарських культур;
- супутникових знімків для проведення масштабного аналізу полів.

Завдяки цим технологіям аграрії отримують у режимі реального часу актуальну інформацію про стан сільськогосподарських культур, характеристики ґрунту та кліматичні умови, що дає змогу здійснювати точні втручання у процесах зрошення, боротьби зі шкідниками та внесення добрив. Такі втручання суттєво зменшують обсяги використання хімічних препаратів і мінімізують негативний вплив на навколишнє середовище [11,12].

Крім того, автоматизація завдань із моніторингу стану сільськогосподарських культур та виявлення захворювань за допомогою сучасних моделей машинного навчання забезпечила як підвищення ефективності, так і зростання точності порівняно з традиційними методами -ручними перевірками та візуальними оцінками [13,14].

У сфері тваринництва застосування технологій ІІІ, зокрема моделей глибокого навчання для моніторингу поведінки та стану здоров'я тварин, значно розширило можливості оцінювання добробуту тварин. Згорткові нейронні мережі (CNN, Convolutional Neural Networks), які демонструють високу точність в аналізі зображень, забезпечують раннє виявлення проблем зі здоров'ям шляхом моніторингу змін у поведінці тварин, що дозволяє впроваджувати проактивні стратегії управління [12,13].

Прогностичні можливості ШІ у сфері прогнозування врожайності та адаптації до кліматичних змін оптимізують використання ресурсів, зменшують кількість відходів і підвищують точність прогнозів урожайності. Це сприяє ефективнішій адаптації сільського господарства до кліматичних коливань і ресурсних обмежень [9,14] та виступає інструментом модернізації й розвитку агробізнесу, зокрема у сферах сучасного птахівництва, тваринництва тощо. Еволюція великих мовних моделей (LLM) показано на рисунку 1.1.

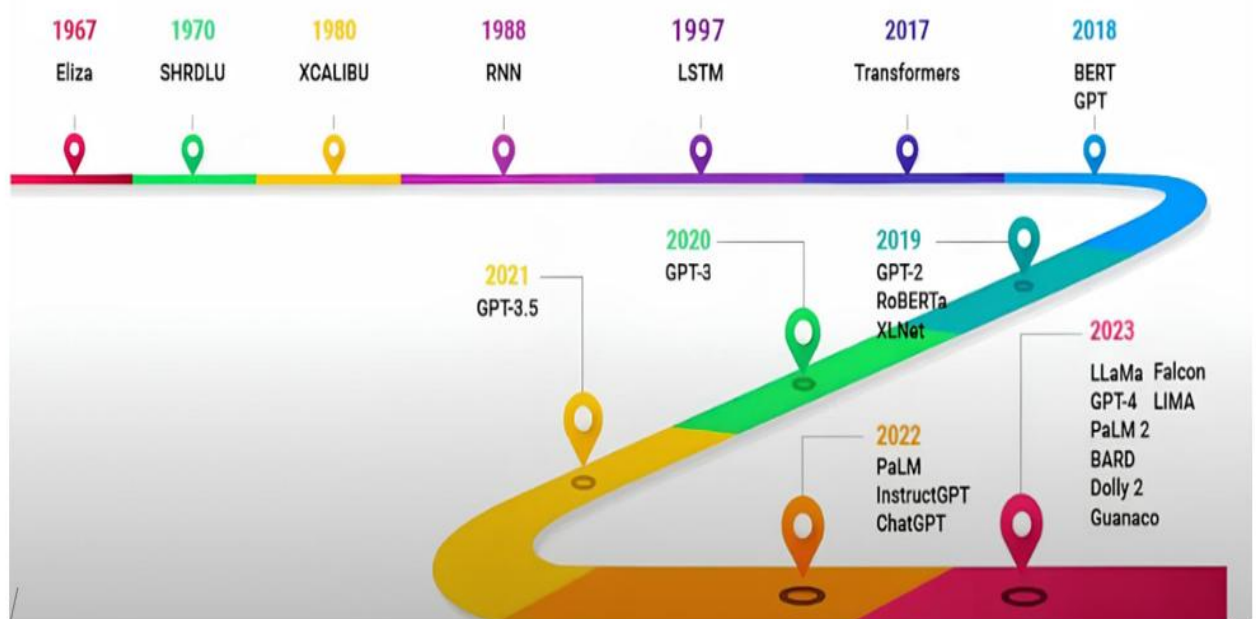


Рисунок 1.1. Еволюція великих мовних моделей (LLM)

Цей рисунок представляє хронологічну еволюцію мовних моделей та архітектур у сфері обробки природної мови (NLP) - від перших експериментальних програм до сучасних великих мовних моделей (LLM). Від простих правил (Eliza) до багатомільярдних моделей (GPT-4, PaLM 2). Ключовий прорив відбувся у 2017 році з появою які лягли в основу всіх сучасних LLM.

Щороку з'являються нові, потужніші та більш адаптивні моделі, орієнтовані на генерацію тексту, розуміння контексту, інтеграцію в застосунки, а також на безпечну взаємодію з користувачами.

1.4 Огляд та вибір технологій для реалізація застосунку

1.4.1 Структура та компоненти серверної архітектури

Сучасні серверні архітектури відіграють ключову роль у забезпеченні надійної, масштабованої та ефективної роботи програмних систем. Від правильного проектування серверної частини залежить стабільність, продуктивність і можливість подальшого розвитку будь-якого застосунку. Розглянемо основні структурні елементи та компоненти типової серверної архітектури на прикладі системи для обробки та аналізу відео.

1. Основні рівні архітектури

1.1. API-рівень (Контролери та маршрутизація)

Цей рівень відповідає за прийом зовнішніх запитів від клієнтів (наприклад, веб-інтерфейсу або мобільного застосунку). Контролери обробляють HTTP-запити, виконують валідацію даних та передають їх на наступні рівні для обробки.

1.2. Сервісний рівень (Бізнес-логіка)

Сервіси містять основну бізнес-логіку застосунку. Вони відповідають за обробку даних, взаємодію з моделями, виконання складних операцій (наприклад, обробка відео, виділення кадрів, аналіз зображень).

1.3. Рівень моделей (Дані та схеми)

Моделі визначають структуру даних, які використовуються в системі. Вони можуть включати схеми для валідації, а також описувати взаємозв'язки між різними сутностями.

1.4. Рівень зберігання (Сховище)

Цей рівень відповідає за зберігання відеофайлів, кадрів та інших даних. Для цього можуть використовуватись файлові системи, хмарні сховища або бази даних.

2. Ключові компоненти серверної архітектури

2.1. Контролери (Controllers)

Приймають запити, виконують первинну обробку та передають дані у відповідні сервіси.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		13

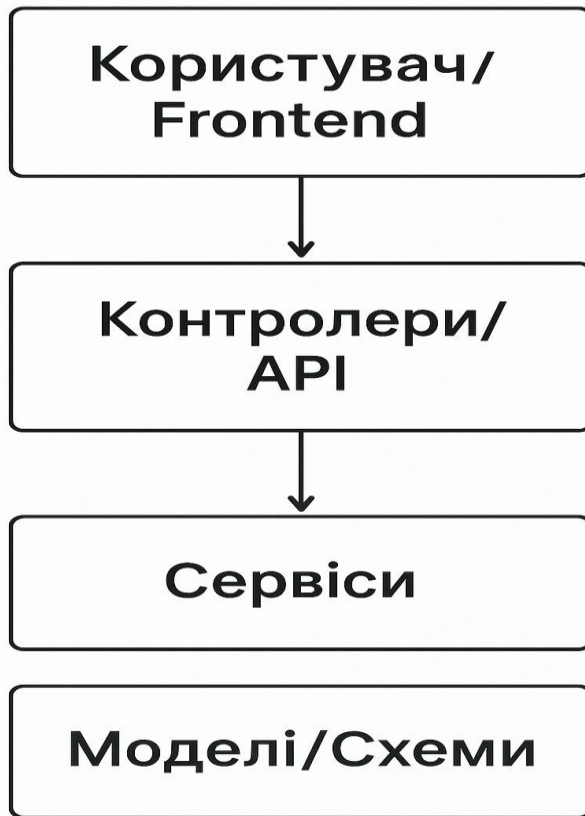


Рисунок 1.2. Структура та компоненти серверної архітектури

2.2. Сервіси (Services)

Реалізують бізнес-логіку: обробка відео, виділення кадрів, взаємодія з зовнішніми API (наприклад, YouTube, OpenAI).

2.3. Моделі та схеми (Models & Schemas)

Визначають структуру даних, що використовуються у запитах, відповідях та при зберіганні.

2.4. Утиліти (Utils)

Допоміжні функції для повторюваних операцій (наприклад, конвертація форматів, логування, обробка помилок).

2.5. Сховище (Storage)

Фізичне або хмарне місце для зберігання відео та кадрів.

2.6. Інтерфейс адміністратора/користувача (Frontend)

Хоча це не частина серверної архітектури, фронтенд взаємодіє із сервером через API, забезпечуючи зручний інтерфейс для користувача.

3. Додаткові компоненти

Документація API (Swagger, ReDoc) - автоматично генерується для зручності розробників.

Docker-контейнери - забезпечують ізольоване середовище для запуску сервісів.

CI/CD - автоматизація тестування та розгортання.

Моніторинг та логування - для відстеження стану системи та швидкого реагування на помилки.

Грамотно спроектована серверна архітектура забезпечує гнучкість, масштабованість і надійність програмної системи. Чітке розділення відповідальностей між компонентами дозволяє легко розширювати функціонал, підтримувати та оптимізувати систему відповідно до зростаючих вимог бізнесу.

1.4.2 Використання Python FastAPI та фронтенд-фреймворку Streamlit

У сучасній розробці програмного забезпечення важливу роль відіграють інструменти, які дозволяють швидко створювати масштабовані серверні додатки та зручні інтерфейси для користувачів. Python пропонує широкий вибір таких інструментів, серед яких FastAPI та Streamlit займають провідні позиції завдяки своїй простоті, гнучкості та продуктивності.

FastAPI - це сучасний, високопродуктивний веб-фреймворк для створення RESTful API на Python. Його основні переваги:

Висока швидкість так як FastAPI побудований на базі асинхронного сервера Uvicorn та бібліотеки Starlette, що забезпечує високу продуктивність.

Автоматична генерація документації, Swagger UI та ReDoc інтегруються "з коробки", що спрощує тестування та інтеграцію API.

Вбудована валідація даних: Використання Pydantic для типізації та перевірки вхідних/вихідних даних.

Зручність для розробників: Простий синтаксис, підтримка сучасних можливостей Python (type hints, async/await).

Streamlit - це open-source фреймворк для створення веб-інтерфейсів на Python, орієнтований на швидку розробку прототипів, аналітичних панелей та візуалізацій.

Простота: Інтерфейс створюється за допомогою звичайного Python-коду без необхідності знати HTML, CSS чи JavaScript.

Інтерактивність: Підтримка віджетів (кнопки, слайдери, завантаження файлів тощо) для взаємодії з користувачем.

Інтеграція з науковими бібліотеками: Легко виводити графіки, таблиці, зображення.

Сценарій інтеграції FastAPI та Streamlit

У типовому проєкті FastAPI використовується для реалізації серверної логіки, обробки даних, взаємодії з базами даних чи зовнішніми сервісами. Streamlit, у свою чергу, забезпечує зручний інтерфейс для користувача, який взаємодіє з бекендом через HTTP-запити до FastAPI.

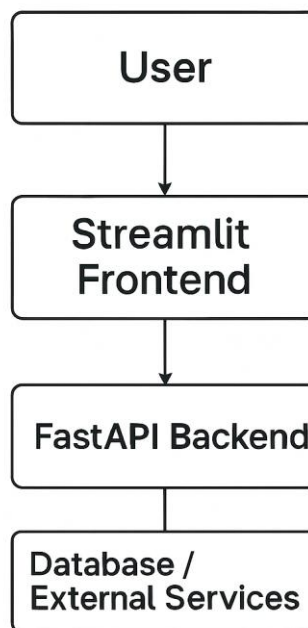


Рисунок 1.3. Сценарій інтеграції FastAPI та Streamlit

Переваги такого підходу:

1. Чітке розділення відповідальностей між фронтендом і бекендом
2. Можливість масштабування кожного компонента окремо
3. Гнучкість у розширенні функціоналу

Використання FastAPI та Streamlit у комплексі дозволяє швидко створювати сучасні, масштабовані та зручні для користувача програмні рішення. Такий підхід ідеально підходить для наукових, аналітичних, освітніх та бізнес-проектів, де важлива швидкість розробки, інтерактивність та простота підтримки.

1.4.3 Swagger для документації API

Swagger - це фреймворк з відкритим кодом, який використовується для створення, опису, документування та візуального тестування RESTful API. У 2016 році Swagger став частиною проекту OpenAPI Specification (OAS), і зараз терміни Swagger та OpenAPI часто вживаються як синоніми, хоча технічно Swagger - це набір інструментів, що реалізують OpenAPI. Swagger дозволяє:

У проєкті для документування та тестування RESTful API використовується інтеграція FastAPI з OpenAPI (Swagger). FastAPI автоматично генерує машиночитабельну документацію для всіх ендпоінтів у форматі OpenAPI Specification (OAS), що забезпечує прозорість, зручність тестування та легкість інтеграції для розробників.

Основні можливості Swagger/OpenAPI у проєкті:

Автоматична генерація документації:

FastAPI створює специфікацію API у форматі JSON або YAML на основі опису маршрутів, параметрів, типів запитів і відповідей.

Інтерактивний веб-інтерфейс (Swagger UI):

Документація доступна за адресою /docs і дозволяє переглядати всі ендпоінти, їх параметри, а також тестувати запити безпосередньо у браузері.

Підтримка OpenAPI:

Вся документація відповідає стандарту OpenAPI, що дозволяє легко генерувати клієнтський або серверний код для різних мов програмування за допомогою сторонніх інструментів (наприклад, Swagger Codegen).

Зручність для розробників:

Документація оновлюється автоматично при зміні коду, що мінімізує ризик

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		17

неактуальної інформації та спрощує командну роботу. Ключові компоненти Swagger/OpenAPI, які використовуються у проєкті:

OpenAPI Specification (OAS): Формат опису REST API, який визначає всі маршрути, методи, параметри та структури даних.

Swagger UI: Інтерактивний веб-інтерфейс для перегляду та тестування API, доступний за замовчуванням у FastAPI.

ReDoc: Альтернативний інтерфейс документації, доступний за адресою /redoc.

Переваги такого підходу:

Відсутність необхідності вручну підтримувати документацію — все генерується автоматично.

Можливість швидко тестувати API під час розробки.

Легкість інтеграції з іншими системами та генерації SDK.

Висновок:

Завдяки використанню FastAPI та вбудованої підтримки Swagger/OpenAPI, проєкт отримує сучасну, зручну та завжди актуальну документацію, що значно спрощує розробку, тестування та подальшу підтримку API.

1.4.4 Можливості OpenAI API

OpenAI API надає доступ до передових моделей штучного інтелекту, зокрема мовних моделей GPT (Generative Pre-trained Transformer), які здатні виконувати широкий спектр завдань з обробки природної мови. Інтеграція OpenAI API у серверну архітектуру відкриває нові горизонти для створення інтелектуальних, адаптивних та інтерактивних застосунків.

Основні можливості OpenAI API:

Генерація тексту:

Автоматичне створення осмислених текстів на основі заданого запиту, що може використовуватись для написання статей, резюме, генерації описів, сценаріїв тощо.

Аналіз тексту:

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		18

Визначення тональності, класифікація, реферування, витяг ключових слів, тематичний аналіз, виявлення фактів та інше.

Відповіді на запитання:

Моделі можуть знаходити та формулювати відповіді на питання на основі наданого контексту або знань, що робить їх ідеальними для чат-ботів, довідкових систем, цифрових помічників.

Переклад тексту:

Підтримка багатьох мов для автоматичного перекладу, що дозволяє створювати багатомовні застосунки.

Розпізнавання та генерація коду:

Моделі OpenAI здатні генерувати, пояснювати та виправляти програмний код, що корисно для автоматизації розробки, навчання та підтримки користувачів.

Обробка зображень (DALL-E, CLIP):

Деякі моделі OpenAI дозволяють генерувати зображення за текстовим описом або аналізувати вміст зображень.

Аналіз діалогів та контексту:

Підтримка багатокрокових діалогів, збереження контексту розмови, персоналізація відповідей.

Практичне застосування у серверній архітектурі:

Інтелектуальні чат-боти та віртуальні помічники

Автоматизація підтримки користувачів, відповіді на типові питання, персоналізовані рекомендації.

Автоматичний аналіз відео та текстових даних

Генерація описів до відео, автоматичне реферування, пошук за змістом.

Мовна аналітика для бізнесу

Аналіз відгуків, соціальних мереж, автоматичне виявлення трендів та проблем.

Освітні та навчальні платформи.

Генерація тестових завдань, пояснення складних тем, автоматичне оцінювання відповідей.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		19

У даному проєкті OpenAI API може використовуватись для:
Автоматичного опису відео та кадрів (генерація текстових підписів)
Класифікації поведінки на основі текстових або візуальних даних
Створення інтелектуального інтерфейсу для взаємодії з користувачем
(наприклад, через Streamlit)

Підтримки багатомовності та перекладу результатів

Переваги використання OpenAI API:

Масштабованість: API працює у хмарі, не потребує локального розгортання моделей.

Гнучкість: Можливість налаштування параметрів генерації, вибору моделей, обмеження контексту.

Швидкість впровадження: Простий RESTful інтерфейс, підтримка багатьох мов програмування.

Безперервне оновлення: Доступ до найновіших моделей та функцій без необхідності самостійного навчання.

Висновок:

Інтеграція OpenAI API у серверну архітектуру значно розширює функціональні можливості застосунку, дозволяє створювати інтелектуальні сервіси нового покоління та забезпечує конкурентні переваги у сфері обробки природної мови та штучного інтелекту.

1.5 Проектування схеми серверної архітектури для системи

Проектування серверної архітектури є ключовим етапом створення сучасних інформаційних систем, що забезпечують обробку, зберігання та аналіз великих обсягів даних. У даному проєкті реалізовано багаторівневу серверну архітектуру, яка поєднує ефективну обробку відео, інтеграцію з зовнішніми сервісами та зручний інтерфейс для користувачів.

Основні компоненти архітектури:

FastAPI Backend

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		20

Відповідає за прийом, обробку та маршрутизацію HTTP-запитів, реалізацію бізнес-логіки, взаємодію з моделями та зовнішніми API (наприклад, OpenAI).

Сервіси обробки відео та кадрів

Реалізують функції завантаження, збереження, декодування відео, виділення кадрів з використанням OpenCV, а також подальший аналіз даних.

Моделі та схеми даних

Описують структуру основних сутностей системи (відео, кадр, результати аналізу), забезпечують валідацію та серіалізацію даних.

Файлове сховище

Використовується для зберігання відеофайлів та виділених кадрів. Може бути реалізовано як локально, так і у хмарі (наприклад, AWS EFS).

Streamlit Frontend

Інтерактивна панель для користувачів, яка дозволяє завантажувати відео, переглядати результати аналізу та взаємодіяти із системою через API.

Документація API (Swagger/OpenAPI)

Автоматично генерується FastAPI для зручності тестування та інтеграції.

Docker-контейнери

Забезпечують ізольоване середовище для розгортання бекенду та фронтенду, спрощують масштабування та супровід.

Пояснення до схеми архітектури.

Користувач взаємодіє із системою через Streamlit Frontend, який надсилає запити до FastAPI Backend.

FastAPI Backend приймає запити, викликає відповідні сервіси для обробки відео, аналізу кадрів, інтеграції з OpenAI API.

Сервіси працюють із моделями та схемами даних, забезпечують валідацію та обробку інформації.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		21

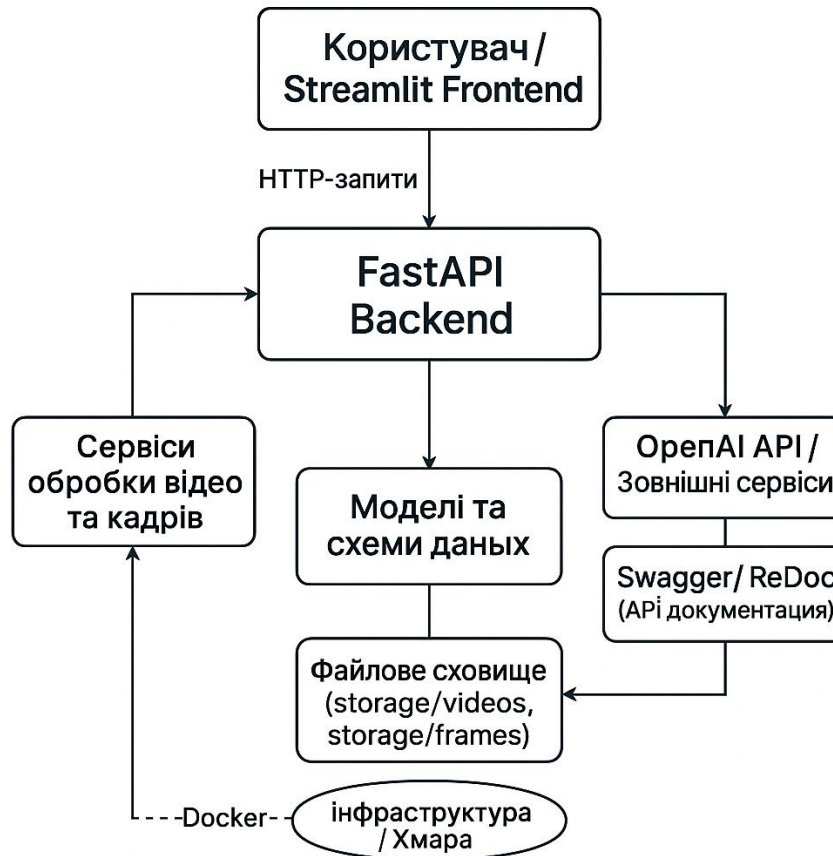


Рисунок 1.4. Схема архітектури

Файлове сховище використовується для зберігання відео та кадрів, доступне для подальшого аналізу.

Swagger забезпечують автоматичну документацію API.

Docker використовується для контейнеризації всіх компонентів, що спрощує розгортання як локально, так і у хмарі. Запропонована архітектура забезпечує гнучкість, масштабованість та простоту супроводу системи, дозволяє легко інтегрувати нові сервіси та адаптувати систему під різні сценарії використання.

1.6 Впровадження штучного інтелекту через API

API (Application Programming Interface) - це стандартизовані інтерфейси, що забезпечують взаємодію компонентів системи. Вони приховують внутрішню складність, забезпечують модульність, інтеграцію, безпеку, гнучкість та масштабованість. Нижче розглядаються основні переваги й практичні аспекти впровадження ШІ через API.

1. Абстрагування складності

API надають готові ШІ-функції - наприклад, GPT-4, DALL·E, Whisper - без необхідності розробляти моделі з нуля. Це суттєво спрощує інтеграцію інтелектуальних можливостей навіть для команд без глибоких знань у сфері ML.

2. Підвищення модульності

API дають змогу компонувувати сервіси як окремі блоки (микросервіси), які можна незалежно розробляти, тестувати та масштабувати. Це значно знижує складність і дає можливість гнучкого оновлення .

3. Інтеграція зовнішніх систем

API дозволяють легко інтегрувати зовнішні сервіси - такі як OpenAI, Google Cloud AI, Meta LLaMA API — використовуючи стандартизовані HTTP-запити. Це спрощує обмін даними між СУБД, фронтендом, OpenAI та іншими інструментами (Swagger/ReDoc, хмарна інфраструктура).

4. Швидкість впровадження ШІ

API дозволяють швидко додати інтелектуальні функції — генерацію тексту, переклад, обробку зображень, автоматичне програмування — завдяки готовим інтерфейсам і SDK для популярних мов (Python, JS, Java) .

5. Зниження витрат на розробку та інфраструктуру

Власний розвиток моделей потребує значних обчислювальних ресурсів і експертного часу. Використання API дозволяє перекласти ці витрати на провайдера, суттєво скорочуючи витрати компанії .

6. Масштабованість та гнучкість

API провайдери забезпечують автоматичне масштабування обчислень залежно від навантаження (через Kubernetes, serverless) . Це дозволяє безболісно обробляти пікові запити, навіть під час стрімкої зростання користувачів.

7. Інтерактивність через природну мову

ШІ через API дозволяє користувачам комунікувати з системою мовою, що значно підвищує UX та доступність функціоналу .

8. Автоматизація документації та контролю якості

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		23

AI-інструменти можуть публічно оновлювати Swagger/ReDoc опис API, генерувати тест-кейси, знаходити помилки у логах та коді, що пришвидшує розробку й підвищує стабільність.

9. Безпека та захист

AI-інструменти фільтрують небезпечні запити, виявляють аномальні шаблони (DDoS), автоматично блокують підозрілі активності, знижуючи кількість фальшивих спрацьовувань і підвищуючи захист API .

10. AI-управління і відповідність

AI допомагає формувати політики доступу, відстежувати контроль доступу (Identity & Access), виконувати аудит запитів — необхідне для відповідності GDPR, CCPA та іншим стандартам.

11. Прогнозування навантаження та оптимізація

AI аналізує трафік і прогнозує майбутні піки, автоматично коригуючи масштаби інфраструктури й обмеження використання.

12. Агенційний AI для оркестрації

Агенційні AI-системи можуть самостійно маршрутизувати запити, керувати оркестрацією мікросервісів, динамічно оптимізувати продуктивність API.

13. Аналітика та обробка даних у реальному часі

AI API допомагають аналізувати дані користувачів, оцінювати тон повідомлень, класифікувати дзвінки або обробляти соціальні мережі прямо в реальному часі, інтегруючи висновки в CRM, маркетинг чи служби підтримки .

14. Приватність і контроль даних

Cloud-провайдери (наприклад, Azure OpenAI) гарантують, що ваші дані залишаються конфіденційними, не використовуються для тренування інших моделей і зберігаються згідно з політиками безпеки .

Висновок.

Використання ШІ через API - це ефективний і сучасний шлях:

1. швидкого впровадження інтелектуальних функцій,
2. зниження витрат на інфраструктуру й розробку,
3. масштабованості й автоматичної оптимізації,

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		24

4. покращення безпеки та відповідності,

5. покращення користувацького досвіду.

Це дозволяє будувати конкурентоздатні рішення - від чат-ботів і генерації контенту до складного аналітичного процесу та безпекового моніторингу.

1.6.1 Використання OpenAI для інтеграції LLM моделі

OpenAI надає сучасні та гнучкі інструменти для інтеграції великих мовних моделей (LLM) через API, які забезпечують високу якість відповідей, мультимодальність і масштабованість. Найактуальніша інформація представлена нижче:

1. Простота інтеграції

1. OpenAI API має RESTful-архітектуру, яка дозволяє швидко інтегрувати LLM у будь-які веб- або серверні додатки.

2. Підтримуються SDK для Python, JavaScript та інших мов, а також готові приклади інтеграції.

2. Актуальні моделі

1. GPT-4o (Omni) - мультимодальна модель, яка обробляє текст, зображення та аудіо в реальному часі.

2. GPT-4.1 - найновіша текстова модель з підтримкою до 1 млн токенів контексту.

3. Sora - окрема модель для генерації відео з тексту (не через стандартний API).

3. Висока якість результатів

1. GPT-4o забезпечує природну мову, високу точність, емоційне розуміння та швидкість відповіді.

2. GPT-4.1 демонструє найкращі результати у тестах на довгі контексти та обробку відео-даних (Video-MME).

4. Гнучкість параметрів

1. Параметри temperature, top_p, max_tokens, stop дозволяють налаштовувати креативність, довжину та стиль відповіді.

2. Доступна можливість system prompt для управління поведінкою асистента.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		25

5. Масштабованість та продуктивність

1. OpenAI API побудований на serverless-інфраструктурі з автоматичним масштабуванням.

2. GPT-4o та GPT-4.1 підтримують велику кількість одночасних користувачів без втрати продуктивності.

6. Постійне оновлення

1. GPT-4o представлений у травні 2024 року як універсальна мультимодальна модель.

2. GPT-4.1 - найновіша з текстових моделей (2025), рекомендована замість GPT-4.5, який знімається з API у липні 2025 року.

7. Сфери застосування: чат-боти, служба підтримки, генерація тексту, маркетинг, реклама, SEO, навчальні системи та персональні помічники, обробка зображень, аудіо та відео (через GPT-4o), аналіз великих масивів даних (GPT-4.1 з довгим контекстом).

8. Навчання з підкріпленням (RLHF)

GPT-4o та GPT-4.1 навчені за допомогою людських оцінок (Reinforcement Learning from Human Feedback), що забезпечує точність і природність відповідей.

9. Ціни та ефективність

GPT-4o дешевший за GPT-4.1, працює швидше і охоплює ширший спектр задач.

GPT-4.1 підтримує великі контексти до 1 млн токенів - корисно для обробки довгих документів.

Зробивши аналіз великих мовних моделей побудував діаграму. Діаграма параметрів трансформерних мовних моделей показана на рисунку 1.5. На діаграмі параметрів трансформерних мовних моделей показано кількість параметрів (у мільйонах або мільярдах) у різних моделей LLM (Large Language Models), щоб порівняти їхній масштаб, розвиток і потужність.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		26

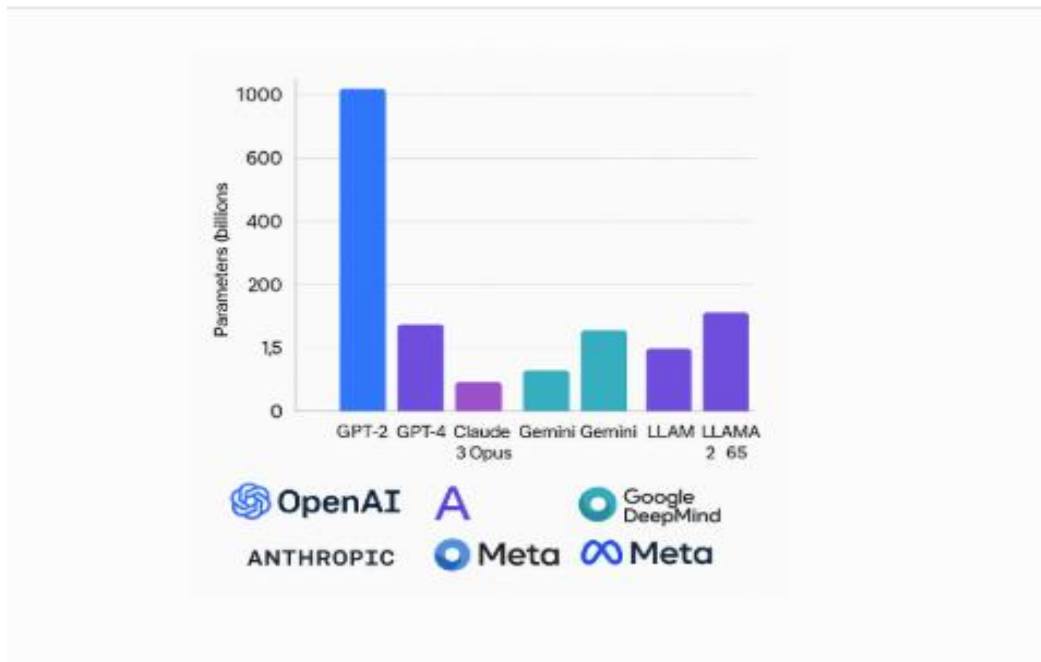


Рисунок 1.5. Діаграма параметрів трансформерних мовних моделей

1.7 Реалізація серверної частини

1.7.1 Планування проекту

На початковому етапі проектування серверної інфраструктури визначаю технічні вимоги, що базуються на функціональних характеристиках системи, обраній архітектурі та очікуваному навантаженні. При цьому враховую обсяг та тип оброблюваних даних, швидкість обробки запитів, вимоги до масштабованості та стійкості до збоїв. Важливу роль відіграють також особливості інтеграції з зовнішніми сервісами, зокрема API великих мовних моделей та векторної бази даних. У рамках даного проекту серверна частина має забезпечувати стабільну взаємодію з клієнтським інтерфейсом, виконання операцій обробки відеоданих у режимі реального часу, управління векторними embedding-даними та підтримку інтерактивної взаємодії користувача із системою. Планування реалізації серверної частини включає:

1.Визначення функціональних вимог.

Формування переліку основних функцій, які має виконувати серверна частина

(завантаження відео, виділення кадрів, інтеграція з OpenAI API, зберігання результатів тощо).

2. Вибір технологій та інструментів.

Обґрунтування вибору FastAPI для створення RESTful API, OpenCV для обробки відео, Streamlit для фронтенду, Docker для контейнеризації, а також структурування проєкту відповідно до принципів чистої архітектури.

3. Проектування структури директорій та модулів.

Розробка логічної структури проєкту (розділення на контролери, сервіси, моделі, схеми, утиліти, сховище тощо) для забезпечення зручності розробки, тестування та масштабування.

4. Планування інтеграції з зовнішніми сервісами.

Визначення способів взаємодії з OpenAI API, організація зберігання відео та кадрів, підготовка до можливого розгортання у хмарному середовищі (AWS).

5. Визначення етапів розробки та тестування.

Розбиття процесу на окремі етапи (розробка API, реалізація обробки відео, інтеграція з фронтендом, написання тестів, підготовка документації).

Цей підхід до планування дозволяє забезпечити системність, передбачуваність та якість реалізації серверної частини проєкту, а також створює основу для подальшого розвитку та масштабування системи.

1.7.2 Апаратні вимоги

Для ефективної роботи серверної частини системи, виконуючу обробку відео, аналіз кадрів та інтеграцію з зовнішніми сервісами, необхідно враховувати такі апаратні вимоги для ефективної роботи та масштабування:

1.Процесор: 8-ядерний або серверний CPU (Intel Xeon W, AMD Threadripper Pro/EPYC) - особливо важливо для обробки кількох потоків чи користувачів (proхрс.com).

2.RAM: мінімум 16 ГБ, краще 32 ГБ - рекомендовано 4 ГБ RAM на кожен потік GPU для моделей, що завантажуються у відеоаналіз.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		28

3.Диск: від 200 ГБ NVMe SSD або більше, залежно від обсягу відеоархіву і кількості кадрів.

4.GPU (опціонально): NVIDIA зі підтримкою CUDA (наприклад, RTX 4090 або дата-центр GPU A100/Blackwell) - суттєво прискорює обробку відео, аналіз кадрів і глибоке навчання (gcore.com).

5.Мережа: високошвидкісне підключення (10 GbE або більше), особливо для роботи з великими відеофайлами та віддаленими API.

Додаткові вимоги:

ОС: Linux (Ubuntu 22.04 LTS або схожа), можливе використання Windows Server/macOS для розробки.

Контейнеризація: Docker-сумісність для легкого розгортання.

Резервне копіювання: зовнішнє або хмарне сховище для збереження відеоданих і кадрів.

Розширення VRAM (при локальних МЛ-моделях): мінімум 12 ГБ VRAM (RTX 3060); для серйозного deep learning - 24–40 ГБ (RTX 3090, A100).

У межах розробки системи підтримки роботи з великими мовними моделями (LLM) особливу увагу було приділено вибору оптимального технологічного стеку, здатного забезпечити високу продуктивність, гнучкість, легку масштабованість та ефективну інтеграцію з сучасними сервісами штучного інтелекту. При формуванні архітектури враховувалися специфічні вимоги до систем взаємодії з LLM, зокрема потреба в обробці великих обсягів векторних даних, забезпечення швидкого відгуку сервера, а також підтримка асинхронної взаємодії з API моделей.

У підсумку, обраний технологічний стек забезпечує оптимальну платформу для побудови сучасної системи роботи з великими мовними моделями. Його використання дозволяє досягти високої продуктивності, забезпечити гнучкість і розширюваність архітектури, а також реалізувати складні сценарії інтерактивної взаємодії з LLM у реальному часі. Завдяки цьому система може ефективно обробляти як текстові, так і мультимодальні дані, забезпечуючи широкий спектр можливостей для кінцевих користувачів.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		29

1.7.3 Програмні вимоги

Для коректної роботи серверної частини системи необхідно забезпечити відповідне програмне середовище, яке включає операційну систему, інтерпретатор мови програмування, бібліотеки, фреймворки та допоміжні інструменти. Основні програмні вимоги:

1. Операційна система Linux (рекомендовано Ubuntu 22.04 LTS) можлива підтримка Windows 10/11, macOS для розробки.

2. Мова програмування Python 3.10.

3. Основні бібліотеки та фреймворки, FastAPI - для створення RESTful API; Uvicorn - ASGI сервер для запуску FastAPI; OpenCV (cv2) - для обробки відео та виділення кадрів ; Streamlit - для створення інтерактивного фронтенду; Requests - для взаємодії з зовнішніми API; python-dotenv - для роботи з конфігураційними файлами .env; pytest - для автоматизованого тестування.

4. Інтеграції та зовнішні сервіси, OpenAI API - для використання мовних моделей YouTube API (опціонально) - для завантаження відео з YouTube.

5. Інструменти для розгортання та супроводу, Docker - для контейнеризації застосунків; Docker Compose - для оркестрації декількох сервісів; Git - для контролю версій; Swagger UI / ReDoc - для автоматичної генерації документації API. Додаткові вимоги, встановлені драйвери для роботи з відео (ffmpeg, libsm6, libxext6 - для OpenCV), наявність віртуального середовища Python (venv, virtualenv) для ізоляції залежностей.

Дотримання зазначених програмних вимог гарантує стабільну роботу серверної частини системи, спрощує розгортання, тестування та подальший супровід проєкту.

Для зручності всі залежності перелічені у файлі `requirements.txt`, що дозволяє швидко встановити необхідне програмне забезпечення за допомогою однієї команди.

					КГ 08. 27 001. 00 ДП ПЗ	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		30

1.7.4 Вимоги до безпеки

Шифрування. Усі передавані дані мають бути захищеними за допомогою протоколу HTTPS із використанням сучасної версії транспортного рівня безпеки (TLS 1.2 або вище). Це забезпечує конфіденційність і цілісність інформації під час її передавання через мережу.

2.Автентифікація та авторизація. Для забезпечення надійної автентифікації та контролю доступу застосовуються сучасні галузеві стандарти, зокрема OAuth 2.0 для авторизації та JSON Web Token (JWT) для передачі підтверджених і захищених токенів користувачів. Це дозволяє ефективно керувати сесіями користувачів та захищати доступ до ресурсів системи.

3.Контроль доступу. Права доступу до системних ресурсів реалізуються на основі ролей користувачів за моделлю Role-Based Access Control (RBAC). Такий підхід дозволяє гнучко визначати рівні доступу для різних категорій користувачів та мінімізує ризики несанкціонованого доступу.

4.Резервне копіювання. Для забезпечення збереження та відновлення критично важливих даних у разі збоїв або інцидентів безпеки, у системі передбачено регулярне автоматизоване створення резервних копій. Резервні копії виконуються відповідно до затвердженої політики резервного копіювання та зберігаються у захищених середовищах.

5.Моніторинг безпеки. У систему інтегруються інструменти для постійного моніторингу безпеки, які дозволяють виявляти потенційні загрози та оперативно реагувати на підозрілу активність. Це включає використання систем виявлення вторгнень (IDS), систем сповіщення про інциденти, а також механізмів збору та аналізу логів.

1.7.5 Технологічний стек для системи роботи з LLM

Для реалізації системи, що інтегрує великі мовні моделі (LLM, Large Language Models), використовується сучасний технологічний стек, який забезпечує ефективну обробку запитів, масштабованість, безпеку та зручність розробки.

					КГ 08. 27 001. 00 ДП ПЗ	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		31

Основні компоненти технологічного стеку:

1. Мова програмування Python 3.10+ - основна мова для серверної логіки, обробки даних та інтеграції з LLM.

2. Фреймворки та бібліотеки: FastAPI - високопродуктивний фреймворк для створення RESTful API, дозволяє швидко інтегрувати LLM у серверну частину. OpenAI Python SDK - офіційна бібліотека для взаємодії з OpenAI API та іншими LLM-провайдерами. Pydantic - для типізації, валідації та серіалізації даних. Requests / httpx - для виконання HTTP-запитів до зовнішніх API. Streamlit - для створення інтерактивного користувацького інтерфейсу, який дозволяє взаємодіяти з LLM у реальному часі.

3. Інфраструктура та розгортання, Docker - для контейнеризації застосунків, що спрощує розгортання та масштабування. Docker Compose - для одночасного запуску декількох сервісів (бекенд, фронтенд, сховище). Git - для контролю версій та командної розробки. CI/CD (наприклад, GitHub Actions) - для автоматизації тестування та розгортання.

4. Зовнішні сервіси та API, OpenAI API - для доступу до GPT-3, GPT-4 та інших LLM.

YouTube API (опціонально)-для завантаження відео, якщо потрібна обробка відеоконтенту.

5. Сховище даних-Файлова система (storage/videos, storage/frames)-для зберігання відео та кадрів. Хмарні сервіси (AWS S3, EFS) - для масштабованого зберігання та резервного копіювання (за потреби).

6. Документація та тестування: Swagger UI/ReDoc - для автоматичної генерації та тестування API, pytest - для написання та запуску автоматизованих тестів.

Обраний технологічний стек дозволяє ефективно інтегрувати LLM у серверну архітектуру, забезпечує гнучкість, масштабованість та простоту підтримки системи. Використання сучасних інструментів і бібліотек гарантує швидку розробку, легку інтеграцію нових функцій та високу якість кінцевого продукту.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		32

1.7.6 Розробка сервера та інтеграція

Розробка серверної частини системи передбачає створення надійної, масштабованої та легко підтримуваної архітектури, яка забезпечує обробку відео, інтеграцію з великими мовними моделями (LLM) та взаємодію з користувачем через API та фронтенд. Основні етапи розробки сервера:

1. Створення структури проєкту це організація директорій відповідно до принципів чистої архітектури: виділення окремих модулів для контролерів, сервісів, моделей, схем, утиліт та сховища. Налаштування конфігураційних файлів та середовища (файл .env, налаштування Docker).

2. Реалізація RESTful API з використанням FastAPI:

1. Опис основних ендпоінтів для завантаження відео, отриманих результатів аналізу, інтеграції з зовнішніми сервісами.

2. Валідація вхідних та вихідних даних за допомогою Pydantic-схем.

3. Автоматична генерація документації API (Swagger UI, ReDoc).

3. Розробка сервісів для обробки відео та кадрів

1. Реалізація логіки завантаження, збереження та обробки відеофайлів.

2. Виділення кадрів із відео з використанням OpenCV.

3. Зберігання результатів у файловій системі або хмарному сховищі.

4. Інтеграція з LLM через OpenAI API

1. Розробка окремого сервісу для взаємодії з OpenAI API.

2. Передача текстових або візуальних даних для аналізу, генерації описів, класифікації тощо.

4. Обробка та збереження результатів, отриманих від LLM.

5. Забезпечення взаємодії з фронтендом (Streamlit)

1. Реалізація API для обміну даними між сервером і Streamlit-додатком.

2. Тестування інтеграції та забезпечення коректної роботи всіх компонентів.

6. Контейнеризація та розгортання

1. Створення Dockerfile та docker-compose.yml для ізольованого запуску серверної частини та супутніх сервісів.

2. Підготовка до розгортання у хмарному середовищі (AWS, Google Cloud) або на локальному сервері.

Особливості інтеграції. Кожен компонент (обробка відео, інтеграція з LLM) реалізовано у вигляді окремого сервісу або модуля, що спрощує супровід та масштабування.

1.7.7 Реалізація сервера

Серверна частина системи реалізує із застосуванням сучасного стеку технологій на основі мови програмування Python. В якості основного веб-фреймворку використовую FastAPI, що дозволяє створювати високопродуктивні та масштабовані RESTful API, орієнтовані на обробку відео, інтеграцію з великими мовними моделями (LLM) та взаємодію з користувачем. Етапи реалізації:

1. Ініціалізація проекту

Формування структури директорій: app/api, app/services, app/models, app/core, app/utils, storage/. Створення віртуального середовища та встановлення залежностей із файлу requirements.txt.

2. Розробка API з використанням FastAPI

Створюю основні маршрути (ендпоінти) для завантаження відео, обробки кадрів, запуску аналізу та інтеграції з LLM. Валідація запитів із використанням Pydantic-схем. Генерація інтерактивної документації API (Swagger UI, ReDoc).

3. Реалізація відеосервісів

1. Завантаження відеофайлів через API та збереження відео у файльовій системі (storage/videos/), виділення кадрів із використанням бібліотеки OpenCV та збереження їх у storage/frames/.

4. Інтеграція з OpenAI API (LLM)

Розроблюю окремий сервісний модуль для надсилання текстових або візуальних запитів до OpenAI API та обробка та збереження результатів аналізу для подальшого використання.

5. Контейнеризація - створення Dockerfile для серверної частини, та налаштування docker-compose для одночасного запуску бекенду, фронтенду та супутніх служб.

Розроблений сервер є надійною, масштабованою та гнучкою платформою, здатною ефективно обробляти відеоконтент, здійснювати інтеграцію з LLM та забезпечувати взаємодію з користувачем через сучасні API. Обрана архітектура сприяє легкому розширенню функціональності та адаптації системи до нових задач і сценаріїв використання.

1.8 Програмна реалізація застосунку

У сучасних умовах автоматизація збору та аналізу відеоданих відіграє ключову роль у розвитку технологій машинного навчання та комп'ютерного зору. Розроблений застосунок призначений для обробки відеоматеріалів і аналізу окремих кадрів з метою вивчення поведінкових патернів курей та інших тварин на основі відеоспостереження.

У рамках демонстрації функціональних можливостей системи було оброблено відео з участю курей та інших тварин. Водночас архітектура рішення є універсальною та здатна адаптуватися до обробки відеозаписів із будь-якими видами тварин, що відкриває широкі можливості для застосування в зоологічних дослідженнях, тваринництві, агропромисловості, ветеринарній медицині та суміжних сферах.

Архітектура застосунку побудована за модульним принципом, що забезпечує гнучкість, масштабованість та можливість подальшого розширення. До складу системи входять модулі збору та обробки відеоданих, аналітичний модуль для поведінкового аналізу, а також інтерфейс користувача для взаємодії з результатами.

Система використовує сучасні технології машинного навчання, комп'ютерного зору, обробки відеопотоків та великі мовні моделі (LLM), що дозволяє не лише детально аналізувати відеокадри, а й автоматично формувати звіти й рекомендації на основі виявлених поведінкових шаблонів.

Компоненти системи взаємодіють між собою через внутрішні RESTful API, що забезпечує ефективну обробку великих обсягів даних у реальному часі. Завдяки використанню контейнеризації (Docker) та можливості оркестрації (наприклад, за

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		35

допомогою Kubernetes) система легко розгортається, масштабується та підтримується відповідно до потреб користувача або організації.

1.8.1 Загальна структура програми

У цьому розділі представлено загальну архітектуру розробленої системи, що забезпечує автоматизовану обробку відеоданих з метою аналізу поведінки тварин із використанням сучасних технологій комп'ютерного зору та великих мовних моделей (LLM). Архітектура побудована за модульним принципом та включає у себе взаємопов'язані компоненти, кожен з яких виконує окремі функціональні завдання.

Основними елементами системи є: інтерфейс користувача (Frontend), серверна частина (Backend), а також зовнішні інтелектуальні сервіси - великі мовні моделі та спеціалізовані агенти (Agents).

Взаємодія між компонентами системи організована таким чином, щоб забезпечити ефективний аналіз відео, виявлення аномалій у поведінці тварин, генерацію текстових описів та автоматичне формування аналітичних звітів для підтримки процесів моніторингу та прийняття рішень.

1.Фронтенд (Streamlit)

Фронтенд реалізований з використанням бібліотеки Streamlit та взаємодіє з бекендом через API. Він надсилає запити на обробку відео та отримує відповіді - як результати аналізу, так і текстові відповіді на запитання користувача.

2.Бекенд (Python).

Бекенд розроблений на базі мови програмування Python та складається з двох ключових компонентів:

Розбиття відео на кадри - модуль, що здійснює автоматичне розділення відео на окремі кадри з заданим часовим інтервалом.

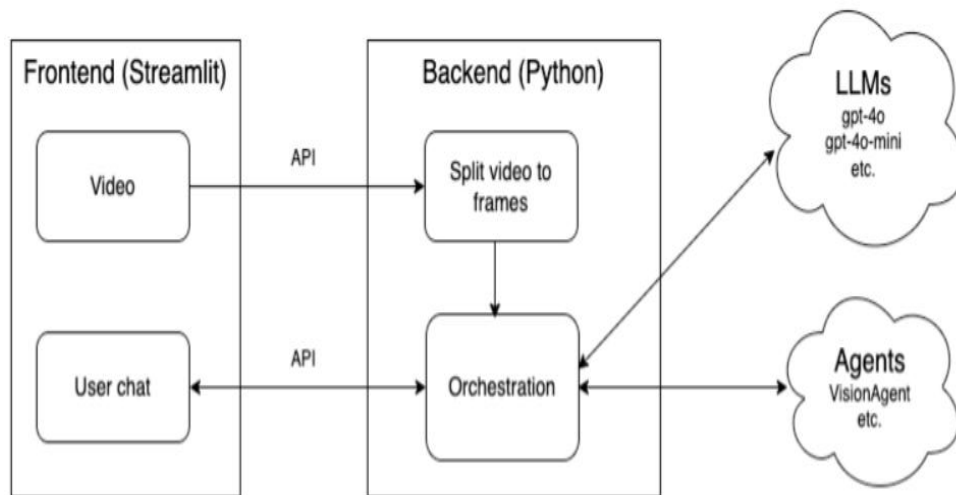


Рисунок 1.6. Структура програми

Оркестратор - центральний логічний модуль, який координує роботу усієї системи. Він приймає запити від фронтенду, викликає відповідні сервіси (модулі обробки відео, LLM, агенти тощо), формує результати аналізу та передає їх назад на фронтенд. Оркестратор також здійснює виклики до зовнішніх сервісів для виконання таких завдань:

1. аналіз зображень;
2. генерація текстових відповідей;
3. виявлення аномалій.
3. Великі мовні моделі (LLM)

Великі мовні моделі використовуються для виконання наступних завдань:

1. генерація тексту;
2. створення описів та аналіз зображень;
3. відповіді на запитання користувачів.

Бекенд викликає LLM для обробки текстових запитів та зображень, отриманих з відео.

4. Агенти (VisionAgent)

VisionAgent - це спеціалізовані модулі або мікросервіси, які виконують специфічні функції, пов'язані з обробкою та аналізом зображень та відео. Ці агенти застосовують методи комп'ютерного зору, машинного навчання та інші алгоритми для розв'язання завдань в області моніторингу поведінки тварин.

Основні функції VisionAgent:

Виявлення аномалій - аналіз кадрів для виявлення нетипових моделей поведінки, таких як ознаки стресу чи незвичайна моторика тварин.

Класифікація об'єктів - використання алгоритмів машинного навчання для класифікації об'єктів на кадрах відео, визначення ключових характеристик та розрізнення різних елементів сцени.

Аналіз поведінки - застосування комп'ютерного зору для оцінки поведінки курей, підтримка виявлення ознак стресу, агресії або відхилень від природної активності.

Інтеграція з іншими сервісами - VisionAgent спроектовано таким чином, щоб він міг безперешкодно інтегруватися з бекендом системи та іншими компонентами штучного інтелекту (LLM, оркестраційний модуль), забезпечуючи комплексний аналіз даних.

Обробка в реальному часі-VisionAgent оптимізовано для високої продуктивності та здатен здійснювати обробку відеокадрів у режимі реального часу, що дозволяє своєчасно здійснювати моніторинг і ухвалювати управлінські рішення в автоматизованих системах.

Завдяки цим функціям VisionAgent виступає ключовим компонентом для створення надійних рішень з автоматизованого аналізу відео у таких сферах, як моніторинг поведінки курей та забезпечення добробуту тварин у птахівництві.

1.8.2 Взаємодія компонентів системи

На рисунку 1.7. представлений скріншот структури проекту. Структура проекту розроблена за принципами модульної архітектури, що забезпечує чітке розділення відповідальностей та спрощує масштабування і підтримку коду.

Основні компоненти проекту: app/ - основна серверна частина додатку;
api/ - реалізація REST API, що включає контролери, залежності та маршрути.
core/ - модулі конфігурації додатку та обробки виняткових ситуацій.
models/ - опис моделей бізнес-логіки та схем для валідації даних.
services/ - бізнес-логіка обробки відеофайлів і кадрів (frames).

utils/ - допоміжні утиліти, що використовуються в різних модулях проєкту.

frontend/ - інтерактивний користувацький інтерфейс, реалізований за допомогою бібліотеки Streamlit, що дозволяє здійснювати завантаження відео, перегляд оброблених даних та взаємодію з результатами роботи системи.

tests/ - модуль автоматизованого тестування, що забезпечує контроль якості коду та коректності бізнес-логіки.

storage/ - тимчасове сховище для вхідних відеофайлів та кадрів, що генеруються

Інфраструктурні файли:

.env.example - шаблон файлу змінних середовища.

Dockerfile - конфігурація контейнера для розгортання додатку.

docker-compose.yml - налаштування контейнерної інфраструктури.

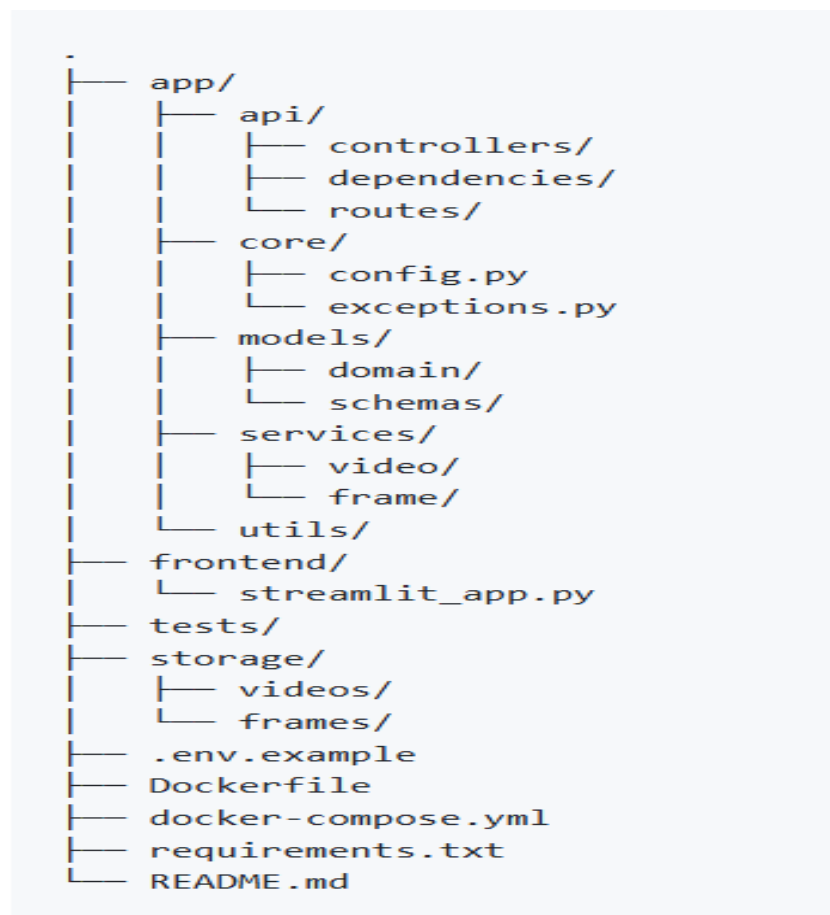


Рисунок 1.7. Структура проєкту

requirements.txt - перелік програмних залежностей проєкту.

README.md - технічна документація та інструкція з використання додатку.

Таким чином, загальна архітектура системи забезпечує ефективну взаємодію між інтерфейсом користувача, серверною логікою та зовнішніми інтелектуальними сервісами, що дозволяє реалізувати повний цикл автоматизованого аналізу відеоданих. Використання модульного підходу та сучасних технологій штучного інтелекту дає змогу створювати гнучке, масштабоване рішення, яке може бути адаптоване для різноманітних сценаріїв аналізу поведінки тварин.

Система надає користувачам можливість завантажувати відео як з онлайн-відеосервісів, так і з локальних файлів, обирати мову взаємодії та використовувати великі мовні моделі (LLM) для подальшої обробки даних. Крім того, у системі реалізовано можливість налаштування параметрів сегментації відео, що дає змогу користувачеві задати інтервал генерації кадрів (зображень, вилучених з відео), тобто визначити кількість секунд між сусідніми кадрами.

1.9 Приклад роботи застосунку

У даній підсистемі реалізовано веб-інтерфейс користувача на основі фреймворку Streamlit, що забезпечує інтерактивну взаємодію із серверною частиною системи. Інтерфейс дозволяє здійснювати завантаження відео, налаштування параметрів обробки, вибір мовної моделі та перегляд результатів аналізу. Головна стонка програми показана на рисунку 1.8. Розглянемо принципи роботи системи.

Ліва панель: Опції завантаження, на рисунку 1.8.

1.Вибір способу завантаження: користувач може обрати джерело відео для завантаження - локальний файл або посилання на відео з YouTube.

2.Інтервал кадрів (секунди): задається інтервал у секундах, з яким система автоматично генеруватиме кадри з відео.

3.Вибір мови: дозволяє користувачеві обрати мову для спілкування з системою або для подальшого аналізу даних.

4.Вибір моделі: дає змогу обрати велику мовну модель (LLM), яка буде задіяна для обробки текстової та мультимодальної інформації.

Права панель: Панель обробки відео

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		40

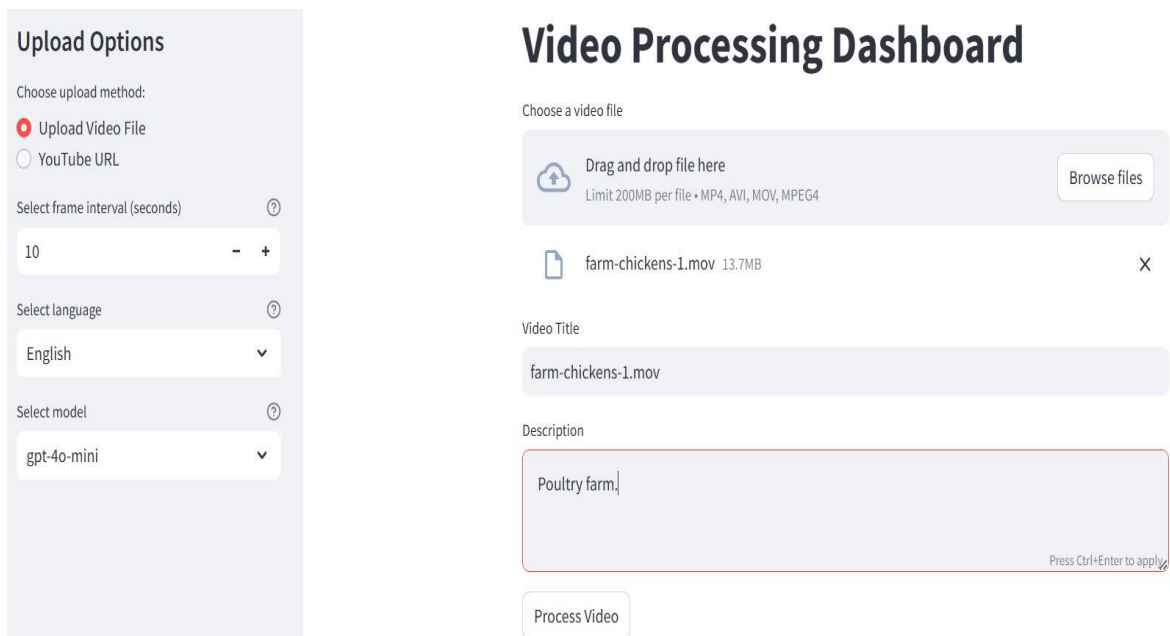


Рисунок 1.8. Скріншот головної сторінки

Права частина інтерфейсу (Dashboard) використовується для моніторингу процесу обробки відео. Тут відображаються результати аналізу, проміжні дані та підсумкові висновки, отримані у ході обробки відео та взаємодії з LLM і спеціалізованими агентами системи.

Однією з основних функцій розробленої системи є автоматизоване виділення кадрів (frame extraction) із завантажених відеофайлів, надано на рисунку 1.6. Цей процес дозволяє отримати набір зображень, які у подальшому використовуються для аналізу поведінки тварин за допомогою комп'ютерного зору та великих мовних моделей.

1. Інтерфейс для завантаження відео

На початковому етапі користувач завантажує відеофайл через інтерактивний інтерфейс (див. рис. 1.6). Підтримуються формати MP4, AVI, MOV, MPEG4, з обмеженням на розмір файлу до 200 МБ. Завантаження можливе як через кнопку Browse files, так і шляхом перетягування файлу (Drag and drop).

Video Processing Dashboard

Choose a video file



Drag and drop file here

Limit 200MB per file • MP4, AVI, MOV, MPEG4

Browse files

Processing Status

Status: completed - Processing completed. 3 frames extracted

Extracted Frames



Frame 1



Frame 2



Frame 3

Рисунок 1.9. Скріншот виділення кадрів із відео

2. Статус обробки

Після завантаження відеофайлу система автоматично переходить до етапу його обробки. На панелі Processing Status відображається хід виконання процесу. Статус "completed" свідчить про успішне завершення обробки та про кількість виділених кадрів (у даному прикладі -3 кадри).

4. Алгоритм виділення кадрів

Відеофайл розбивається на окремі кадри з використанням модуля оркестрації. Користувач заздалегідь визначає інтервал між кадрами (у секундах) під час налаштування параметрів сегментації відео. Наприклад, при інтервалі у 5 секунд з 30-секундного відео буде виділено 6 кадрів. Для кожного зазначеного інтервалу система автоматично "захоплює" відповідний кадр і зберігає його у форматі зображення.

5. Результати виділення

У секції Extracted Frames відображаються отримані кадри у вигляді мініатюр

					КГ 08. 27 001. 00 ДП ПЗ	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		42

(див. рис. 1.6). У прикладі було згенеровано три кадри -Frame 1, Frame 2, Frame 3. Ці кадри зберігаються у системі для подальшого використання у модулях аналізу зображень (VisionAgent) та генерації текстових описів (LLM).

Переваги підходу

1. Виділення кадрів здійснюється автоматично без участі користувача після завантаження відео.
2. Підтримується налаштування гнучкого інтервалу для генерації кадрів.
3. Система оптимізована для обробки кадрів у режимі реального часу.
4. Отримані кадри одразу готові для інтеграції у подальші етапи аналізу.



Рисунок 1.10. Скріншот кадру з обробки відео

Завдяки автоматизованому процесу виділення кадрів система забезпечує ефективну підготовку вхідних даних для подальшого використання у задачах аналізу поведінки тварин.

Таким чином, модуль обробки відео забезпечує автоматизоване та ефективне виділення кадрів з відеоматеріалів, що створює основу для подальшого аналізу зображень за допомогою алгоритмів комп'ютерного зору та великих мовних моделей.

Гнучкість налаштувань інтервалу сегментації та інтеграція у загальну архітектуру системи дозволяють адаптувати процес обробки до різних сценаріїв застосування.

Після завершення етапу виділення кадрів з відео кожен кадр передається на подальшу обробку, яка виконується за участі великих мовних моделей (LLM) та спеціалізованих агентів комп'ютерного зору. Даний процес дозволяє автоматично генерувати опис зображення, що значно спрощує аналіз контенту для користувача.



Frame 3



Опиши що зображено на картинці?



На картинках зображено багато курчат, які знаходяться у приміщенні для вирощування пташиного поголів'я. Курчата predominantly білі, і вони розташовані на підлозі, яка покрита землею або стружкою. У центрі кадру видно червоні годівниці та поїлки, які використовуються для годування та забезпечення водою птахів. Приміщення освітлене природним світлом з вікон, а також має синій дах або накриття. Картинки демонструють активну та спокійну атмосферу, де курчата пересуваються в пошуках їжі та води.

Рисунок 11. Скріншот обробки фрагменту відео

На рисунку 1.11 представлено приклад обробки одного з кадрів (Frame 3), де система пропонує користувачеві функціонал "Опиши що зображено на картинці?". Користувач має можливість ініціювати процес генерації текстового опису для обраного кадру.

Процес обробки включає такі етапи:

1. Попередня підготовка кадру:

Відібраний кадр передається на сервер для попередньої обробки (масштабування, нормалізація, приведення у відповідний формат для взаємодії з LLM).

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		44

2.Інтеграція з великою мовною моделлю: Опрацьований кадр передається як вхідний параметр до мовної моделі, яка здійснює мультимодальний аналіз (зображення + запит користувача у вигляді тексту).

3.Генерація текстового опису:

Мовна модель аналізує вміст зображення та генерує осмислений текстовий опис сцени.

У наведеному прикладі модель ідентифікувала, що на зображенні показано велику кількість курчат у приміщенні для вирощування пташиного поголів'я, описала особливості середовища, елементи інфраструктури (червоні годівниці, природне освітлення) та зафіксувала загальну поведінку птахів.

4.Вивід результату:

Згенерований текст автоматично виводиться у графічний інтерфейс (Streamlit) під обраним кадром для зручного ознайомлення користувача з результатами.

Переваги підходу:

Автоматизація: користувач не потребує спеціальних знань для аналізу зображень - система автоматично генерує зрозумілий опис.

Мультимодальність: LLM дозволяє поєднувати зображення і текст, що забезпечує якісний контекстуальний аналіз.

Гнучкість: система може бути застосована до будь-яких кадрів, незалежно від їхнього змісту чи джерела.

Швидкість: обробка виконується у режимі, наближеному до реального часу.

На рисунку 1.12 наведено приклад роботи програми по підрахунку тварин на прикладі обробки відео з курами.



Рисунок 1.12. Скріншот роботи системи по підрахунку курей

Завдяки такому підходу користувач отримує зрозумілу текстову інтерпретацію складної візуальної інформації, що значно полегшує аналіз поведінки тварин та стану середовища у птахівничих господарствах.

Таким чином, модуль аналізу кадрів дозволяє автоматизовано формувати текстові описи візуального контенту, що значно спрощує процес інтерпретації відеоданих. Використання мультимодальних можливостей великих мовних моделей забезпечує якісний та контекстуально точний аналіз зображень у системі.

На рисунку 1.13 представлено приклад інтерактивної взаємодії користувача із системою під час аналізу відео. Демонструється процес завантаження відеофайлу, налаштування параметрів обробки та генерації текстових відповідей за допомогою великої мовної моделі (LLM).

1.Завантаження відео та налаштування параметрів

У лівій панелі Upload Options користувач обрав метод завантаження - локальний відеофайл (Upload Video File), встановив інтервал генерації кадрів 2 секунди, мову спілкування - українська та обрав для обробки модель gpt-4o.

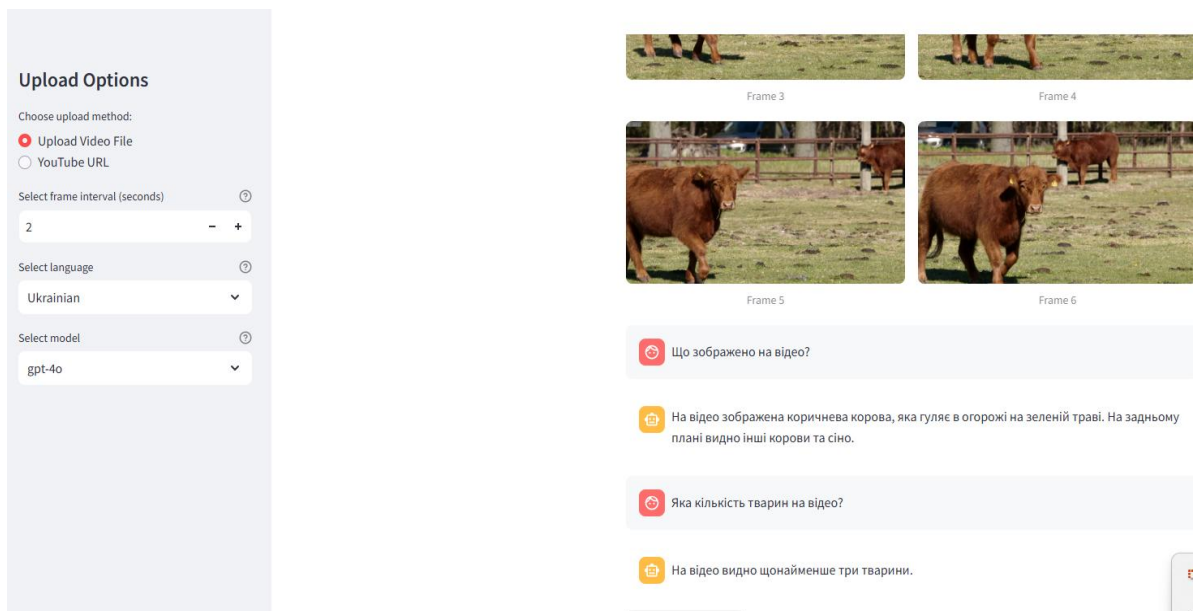


Рисунок 1.13. Скріншот обробки фрагменту відео з тваринами

2. Процес аналізу кадрів

Система автоматично розбила відео на окремі кадри з кроком у 2 секунди. Витягнуті кадри відображено у правій частині інтерфейсу (Frame 3 - Frame 6), де користувач може переглядати вміст зображень.

3. Генерація опису та відповіді на запитання

Після генерації кадрів користувач взаємодіє з системою у діалоговому режимі. Зокрема, на основі обраного кадру було сформульовано такі запити:

1. Що зображено на відео?

Відповідь моделі: На відео зображена коричнева корова, яка гуляє в огорожі на зеленій траві. На задньому плані видно інші корови та сіно.

2. Яка кількість тварин на відео?

Відповідь моделі: На відео видно щонайменше три тварини.

Переваги інтерактивного аналізу

Користувач має змогу не лише переглядати згенеровані кадри, але й задавати довільні текстові запитання щодо вмісту кадрів.

Система автоматично формує осмислені, контекстуально точні текстові відповіді за допомогою великої мовної моделі.

Мультимодальна взаємодія дозволяє об'єднати аналіз зображень та генерацію текстових описів у єдиному інтерфейсі.

Таким чином, представлений приклад демонструє інтеграцію автоматизованого аналізу відеоконтенту та можливостей великих мовних моделей у процесі роботи системи.

Таким чином, інтерактивний механізм аналізу кадрів із застосуванням великих мовних моделей забезпечує користувачеві зручний спосіб отримання інформативних описів та відповідей на запитання щодо вмісту відео. Це значно розширює функціональні можливості системи та підвищує ефективність аналізу відеоданих у завданнях моніторингу поведінки тварин.

У лівій частині інтерфейсу (Upload Options) користувач обирає спосіб завантаження відео - у даному випадку використовується завантаження локального файлу (Upload Video File).

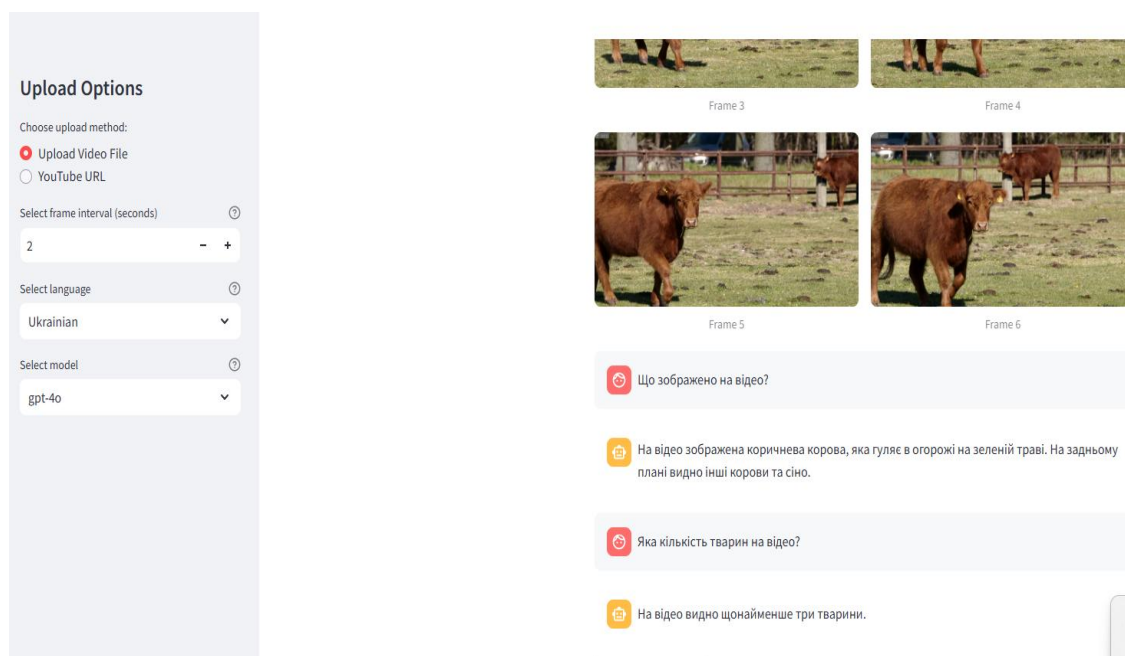


Рисунок 1.14. Скріншот відео по підрахунку тварин

Для розбиття відео на кадри задано інтервал 2 секунди (Select frame interval), обрано мову аналізу - українська (Select language), а для обробки текстових запитів і генерації описів використано модель gpt-4o.

1. Аналіз кадрів

Після обробки відео система автоматично виділяє кадри (на прикладі -Frame 3, Frame 4, Frame 5, Frame 6), які відображаються у центральній частині інтерфейсу.

2. Взаємодія користувача з LLM

Інтерактивна панель дозволяє користувачеві ставити текстові запитання стосовно вмісту кадрів.

У прикладі наведено такі запити:

1. Що зображено на відео?

Відповідь системи: на відео зображена коричнева корова, яка гуляє в огорожі на зеленій траві. На задньому плані видно інші корови та сіно.

2. Яка кількість тварин на відео?

Відповідь системи: на відео видно щонайменше три тварини.

Основні переваги:

1. Система дозволяє інтерактивно аналізувати будь-який кадр з відео за допомогою LLM, що дає змогу формувати осмислені текстові описи вмісту.

2. Підтримується гнучкий вибір параметрів аналізу: інтервал кадрів, мова спілкування, модель LLM.

3. Взаємодія з системою реалізована у простому для користувача інтерфейсі, що не потребує спеціальної підготовки.

Демонстрований приклад підтверджує здатність системи інтегрувати комп'ютерний зір та мультимодальні мовні моделі для автоматизованого аналізу відеоданих у реальному часі.

За результатами проведених експериментів було підтверджено працездатність та ефективність розробленої системи для автоматизованого аналізу відеоданих за участі технологій комп'ютерного зору та великих мовних моделей (LLM). Система забезпечує стабільну обробку відео з можливістю гнучкого налаштування параметрів сегментації, автоматичного виділення кадрів та інтерактивного аналізу їх вмісту у текстовому вигляді.

Під час експериментів було успішно продемонстровано генерацію якісних описів кадрів та коректні відповіді на запитання користувача щодо вмісту відео. Інтерфейс системи є інтуїтивно зрозумілим та зручним у використанні. Отримані результати свідчать про практичну придатність створеного рішення для

задач моніторингу поведінки тварин та аналізу візуальних даних у сільському господарстві й суміжних сферах.

У перспективі можливим напрямом розвитку системи є розширення її функціональних можливостей, оптимізація продуктивності та інтеграція додаткових моделей комп'ютерного зору для поглибленого аналізу відеоконтенту.

Застосування технологій штучного інтелекту та великих мовних моделей створює нові можливості для автоматизації, підвищення ефективності та сталого розвитку тваринництва. Подальше удосконалення інфраструктури, якісна підготовка даних та інтеграція інтелектуальних систем у виробничі процеси відкривають перспективи для широкого впровадження цих рішень у практичну діяльність відповідних підприємств.

1.10 Підключення та використання OpenAI API

1.10.1 Налаштування доступу до OpenAI API

Для інтеграції з сервісами OpenAI було виконано наступні кроки:

1. Зареєстровано обліковий запис на офіційному веб-сайті OpenAI за адресою: <https://platform.openai.com>.
2. Після реєстрації створено новий проєкт, у межах якого згенеровано API-ключ, що надає доступ до функціоналу OpenAI API.
3. API-ключ було отримано у вкладці "API Keys" в особистому кабінеті користувача. Цей ключ використовується для автентифікації запитів до OpenAI API.

Для забезпечення безпеки:

1. API-ключ та інші чутливі дані не зберігаються у відкритому коді;
2. Використовую стандартний механізм NestJS для керування конфігураційними змінними;
3. Ключі зберігаються у файлі .env, який додано до .gitignore для запобігання потраплянню до публічного репозиторію.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		50

```

example
PORT=3000

OPEN_AI_KEY=sk- ...

#embedding model
modelName=text-embedding-ada-002

MONGODB_URI=mongodb://localhost:27017/mongodb

PINECONE_INDEX=
PINECONE_API_KEY=
PINECONE_ENVIRONMEN=gcp-starter

```

Рисунок 1.12. Скриншот файлу env.example

1.10.2 Файли .env.example - шаблону змінних середовища

.env.example є шаблонним файлом, який демонструє структуру конфігураційного файлу .env. Цей файл використовується у програмних проєктах для зберігання змінних середовища - параметрів, що можуть змінюватися залежно від середовища запуску: розробка, тестування, продакшн.

Опис змінних:

PORT=3000, вказує порт, на якому працює застосунок. У цьому випадку - 3000.

OPEN_AI_KEY=sk-...

API-ключ для авторизованого доступу до сервісів OpenAI. Значення "sk-..." потрібно замінити на ваш реальний ключ.

#embedding model

Це коментар, який пояснює, що наступна змінна стосується налаштування моделі вбудовування. modelName=text-embedding-ada-002

Назва моделі OpenAI, яка використовується для генерації embedding-векторів. MONGODB_URI=mongodb://localhost:27017/mongodb

Рядок підключення до локальної бази даних MongoDB на порту 27017 з назвою бази "mongodb". PINECONE_INDEX=ai-course

Назва індексу у базі Pinecone, що використовується для зберігання або пошуку векторних даних. PINECONE_API_KEY=

Поле для введення вашого API-ключа Pinecone.

PINECONE_ENVIRONMENT=gcp-starter

Середовище, у якому запущено Pinecone. Значення "gcp-starter" вказує на початковий кластер у Google Cloud.

Примітка: Файл. env.example слугує шаблоном для створення конфігураційного файлу. env. Після копіювання шаблону необхідно ввести реальні значення для змінних. Файл. env не слід додавати до репозиторію, оскільки він може містити конфіденційну інформацію.

```

└─ openai
  └─ dto
     | TS completion.dto.ts
     TS openai.controller.spec.ts
     TS openai.controller.ts
     TS openai.module.ts
     TS openai.service.spec.ts
     TS openai.service.ts

```

Рисунок 1.13. Скриншот структури папки модуля OpenAI

Зображення показує структуру папки модуля OpenAI у проекті, розробленому з використанням TypeScript та NestJS. Ось детальний опис файлів, представлених у папці:

1. Папка dto (Data Transfer Object)

completion.dto.ts: Цей файл містить клас або інтерфейс, який визначає структуру даних для запитів генерації тексту в OpenAI API. DTO використовується для передачі даних між клієнтом і сервером, забезпечуючи валідацію та типізацію вхідних даних для API.

2. Файли специфікації та імплементації:

`openai.controller.spec.ts`: Цей файл містить юніт-тести для контролера OpenAI. Тести перевіряють, що контролер коректно обробляє вхідні запити та відповіді.

`openai.controller.ts`: Файл контролера, який обробляє HTTP-запити до OpenAI API. Контролер приймає запити, передає дані до сервісного шару для обробки та повертає відповіді клієнту.

`openai.module.ts`: Модуль, що інкапсулює всі компоненти OpenAI (сервіси, контролери) і можливі імпорти з інших модулів. Модуль забезпечує ізоляцію та організацію функціоналу, пов'язаного з OpenAI.

`openai.service.spec.ts`: Файл з тестами для сервісу OpenAI. Ці тести забезпечують перевірку логіки, що використовується сервісом для взаємодії з API OpenAI.

`openai.service.ts`: Сервіс, що містить бізнес-логіку для взаємодії з OpenAI API. Сервіс виконує запити до API, обробляє відповіді та передає їх до контролера.

Ця структура папок і файлів демонструє чітке відділення відповідальності між різними компонентами додатку, дозволяючи легше управління кодом і підтримку.

1. Імпорт модулів, код починається з імпорту необхідних модулів та компонентів:

`NestFactory`: Використовується для створення інстанції додатку NestJS.

`AppModule`: Це основний модуль додатку, який інтегрує всі інші модулі.

`ConfigService`: Модуль для завантаження та використання конфігураційних змінних.

`SwaggerModule` і `DocumentBuilder`: Модулі, використовувані для створення документації API за допомогою OpenAPI (раніше відомого як Swagger).

2. Функція `bootstrap`, це асинхронна функція, яка ініціалізує та запускає додаток: створення екземпляра додатку з використанням `AppModule`.

Отримання екземпляра `ConfigService` через метод `app.get()`.

Використання `ConfigService` для отримання номеру порту, на якому має слухати додаток.

```

main.ts > ...
import { NestFactory } from '@nestjs/core';
import { AppModule } from './app.module';
import { ConfigService } from '@nestjs/config';
import { SwaggerModule, DocumentBuilder } from '@nestjs/swagger';

async function bootstrap() {
  const app = await NestFactory.create(AppModule);
  const configService = app.get(ConfigService);
  const port = configService.get<number>('port');

  const config = new DocumentBuilder()
    .setTitle('Diplom test API')
    .setDescription('Diplom test API')
    .setVersion('1.0')
    .addTag('test')
    .build();
  const document = SwaggerModule.createDocument(app, config);
  SwaggerModule.setup('api', app, document);

  await app.listen(port);
}
bootstrap();

```

Рисунок 1.14. Скриншот коду конфігурація Swagger модуля

2. Налаштування Swagger

DocumentBuilder, конфігуратор для Swagger, в якому встановлюються заголовок, опис, версія документації та теги.

Створення документа Swagger з використанням SwaggerModule.createDocument(), передаючи туди додаток та конфігурацію Swagger.

Налаштування Swagger UI через SwaggerModule.setup(), вказуючи шлях до документації та асоціюючи його з додатком.

3. Запуск додатку.

Додаток запускається на вказаному порту з використанням app.listen(), що дозволяє обслуговувати вхідні HTTP-запити.

5. Виклик функції bootstrap

Виклик bootstrap() розпочинає виконання функції та ініціацію додатку.

Цей код є типовим прикладом налаштування додатку з інтеграцією Swagger для документації API, що робить його доступним для розробників і сторонніх сервісів для легкого взаємодії з додатком. Swagger забезпечує чітке визначення ендпойнтів, параметрів запитів і форматів відповідей, що сприяє швидкій розробці та інтеграції.

Усі компоненти системи успішно протестовано, що підтверджує її працездатність та придатність до подальшого використання й розвитку. Реалізована архітектура дозволяє швидко адаптувати рішення під нові задачі, що робить її придатною для широкого спектра прикладних застосувань у галузях, пов'язаних з обробкою відео та штучним інтелектом.

Розроблена система є гнучкою, надійною та адаптованою до потреб кінцевого користувача. Її архітектура дозволяє легко розширювати функціонал, додавати нові модулі аналітики, підтримувати різні формати даних та інтегруватися з іншими сервісами.

Отримані результати демонструють ефективність запропонованого підходу та підтверджують доцільність застосування обраних технологій для створення сучасних інтелектуальних рішень у сфері відеоаналітики та штучного інтелекту.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		55

2 ЕКОНОМІЧНИЙ РОЗДІЛ

2.1 Резюме

У результаті виконання дипломного проєкту на тему «Розробка застосунку для аналізу відео та поведінки тварин на базі LLM» було створено інтерактивну систему для автоматизованого оброблення відеоданих, генерації текстових описів та аналізу поведінки тварин.

Розроблена система включає інтерактивний вебінтерфейс на базі Streamlit, серверну частину на Python із REST API, модуль для автоматичного виділення кадрів із відео та модулі інтеграції з великими мовними моделями.

Система забезпечує повний цикл роботи - від завантаження відеофайлу та налаштування параметрів обробки до автоматичного аналізу зображень, формування текстових описів та відповіді на запитання користувача.

Розроблений застосунок є практичним прикладом сучасного підходу до побудови систем мультимодального аналізу даних. Інтеграція технологій штучного інтелекту, методів комп'ютерного зору та гнучкої клієнт-серверної архітектури забезпечила створення ефективної, масштабованої та адаптивної платформи.

2.2. Визначення трудомісткості розробки програмного забезпечення.

Тривалість розробки програмного продукту визначається низкою чинників, зокрема: обсягом запланованих робіт, рівнем трудомісткості окремих етапів, кваліфікацією розробників, а також часовими обмеженнями, продиктованими ринковими вимогами. Для оцінки масштабів програмного забезпечення у даному проєкті використано метод структурної аналогії, який передбачає визначення орієнтовної складності продукту шляхом порівняння з аналогічними програмними рішеннями. На основі спеціалізованих каталогів аналогічних систем розраховується кількість умовних машинних команд, необхідних для реалізації подібного продукту (у тисячах умовних команд), що дозволяє обґрунтовано оцінити обсяг робіт та очікувані строки розробки.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		56

У таблиці 2.1 представлені аналоги програмного забезпечення, функції яких, у більшому або меншому ступені, виконує розроблений програмний продукт.

Таблиця 2.1 Каталог аналогів

Найменування ПП	Обсяг функції ПП - V _о , усл. машинних командах.
1. ПП автоматизації засобів по каталогу	680 - 7000
2. ПП автоматизованих розрахунків	1300 - 8600
3. ПП оптимізації розрахунків	1300 - 4200

У таблиці 2.1 представлені аналоги програмного забезпечення, функції яких, у більшому або меншому ступені, виконує розроблений програмний продукт. Для нашого варіанта виділено сірим кольором.

Вибравши аналог ПЗ, що містить V_о в умовних машинних командах, трудомісткість визначаємо на основі табл.2.

Таблиця.2.2

Обсяг ПП, тис.умов.машинних команд	Норма часу, люд/год
1.00	229
2.00	254
3.00	262

На підставі отриманого значення, по довіднику, визначаємо укрупнену норму часу на розробку аналога програмного забезпечення (коректується поправочним коефіцієнтом враховуючої умови розробки ПЗ, тобто в умовах комп'ютера, K_к=0,7÷0,8), для нашого варіанта виділено сірим кольором:

$$T_{ар} = 229 \times 0,8 = 183,2 \text{ (люд/годин)}.$$

Трудомісткість програмного продукту визначається по кожному етапу розробки окремо на підставі трудомісткості аналога з урахуванням складності

розробки, ступеня новизни і ступеня використання в розробці стандартних модулів на підставі формул:

$$T_{ТЗ} = T^a p \times L_1 \times K_H \quad (2.1)$$

$$T_{ТП} = T^a p \times L_2 \times K_H \quad (2.2)$$

$$T_{РП} = T^a p \times L_3 \times K_H \times K_T \quad (2.3)$$

Для розрахунку необхідні наступні коефіцієнти:

L_i - питома вага і-го етапу розробки (див. табл. 2.2.);

K_н - поправочний коефіцієнт, що враховує ступінь новизни (див. табл. 2.3.);

Кт - поправочний коефіцієнт, що враховує ступінь використання в розробці типових програм (див. табл. 2.4.).

Таблиця 2.2. Значення питомих коефіцієнтів трудомісткості стадії в загальній трудомісткості розробки ПП.

Код стадії	Ступінь новизни		
	А	Б	В
ТЗ (L ₁)	0,15	0,12	0,12
ТП (L ₂)	0,16	0,15	0,11
РП (L ₃)	0,55	0,58	0,61

Для нашого варіанта виділено сірим кольором.

Таблиця 2.3. Значення поправочного коефіцієнта, що враховує ступінь новизни

Код ступеня новизни	Ступінь новизни	Значення К _н
А	Принципово нові ПО	1,75 - 1,2
Б	ПО - розвиток визначеного параметричного ряду	1,0 - 0,8
В	ПО маючий аналог	0,7

Для нашого варіанта виділено сірим кольором.

Таблиця 2.4. Значення коефіцієнта ступеня використання в розробці ТП

Ступінь охоплення реалізованих функцій розроблювального ПО типовими програмами, %	Значення К _т
60 і вище	0,6
40-60	0,7
20-40	0,8
До 20	0,9

Для нашого варіанта виділено сірим кольором.

Тепер розраховуємо трудомісткість по кожному етапу окремо:

Трудомісткість технічного завдання

$$T_{ТЗ} = T^a * L_1 * K_n = 183,2 * 0,12 * 0,7 = 15,38 \text{ (люд/годин)} \quad (2.4)$$

Трудомісткість розробки технічного проекту

$$T_{ТП} = T^a * L_2 * K_n = 183,2 * 0,11 * 0,7 = 14,11 \text{ (люд/годин)} \quad (2.5)$$

Трудомісткість розробки робочого проекту

$$T_{РП} = T^a * L_3 * K_n * K_t = 183,2 * 0,61 * 0,7 * 0,6 = 46,94 \text{ (люд/годин)} \quad (2.6)$$

Для подальших розрахунків визначили кількість папера, витраченого на кожен етап: технічне завдання N_{ТЗ}= 2 (стор), розробка ТП N_{ТП}=22 (стор), розробка робочого проекту N_{РП}=10(стор), пояснювальна записка відповідно N_{ПЗ}=26 (стор)

Розрахунок зведений у таблицю 2.5

Таблиця 2.5. Розрахунок трудомісткості ПП

Найменування етапів	Розрахунок, годин.		
	1	2	3
1.ТЗ	$T_{PT3}=15,38$	$T_{KK}=0,7*N_{T3}=0,7*2=1,4$	$T_{HK}=0,15*N_{T3}=0,15*2=0,3$
2.Розробка ПП	$T_{PTP}=14,11$	$T_{KK}=0,7*N_{TP}=0,7*22=15,4$	$T_{HK}=0,15*N_{TP}=0,15*22=3,3$
3.Розробка РП	$T_{PRP}=46,94$	$T_{KK}=0,7*N_{RP}=0,7*10=7,0$	$T_{HK}=0,15*N_{RP}=0,15*10=1,5$
4.Розробка ПЗ	$T_{PZ}=1,5*N_{PZ}=1,5*26=39$	$T_{KK}=0,7*N_{T3}=0,7*26=18,2$	$T_{HK}=0,15*N_{PZ}=0,15*26=3,9$
Усього, в т.ч.:	166,43		
- на розробку	$\Sigma T_p=115,43$		
- контроль керівника		$\Sigma T_{KK}=42,0$	
- нормоконтроль			$\Sigma T_{HK}=9,0$

2.3 Розрахунок ціни програмного продукту.

Для визначення загальної вартості розробки програмного забезпечення здійснено розрахунок основної заробітної плати виконавців, матеріальних витрат, а також загальних витрат, пов'язаних із процесом створення ПЗ. Розрахунок основної заробітної плати виконавців представлено у таблиці 2.7. Відповідно до ст. 8 Закону України "Про Державний бюджет України на 2025 рік" встановлено мінімальну заробітну плату у місячному розмірі з 01.01.25 року - 8000 гривень. Мін. погодинна тарифна ставка становить 48,00 гривень.

Таблиця 2.6 Розрахунок основної заробітної плати виконавців

Найменування робіт	Трудомісткість робіт, години	Погодинна тарифна ставка, грн.	Розрахунок, грн.
1.Розробка ПП	115,43	48,00	5540,64
2.Контроль керівника	42,0	115,00	4830,00
3.Нормоконтроль	9,0	120,00	1080,00
Усього	-	-	$\Sigma Z_o=11450,64$

Зробимо розрахунок матеріальних витрат на розробку ПП. Розрахунок зведемо в таблицю 2.7

					КГ 08. 27 001. 00 ДП ПЗ	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		59

Таблиця 2.7 Розрахунок матеріальних витрат на розробку ПО

Найменування матеріальних витрат	Тип, модель	Кількість	Ціна одиниці, грн.	Вартість, грн.
Папір	Лист А4	60	5.0	300
Разом	-	-	-	$V_{Mi}=300$
Транспортно-заготівельні витрати (10%)				$V_{тр_з} = 0,1 \times V_{M1} = 30$
Усього				$V_M = V_{Mi} + V_{тр_з} = 330$

На підставі отриманих даних по окремих статтях витрат складена калькуляція планової собівартості в цілому ПП за формою, приведеною в таблиці 2.8.

Таблиця 2.8. Розрахунок статей витрат планової собівартості

Стаття витрат	Значення, грн.	Формула розрахунку
1. Матеріали	330,0	V_M (див. табл. 2.8.)
2. Основна заробітна плата	11450,64	Z_o (див. табл. 2.7.)
3. Додаткова заробітна плата	1145,06	$Z_d = 0,1 \times Z_o = 11450,64 \times 0,1$
4. Відрахування до єдиного фонду соціального внеску	2771,05	$V_{е.с.в.} = 0,22 \times (Z_o + Z_d) = 0,22 * (11450,64 + 1145,06)$
5. Накладні витрати	4580,26	$V_{нак.} = 0,4 \times Z_o = 0,4 * 11450,64$
6. Повна собівартість	20277,01	$C_{пов} = V_M + Z_o + Z_d + V_{е.с.в.} + V_{нак.} = 330,0 + 11450,64 + 1145,06 + 2771,05 + 4580,26$

Розмір прибутку, що включається в ціну, визначаємо по наступній формулі:

$$П = (C_{п} * P) / 100 = (20277,01 * 10) / 100 = 2027,7 \text{ грн} \quad (2.6)$$

Де p - плановий рівень рентабельності (10-15%).

Оптова ціна (кошторисна вартість) визначається по формулі:

$$Ц_o = C_{пов} + П = 20277,01 + 2027,7 = 22304,71 = 25920,05 \text{ грн} \quad (2.7)$$

Податок на додану вартість визначаємо по наступній формулі:

$$ПДВ = 22304,71 + 22304,71 * 0,2 = 26765,65 = 5184,01 \text{ грн} \quad (2.8)$$

Виходячи з отриманих даних, ціна реалізації розробленого програмного продукту на основі наступної формули, становитиме:

$$Ц_p = Ц_o + ПДВ = 25920,05 + 5184,01 = 31104,06 \text{ грн} \quad (2.9)$$

3 РОЗДІЛ ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

Охорона праці на підприємстві завжди має ключове значення, оскільки саме завдяки дотриманню відповідних рекомендацій працівники формують безпечні алгоритми виконання службових обов'язків. Головною метою охорони праці є забезпечення безпеки життя, збереження працездатності та здоров'я працівників у процесі виконання ними трудових функцій.

У сфері роботи з комп'ютерною технікою, як і в інших галузях, необхідно враховувати такі параметри, як:

- рівень освітлення;
- температура повітря;
- відносна вологість;
- рівень вібрації.

Особливу увагу при цьому слід приділяти дотриманню вимог пожежної безпеки, зокрема:

- вогнестійкості приміщення;
- допустимому рівню шумового навантаження;
- характеристикам електромагнітного, ультрафіолетового та інфрачервоного випромінювання.

У межах дипломного проєкту розглянуто питання охорони праці, зосереджуючи увагу на аналізі безпеки професійної діяльності розробника веб-сторінок в умовах офісного середовища.

3.1 Аналіз та безпека умов праці працівника на робочому місці

Під час будь-якого виду роботи за комп'ютером, працівник наражає себе на небезпечні фактори виробничого середовища, а саме: фізичні та психофізіологічні небезпечні й шкідливі виробничі фактори. Це підвищена температура повітря робочої зони, підвищений рівень шуму, знижена вологість повітря. У нервово-психічних перевантаженнях програміст зазнає перенапругу аналізаторів та монотонність праці, інколи, ще й розмовну перенапругу, коли розробнику потрібно скласти технічне завдання разом з клієнтом.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		61

Це виробничі фактори, які виникають при роботі у приміщеннях з комп'ютерами,

Окрім цього, комп'ютер випромінює електростатичні та електромагнітні поля у діапазоні від 5 Гц до 2 кГц та від 2 до 400 кГц, тож робота за комп'ютером включає ще підвищений рівень електромагнітний випромінювання та підвищений рівень статичної електрики.

У офісних приміщеннях не завжди є достатня кількість природного освітлення у такому разі присутня велика кількість штучного освітлення, яке у свою чергу не завжди правильно налаштоване, з цього виникає, що світло може бути недостатньо яскравим або дуже яскравим.

3.2 Розробка заходів з охорони праці

Виробниче освітлення

Штучне освітлення в приміщеннях з робочими місцями, обладнаними ВДТ має здійснюватися системою загального рівномірного освітлення. У виробничих та адміністративно-громадських приміщеннях, у разі переважної роботи з документами, допускається застосування системи комбінованого освітлення (крім системи загального освітлення, додатково встановлюються світильники місцевого освітлення).

Зазначення освітленості на поверхні робочого столу в зоні розміщення документів має становити 300-500лк , а освітленість екрана має не перевищувати 300лк

Мікроклімат

При роботі у приміщеннях з великою кількістю комп'ютерів, приміщення класифікуються як приміщення з підвищеною небезпекою електротравм, температура повітря влітку може становити більше 35 С, що дуже погано впливає на здоров'я людини, тож у таких приміщеннях повітря повинне охолоджуватись та понижена вологість повітря повинна регулюватись спеціальним обладнанням. Відповідно до норм ДСН 3.3.6.042-99 температура повітря в офісі повинна становити 22-25 С, вологість повітря 40-60%, швидкість руху повітря не більше 0,1 м/с.

					КГ 08. 27 001. 00 ДП ПЗ	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		62

Якщо ці норми перевищені, робочій день працівника повинен бути скорочений на 10%.

3.3 Організація робочого місця користувача ПК

Конструкція робочого місця користувача ПК й взаємне розташування всіх його елементів (сидіння, органи керування, засобу відображення інформації) відповідають антропометричним, фізіологічним і психологічним вимогам, а також характеру роботи.

Конструкція робочих меблів повинна забезпечувати можливість індивідуального регулювання відповідно росту працюючих для підтримки зручної пози.

Робочий стіл повинен бути пофарбований матовою фарбою. Дисплей розташований так, що його верхній край перебуває на рівні очей на відстані близько 70 см, що укладається в у припустимі рамки від 60 до 90 см. Частота мерехтіння екрана $f_{\text{мер}}=100$ Гц, що відповідає умові $f_{\text{мер}}>70$ Гц.

Робоче місце розташоване перпендикулярно віконним прорізам, це зроблено з тією метою, щоб виключити пряму й відбиту мерехтливність екрана від вікон і приладів штучного освітлення.

Для офісів та приміщень, обладнаних персональними комп'ютерами або технікою для бізнесу допустимий рівень шуму цілодобово - 50 дБА, а максимальний- 65 дБА

Згідно темі дипломного проекту робоче місце програміста укомплектовано пристроями з електромагнітним випромінюванням.

Забезпечення здорових і безпечних умов праці покладається на адміністрацію підприємств, установ, організацій. Вона зобов'язана запроваджувати сучасні засоби безпеки праці, які мають попереджувати виробничий травматизм і забезпечувати санітарно-гігієнічні умови, що запобігають виникненню професійних захворювань.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		63

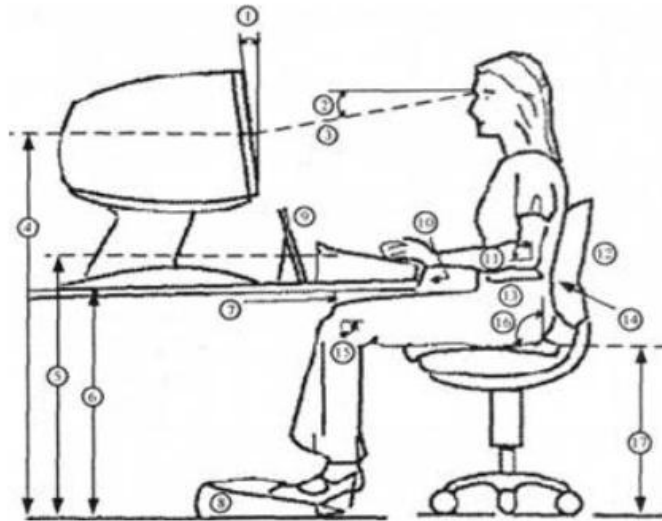


Рисунок 3.1 Робоче місце користувача ПК

3.4 Пожежна безпека

Забезпечення пожежної безпеки на об'єкті праці є важливою частиною роботи по створенню безпечних та здорових умов праці.

Прохід до аварійних виходів повинен бути вільний, шириною не менше 1 метру, у разі великої кількості горючих відходів потрібно використовувати відведені сміттєзбірники. Електроприлади повинні використовуватися тільки для їхнього прямого призначення, а у разі пошкодження приладів, слід вимкнути їх живлення та привести до пожежобезпечного стану. Первинні засоби пожежогасіння застосовуються для боротьби з пожежами на початковій стадії. До них належать: пожежні кран-комплекти, вогнегасники, пожежний інвентар (резервуари з водою, ящики з піском, пожежні відра, лопати), а також різний переносний пожежний інструмент (кирки, сокири, багри, ломи ін.).

Для гасіння пожеж промисловість випускає різні вогнегасники. Найбільшого поширення набули водопінні, водяні, газові (вуглекислотні) і порошкові. За ефективністю пожежогасіння гасіння, економічністю та іншими показниками більш перспективними вважаються порошкові вогнегасники. Первинні засоби пожежогасіння розміщують на пожежних щитах, які встановлюють на виробничій території з розрахунку один щит на 5000 м². Вони фарбуються у червоний колір.

Згідно Правил, на кожному поверсі будинку адміністративного призначення повинно знаходитися не менше двох вогнегасників з масою заряду вогнегасної речовини 5 кг і більше. Експлуатація вогнегасників без призначення відповідального за організацію цієї роботи не допускається. Забороняється палити на підприємстві, крім спеціально виведених для цього місцях, забороняється зберігати легкозаймисті матеріали, такі як папір ближче ніж 1 метр від електрощитів, 0,15 м від приладів центрального водяного опалення та 0,6 м від сповіщувачів автоматичної пожежної сигналізації, також документація повинна зберігатися у спеціально відведених для цього шафах.

Для запобігання розповсюдження пожежі встановлюють протипожежні системи, які складаються з датчиків, звукових сповіщувачів, аварійних кнопок, приймально-контрольної панелі, яка виступає як аналізатор інформації, яку отримали датчики і відправляє ці данні на пульт пожежної охорони. Протипожежна сигналізація призначення для виявлення пожежі на початковому етапі. Підприємство крім установки пожежної сигналізації на своєму об'єкті, має укласти договір на обслуговування даної системи з фірмою, що має на це ліцензію. В обслуговування входить проведення встановлених нормами регламентних робіт, а так само усунення несправностей в роботі системи. Періодичність перевірки узгоджується з замовником, але повинна бути не рідше ніж один раз на місяць.

У разі, якщо пожежі не вдалось уникнути, необхідно:

1. терміново повідомити пожежну охорону по телефону 101, вказати при цьому адресу, кількість поверхів, місце виникнення пожежі, наявність людей, своє прізвище;
2. організувати евакуацію людей та матеріальних цінностей;
3. повідомити про виникнення пожежі адміністрацію та чергового (за його наявності);
4. вимкнути, у разі необхідності, струмоприймачі та вентиляцію;
5. розпочати гасіння пожежі наявними первинними засобами пожежогасіння;
6. організувати зустріч підрозділів пожежної охорони й надати їм консультаційну та іншу допомогу в процесі гасіння пожежі.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		65

ВИСНОВКИ

Під час виконання дипломного проєкту на тему «Розробка застосунку для аналізу відео та поведінки тварин на базі LLM» було спроектовано та реалізовано інтерактивну систему, що дозволяє автоматизовано здійснювати обробку відеоданих, аналізувати поведінкові характеристики тварин та формувати текстові описи за допомогою сучасних технологій штучного інтелекту. В процесі роботи над проєктом було виконано:

-розроблено клієнт-серверну архітектуру системи з використанням Python, Streamlit та REST API;

-реалізовано модуль автоматизованого виділення кадрів із відео із гнучким налаштуванням параметрів сегментації;

-інтегровано взаємодію з великими мовними моделями (LLM) через OpenAI API для генерації текстових описів кадрів та відповіді на запитання користувача;

-впроваджено модулі комп'ютерного зору для попередньої обробки зображень та виявлення ключових елементів сцени;

-реалізовано зручний інтерактивний вебінтерфейс для взаємодії кінцевого користувача із системою;

-проведено тестування працездатності системи у різних сценаріях (аналіз відео з птахівничих та тваринницьких господарств).

Результати експериментальної частини показали, що система є стабільною у роботі, забезпечує високу якість згенерованих текстових описів та адекватно справляється із завданням аналізу відеоданих. Інтеграція технологій LLM та комп'ютерного зору дозволяє ефективно поєднувати аналіз зображень та генерацію текстових висновків у режимі реального часу. Практична цінність розробленого рішення полягає у можливості його застосування в агропромисловому комплексі, крім того, платформа може бути адаптована для використання в інших галузях, де необхідний аналіз відеоінформації та мультимодальна взаємодія з кінцевим користувачем. Таким чином, поставлені у дипломному проєкті мета та завдання було повністю виконано, що підтверджується результатами практичної реалізації системи та успішно проведеного тестування.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		66

ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Бондаренко О. С. Захист інформації в комп'ютерних системах: навчальний посібник. - К.: КНЕУ, 2020. - 256 с.
2. Коваль І. В. Проблеми безпеки штучного інтелекту в умовах інформаційної війни // Інформаційна безпека. - 2023. - № 2. - С. 42–48.
3. Мороз С. В. Кібербезпека та інформаційна безпека: теорія і практика. - К.: Центр учбової літератури, 2021. - 312 с.
4. Олійник В. В. Архітектура комп'ютерних систем: підручник. - Львів: Львівська політехніка, 2020. - 384 с.
5. Пархоменко Л. М. Штучний інтелект і машинне навчання: навчальний посібник. - Харків: ХНУРЕ, 2022. - 198 с.
6. Теленко О. В. Використання LLM у державному секторі: ризики та рекомендації // Політичні студії. - 2023. - № 3(41). - С. 105–112.
7. OpenAI API Documentation. - OpenAI [Електронний ресурс]. - Режим доступу: <https://platform.openai.com/docs>
8. Streamlit Documentation. - Streamlit.io [Електронний ресурс]. - Режим доступу: <https://docs.streamlit.io/>
9. Aggarwal C.C. Neural Networks and Deep Learning. - Springer, 2021.
10. Singh D., Bhatia R. A survey on object detection using deep learning techniques. // International Journal of Multimedia and Ubiquitous Engineering. - 2021. - Vol. 16, No. 3. - P. 1-16.
11. Zhang C. et al. Computer vision for animal health monitoring: a review. // Computers and Electronics in Agriculture. - 2021. - Vol. 182. - 105981.
12. McKinney W. Python for Data Analysis. - O'Reilly Media, 2022.
13. Documentation of YOLO Object Detection Framework. - <https://github.com/AlexeyAB/darknet>
14. Krizhevsky A., Sutskever I., Hinton G.E. ImageNet classification with deep convolutional neural networks. // Communications of the ACM. - 2022. - Vol. 60, No. 6. - P. 84-90.

					<i>КГ 08. 27 001. 00 ДП ПЗ</i>	Арк.
Ізм.	Лист	№ докум.	Підпис	Дата		67

Додаток А. Програмний код «Опис ендпоїнтів»

```
// Імпортуємо бібліотеки
const express = require('express');
const bodyParser = require('body-parser');
const app = express();
const PORT = 3000;
// Підключення middleware
app.use(bodyParser.json());

// Тимчасовий масив для зберігання користувачів
let users = [];

/**
 * GET /api/v1/users
 * Повертає список усіх користувачів.
 */
app.get('/api/v1/users', (req, res) => {
  res.status(200).json(users);
});

/**
 * POST /api/v1/users
 * Додає нового користувача у систему.
 * Тіло запиту має містити name, email, password.
 */
app.post('/api/v1/users', (req, res) => {
  const { name, email, password } = req.body;

  // Перевірка на обов'язкові поля
  if (!name || !email || !password) {
    return res.status(400).json({ error: 'Missing required fields' });
  }

  // Перевірка на існуючого користувача
  if (users.find(user => user.email === email)) {
    return res.status(409).json({ error: 'User with this email already exists' });
  }

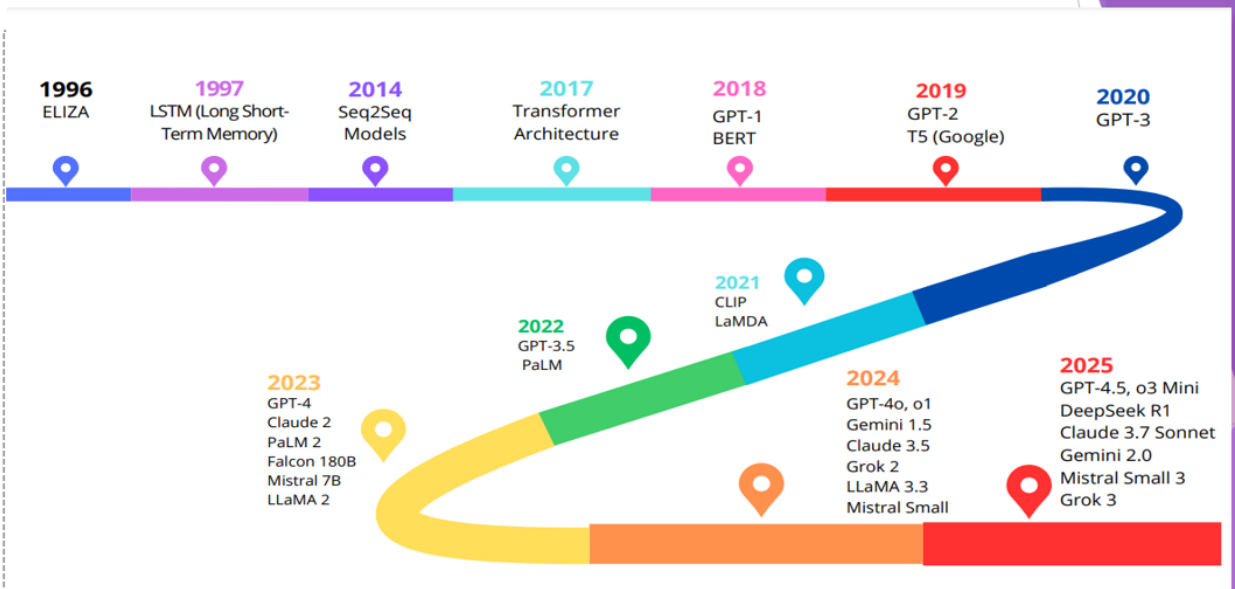
  // Створення нового користувача
  const newUser = {
    id: Date.now().toString(),
    name,
    email
  };
  users.push(newUser);
  res.status(201).json(newUser);
});

// Запуск сервера
app.listen(PORT, () => {
  console.log(`Server is running on http://localhost:${PORT}`);
});
```

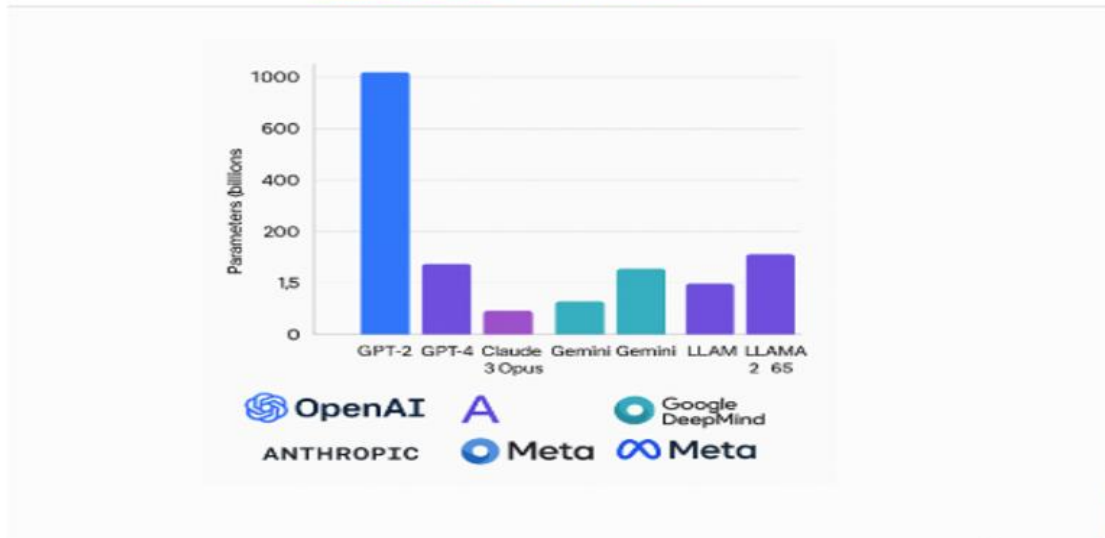
«РОЗРОБКА ЗАСТОСУНКУ ДЛЯ АНАЛІЗУ ПОВЕДІНКИ ТВАРИН НА БАЗІ LLM»

Виконав: Усатюк О.Г.

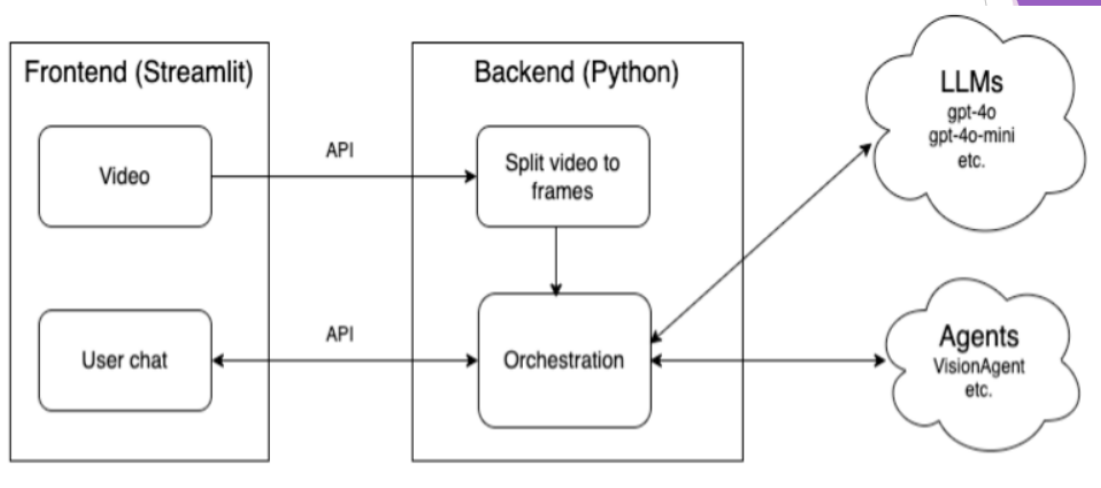
ЕВОЛЮЦІЯ ВЕЛИКИХ МОВНИХ МОДЕЛЕЙ



Діаграма параметрів трансформерних мовних моделей



СТРУКТУРА СИСТЕМИ



СКРІНШОТ ГОЛОВНОЇ СТОРІНКИ СИСТЕМИ

Upload Options

Choose upload method:

Upload Video File

YouTube URL

Select frame interval (seconds) ?

10 - +

Select language ?

English ▾

Select model ?

gpt-4o-mini ▾

Video Processing Dashboard

Choose a video file

Drag and drop file here Browse files

Limit 200MB per file • MP4, AVI, MOV, MPEG4

farm-chickens-1.mov 13.7MB ×

Video Title

farm-chickens-1.mov

Description

Poultry farm | Press Ctrl+Enter to apply

Process Video

СКРІНШОТ ВИДІЛЕННЯ КАДРІВ ІЗ ВІДЕО

Video Processing Dashboard

Choose a video file

Drag and drop file here Browse files

Limit 200MB per file • MP4, AVI, MOV, MPEG4

Processing Status

Status: completed - Processing completed. 3 frames extracted

Extracted Frames



Frame 1



Frame 2



Frame 3

Скріншот кадру з обробки відео



Скріншот обробки запиту



Frame 3



Опиши що зображено на картинці?



На картинках зображено багато курчат, які знаходяться у приміщенні для вирощування пташиного поголів'я. Курчата predominantly білі, і вони розташовані на підлозі, яка покрита землею або стружкою. У центрі кадру видно червоні годівниці та поїлки, які використовуються для годування та забезпечення водою птахів. Приміщення освітлене природним світлом з вікон, а також має синій дах або накриття. Картинки демонструють активну та спокійну атмосферу, де курчата пересуваються в пошуках їжі та води.

СКРІНШОТ ОБРОБКИ ЗАПИТУ З КАДРІВ ПО ПІДРАХУНКУ КУРЕЙ



СКРІНШОТ ОБРОБКИ ЗАПИТУ

Upload Options

Choose upload method:

Upload Video File

YouTube URL

Select frame interval (seconds) ⓘ

2 - +

Select language ⓘ

Ukrainian ▼

Select model ⓘ

gpt-4o ▼

Frame 3

Frame 4

Frame 5

Frame 6

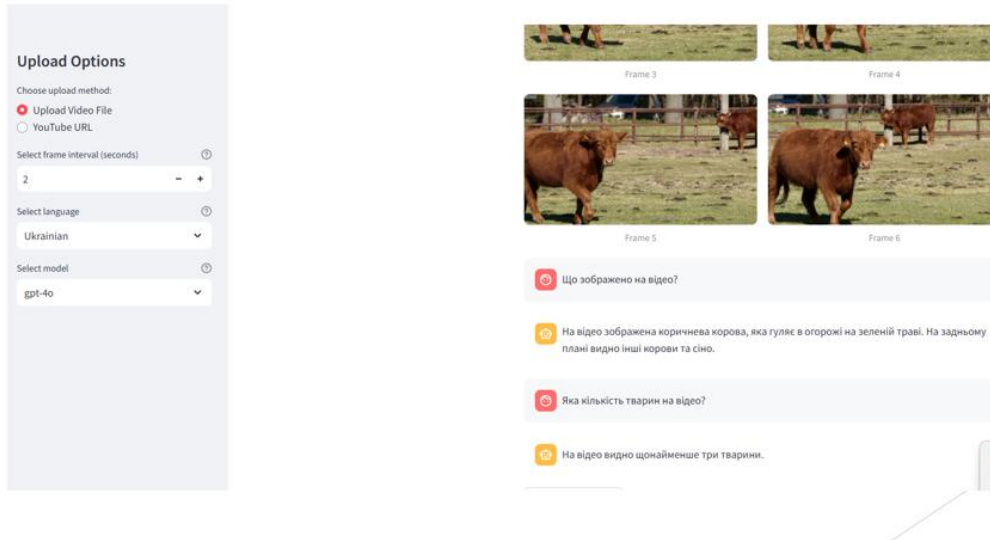
Що зображено на відео?

На відео зображена коричнева корова, яка гуляє в огорожі на зеленій траві. На задньому плані видно інші корови та сіно.

Яка кількість тварин на відео?

На відео видно щонайменше три тварини.

СКРІНШОТ ОБРОБКИ ЗАПИТУ З КАДРІВ ПО ПІДРАХУНКУ ТВАРИН



Висновки

Під час виконання дипломного проєкту було спроектовано та реалізовано інтерактивну систему, що дозволяє автоматизовано здійснювати обробку відеоданих, аналізувати поведінкові характеристики тварин та формувати текстові описи за допомогою сучасних технологій штучного інтелекту. В процесі роботи над проєктом було виконано:

1. розроблено клієнт-серверну архітектуру системи з використанням Python, Streamlit та REST API;
2. реалізовано модуль автоматизованого виділення кадрів із відео із гнучким налаштуванням параметрів сегментації;
3. інтегровано взаємодію з великими мовними моделями (LLM) через OpenAI API для генерації текстових описів кадрів та відповіді на запитання користувача;
4. реалізовано зручний інтерактивний вебінтерфейс для взаємодії кінцевого користувача із системою;
5. проведено тестування працездатності системи у різних сценаріях.

РЕЦЕНЗІЯ

на дипломний проект (роботу) здобувача (здобувачки) освіти
відділення комп'ютерних систем

Усатюка Олександра Геннадійовича

(прізвище, ім'я та по батькові)

Спеціальність 123 Комп'ютерна інженерія

Освітньо-професійна програма Комп'ютерна графіка і Web-дизайн

Керівник дипломного проекту (роботи) Ломовцев Павло Борисович

(прізвище, ім'я та по батькові)

Тема дипломного проекту (роботи) Розробка застосунку для аналізу поведінки тварин на базі LLM

Обсяг розрахунково-пояснювальної записки 76 сторінок

Обсяг графічної (презентаційної) частини 12 аркушів (слайдів)

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ)

а) заключення про ступінь відповідності виконаного дипломного проекту (роботи) завданню
Робота відповідає технічному завданню до дипломного проекту.
Виконана у відповідності з вимогами

б) характеристика виконання кожного розділу дипломного проекту (роботи) _____
При виконанні дипломного проекту здобувач продемонстрував уміння використовувати останні досягнення науки та техніки, уміння працювати з літературою. Так, здобувач грамотно дослідила та проаналізувала усі етапи проектування застосунку для аналізу поведінки тварин на базі LLM

в) оцінка якості виконання пояснювальної записки та графічної частини дипломного проекту (роботи) _____
Пояснювальна записка та графічна частина відповідає вимогам, виконана якісно та відображає основні елементи проектування застосунку для аналізу поведінки тварин на базі LLM

г) перелік позитивних якостей дипломного проекту (роботи)

Розроблено застосунок для аналізу поведінки тварин на базі LLM із використанням мови програмування Python. Запропонована система ґрунтується на застосуванні мультимодальних LLM на прикладі для аналізу поведінкової активності тварин на основі потокового відео

д) основні недоліки дипломного проекту (роботи) Для подальшого вдосконалення потрібне доопрацювання, зокрема в галузі оптимізації обчислювальних процесів, покращення інтерпретації результатів та забезпечення конфіденційності даних. Тривалий час обробки великих обсягів даних, оскільки розбиття відео на кадри, їх аналіз та генерація текстових описів потребують значних обчислювальних потужностей, особливо для довгих відеозаписів

Оцінка розрахункової частини Добре

Оцінка графічної частини Відмінно

Загальна оцінка Добре

Прізвище, ім'я, по батькові рецензента к.т.н. Рудніченко Микола Дмитрович

Місце роботи і посада рецензента Національний університет «Одеська політехніка»,
доцент кафедри інформаційних технологій



Підпис

« 27 » 06 2025 р.

ВІДГУК

керівника на дипломний проєкт здобувача (здобувачки) освіти
відділення комп'ютерних систем

Усатюка Олександра Геннадійовича

(прізвище, ім'я та по батькові)

Спеціальність: 123 "Комп'ютерна інженерія"

Освітньо-професійна програма: «Комп'ютерна графіка і Web-дизайн»

Тема дипломного проєкту: Розробка застосунку для аналізу поведінки
тварин на базі LLM

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЄКТУ

а) обсяг і якість виконання проєкту (графічного матеріалу і розрахунково-пояснювальної записки) Дипломний проєкт виконано відповідно технічному завданню. Пояснювальна записка містить 76 сторінки. У пояснювальній записці виконано опис етапів, відповідно до індивідуального завдання, розділи пояснювальної записки відповідають етапам рішення завдання, поставленого у дипломному проєкті. Графічна частина складається з 12 слайдів мультимедійної презентації. Якість виконання пояснювальної записки та графічної частини добра, розробку виконано в повному обсязі.

б) самостійність роботи над проєктом: _____

Протягом всього строку дипломного проєктування та переддипломної практики здобувач освіти Усатюк Олександр Геннадійович поступово та послідовно виконував всі етапи розробки. Всі роботи здобувач освіти виконував самостійно, з оглядом на рекомендації керівника

в) теоретична підготовка випускника (випускниці): Здобувач освіти Усатюк Олександр Геннадійович під час роботи над дипломним проєктом вивчив достатню кількість літературних джерел та матеріалів за даною тематикою.

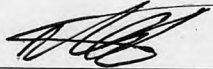
Вважаю, що теоретична підготовка дипломника добра і він готовий до захисту дипломного проєкту

г) вміння розв'язувати виробничі та конструкторські питання _____
*Під час дипломного проектування здобувач освіти Усатюк Олександр
Геннадійович розробив для аналізу поведінки тварин на базі LLM.
Дипломник володіє навичками написання веб-сайту за допомогою таких
технологій як CMS OpenCard, СУБД MySQL, HTTP*

Оцінка розрахункової частини _____ *Добре*
Оцінка графічної частини _____ *Добре*
Загальна оцінка _____ *Добре*

Прізвище, ім'я, по батькові керівника дипломного проекту _____
Ломовцев Павло Борисович

Місце роботи і посада керівника дипломного проекту _____
*Одеський національний технологічний університет, к.т.н, доцент,
завідувач кафедри інформаційних технологій та кібербезпеки*

Підпис _____ 

«18» _____ *06* 2025 р.

**ДОЗВІЛ
НА РОЗМІЩЕННЯ
ВИПУСКНОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
(ДИПЛОМНОГО ПРОЕКТУ)
В ЕЛЕКТРОННОМУ РЕПОЗИТАРІЇ ВСП «ОТФК ОНТУ»**

Ми, що нижче підписалися,

Усатюк Олександр Геннадіович
здобувач освіти гр. 4КГ-08, та

Ломовцев Павло Борисович,
керівник дипломного проекту,

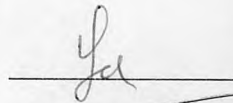
не заперечуємо щодо розміщення електронного варіанту пояснювальної записки до дипломного проекту фахового молодшого бакалавра на тему:

«Розробка застосунку для аналізу поведінки тварин на базі LLM» (автор роботи – Усатюк О.Г., керівник роботи – Ломовцев П.Б.)

виконаного у ВСП «Одеський технічний фаховий коледж Одеського національного технологічного університету» в 2025 році, у повному обсязі в електронному репозитарії ВСП «ОТФК ОНТУ» для вільного доступу через мережу Інтернет.

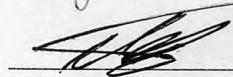
Несемо відповідальність за ідентичність електронного та друкованого варіантів випускної кваліфікаційної роботи і даємо згоду на обробку персональних даних.

Виконавець



/ Усатюк О.Г. /

Керівник



/ Ломовцев П.Б. /

«16» червня 2025 р.

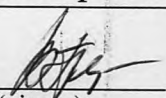
Д О В І Д К А

циклової комісії КТ та ПІ
про допуск до захисту дипломного проєкту
здобувача (здобувачки) освіти ІV курсу
відділення комп'ютерних систем групи 4КГ-08

Усатюка Олександра Генадійовича

на тему Розробка застосунку для аналізу поведінки тварин
на базі LLM

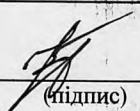
Висновок відповідальної особи за проведення нормоконтролю:
пояснювальна записка до дипломного проєкту виконана з деякими
порушеннями ДСТУ та оформлена відповідно до вимог Положення про
дипломне проєктування


(підпис)

18.06.2025
(дата)

Петрашова В.І.
(П.І.Б.)

Висновок відповідальної особи за перевірку роботи на наявність академічного
плагиату згідно звіту про перевірку від 13.06.2025 р. значення коефіцієнту
подібності в роботі становить 40,16%, коефіцієнт цитування – 1,38%.


(підпис)

18.06.2025
(дата)

Краснокутська К.Г.
(П.І.Б.)

Попередня експертиза (малий захист) дипломного проєкту

здобувача (здобувачки) освіти

Усатюка О.Г.
(П.І.Б.)

проведена « 18 » червня 2025 р.

Висновки Пояснювальна записка до дипломного проєкту виконана у повному
обсязі. Випускна кваліфікаційна робота (дипломний проєкт) відповідає
вимогам Положення про дипломне проєктування та рекомендована до
захисту.

Голова ЦК КТ та ПІ


(підпис)

Кривченко Ю.В.
(П.І.Б.)

Звіт подібності

метадані

Назва організації

Odesa Technical Professional College of Odesa National University of Technology

Заголовок

Розробка застосунку для аналізу поведінки тварин на базі LLM

Автор

Науковий керівник / Експерт

Усатюк Олександр Геннадійович Ломовцев Павло Борисович

підрозділ

Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету"

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



КП 1



КЦ

25

Довжина фраз для коефіцієнта подібності 2

10921

Кількість слів

87630

Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про **МОЖЛИВІ** маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		18
Інтервали		0
Мікропробіли		2
Білі знаки		0
Парафрази (SmartMarks)		114

Подібності за списком джерел

Нижче наведено список джерел. В цьому списку є джерела із різних баз даних. Копії тексту означає в якому джерелі він був знайдений. Ці джерела і значення коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз

Копія тексту

порядковий номер	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Створення web-застосунку цифрового помічника з використанням Open AI 5/28/2025 Odesa Technical Professional College of Odesa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	591 5.41 %

2	Створення web-застосунку цифрового помічника з використанням Open AI 5/28/2025 Odessa Technical Professional College of Odessa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	437 4.00 %
3	Створення web-застосунку цифрового помічника з використанням Open AI 5/28/2025 Odessa Technical Professional College of Odessa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	238 2.18 %
4	Створення web-застосунку цифрового помічника з використанням Open AI 5/28/2025 Odessa Technical Professional College of Odessa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	236 2.16 %
5	Створення web-застосунку цифрового помічника з використанням Open AI 5/28/2025 Odessa Technical Professional College of Odessa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	218 2.00 %
6	Створення web-застосунку цифрового помічника з використанням Open AI 5/28/2025 Odessa Technical Professional College of Odessa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	159 1.46 %
7	Створення web-застосунку цифрового помічника з використанням Open AI 5/28/2025 Odessa Technical Professional College of Odessa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	153 1.40 %
8	Створення web-застосунку цифрового помічника з використанням Open AI 5/28/2025 Odessa Technical Professional College of Odessa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	143 1.31 %
9	Створення web-застосунку цифрового помічника з використанням Open AI 5/28/2025 Odessa Technical Professional College of Odessa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	127 1.16 %
10	Створення web-застосунку цифрового помічника з використанням Open AI 5/28/2025 Odessa Technical Professional College of Odessa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	123 1.13 %

з домашньої бази даних (27.39 %)

порядковий номер	заголовок	кількість ідентичних слів (фрагментів)
1	Створення web-застосунку цифрового помічника з використанням Open AI 5/28/2025 Odessa Technical Professional College of Odessa National University of Technology (Відокремлений структурний підрозділ "Одеський технічний фаховий коледж Одеського національного технологічного університету")	2991 (26) 27.39 %

з програми обміну базами даних (0.10 %)

ПОРЯДКОВЫЙ НОМЕР	ЗАГОЛОВОК	КОЛИЧЕСТВО ИДЕНТИЧНЫХ СЛОВ (ФРАГМЕНТОВ)
1	Osnovna_chastina_2024_Perevoznyk_K_M 12/15/2024 National Technical University "Kharkiv Polytechnic Institute" students papers (National Technical University "Kharkiv Polytechnic Institute" students papers)	11 (1) 0.10 %

3 Интернету (12.67 %)

ПОРЯДКОВЫЙ НОМЕР	ДЖЕРЕЛО URL	КОЛИЧЕСТВО ИДЕНТИЧНЫХ СЛОВ (ФРАГМЕНТОВ)
1	https://card-file.ontu.edu.ua/bitstreams/e4fafe26-0a7e-4a4d-afc2-94341838de2a/download	620 (19) 5.68 %
2	https://card-file.ontu.edu.ua/bitstreams/c58b0ff5-46e0-49f8-8cbe-65c32256665d/download	101 (3) 0.92 %
3	https://card-file.ontu.edu.ua/bitstreams/29489599-0581-4ce6-8890-c3b13d9f2e0e/download	80 (5) 0.73 %
4	https://card-file.ontu.edu.ua/bitstreams/341a820e-d025-42f3-b7dc-27e831d6c66f/download	76 (4) 0.70 %
5	https://card-file.ontu.edu.ua/bitstreams/1dff552d-7200-49b8-ae1d-ba76a1335685/download	76 (5) 0.70 %
6	https://card-file.ontu.edu.ua/server/api/core/bitstreams/ead3fa83-2e3d-4cd7-bfbd-1d5ed04c1ce4/content	62 (4) 0.57 %
7	https://card-file.ontu.edu.ua/bitstreams/53ed22ad-8700-4162-b97a-082a1ad472d6/download	41 (3) 0.38 %
8	https://card-file.ontu.edu.ua/server/api/core/bitstreams/a141b658-5fa7-4f90-b0bd-7f0ccaed21e5/content	39 (2) 0.36 %
9	https://card-file.ontu.edu.ua/server/api/core/bitstreams/995bdcec-4e4d-4321-8070-4d6badcb8e49/content	33 (2) 0.30 %
10	https://oppb.com.ua/sites/default/files/instrukciya_pozh_bezpeka_sklady.pdf	30 (3) 0.27 %
11	https://card-file.ontu.edu.ua/bitstreams/035f6436-20b4-4ee6-8e99-bede670e308b/download	28 (1) 0.26 %
12	https://dev.to/abidullah786/building-a-restful-api-with-nodejs-a-step-by-step-guide-27op	21 (2) 0.19 %
13	https://cyberleninka.ru/article/n/evolyutsiya-algoritmov-mashinnogo-obucheniya-ot-klassicheskikh-modeley-k-nevrosetevym-metodam	20 (1) 0.18 %
14	https://card-file.ontu.edu.ua/bitstreams/5240e379-7721-49f0-8ee6-27140b0b473a/download	19 (2) 0.17 %
15	https://dev.to/henryshi101/online-chat-room-with-nodejs-3dm8	15 (2) 0.14 %
16	https://github.com/Skyeaeer/GreatestShow/blob/main/index.ts	15 (2) 0.14 %
17	https://apidog.com/blog/bearer-token-nodejs-express/	15 (2) 0.14 %
18	https://cyberleninka.ru/article/n/sravnienie-prediktivnoy-sposobnosti-modeley-mashinnogo-obucheniya-s-ispolzovaniem-razlichnoy-struktury-dannyh	14 (2) 0.13 %
19	https://dl.acm.org/doi/10.1016/j.asoc.2019.105716	14 (1) 0.13 %
20	https://ela.kpi.ua/jspui/bitstream/123456789/34802/1/Savelieva-A-A_bakalavr.pdf	12 (1) 0.11 %
21	https://card-file.ontu.edu.ua/server/api/core/bitstreams/6e367988-fd72-47a3-ac3d-c5748c863588/content	12 (1) 0.11 %
22	https://card-file.ontu.edu.ua/bitstreams/0e72a3b9-bdd7-4711-a3c6-dedc1d4287cc/download	10 (1) 0.09 %
23	https://9lib.org/document/zx50en4w-1-2-2-3-3.html	10 (1) 0.09 %
24	https://card-file.ontu.edu.ua/bitstreams/bbed74c8-2ea7-44c5-8d00-0fe3fd9790ee/download	9 (1) 0.08 %
25	https://card-file.ontu.edu.ua/server/api/core/bitstreams/6eb6bf1c-5813-45e6-93c5-25539b4709d3/content	6 (1) 0.05 %

Список прийнятих фрагментів (немає прийнятих фрагментів)

ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДИНОЗВІК СЛІВ (ФРАГМЕНТІВ)
------------------	-------	---------------------------------------

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП **ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ**

Спеціальність: 123 **Комп'ютерна інженерія**

Освітня програма: **Комп'ютерна графіка та Web-дизайн** Група: **4КГ-08**

Дипломний проект здобувача освіти денної форми навчання **КГ_08.27.000_ДП**

УСАТЮКА

ОЛЕКСАНДРА ГЕННАДІЙОВИЧА

м. Одеса

2025 р. МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ ВСП **ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ**

Спеціальність: 123 **Комп'ютерна інженерія**

Освітньо-професійна програма: **Комп'ютерна графіка та Web-дизайн**

Група: **4КГ-08**

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту на тему:

Розробка застосунку для аналізу поведінки тварин на базі LLM

Проектний матеріал складається з пояснювальної записки на _____ сторінках та графічного (перзентаційного) матеріалу на _____ аркушах (слайдах).

Дипломник _____ (Усатюк О.Г.)

Керівник(и) _____ (Помовцев)

Консультанти: з економічного розділу _____ (Іванченко В. С.)

з розділу охорони праці та техніки безпеки _____ (Чорновол Н. І.) з нормоконтролю _____

(Петлашова В. І.) старший консультант _____ (Кривченко Ю. В.)

До захисту допущений Голова циклової комісії _____ (Кривченко Ю. В.) Завідувач відділення _____ (Краснохутська К. Г.)

Захист « _____ » _____ 2025 р. Протокол ЕК № _____

Оцінка ЕК _____

Секретар ЕК _____

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП **ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ**

Відділення **Комп'ютерних систем** Комісія **КТ та ПІ**

Спеціальність 123 - **Комп'ютерна інженерія**

Освітня програма **Комп'ютерна графіка та Web-дизайн**

ЗАТВЕРДЖУЮ Заст. дир. з НР Беркань І. В. « _____ » _____ 2025 року ЗАВДАННЯ на дипломний проект (роботу).

Усатюка Олександра Геннадійовича

1. Тема проекту (роботи) **Розробка застосунку для аналізу поведінки тварин на базі LLM**

Затверджена наказом по коледжу від « _____ » _____ 11 2025 р., наказ № **246**

2 Термін здачі **закінченого проекту (роботи)** 3. Вихідні дані до проекту (роботи)

Розробка застосунку для аналізу поведінки тварин на базі LLM; Великі мовні моделі (LLM);

Мова програмування Python, фреймворк Streamlit та REST AP Мова програмування JavaScript

з використанням платформи з відкритим кодом; Nodejs, фреймворк для створення серверних додатків NestJS; векторна базаю даних Pinecone.