**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**
**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**
**імені ІГОРЯ СІКОРСЬКОГО»**
ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ
(повна назва інституту/факультету)

кафедра БІОМЕДИЧНОЇ КІБЕРНЕТИКИ
(повна назва кафедри)

«На правах рукопису»
УДК _____

**«До захисту допущено»**
В.о.завідувача кафедри БМК

_____ Світлана АЛХІМОВА
(підпис) (ініціали, ПРІЗВИЩЕ

"___" __*грудня*__2025р.

# Магістерська дисертація

**на здобуття ступеня магістра**
**за освітньо-професійною програмою *«Комп'ютерні технології в біології та медицині»***
**зі спеціальності *122 «Комп'ютерні науки»***

на тему: **Система диференціальної діагностики раку легень за даними комп'ютерної томографії**

Виконав: студент **II** курсу, групи _ЗК-41мп__

**ГЛОМОЗДА КОСТЯНТИН ЮРІЙОВИЧ** _____
(прізвище, ім'я, по батькові) (підпис)

Науковий керівник: *професор кафедри біомедичної кібернетики д.б.н., с.н.с., професор Настенко Євген Арнольдович* _____
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по ініціали) (підпис)

Рецензент: *доцент кафедри БМІ, к.ф.-м.н., доцент Соломін Андрій В'ячеславовович* _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Київ – 2025 року

**Національний технічний університет України**
**«Київський політехнічний інститут імені Ігоря Сікорського»**
**Факультет біомедичної інженерії**
**Кафедра біомедичної кібернетики**

Рівень вищої освіти – другий (магістерський)

Спеціальність – 122 «Комп'ютерні науки»

Освітньо-професійна програма «Комп'ютерні технології в біології та медицині»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри БМК
_____ Світлана АЛХІМОВА
«_____»__*листопада*___2025 р.

**ЗАВДАННЯ**
**на магістерську дисертацію студентці**

**ГЛОМОЗДІ КОСТЯНТИНУ ЮРІЙОВИЧУ**
(прізвище, ім'я, по батькові)

1. Тема дисертації **Система диференціальної діагностики раку легень за даними комп'ютерної томографії**

науковий керівник дисертації
***Настенко Євген Арнольдович, д.б.н., с.н.с., професор***
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «03»_*листопада*_ 2025 р. №***4751-с***_

2. Термін подання студентом дисертації ***20-22 листопада 2025 року***

3. Об'єкт дослідження: процес автоматизованої діагностики онкологічних захворювань легень на основі аналізу комп'ютерної томографії з використанням інтелектуального агента на базі великих мовних моделей та ансамблевих методів класичного машинного навчання для класифікації патологічних утворень.

4. Вихідні дані: датасет комп'ютерної томографії імуногістохімії пацієнтів з різними видами пухлин легень у форматі DICOM, організований у шістнадцять класів онкологічних патологій.

5. Перелік завдань, які потрібно розробити: розробити інтерфейс користувача для перегляду медичних зображень з інтерактивним розмічуванням областей інтересу; створити модуль екстракції характеристик інтенсивності, текстури та форми з медичних зображень; навчити три алгоритми класифікації та розробити систему ансамблевого об'єднання результатів; реалізувати автоматичне виявлення підозрілих областей на зображеннях; інтегрувати велику мовну

модель та побудувати агентну систему з динамічним вибором інструментів; додати конверсаційний інтерфейс для діалогу з агентом та відображення результатів діагностики.
6. Орієнтовний перелік графічного (ілюстративного) матеріалу: 15 рисунків, 0 таблиць, презентація з захисту МД на 12 слайдах.
7. Орієнтовний перелік публікацій: не передбачено
8. Консультанти розділів дисертації: не передбачено
9. Дата видачі завдання ***28 серпня 2025р.***

**Календарний план**

| № з/п | Назва етапів виконання магістерської дисертації | Термін виконання етапів МД | Примітка |
|---|---|---|---|
| 1 | Отримати завдання на МД | До 28.08.2025 | ***виконано*** |
| 2 | Завершення виконання практичної частина МД | До 25.10.2025 | ***виконано*** |
| 3 | Завершення оформлення розділів МД (Вступ, Основні розділи та висновки до них, Загальні висновки, Список використаних джерел) | До 24.11.2025 | ***виконано*** |
| 4 | Апробація та публікація результатів дослідження МД (отримати підтвердження про прийняття до друку) | До 01.12.2025 | ***виконано*** |
| 5 | Перевірка МД науковим керівником | 24.11.2025 | ***виконано*** |
| 6 | Подання в електронному вигляді МД на перевірку нормоконтролера | До 01.12.2025 | |
| 7 | Подання в електронному вигляді МД на перевірку подібності Strike Plagiarism com. Отримати позитивний звіт подібності. | До 05.12.2025 | |
| 8 | Підготовка Експертної оцінки до звіту подібності. | До 10.12.2025 | |
| 9 | Отримати відгук наукового керівника | 10.12.2025 | |
| 10 | Надати на кафедру пакет документів в паперовому та електронному вигляді (МД, відгук керівника, звіт подібності, Експертної оцінки до звіту подібності ) | 10-11.12.2025 | |
| 11 | Отримати допуск до захисту МД в ЕК та направлення до рецензента (засідання кафедри) | Засідання кафедри | |
| 12 | Подання МД рецензенту. Отримати на надання рецензію до ЕК / на кафедру | До 19.12.2025 | |
| 13 | Подання супровідного пакету документів по МД до захисту в ЕК[1] | До 19.12.2025 | |
| 14 | Захист МД в ЕК | 22 – 26.12.2025 | |

Студент _____ **Костянтин ГЛОМОЗДА**
(підпис) (ім'я, ПРІЗВИЩЕ)

Науковий керівник _____ **Євген НАСТЕНКО**
(підпис) (ім'я, ПРІЗВИЩЕ)

Нормоконтролер _____ **Галина КОРНІЄНКО**
(підпис) (ім'я, ПРІЗВИЩЕ)

[1] не пізніше ніж за один тиждень до затвердженої дати захисту МД в ЕК

# РЕФЕРАТ

Магістерська дисертація за темою «Система диференціальної діагностики раку легень за даними комп'ютерної томографії» виконана студентом кафедри біомедичної кібернетики ФБМІ *Гломоздою Костянтином Юрійовичем зі спеціальності 122 «Комп'ютерні науки» за освітньо-професійною програмою «Комп'ютерні технології в біології та медицині»* та складається зі вступу; чотирьох розділів (огляд сучасних систем диференціальної діагностики раку легень за даними комп'ютерної томографії, засоби реалізації програмного застосунку для диференціальної діагностики раку легень, обґрунтування вибору технічних рішень для програмного застосунку для диференціальної діагностики раку легень, програмна реалізація та методика роботи програмного застосунку для диференціальної діагностики раку легень; розділу узагальнення результатів роботи); розділу зі стартап-проєкту; висновків до кожного з цих розділів; загальних висновків; списку використаних джерел, який налічує 142 джерела. Загальний обсяг роботи 120 сторінки.

***Актуальність теми.***

Рак легень залишається провідною причиною онкологічної смертності у світі, при цьому рання діагностика критично впливає на виживаність пацієнтів. Комп'ютерна томографія є основним методом скринінгу та діагностики, однак інтерпретація зображень вимагає високої кваліфікації спеціалістів та значних часових ресурсів. Сучасні системи автоматизованої діагностики на основі глибокого навчання демонструють високу точність, але мають обмежену інтерпретованість результатів, що ускладнює їх прийняття медичною спільнотою. Водночас класичні методи машинного навчання забезпечують прозорість прийняття рішень через аналіз конкретних ознак зображень. Інтеграція інтелектуальних агентів на базі великих мовних моделей дозволяє створити природномовний інтерфейс взаємодії, який знижує поріг входження для медичних працівників. Поєднання ансамблевих методів

класифікації з конверсаційними агентами становить актуальний напрямок розвитку медичних діагностичних систем.

***Мета і задачі дослідження.***

Метою роботи є підвищення ефективності диференціальної діагностики раку легень шляхом створення автоматизованої платформи на основі інтелектуального агента та ансамблевих методів машинного навчання.

Сформульовано наступні **задачі**:

1. Розробити інтерфейс користувача для перегляду медичних зображень з інтерактивним розмічуванням областей інтересу;
2. Створити модуль екстракції характеристик інтенсивності, текстури та форми з медичних зображень;
3. Навчити три алгоритми класифікації та розробити систему ансамблевого об’єднання результатів;
4. Реалізувати автоматичне виявлення підозрілих областей на зображеннях; інтегрувати велику мовну модель та побудувати агентну систему з динамічним вибором інструментів;
5. Додати конверсаційний інтерфейс для діалогу з агентом та відображення результатів діагностики.

***Об’єкт дослідження*.**

Процес автоматизованої діагностики онкологічних захворювань легень на основі аналізу комп’ютерної томографії з використанням інтелектуальних агентів та ансамблевих методів машинного навчання.

***Предмет дослідження***.

Методи класифікації патологічних утворень легень на зображеннях комп'ютерної томографії з використанням ансамблевих алгоритмів машинного навчання та оркестрації робочих процесів інтелектуальних агентів.

***Методи дослідження***.

Машинне навчання (Random Forest, Support Vector Machine, XGBoost), текстурний аналіз (матриця співвиникнення рівнів сірого, локальні бінарні патерни), морфологічний аналіз форми (моменти Ху), порогова сегментація

методом Otsu, ансамблеве голосування, оркестрація робочих процесів через LangGraph, function calling великих мовних моделей, асинхронне програмування, мікросервісна архітектура.

***Практичне значення одержаних результатів.***

Робота була виконана в рамках ініціативної наукової роботи кафедри БМК НДР д/р № 0123U100866 «Методи та моделі ідентифікації станів об’єктів у задачах прийняття медичних рішень». Зареєстровано 06-02-2023 [47]. Договір про співпрацю №Д/0002.01/3400.02/5/2023 від 05 січня 2023 року між КПІ ім. Ігоря Сікорського та ДУ "Національний інститут фтизіатрії і пульмонології ім. Ф.Г. Яновського НАМН України" [142]. Довідка надається.

***Апробація результатів дослідження*** непередбачено

***Публікації.*** непередбачено

***Ключові слова.***

Диференціальна діагностика, комп’ютерна томографія, рак легень, інтелектуальні агенти, ансамблеве машинне навчання, великі мовні моделі, класифікація медичних зображень, текстурний аналіз, LangGraph, конверсаційний інтерфейс.

**Бібліографічний опис МД.**

Гломозда, К. Ю. Система диференціальної діагностики раку легень на основі комп’ютерної томографії з використанням інтелектуальних агентів та ансамблевих методів машинного навчання : магістерська дис. : 122 Комп’ютерні науки / Гломозда Костянтин Юрійович. – Київ, 2025. – 153 с.

# ABSTRACT

Master’s thesis on the topic "System for differential diagnosis of lung cancer based on computed tomography data" was completed by student of the Department of Biomedical Cybernetics, Faculty of Biomedical Engineering, Hlomozda Kostiantyn in specialty 122 "Computer Science" under the educational and professional program "Computer Technologies in Biology and Medicine" and consists of an introduction; four chapters (review of modern systems for differential diagnosis of lung cancer based on computed tomography data, means of implementing a software application for differential diagnosis of lung cancer, justification of the choice of technical solutions for a software application for differential diagnosis of lung cancer, software implementation and methodology of the software application for differential diagnosis of lung cancer; section summarizing the results of the work); startup project chapter; conclusions for each chapter; general conclusions; list of references containing 142 sources. Total volume of the work is 120 pages.

**Relevance of the topic.**

Lung cancer remains the leading cause of oncological mortality worldwide, with early diagnosis critically affecting patient survival rates. Computed tomography serves as the primary screening and diagnostic method, however image interpretation requires high specialist qualification and significant time resources. Modern automated diagnostic systems based on deep learning demonstrate high accuracy but have limited interpretability of results, complicating their acceptance by medical community. Meanwhile, classical machine learning methods ensure decision-making transparency through analysis of specific image features. Integration of intelligent agents based on large language models enables creation of natural language interaction interface, lowering entry barrier for medical professionals. Combination of ensemble classification methods with conversational agents represents relevant direction in medical diagnostic systems development.

**Research goal and objectives.**

The aim of the work is to improve the efficiency of lung cancer differential diagnosis by creating an automated platform based on an intelligent agent and ensemble machine learning methods.

Following objectives were formulated:

1. Develop user interface for medical image viewing with interactive region of interest annotation;
2. Create module for extracting intensity, texture and shape characteristics from medical images;
3. Train three classification algorithms and develop ensemble result combination system;
4. Implement automatic suspicious region detection on images; integrate large language model and build agent system with dynamic tool selection;
5. Add conversational interface for agent dialogue and diagnostic results display.

**Research object.**

Process of automated lung oncological disease diagnosis based on computed tomography analysis using intelligent agents and ensemble machine learning methods.

**Research subject.**

Methods of lung pathological formation classification on computed tomography images using ensemble machine learning algorithms and intelligent agent workflow orchestration.

**Research methods.**

Machine learning (Random Forest, Support Vector Machine, XGBoost), texture analysis (gray level co-occurrence matrix, local binary patterns), morphological shape analysis (Hu moments), Otsu threshold segmentation, ensemble voting, workflow orchestration via LangGraph, large language model function calling, asynchronous programming, microservice architecture.

**Practical significance of obtained results.**

The work was performed within the framework of the initiative research work of the Department of Biomedical Cybernetics research project No. 0123U100866 "Methods and Models for Object State Identification in Medical Decision-Making Tasks". Registered on 06-02-2023. Cooperation agreement No. D/0002.01/3400.02/5/2023 dated January 5, 2023 between Igor Sikorsky Kyiv Polytechnic Institute and State Institution "National Institute of Phthisiology and Pulmonology named after F.G. Yanovskyi of the National Academy of Medical Sciences of Ukraine" [142]. Certificate is provided.

**Approbation of research results.** Not planned.

**Publications.** Not planned.

**Keywords.**

Differential diagnosis, computed tomography, lung cancer, intelligent agents, ensemble machine learning, large language models, medical image classification, texture analysis, LangGraph, conversational interface.

# ЗМІСТ

# ВСТУП

Рак легень залишається провідною причиною онкологічної смертності у світі, при цьому рання діагностика критично впливає на виживаність пацієнтів. Комп’ютерна томографія є основним методом скринінгу та діагностики, однак інтерпретація зображень вимагає високої кваліфікації спеціалістів та значних часових ресурсів. Сучасні системи автоматизованої діагностики на основі глибокого навчання демонструють високу точність, але мають обмежену інтерпретованість результатів, що ускладнює їх прийняття медичною спільнотою. Водночас класичні методи машинного навчання забезпечують прозорість прийняття рішень через аналіз конкретних ознак зображень. Інтеграція інтелектуальних агентів на базі великих мовних моделей дозволяє створити природномовний інтерфейс взаємодії, який знижує поріг входження для медичних працівників. Поєднання ансамблевих методів класифікації з конверсаційними агентами становить актуальний напрямок розвитку медичних діагностичних систем, що робить дану роботу своєчасною та практично значущою.

***Мета і завдання роботи.***

Метою роботи є підвищення ефективності диференціальної діагностики раку легень шляхом створення автоматизованої платформи на основі інтелектуального агента та ансамблевих методів машинного навчання.

Її досягнення передбачає вирішення наступних завдань:

1. Аналіз вітчизняних та зарубіжних джерел щодо сучасних систем диференціальної діагностики раку легень на основі комп'ютерної томографії та технологій штучного інтелекту;
2. Розробка інтерфейсу користувача для перегляду медичних зображень з інтерактивним розмічуванням областей інтересу;
3. Створення модуля екстракції характеристик інтенсивності, текстури та форми з медичних зображень;

4. Навчання трьох алгоритмів класифікації та розробка системи ансамблевого об'єднання результатів;

5. Реалізація автоматичного виявлення підозрілих областей на зображеннях;

6. Інтеграція великої мовної моделі та побудова агентної системи з динамічним вибором інструментів;

7. Додавання конверсаційного інтерфейсу для діалогу з агентом та відображення результатів діагностики.

***Використані методи.***

У роботі застосовано методи машинного навчання, зокрема Random Forest, Support Vector Machine з радіально-базисним ядром та XGBoost для класифікації патологічних утворень легень. Для екстракції характеристик зображень використано текстурний аналіз через матрицю співвиникнення рівнів сірого та локальні бінарні патерни, морфологічний аналіз форми через моменти Ху, а також статистичний аналіз інтенсивності. Автоматичне виявлення областей інтересу реалізовано методом порогової сегментації Otsu з морфологічними операціями. Для об'єднання результатів класифікації застосовано ансамблеве голосування з оптимізованими ваговими коефіцієнтами. Оркестрацію робочих процесів інтелектуального агента здійснено через LangGraph з використанням function calling великих мовних моделей Azure OpenAI GPT-5. Архітектура системи побудована на принципах мікросервісного підходу з асинхронним програмуванням через Python asyncio для забезпечення високої продуктивності.

***Отримані результати.***

Розроблено повнофункціональну систему диференціальної діагностики раку легень, що включає переглядач комп’ютерної томографії формату DICOM з інструментами інтерактивного маскування областей інтересу, модуль екстракції від дев’яноста до ста характеристик медичних зображень та ансамблевий класифікатор з паралельним виконанням трьох алгоритмів

машинного навчання. Реалізовано інтелектуального агента на базі LangGraph з динамічним вибором діагностичних інструментів та конверсаційним інтерфейсом українською мовою. Система забезпечує повний цикл діагностики від завантаження зображень до отримання інтерпретованих результатів класифікації шістнадцяти типів онкологічних патологій за шістсот-дві тисячі двохсот мілісекунд. Ансамблевий підхід демонструє підвищену надійність передбачень порівняно з окремими алгоритмами через балансування специфічних переваг Random Forest, Support Vector Machine та XGBoost. Мікросервісна архітектура з трьома незалежними сервісами забезпечує масштабованість компонентів та гнучкість розвитку системи.

***Апробація результатів роботи.*** Не заплановано.

***Публікації.*** Не заплановано.

***Структура роботи.***

Магістерська дисертація за темою "Система диференціальної діагностики раку легень за комп'ютерною томографією з використанням інтелектуальних агентів" виконана студентом *Гломоздою Костянтином Юрійовичем зі спеціальності 122 «Комп'ютерні науки» за освітньо-професійною програмою «Комп'ютерні технології в біології та медицині»*, побудована за класичним типом та викладена на 151 сторінці машинописного тексту. Вона складається зі вступу; чотирьох розділів: огляд сучасних систем диференціальної діагностики раку легень за даними комп’ютерної томографії, засоби реалізації програмного застосунку для диференціальної діагностики раку легень, обґрунтування вибору технічних рішень для програмного застосунку для диференціальної діагностики раку легень, програмна реалізація та методика роботи програмного застосунку для диференціальної діагностики раку легень; розділу узагальнення результатів роботи); розділу зі стартап-проєкту; висновків до кожного з цих розділів; загальних висновків; списку використаних джерел, який налічує 142 джерела. У роботі представлено 9 рисунків.

# РОЗДІЛ 1
# ОГЛЯД СУЧАСНИХ СИСТЕМ ДИФЕРЕНЦІАЛЬНОЇ ДІАГНОСТИКИ РАКУ ЛЕГЕНЬ ЗА ДАНИМИ КОМП'ЮТЕРНОЇ ТОМОГРАФІЇ

У даному розділі здійснено систематичний огляд сучасного стану розвитку систем штучного інтелекту для діагностики патологій легень, розроблених провідними світовими технологічними компаніями та науково-дослідними центрами. Розглянуто технічні характеристики, архітектурні особливості, набори даних для навчання та результати клінічної валідації десяти ключових платформ, що сформували ландшафт сучасної автоматизованої радіології. Особливу увагу приділено аналізу еволюції алгоритмічних підходів від дослідницьких прототипів до сертифікованих медичних виробів, а також оцінці їхньої функціональності в реальних клінічних умовах. Це дозволяє сформувати цілісне уявлення про існуючі методи автоматизації аналізу комп'ютерної томографії та рентгенографії, а також визначити базові принципи побудови сучасних діагностичних систем.

## 1.1 Наскрізна система скринінгу раку легень від Google Health

Система скринінгу раку легень від Google Health та Northwestern Medicine, представлена в Nature Medicine у дві тисячі дев'ятнадцятому році, продемонструвала результативність на рівні або вище провідних радіологів. Ця розробка стала значним досягненням у використанні штучного інтелекту для високоточної медичної діагностики [1, 2, 3].

Архітектура поєднує модуль Inflated Inception для аналізу тривимірних об'ємів та Mask R-CNN для точної локалізації підозрілих ділянок у легеневій тканині. Система функціонує як статичний конвеєр обробки даних і не має

агентних можливостей для динамічної адаптації під конкретні клінічні випадки [1, 2].

Модель навчалася на масиві з сорока двох тисяч досліджень, що забезпечило високу надійність розпізнавання різноманітних патологій. Під час валідації на шести тисячах незалежних випадків система досягла показника точності дев'яносто чотири цілих чотири десятих відсотка [1, 3, 4].

Штучний інтелект виявив на п’ять відсотків більше онкологічних випадків та на одинадцять відсотків рідше помилявся у здорових пацієнтів порівняно з досвідченими лікарями. Попри успішні результати, система поки не схвалена регулятором для клінічного застосування і залишається виключно дослідницьким інструментом [1, 2, 3].

## 1.2 Платформа Alibaba DAMO Academy для виявлення легеневих вузлів

Система виявлення легеневих вузлів від Alibaba DAMO Academy здобула першість на змаганні LUNA16 у липні дві тисячі сімнадцятого року. Алгоритм продемонстрував повноту відгуку вісімдесят дев’ять цілих сім десятих відсотка, що стало найкращим показником серед учасників. Цей результат підтвердив високу ефективність глибокого навчання у медичній діагностиці, яка відповідає рівню кваліфікованих спеціалістів [6, 7].

Архітектура базується на тривимірній згортковій мережі з використанням модулів ResNet, ResNeXt та U-Net для точної сегментації. Система застосовує мультимасштабне прогнозування та двоетапну фільтрацію для зменшення кількості хибнопозитивних висновків. Це статичний конвеєр обробки даних, який не містить елементів агентної архітектури та не здатен динамічно адаптувати стратегію аналізу [6, 8].

Навчання моделі проводилося на стандартизованому наборі даних LUNA16, що містить вісімсот вісімдесят вісім сканів із розміченими вузлами. Згодом технологію успішно адаптували для діагностики COVID-19,

досягнувши точності дев’яносто шість відсотків. Система була впроваджена у ста сімдесяти лікарнях Китаю, де обробила понад триста сорок тисяч випадків під час пандемії [8, 9, 10].

Рішення доступне медичним закладам через хмарну платформу Alibaba Cloud Healthcare. Попри масштабне використання в Китаї для боротьби з вірусом, система не отримала схвалення американського регулятора для діагностики раку легень. Це обмежує її комерційне застосування на ринку Сполучених Штатів Америки, хоча технічний потенціал залишається високим [6].

### 1.3 AI-Rad Companion Chest CT від Siemens Healthineers

Платформа AI-Rad Companion Chest CT від Siemens Healthineers для аналізу КТ грудної клітки отримала схвалення FDA двадцять шостого вересня дві тисячі дев’ятнадцятого року. Система має модульну архітектуру з трьома окремими реєстраційними номерами, що вимагало незалежної валідації компонентів. Це один із перших комерційних ШІ-продуктів, допущених до рутинного клінічного використання у світовій радіології [11, 12, 13, 14].

Технічна архітектура є фіксованим конвеєром зі спеціалізованих згорткових нейромереж. Для сегментації легень використовується V-Net, а виявлення вузлів базується на каскадній тривимірній мережі. Патології тканини аналізує DenseUNet, тоді як серцевий модуль оцінює коронарний кальцій. Система не має агентних властивостей, функціонуючи як послідовний процес без динамічної адаптації алгоритмів [12, 13, 11].

Навчання моделей базувалося на тисячах клінічних випадків для забезпечення високої узагальнюючої здатності. Дослідження Chamberlin продемонструвало стовідсоткову чутливість виявлення легеневих вузлів при специфічності сімдесят цілих вісім десятих відсотка. Система досягла площі під кривою нуль цілих дев’ятсот сорок дві тисячних у прогнозуванні злоякісності утворень [12, 13, 14, 15].

Комерційний продукт сертифіковано FDA та CE Mark для глобального клінічного використання. Впровадження системи скоротило час аналізу складних випадків на сімдесят вісім відсотків. Платформа автоматично генерує структуровані звіти з кількісними показниками для перевірки лікарем, дозволяючи радіологам зосередитися на прийнятті клінічних рішень [12, 13, 16, 17].

### 1.4 InferRead CT Lung від Infervision

InferRead CT Lung від Beijing Infervision Technology стала першою системою автоматичного виявлення легеневих вузлів, яку схвалило Управління з контролю за продуктами та ліками Сполучених Штатів Америки дев'ятого липня дві тисячі двадцятого року. Це дозволило проводити аналіз без попередньої ручної розмітки. У травні дві тисячі двадцять п’ятого року функціонал було розширено покращеними алгоритмами оцінки ризику злоякісності [18, 19, 20].

Технічна основа платформи складається з трьох нейронних мереж у фіксованому конвеєрі. Для отримання ознак застосовується DenseNet, а Faster R-CNN виконує локалізацію та класифікацію об’єктів. Прогнозування ступеня ризику реалізовано на базі архітектури ResNet-34. Система функціонує як послідовний алгоритм без можливостей автономної агентної поведінки чи динамічного вибору стратегій [18, 19, 20].

Навчання проводилося на масиві з одинадцяти тисяч досліджень, зібраних у різних лікарнях, що гарантує стабільну роботу з різними томографами. Окрема вибірка для оцінки ризику містила понад вісім тисяч верифікованих вузлів. Валідація підтвердила рівень виявлення дев'яносто два цілих шість десятих відсотка та високу дискримінаційну здатність із площею під кривою до нуль цілих дев’яносто п’яти сотих [21, 22, 23].

Система активно використовується у трьохстах вісімдесяти клініках світу, обробляючи щоденно близько п’ятдесяти п’яти тисяч випадків. Продукт

має сертифікати відповідності для європейського ринку, Великої Британії та Китаю. Це підтверджує надійність технології та її статус провідного комерційного рішення для діагностики легеневих патологій [22, 23].

### 1.5 Lunit INSIGHT CXR для виявлення легеневих вузлів

Lunit INSIGHT CXR, розроблена у дві тисячі вісімнадцятому році південнокорейською компанією Lunit, є системою штучного інтелекту для аналізу рентгенограм. У дві тисячі двадцять першому році вона отримала схвалення Управління з контролю за продуктами та ліками Сполучених Штатів Америки для автоматичного пріоритизування досліджень. Система одночасно виявляє десять типів патологій, включаючи вузли та пневмоторакс, що робить її універсальним інструментом для скринінгу та невідкладної діагностики [22, 23].

Технічна архітектура базується на мережі ResNet-34 (див. рис. 1.1) із мультиканальним вихідним шаром для незалежних прогнозів аномалій. Навчання проходить у два етапи: спочатку на рівні фрагментів для локальних ознак, а потім на повному зображенні для врахування глобального контексту. Модель має фіксований конвеєр обробки без агентних елементів та можливості динамічної адаптації стратегії аналізу [22, 23].

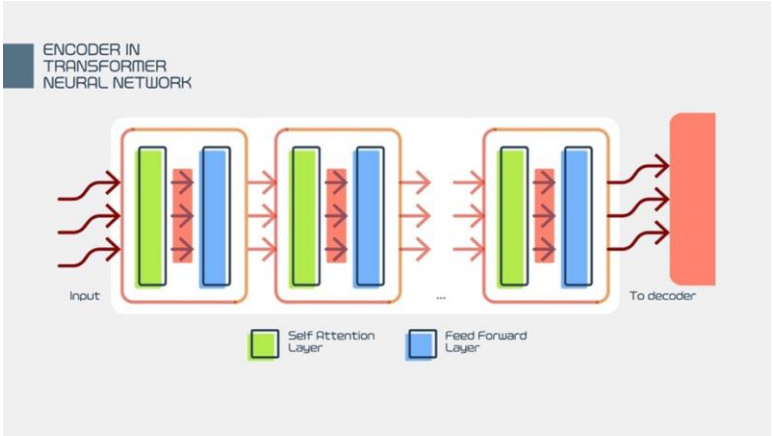


Рисунок 1.1 – Технічна архітектура Lunit INSIGHT CXR [143]

Навчання проводилося на трьох із половиною мільйонах рентгенограм, а валідацію виконано на ста шістдесяти тисячах пар знімків. У дослідженні дві тисячі двадцять четвертого року система перевершила сімнадцять із вісімнадцяти радіологів, досягнувши площі під кривою нуль цілих дев'яносто три соті для виявлення вузлів. Чутливість діагностики плеврального випоту становить до дев’яноста шести відсотків при високій специфічності [21, 23].

Lunit INSIGHT CXR ідентифікує вузли від чотирьох міліметрів і використовується у трьох тисячах закладів сорока країн. У Південній Кореї система щороку обробляє чотири мільйони знімків, маючи сертифікацію на ринках Сполучених Штатів Америки, Європи та Азії. Інтеграція з існуючими медичними системами дозволяє впровадження у клінічний процес без суттєвих змін інфраструктури [22, 23].

### 1.6 Arterys MICA Platform для онкології легенів

Arterys MICA Platform з модулем Oncology AI Lung є хмарною системою штучного інтелекту для аналізу раку легенів, яка однією з перших отримала дозвіл FDA на роботу через хмару. Базова платформа була схвалена у листопаді дві тисячі сімнадцятого року, а спеціалізований модуль — у лютому дві тисячі вісімнадцятого. У жовтні дві тисячі двадцять другого року компанію придбала Tempus AI для інтеграції радіологічних даних із мультимодальною клінічною інформацією [24, 25].

Технічна архітектура складається з трьох послідовних етапів обробки даних у фіксованому конвеєрі. Перший компонент використовує двовимірну U-Net для пошуку кандидатів, другий застосовує 2.5D ResNet для класифікації та фільтрації хибних результатів, а третій залучає тривимірну ENet для точної сегментації. Система не має агентних властивостей і не здатна автономно обирати алгоритми під час аналізу [24, 25].

Навчання проводилося на базі даних LIDC-IDRI, що містить понад тисячу КТ-досліджень. Валідація показала чутливість дев’яносто чотири

відсотки для вузлів розміром понад шість міліметрів із середнім показником чотири хибнопозитивних виявлення на сканування. Точність сегментації досягла коефіцієнта Діса нуль цілих вісімдесят три соті, що співмірно з результатами експертів [24, 25].

Платформа функціонує на хмарній інфраструктурі Amazon Web Services і доступна у понад ста країнах із сертифікатами FDA та CE Mark. Вона інтегрується з існуючими медичними архівами через протоколи DICOM без потреби у локальному обладнанні. У складі Tempus AI система отримала додаткові можливості поєднання візуалізації з геномними даними для персоналізованої онкології [24, 25].

### 1.7 CheXNet та CheXNeXt від Стенфордського університету

CheXNet та CheXNeXt є дослідницькими системами штучного інтелекту для аналізу рентгенограм, створеними групою машинного навчання Стенфордського університету. Перша модель з'явилася в листопаді дві тисячі сімнадцятого року для виявлення пневмонії, а друга вийшла через рік із розширеним функціоналом для діагностики чотирнадцяти патологій. Ці розробки стали важливими прототипами, що продемонстрували потенціал автоматизованої медицини та стимулювали наукові дослідження завдяки відкритому доступу до даних [26, 27, 28, 29].

Технічна основа базується на глибокій нейронній мережі DenseNet зі ста двадцяти одного шару, що використовує щільні з'єднання для ефективного навчання. Для прискорення процесу застосовано трансферне навчання на базі ImageNet, а в новішій версії впроваджено двоетапне тренування для покращення якості розмітки. Системи функціонують як фіксовані конвеєри глибокого навчання і не мають елементів агентної архітектури для автономної адаптації стратегії аналізу [26, 27, 28].

Навчання проводилося на масиві ChestX-ray14 зі ста дванадцяти тисяч знімків, а тестування відбувалося на наборі з консенсусними оцінками

експертів. CheXNet перевершила радіологів у діагностиці пневмонії, тоді як CheXNeXt показала високу точність виявлення вузлів та мас із чутливістю, що перевищує людські показники. Швидкість обробки зображень алгоритмами виявилася приблизно у сто шістдесят разів вищою за роботу лікаря [26, 27, 28].

Системи залишаються виключно дослідницькими прототипами і не мають регуляторного схвалення для клінічного використання. Попри це, роботи Стенфордської команди здійснили значний вплив на наукову спільноту та отримали тисячі цитувань. Часткове відкриття коду та даних сприяло демократизації досліджень, дозволивши вченим у всьому світі розвивати технології медичної візуалізації [26, 28, 29].

## 1.8 Qure.ai qXR та qCT LN Quant для діагностики легеневих патологій

Платформа Qure.ai від індійської компанії включає продукти qXR для рентгенограм та qCT LN Quant для аналізу КТ-зображень. Система qXR стала першим у світі інструментом такого типу, що отримав сертифікацію CE Mark у дві тисячі вісімнадцятому році. У серпні дві тисячі двадцять четвертого року модуль qCT LN Quant отримав схвалення FDA, розширивши можливості платформи на тривимірний аналіз даних [30, 31].

Технічна архітектура базується на згорткових нейромережах, оптимізованих для обробки високороздільних зображень на стандартному обладнанні. Це єдина інтегрована модель із фіксованим конвеєром без елементів агентної архітектури. Система qXR здатна виявляти та локалізувати тридцять різних патологій, включаючи легеневі вузли та пневмоторакс, що робить її універсальним інструментом скринінгу [30, 31].

Моделі навчалися на масиві з понад чотирьох мільйонів рентгенограм, зібраних у п’ятнадцяти установах світу. Клінічні дослідження показали точність понад дев'яносто відсотків та зменшення кількості пропущених вузлів на сорок відсотків. Впровадження системи прискорило звітування на

сорок шість відсотків і покращило оцінку ризику злоякісності на двадцять чотири відсотки [30, 31].

Платформа впроваджена у понад тисячі закладів і обробила більше чотирьох з половиною мільйонів досліджень. Система має регуляторні дозволи FDA та CE Mark для глобального використання. Компанія співпрацює з провідними установами, такими як NHS та AstraZeneca, що підтверджує надійність інструменту для клінічної практики та наукових досліджень [30, 31].

### 1.9 Optellum Virtual Nodule Clinic з Lung Cancer Prediction CNN

Optellum Virtual Nodule Clinic з алгоритмом LCP-CNN є системою штучного інтелекту для оцінки ризику раку легень, розробленою британською компанією Optellum Limited. Двадцять третього березня дві тисячі двадцять першого року вона стала першим у світі схваленим FDA інструментом для діагностики цього захворювання. Платформа інтегрує алгоритми з клінічними даними для надання персоналізованих рекомендацій, переходячи від простого виявлення до повноцінної підтримки лікарських рішень [32, 33, 34].

Технічна архітектура базується на модифікованій мережі DenseNet-121, яка аналізує тривимірні ділянки розміром п’ятдесят шість міліметрів. На відміну від бінарних класифікаторів, система генерує безперервну оцінку ризику злоякісності від нуля до ста відсотків. Це статична модель глибокого навчання, що використовує восьмикратну перехресну валідацію, і не має елементів агентної архітектури для автономного вибору алгоритмів [32, 33, 34].

Навчання проводилося на даних Національного дослідження скринінгу раку легень, що містили понад п'ятнадцять тисяч вузлів. Внутрішня валідація показала площу під кривою дев'яносто дві цілих одна десята відсотка, що значно перевищує показники класичної моделі Брока. Зовнішні тестування та

клінічні дослідження підтвердили ефективність системи, яка покращила діагностичну продуктивність лікарів майже на сім пунктів [32, 33, 34].

Система широко впроваджена у медичних центрах США та має регуляторні дозволи FDA, CE-MDR та UKCA. Важливим успіхом стало отримання кодів відшкодування від Medicare, що дозволяє фінансову компенсацію за використання технології. Завдяки клінічній валідації та інтеграції в робочі процеси, Optellum позиціонується як один із найбільш зрілих продуктів на ринку діагностики онкології [32, 33, 34, 35].

### 1.10 VIDA Diagnostics LungPrint для комплексного аналізу легенів

VIDA Diagnostics LungPrint є платформою штучного інтелекту для кількісного аналізу фізіології легенів на основі КТ, розробленою американською компанією. Система була представлена у дві тисячі вісімнадцятому році та отримала схвалення FDA у жовтні дві тисячі двадцятого року. Її особливістю є комплексний підхід, що виходить за межі виявлення вузлів і включає аналіз тканин та дихальних шляхів для діагностики раку, ХОЗЛ та інших захворювань [36, 37].

Технічна архітектура автоматизує квантифікацію параметрів, виконуючи сегментацію часток легенів та виявлення патологій, таких як емфізема чи консолідації. Модуль аналізу дихальних шляхів вимірює діаметри просвітів, а технологія Hyperion View забезпечує тривимірну візуалізацію результатів. Платформа функціонує як статичний набір інструментів без агентної архітектури або динамічного вибору алгоритмів [36, 37].

Деталі навчальних даних є комерційною таємницею, проте клінічні випробування показали зменшення часу інтерпретації знімків на тридцять п’ять відсотків без втрати точності. Система валідована для тридцяти біомаркерів і відіграє важливу роль у плануванні хірургічних втручань при раку легенів. Автоматична характеризація вузлів дозволяє точніше оцінювати їх клінічну значущість у контексті загального стану пацієнта [36, 37].

Система має регуляторні схвалення FDA, CE Mark та інших агенцій, а також сертифікацію якості ISO 13485. Вона доступна через Nuance AI Marketplace та використовується у шести тисячах закладів по всьому світу. Завдяки партнерствам з TeraRecon та Blackford Analysis платформа легко інтегрується в існуючі радіологічні системи [36, 37].

## Висновок з розділу 1

У цьому розділі проаналізовано десять провідних систем штучного інтелекту для діагностики раку легень та виявлено, що існуючі системи діагностики раку легень досягли високого рівня технологічної зрілості, демонструючи діагностичну точність, яка часто перевищує показники лікарів-радіологів. Використання передових архітектур глибокого навчання дозволило створити надійні інструменти для автоматичного виявлення патологій, які успішно інтегровані в клінічну практику по всьому світу. Втім, критичний розгляд архітектури цих систем виявив суттєве спільне обмеження – вони реалізовані як статичні конвеєри з фіксованою логікою обробки даних. Відсутність здатності до динамічної адаптації стратегії аналізу та автономного вибору інструментів під конкретний клінічний випадок вказує на вичерпання потенціалу класичних підходів. Це створює обґрунтовану потребу в розробці нового класу діагностичних систем на основі агентної архітектури, здатних до інтелектуальної оркестрації множинних алгоритмів.

# РОЗДІЛ 2
# ЗАСОБИ РЕАЛІЗАЦІЇ ПРОГРАМНОГО ЗАСТОСУНКУ ДЛЯ ДИФЕРЕНЦІАЛЬНОЇ ДІАГНОСТИКИ РАКУ ЛЕГЕНЬ

У даному розділі здійснено комплексне обґрунтування вибору інструментальних засобів та архітектурних рішень для реалізації інтелектуальної системи діагностики патологій легень. Проведено порівняльний аналіз мов програмування, веб-фреймворків та бібліотек для побудови клієнтської частини, фокусуючись на специфічних вимогах медичних застосунків: швидкодії, надійності та можливості обробки великих масивів візуальних даних. Особливу увагу приділено вибору методів машинного навчання, інструментів оркестрації агентів та платформ великих мовних моделей, що забезпечують баланс між точністю класифікації та інтерпретованістю результатів. Також розглянуто підходи до організації баз даних, паралельних обчислень та розгортання системи, що дозволяє сформувати оптимальний технологічний стек для створення масштабованого діагностичного комплексу.

## 2.1 Порівняльний аналіз мов програмування для платформ медичної діагностики

Вибір мови програмування для створення медичних систем зі штучним інтелектом є ключовим архітектурним рішенням. Сучасна сфера розробки пропонує три основні варіанти: Python, Java та C++, кожен з яких має власні переваги для задач візуалізації та машинного навчання [38].

Python є динамічною мовою з розвиненою екосистемою бібліотек для обробки зображень та алгоритмів класифікації. Вона дозволяє створювати прототипи значно швидше за конкурентів, хоча й поступається у швидкості виконання складних обчислень. Вбудовані механізми асинхронності

забезпечують ефективну обробку запитів у багатокористувацьких веб-сервісах [39, 40].

Java забезпечує незалежність від платформи та високу продуктивність завдяки компіляції коду під час виконання програми. Статична типізація зменшує ризик помилок у критичних системах, проте суттєво ускладнює та уповільнює процес розробки. Цю технологію часто обирають для створення масштабованих корпоративних рішень у сфері охорони здоров'я [41, 42].

C++ надає прямий доступ до апаратного забезпечення для досягнення максимальної швидкості обчислень. Спеціалізовані бібліотеки дозволяють виконувати складну тривимірну обробку та сегментацію медичних зображень з високою ефективністю. Водночас складність синтаксису та ручне керування пам'яттю підвищують ризики збоїв та час створення продукту [43, 44].

Критичною вимогою для сучасних систем є підтримка асинхронного виконання операцій для ефективного використання ресурсів. Python пропонує зручні вбудовані інструменти для неблокуючої обробки, тоді як інші мови реалізують паралелізм через складніші механізми керування потоками [43, 44].

Остаточний вибір технології залежить від балансу між швидкістю розробки та швидкодією системи. Python є оптимальним для швидкого прототипування, тоді як для високих навантажень застосовують гібридні підходи з використанням C++. Java залишається стандартом для надійних корпоративних інфраструктурних рішень [45, 46, 47].

## 2.2 Архітектурні підходи до побудови серверних компонентів медичних систем

Побудова серверної частини вимагає обґрунтованого вибору інструментів для обробки запитів та інтеграції з сервісами штучного інтелекту. Екосистема Пайтон пропонує спектр рішень від комплексних систем до мінімалістичних бібліотек. Головними вимогами для медичних платформ є

швидкість роботи, підтримка паралельних з'єднань, документування та валідація даних [48].

Архітектурним фундаментом виступає вибір між послідовною та асинхронною моделями. Асинхронний підхід дозволяє ефективно обслуговувати тисячі користувачів без блокування ресурсів під час тривалих операцій. Це критично для систем, що працюють з об'ємними знімками томографії та зовнішніми алгоритмами класифікації [49].

Джанго є повнофункціональним рішенням із вбудованими засобами адміністрування та захисту інформації. Його структура гарантує високий рівень безпеки та стандартизацію розробки інтерфейсів. Проте орієнтація на синхронну роботу та значна ресурсоємність можуть знижувати швидкодію у порівнянні з легшими альтернативами [50, 53].

Фласк пропонує гнучку мікроархітектуру з мінімальним набором функцій, залишаючи вибір компонентів за розробником. Це дозволяє створювати оптимізовані сервіси, але вимагає значних зусиль на налаштування інфраструктури. Використання синхронних протоколів обмежує продуктивність при високих навантаженнях, попри існування асинхронних адаптацій [51].

ФастАПІ забезпечує максимальну продуктивність та автоматичну перевірку даних завдяки асинхронній архітектурі. Він самостійно генерує документацію та значно випереджає аналоги у швидкості обробки. Попри відсутність вбудованих інструментів для баз даних, це рішення є оптимальним для високонавантажених систем [48, 52, 53].

Вибір технології має базуватися на вимогах мікросервісної архітектури та специфіці медичних даних. Асинхронна модель є перевагою для обробки важких зображень та взаємодії з повільними алгоритмами аналізу. Автоматична валідація та наявність документації є визначальними для надійності платформи [54].

## 2.3 Підходи до створення клієнтських інтерфейсів медичних діагностичних систем

Розробка інтерфейсу медичних платформ вимагає вибору між JavaScript-фреймворками та спеціалізованими Python-бібліотеками. Перші забезпечують повний контроль над структурою застосунку, тоді як другі дозволяють швидко створювати інтерактивні рішення без написання клієнтського коду. Рішення залежить від пріоритетів швидкості реалізації, гнучкості налаштування та наявності інструментів візуалізації [55].

Традиційні JavaScript-фреймворки надають розробникам повний контроль над архітектурою інтерфейсу. React використовує компонентну модель, Angular гарантує надійність завдяки типізації, а Vue пропонує легку інтеграцію. Використання цих інструментів вимагає створення окремого програмного інтерфейсу та написання значного обсягу коду для реалізації взаємодії з серверною частиною [56, 59].

Python-бібліотеки дозволяють розробляти повнофункціональні застосунки виключно засобами мови Python. Gradio спеціалізується на демонстрації моделей машинного навчання та містить вбудовані інструменти для маскування зображень, що є критичним для медичних задач. Вона також забезпечує просте поширення прототипів через публічні посилання без необхідності складного розгортання [55, 57].

Streamlit фокусується на створенні аналітичних панелей та візуалізації даних за допомогою реактивної моделі програмування. Бібліотека інтегрується з інструментами побудови графіків, проте має обмежені можливості для редагування зображень порівняно з аналогами. Обидва рішення забезпечують пряму взаємодію з екосистемою Python без необхідності створення проміжних шарів серіалізації [58].

Важливим критерієм вибору технології є наявність компонентів для роботи з медичними зображеннями. JavaScript-фреймворки потребують інтеграції сторонніх бібліотек для перегляду томографічних зрізів та реалізації

власної логіки маскування. Python-інструменти пропонують готові рішення, хоча їхня функціональність може поступатися спеціалізованим медичним переглядачам [59].

Швидкість розробки є принциповим фактором для проектів з обмеженими ресурсами. Python-бібліотеки дозволяють створювати робочі прототипи за лічені години без складного налаштування інфраструктури. Натомість JavaScript-фреймворки вимагають більше часу на початковому етапі, але забезпечують гнучкість дизайну для промислових систем [60].

Моделі обробки взаємодій користувача суттєво відрізняються залежно від обраного підходу. Python-бібліотеки використовують автоматичну синхронізацію стану, тоді як JavaScript вимагає явного програмування обміну даними через мережеві запити. Правильний вибір механізму керування станом є критичним для надійного збереження масок регіонів інтересу при навігації [61].

## 2.4 Підходи до оркестрації багатокрокових робочих процесів агентів штучного інтелекту

Створення інтелектуальних діагностичних систем вимагає координації дій від розпізнавання намірів до пояснення результатів. Існують підходи на основі ациклічних графів для послідовних операцій, скінченних автоматів для дискретних станів та графів керування для динамічних процесів. Вибір методу залежить від складності сценарію, необхідності зворотних переходів та вимог до прозорості функціонування [62].

Архітектурні шаблони визначають методи координації агентів. Послідовна модель забезпечує лінійну залежність етапів, тоді як паралельна дозволяє одночасне виконання незалежних алгоритмів класифікації. Ієрархічний підхід використовує керівні модулі для управління підлеглими структурами, а групова взаємодія забезпечує узгодження рішень кількох експертних систем через спільний простір обміну повідомленнями [63].

Сучасні інструменти оркестрації пропонують різні рівні контролю над процесами. ЛангЧейн використовує послідовні ланцюжки з централізованим управлінням, тоді як ЛангГраф впроваджує графову модель з підтримкою циклів для прозорого виконання. АвтоГен базується на асинхронному обміні повідомленнями для дослідницьких цілей, а КрюАІ спрощує командну роботу через рольову модель, хоча й обмежує гнучкість структури [64].

Управління станом забезпечує збереження контексту та відновлення роботи після технічних збоїв. Персистентність реалізується через запис історії та метаданих у бази даних, а контрольні точки дозволяють повертатися до стабільних станів. Для медичних систем це критично важливо, оскільки тривалі діагностичні процеси вимагають фіксації всіх проміжних результатів для юридичного аудиту [65].

Інтеграція людського контролю є обов'язковою вимогою для медичних систем, де фінальне рішення залишається за фахівцем. Механізми участі людини дозволяють призупиняти процес для отримання підтвердження або додаткових даних. Системи надійності забезпечують повторні спроби виконання, альтернативну маршрутизацію завдань та контрольоване зниження функціональності у разі критичних помилок [66].

## 2.5 Критерії вибору платформи великих мовних моделей для медичних діагностичних систем

Інтеграція мовних моделей у медицину вимагає ретельного аналізу технічних та безпекових критеріїв доступних платформ. Ринок пропонує хмарні рішення від технологічних гігантів, що гарантують масштабованість через програмні інтерфейси, або моделі з відкритим кодом для локального розгортання на обладнанні установи. Гібридний підхід дозволяє поєднувати потужність хмарних обчислень для складних задач із захищеністю локальної обробки чутливих даних [67].

Захист медичної інформації є критичною вимогою, що регулюється суворими міжнародними стандартами конфіденційності та прозорості обробки даних. Хмарні платформи корпоративного рівня зазвичай надають сертифікацію відповідності основним регуляторним режимам Європи та Сполучених Штатів Америки. Локальне використання моделей забезпечує повний контроль над інформацією без передачі третім сторонам, проте покладає відповідальність за аудит безпеки безпосередньо на медичну установу [68].

Фізичне розташування серверів безпосередньо впливає на затримку відповідей та якість взаємодії з персоналом. Значні часові затримки порушують плавність діалогу та можуть призвести до втрати контексту роботи системи. Хмарні провайдери дозволяють обирати найближчий регіон для оптимізації швидкості, тоді як локальне розміщення обчислювальних потужностей гарантує мінімально можливий час відгуку [69].

Архітектурна гнучкість системи залежить від механізмів інтеграції зовнішніх інструментів у процес генерації відповідей. Сучасні платформи підтримують концепцію виклику функцій, де модель формує структуровані інструкції для виконання конкретних дій з відповідними параметрами. Провідні хмарні сервіси реалізують цей функціонал на рівні програмних інтерфейсів, тоді як моделі з відкритим кодом часто потребують додаткового налаштування або використання проміжних систем оркестрації [70].

Економічна ефективність рішень суттєво різниться залежно від обраної моделі розгортання та обсягів використання. Хмарні сервіси використовують тарифікацію за обсяг оброблених даних, що є вигідним при змінному або помірному навантаженні на систему. Власні обчислювальні потужності вимагають значних капітальних інвестицій та стають економічно доцільними лише для великих центрів із стабільно високою інтенсивністю діагностичних процесів [71].

Придатність платформи для національної системи охорони здоров’я визначається якістю підтримки державної мови. Більшість моделей

оптимізовані для англійської мови, хоча передові рішення демонструють прийнятні результати для поширених європейських мов. Для забезпечення високої точності медичної термінології часто необхідне додаткове налаштування моделей на спеціалізованих корпусах текстів цільовою мовою [72].

## 2.6 Підходи до організації персистентності даних у медичних діагностичних системах

Організація зберігання даних вимагає вибору архітектури системи керування базами даних відповідно до потреб медичних застосунків. Вбудовані рішення функціонують як локальні бібліотеки, клієнт-серверні системи використовують окремі процеси для мережевої взаємодії, а розподілені платформи забезпечують масштабування через реплікацію. Остаточний вибір залежить від обсягу інформації та вимог до відмовостійкості [73].

Вбудовані системи пропонують спрощене розгортання для установ з обмеженими ресурсами завдяки відсутності складного адміністрування. Зберігання інформації у єдиному файлі полегшує резервне копіювання та гарантує високу швидкодію операцій. Головним недоліком є неможливість ефективної роботи у багатокористувацькому режимі [74].

Клієнт-серверна архітектура забезпечує масштабування та одночасний доступ користувачів завдяки механізмам блокування та керування з’єднаннями. Підтримка реплікації гарантує високу доступність сервісів та збереження критичних даних. Експлуатація таких систем потребує кваліфікованого адміністрування та регулярного моніторингу ресурсів [75, 76].

Транзакційні властивості атомарності, узгодженості, ізоляції та довговічності є фундаментом надійності медичних систем. Вони гарантують цілісність складних записів та коректне завершення операцій навіть у випадку

технічних збоїв. Реалізація цих принципів здійснюється через механізми попереднього журналювання змін [76].

Асинхронний доступ до бази даних підвищує пропускну здатність сервера шляхом неблокуючого виконання операцій введення-виведення. Ця модель є оптимальною для високонавантажених систем, хоча й вимагає складнішої реалізації програмного коду. Різні бібліотеки надають інструменти для адаптації синхронних викликів до асинхронного контексту [77].

Об’єктно-реляційне відображення дозволяє працювати з базами даних через об’єктні конструкції мови програмування без написання прямих запитів. Бібліотеки пропонують декларативні або імперативні методи опису зв’язків між класами та таблицями. Основним викликом залишається структурна невідповідність між об’єктною та реляційною моделями даних [78].

Патерн репозиторію відокремлює логіку доступу до даних від бізнес-правил застосунку через абстрактні інтерфейси. Це спрощує тестування та дозволяє змінювати технології зберігання без модифікації основного коду. Альтернативні підходи, такі як одиниця роботи, допомагають координувати транзакції для групи пов’язаних операцій [79].

Проектування схеми бази даних вимагає балансу між нормалізацією для уникнення дублювання та оптимізацією швидкодії. Медичні системи використовують складну структуру зв’язків для організації діагностичної інформації та результатів класифікації. Використання індексів прискорює пошук даних, проте збільшує час виконання операцій запису [80].

## 2.7 Методи обробки медичних томографічних зображень

Обробка томографічних зображень вимагає комплексного підходу до читання форматів, конвертації даних, сегментації структур та виділення ознак. Вибір інструментів базується на необхідності збереження діагностичної цінності та забезпеченні швидкодії інтерактивних систем (див рис. 2.1).

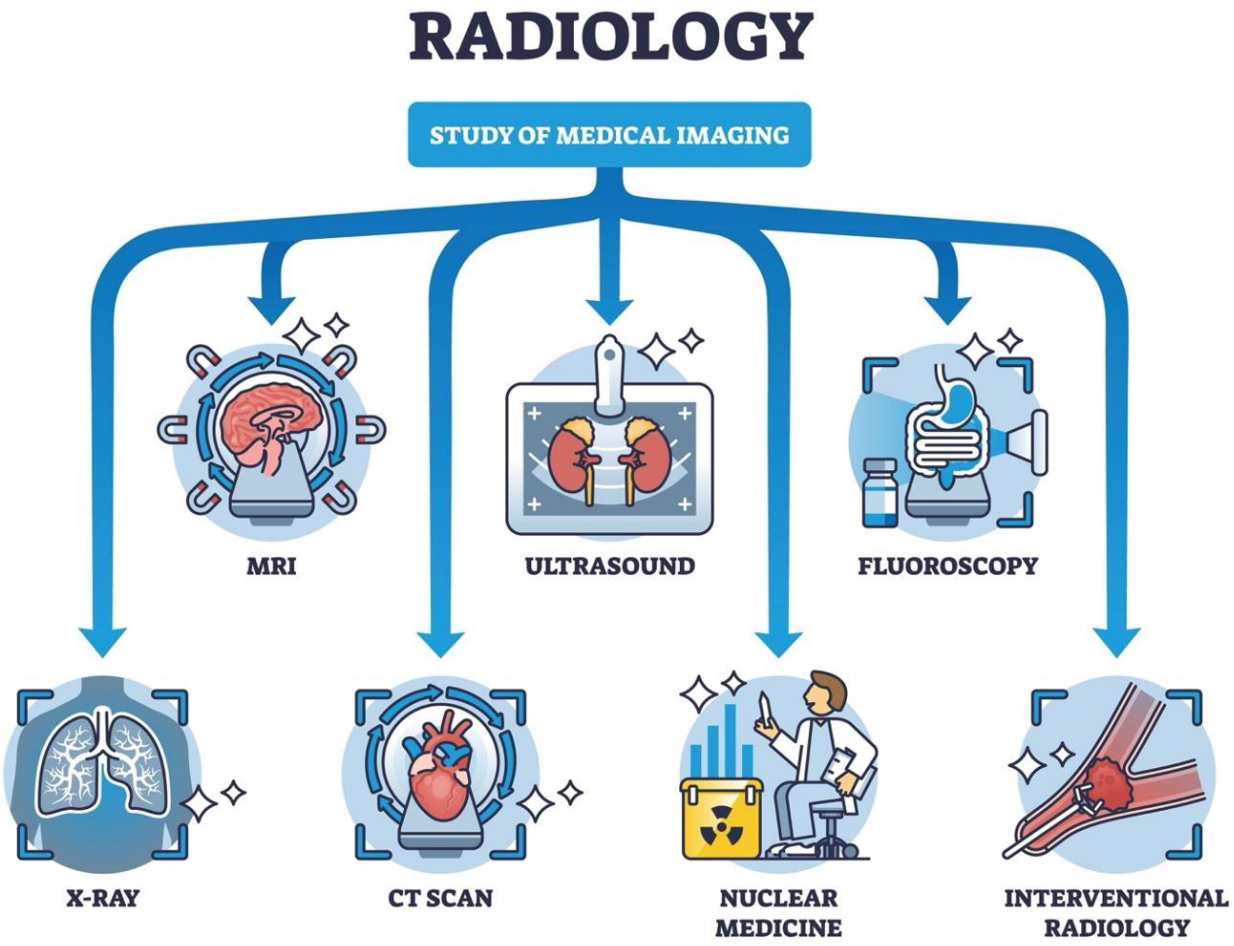


Рисунок 2.1 – Інструменти навчання медичних зображень [144]

Стандарт DICOM є основним у медицині, оскільки зберігає зображення разом із метаданими про пацієнта та параметри сканування. Інші формати, такі як NIfTI або звичайні графічні файли, мають вужчу сферу застосування через відсутність стандартизованої супровідної інформації [81].

Бібліотеки для роботи з медичними форматами різняться мовою реалізації та функціоналом. Пайтон пропонує Pydicom для зручного доступу, SimpleITK фокусується на сегментації, C++ використовує швидку бібліотеку GDCM, а Java застосовує Dcm4che для мережевих завдань [82].

Для веб-візуалізації томографічні дані у шкалі Хаунсфілда конвертують у підтримувані формати з використанням вікон відображення. Різні налаштування ширини та центру вікна дозволяють оптимально візуалізувати легеневі або кісткові тканини за допомогою графічних бібліотек [83].

Методи автоматичної сегментації варіюються від простих порогових алгоритмів до складних адаптивних підходів. Для виділення патологій застосовують методи Отсу, нарощування областей та виявлення країв, що враховують інтенсивність пікселів та їх просторові зв'язки [84].

Морфологічні операції, такі як ерозія та дилатація, використовуються для уточнення форм сегментованих об'єктів. Комбінування цих методів дозволяє видаляти шуми, заповнювати прогалини та відновлювати структуру анатомічних елементів на бінарних зображеннях [85].

Кількісний аналіз регіонів інтересу включає розрахунок статистичних, текстурних та геометричних параметрів. Матриці суміжності описують текстуру, а інваріанти моментів дозволяють розпізнавати форму об'єктів незалежно від їх орієнтації чи масштабу [86].

Спеціалізовані бібліотеки пропонують різні набори інструментів для наукових досліджень та клінічної візуалізації. OpenCV забезпечує високу швидкість обчислень, ITK спеціалізується на тривимірній реєстрації, а Scikit-image надає перевірені алгоритми для наукового аналізу зображень [87].

## 2.8 Підходи до класифікації медичних зображень засобами машинного навчання

Автоматична класифікація патологічних областей на томографічних зображеннях реалізується через різноманітні методи машинного навчання. Медична діагностика висуває специфічні вимоги до алгоритмів, що включають не лише точність, а й пояснюваність рішень, можливість валідації ознак експертами та стійкість до обмежених обсягів даних.

Вибір між класичними алгоритмами та глибокими нейронними мережами визначає архітектуру системи діагностики (див. рис. 2.2). Класичні методи ефективно працюють з малими вибірками та фіксованими векторами ознак, забезпечуючи прозорість процесу прийняття рішень. Глибокі мережі навчаються безпосередньо на пікселях зображень, потребують значних обсягів

даних та потужних графічних процесорів, залишаючись складними для інтерпретації [88, 93].

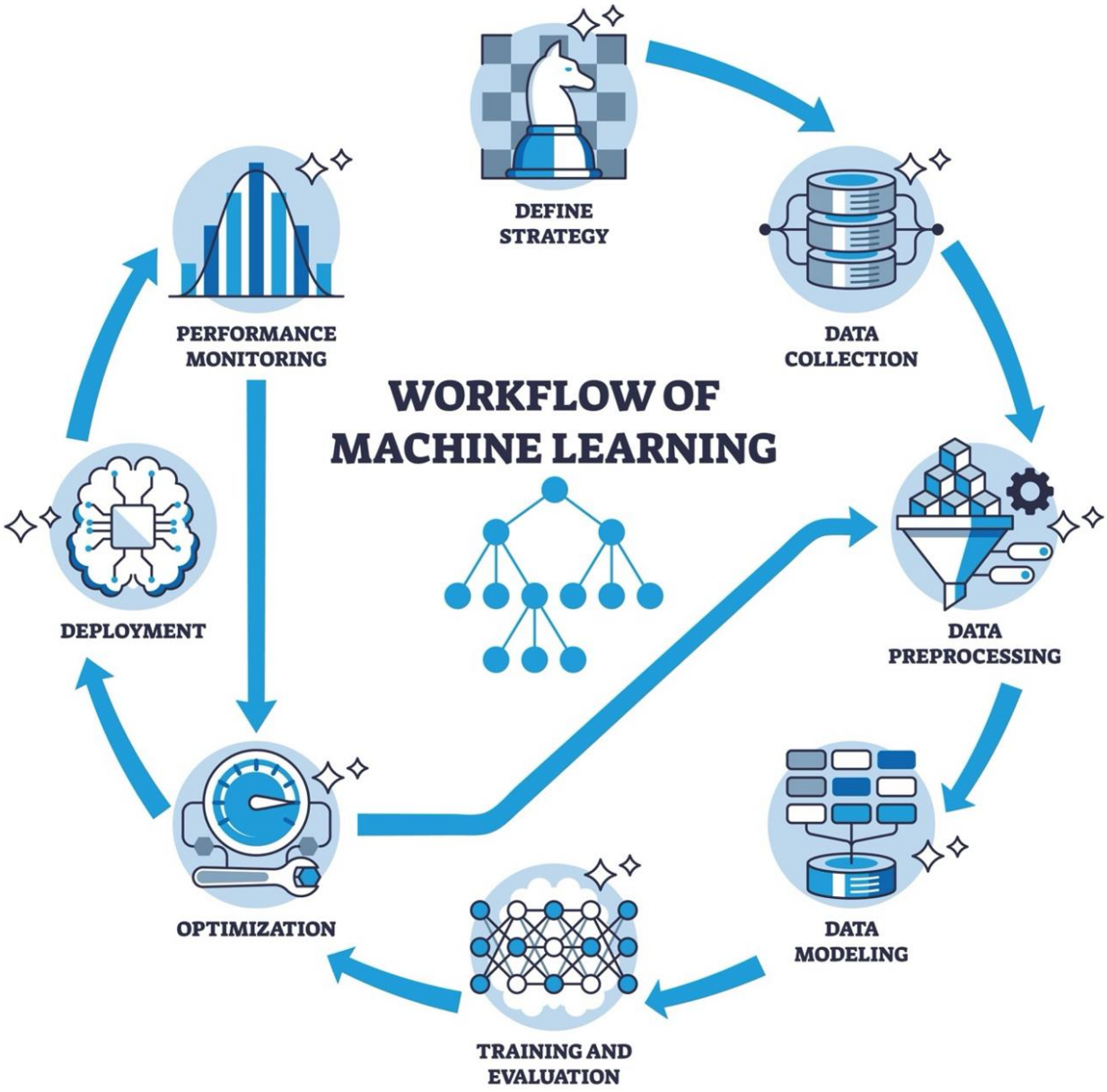


Рисунок 2.2 – Архітектура машинного навчання [145]

Дерева рішень структурують логіку класифікації у вигляді ієрархічних правил розгалуження за критеріями інформативності (див. рис. 2.3). Цей підхід забезпечує повну прозорість для медичних застосунків завдяки можливості відстежити шлях прийняття рішення. Для уникнення перенавчання застосовують обмеження глибини дерева та видалення неефективних гілок [89].

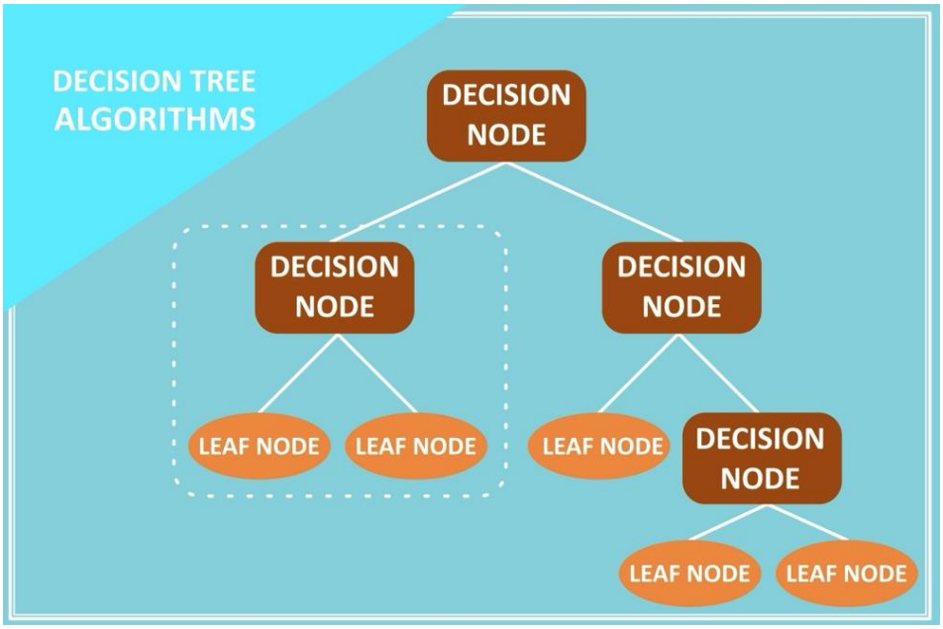


Рисунок 2.3 – Архітектура дерева рішень [146]

Ансамблеві методи об’єднують прогнози декількох моделей для підвищення точності та надійності класифікації. Випадковий ліс зменшує дисперсію помилок шляхом голосування незалежних дерев, побудованих на випадкових підвибірках даних (див. рис. 2.4). Градієнтний бустинг послідовно виправляє помилки попередніх класифікаторів, а стекінг комбінує результати різних алгоритмів через мета-модель [90, 92].

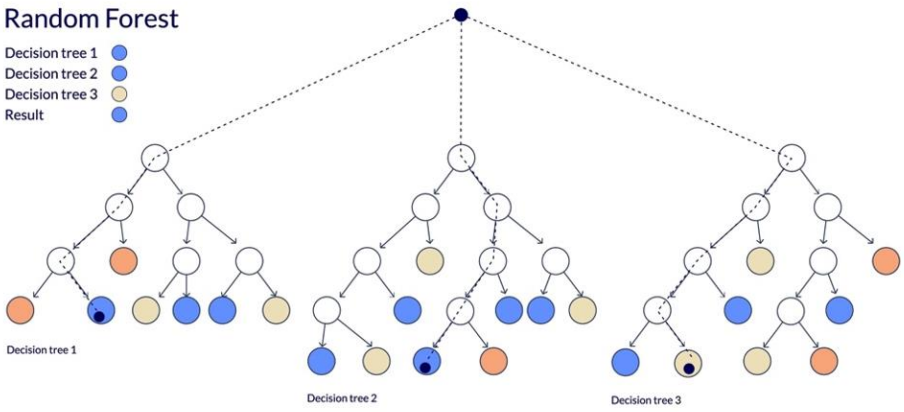


Рисунок 2.4 – Архітектура випадкового лісу [147]

Метод опорних векторів класифікує дані шляхом максимізації відступу між класами у просторі ознак. Використання ядерних функцій дозволяє проєктувати дані у простір вищої розмірності для моделювання складних нелінійних меж рішень. Алгоритм вимагає попередньої нормалізації ознак та налаштування параметрів регуляризації для досягнення балансу між точністю та узагальненням [92, 94].

Реалізація алгоритмів доступна через спеціалізовані бібліотеки з різним рівнем оптимізації та підтримки мов програмування. Пайтон пропонує уніфікований інтерфейс Сайкіт-лерн, тоді як для Джава існує середовище Века. Для обробки великих даних використовують МЛліб, а спеціалізовані бібліотеки забезпечують максимальну продуктивність градієнтного бустингу [92, 94].

Витяг ознак із зображень є критичним етапом підготовки даних для класичних алгоритмів навчання (див. рис. 2.5). Статистичні показники описують розподіл інтенсивності, а текстурні матриці характеризують просторові зв'язки між пікселями. Геометричні параметри та інваріантні дескриптори дозволяють розпізнавати форму сегментованих областей незалежно від їх масштабу та орієнтації [86, 93].

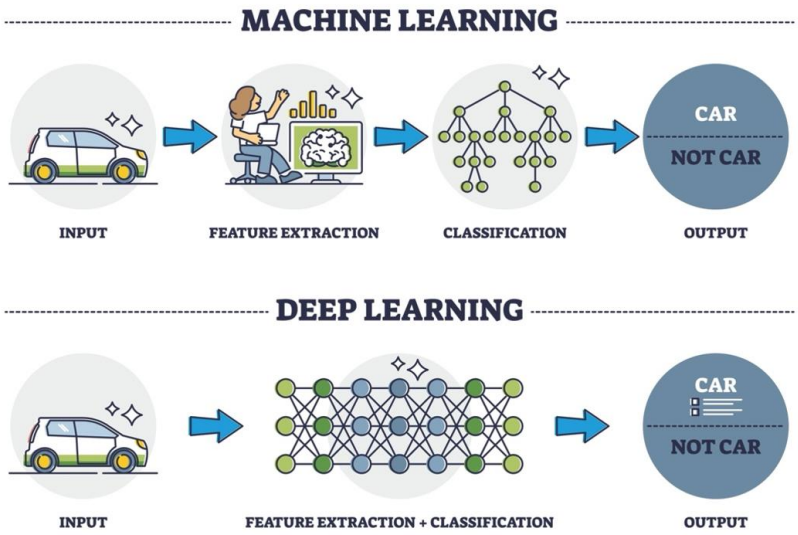


Рисунок 2.5 – Машинне навчання та глибоким навчанням [148]

Серіалізація моделей забезпечує збереження стану навчених класифікаторів для їх повторного використання. Стандартні інструменти мови Пайтон та спеціалізовані формати дозволяють ефективно записувати числові масиви та структури даних. Вибір методу збереження залежить від компромісу між ступенем стиснення файлів та швидкістю їх завантаження в пам’ять. Компроміс між розміром серіалізованих моделей та швидкістю завантаження включає вибір рівня стиснення, де відсутність стиснення забезпечує найшвидше завантаження за рахунок максимального використання дискового простору, тоді як агресивне стиснення зменшує розмір файлів у декілька разів за рахунок додаткових обчислень під час завантаження [94].

## 2.9 Моделі програмування для високопродуктивних медичних застосунків

Архітектура медичних систем вимагає ефективної організації паралельних обчислень для обробки інтенсивних операцій введення-виведення. Специфіка діагностичних програм полягає у перевазі очікування доступу до ресурсів над безпосередньою обробкою даних, що потребує оптимізації використання апаратних потужностей.

Синхронна модель базується на послідовному виконанні, де звернення до зовнішніх ресурсів повністю блокують роботу програми та спричиняють простій процесора. Головною перевагою такого підходу залишається простота реалізації та розуміння коду завдяки лінійній логіці без складного керування станами [95].

Багатопотокова архітектура підвищує ефективність обробки запитів завдяки перемиканню процесора між завданнями під час очікування завершення операцій. Однак використання потоків операційної системи супроводжується значними витратами пам’яті та накладними витратами ресурсів на перемикання контексту [96].

Багатопроцесна модель розподіляє навантаження між ізольованими процесами, що гарантує високу надійність та усуває проблеми синхронізації доступу до пам’яті. Основними недоліками архітектури є значне споживання системних ресурсів та складність організації обміну даними між окремими компонентами [97].

Асинхронна модель використовує кооперативну багатозадачність у межах одного потоку для ефективної обробки тисяч одночасних з’єднань. Головними викликами впровадження є необхідність застосування спеціалізованих неблокуючих бібліотек та ризик повної зупинки циклу подій тривалими обчислювальними операціями [98, 99].

Інструментарій асинхронного програмування включає різноманітні бібліотеки з відмінними механізмами управління циклом подій. У екосистемі Пайтон домінують стандартні рішення на зразок Асінхйо, тоді як інші мови пропонують власні моделі реалізації конкурентності та управління завданнями [99].

Взаємодія з базами даних у асинхронному середовищі потребує використання спеціалізованих неблокуючих драйверів замість традиційних рішень. Більшість сучасних систем підтримують таку роботу нативно, за винятком вбудованих баз даних типу ЕсКьюЛайт, що використовують емуляцію через потоки [77, 99].

Мережева комунікація в асинхронних системах забезпечується бібліотеками з неблокуючою реалізацією протоколів передачі даних. Сучасні фреймворки, такі як ФастАПІ, дозволяють створювати високопродуктивні сервери, а універсальні клієнти забезпечують ефективну обробку зовнішніх запитів [48, 49, 99].

Інтеграція складних обчислень у асинхронний контекст вимагає використання пулів процесів для обходу обмежень інтерпретатора. Спеціалізовані бібліотеки машинного навчання часто реалізують внутрішній паралелізм, що дозволяє виконувати розрахунки без блокування основного циклу виконання програми [100].

## 2.10 Підходи до розгортання та ізоляції компонентів програмних систем

Розгортання медичних систем вимагає ретельного підходу до ізоляції та керування залежностями для уникнення конфліктів. Пряме встановлення на операційну систему є найефективнішим з точки зору використання ресурсів, але створює значні ризики несумісності бібліотек між різними програмами. Віртуалізація гарантує повну ізоляцію завдяки запуску окремих операційних систем, проте потребує значних апаратних потужностей для кожної віртуальної машини [101, 103, 102, 104].

Контейнеризація забезпечує легку ізоляцію на рівні ядра операційної системи, що дозволяє запускати сотні сервісів на одному сервері з мінімальними накладними витратами. Технології на кшталт Докер та Подман пропонують різні архітектурні підходи: від централізованих демонів до повністю розподілених процесів без привілейованого доступу. Це робить контейнери оптимальним вибором для мікросервісних архітектур, хоча рівень їхньої безпеки поступається повній віртуалізації [105, 106].

Оркестрація контейнерів необхідна для координації роботи складних розподілених систем у промисловому середовищі. Докер Компоуз ідеально підходить для локальної розробки завдяки простоті конфігурації, тоді як Кубернетіс забезпечує автоматичне масштабування та самовідновлення сервісів у великих кластерах. Альтернативні платформи, такі як Номад, пропонують спрощене управління різнорідними навантаженнями [101, 107].

Управління залежностями є критичним для стабільності програмного забезпечення, особливо у мові Пайтон. Віртуальні середовища дозволяють ізолювати бібліотеки для кожного проекту, уникаючи конфліктів версій у системі. Інструменти як Поетрі та юві забезпечують детерміноване відтворення оточення завдяки файлам блокування, що фіксують точні версії всіх компонентів та їхні хеш-суми для перевірки цілісності [108, 109, 110].

## Висновок з розділу 2

У цьому розділі систематизовано технологічний фундамент розробки та визначено, що оптимальною архітектурою для агентної системи діагностики є поєднання асинхронної екосистеми Python із мікросервісним підходом. Аналіз показав, що використання ансамблевих методів класичного машинного навчання забезпечує необхідну для медицини прозорість прийняття рішень, у той час як інтеграція з великими мовними моделями через механізми виклику функцій надає системі гнучкості та здатності до динамічної оркестрації діагностичного процесу. Обраний стек технологій, що включає FastAPI для високонавантаженої обробки запитів, LangGraph для керування станами агента та спеціалізовані бібліотеки обробки DICOM-зображень, вирішує виявлену в попередньому розділі проблему статичності діагностичних систем, створюючи надійну базу для програмної реалізації адаптивного медичного асистента.

# РОЗДІЛ 3
# ОБҐРУНТУВАННЯ ВИБОРУ ТЕХНІЧНИХ РІШЕНЬ ДЛЯ ПРОГРАМНОГО ЗАСТОСУНКУ ДЛЯ ДИФЕРЕНЦІАЛЬНОЇ ДІАГНОСТИКИ РАКУ ЛЕГЕНЬ

У даному розділі обґрунтовано вибір програмно-технологічного стека для реалізації платформи інтелектуального аналізу комп’ютерної томографії легень. Детально розглянуто архітектурні рішення щодо побудови мікросервісної взаємодії на базі FastAPI, організації інтерактивного клієнтського інтерфейсу засобами Gradio та оркестрації агентних робочих процесів з використанням бібліотеки LangGraph. Проаналізовано технічні переваги застосування асинхронної моделі виконання Python 3.12, спеціалізованих бібліотек для обробки медичних зображень та ансамблевих методів машинного навчання. Окрему увагу приділено питанням контейнеризації компонентів системи через Docker та забезпечення приватності даних при інтеграції з хмарними сервісами Azure AI Foundry.

## 3.1 Мова програмування

Для реалізації платформи аналізу комп'ютерної томографії легень обрано мову програмування Python 3.12. Цей вибір зумовлений сукупністю технічних факторів, що гарантують ефективну розробку діагностичної системи з елементами штучного інтелекту.

Перевагою мови є потужна екосистема бібліотек для медичної візуалізації та обробки даних. Pydicom забезпечує повноцінну підтримку стандарту DICOM, scikit-learn та XGBoost реалізують алгоритми класифікації, а NumPy та OpenCV дозволяють виконувати швидкі чисельні обчислення та морфологічну обробку зображень [111].

Критичне значення має нативна підтримка асинхронного програмування для неблокуючої обробки запитів у FastAPI. Конструкція asyncio.gather забезпечує паралельне виконання алгоритмів класифікації, суттєво скорочуючи час отримання результату. Асинхронні клієнти оптимізують використання ресурсів під час взаємодії із зовнішніми сервісами та базами даних [112].

Версія 3.12 демонструє підвищену продуктивність завдяки оптимізації інтерпретатора. Це прискорює виконання інтенсивних обчислень при витягу ознак із медичних зображень. Вдосконалена система анотацій типів підвищує надійність коду та спрощує виявлення помилок на етапі розробки [111, 113].

Важливим аспектом є висока швидкість прототипування складних систем. Динамічна типізація дозволяє оперативно тестувати архітектурні рішення та алгоритми без необхідності компіляції. Використання бібліотеки Pydantic поєднує гнучкість розробки з надійною валідацією даних для критичних компонентів [114].

### 3.2 Веб-фреймворки серверної частини

Для побудови серверної частини обрано фреймворк FastAPI, який дозволяє реалізувати архітектуру з трьох незалежних мікросервісів для обробки зображень, штучного інтелекту та фронтенду. Технічні характеристики інструменту повністю відповідають вимогам діагностичної системи реального часу.

Нативна підтримка асинхронності забезпечує ефективне використання ресурсів під час операцій введення-виведення. Це критично для читання файлів DICOM та взаємодії з базою даних, дозволяючи серверу обслуговувати значну кількість запитів одночасно. Асинхронна модель також запобігає блокуванню системи під час тривалих викликів до зовнішніх API [115].

Автоматична генерація документації OpenAPI суттєво спрощує підтримку системи та міжсервісну взаємодію. Вбудовані інтерфейси

дозволяють розробникам тестувати функціонал безпосередньо через браузер без додаткового коду. Наявність актуальних специфікацій полегшує інтеграцію фронтенд-сервісу із серверною логікою [116].

Інтеграція з бібліотекою Pydantic забезпечує автоматичну валідацію вхідних даних без необхідності написання додаткового коду. Кожна кінцева точка перевіряє коректність типів та діапазонів значень, повертаючи структуровані помилки у разі невідповідності. Висока продуктивність нової версії бібліотеки гарантує швидку обробку великих обсягів метаданих [117].

Вбудований механізм впровадження залежностей сприяє створенню чистої архітектури та підвищує тестованість компонентів. Система автоматично керує життєвим циклом ресурсів, таких як з'єднання з базою даних, у межах кожного запиту. Це дозволяє чітко розділити шари додатку та інфраструктури згідно з кращими практиками розробки [118].

### 3.3 Інтерфейс користувача

Для клієнтської частини обрано бібліотеку Gradio 4, що дозволяє створювати інтерактивні інтерфейси виключно засобами мови Python. Такий підхід виключає необхідність використання JavaScript або HTML, суттєво прискорюючи розробку медичних застосунків. Це забезпечує повноцінну функціональність для роботи з томографічними даними при мінімальних витратах ресурсів на створення фронтенду.

Компонент ImageEditor надає повний набір інструментів для маскування патологічних ділянок з використанням одношарової архітектури. Напівпрозоре відображення маски дозволяє візуалізувати виділену область одночасно з оригінальною структурою тканини без візуальних артефактів. Автоматичне масштабування забезпечує коректне відображення знімків у браузері зі збереженням їх первинної роздільної здатності [119].

Система підтримує завантаження файлів DICOM без розширень, що є критичним для сумісності з медичними архівами. Спеціалізований компонент

дозволяє одночасно обробляти серії до тисячі зображень незалежно від наявності явного вказівника формату. Асинхронна модель забезпечує парсинг метаданих та формування повної серії зрізів без блокування інтерфейсу користувача [120].

Подієва архітектура автоматизує синхронізацію стану інтерфейсу та серверної частини через механізм реактивності. Навігація між зрізами та зміна параметрів візуалізації миттєво ініціюють відповідні запити до бекенду без необхідності ручного управління станом. Це гарантує автоматичне збереження та коректне відновлення масок регіонів інтересу під час роботи лікаря [121].

Повна інтеграція з екосистемою Python дозволяє реалізувати логіку взаємодії із зовнішніми сервісами безпосередньо в коді інтерфейсу. Асинхронні обробники подій забезпечують ефективну комунікацію з модулями аналізу даних без створення складних проміжних адаптерів. Вбудований компонент чату структурує історію діагностичних дій з підтримкою ролей та часових міток [122].

### 3.4 Оркестрація робочих процесів агентів

Для управління процесами агента обрано фреймворк LangGraph, що забезпечує контроль стану та умовну маршрутизацію у складних сценаріях. Система координує розпізнавання намірів, виклик інструментів та паралельну класифікацію з генерацією пояснень. Явне визначення графів гарантує прозорість та передбачуваність поведінки на відміну від неявних ланцюжків виконання.

Клас StateGraph дозволяє декларативно визначати стани робочого процесу та правила переходів між ними. Вузли обробки реалізовано як асинхронні функції, що приймають поточний стан і повертають оновлений результат. Централізована структура спрощує візуалізацію та аудит

діагностичного алгоритму, що є критичним для медичної валідації системи [123].

Механізм умовних ребер забезпечує динамічну маршрутизацію виконання на основі контексту діалогу. Система автоматично аналізує наміри користувача та направляє процес до необхідних вузлів, наприклад, ініціюючи детекцію маски перед класифікацією. Це дозволяє адаптувати сценарій взаємодії без необхідності явних вказівок на кожному етапі [124].

Збереження стану в базі даних гарантує відновлення контексту та збереження історії діалогу, координат регіонів інтересу й результатів аналізу. Персистентність є критичною для тривалих діагностичних сеансів та забезпечення стійкості системи до технічних збоїв. Історія станів також формує повноцінний аудиторський слід для подальшого аналізу якості роботи алгоритмів [125].

Інтеграція контрольних точок забезпечує обов'язкову верифікацію критичних рішень медичним фахівцем. Механізм переривання дозволяє призупинити процес для підтвердження коректності виділеної області перед початком ресурсоємної класифікації. Такий підхід гарантує відповідність системи регуляторним вимогам щодо пріоритету лікарського контролю у діагностиці [126].

### 3.5 Сервіс великих мовних моделей

Для реалізації діагностичного агента обрано платформу Azure OpenAI з моделлю GPT-5, що забезпечує якісну обробку природної мови. Вибір зумовлений високим рівнем корпоративної безпеки, наявністю необхідної обчислювальної інфраструктури та підтримкою механізму виклику функцій. Це дозволяє використовувати потужності моделей OpenAI в захищеному середовищі Microsoft Azure.

Ключовою перевагою платформи є відповідність стандарту HIPAA, що гарантує захист конфіденційної медичної інформації. Система забезпечує

шифрування даних, ізоляцію ресурсів та детальний аудит усіх звернень. Критично важливим є те, що дані пацієнтів не використовуються для навчання базових моделей, що виключає ризики витоку інформації [127].

Географічна доступність центрів обробки даних дозволяє мінімізувати мережеві затримки, обираючи найближчий регіон розміщення. Це скорочує час реакції системи до п'ятисот мілісекунд, що є критичним показником для інтерактивного діалогу в реальному часі. Оперативність отримання відповідей безпосередньо впливає на ефективність клінічного процесу [128].

Механізм виклику функцій забезпечує динамічну маршрутизацію інструментів у робочому процесі агента. Модель самостійно визначає необхідність виконання технічних дій, повертаючи структуровані інструкції у форматі JSON для системи оркестрації. Це дозволяє інтегрувати результати класифікації в історію діалогу та генерувати контекстуальні пояснення для лікаря [129].

Модель оплати за фактичне використання ресурсів є економічно вигідною альтернативою традиційному ліцензуванню. Тарифікація на основі кількості оброблених токенів дозволяє уникнути значних початкових інвестицій. Такий підхід забезпечує гнучкість бюджетування та можливість масштабування витрат пропорційно до реального навантаження на систему [130].

### 3.6 Система керування базами даних

Для збереження даних обрано СУБД SQLite з асинхронним розширенням SQLAlchemy 2.0. Вибір зумовлений вимогами надійності та простоти розгортання в умовах обмежених ресурсів. Архітектура без окремого серверного процесу зберігає інформацію в одному файлі, усуваючи складнощі конфігурування мережі та адміністрування доступу.

Підтримка принципів ACID гарантує цілісність діагностичних даних та результатів класифікації навіть у разі збоїв. Механізм транзакцій забезпечує

атомарне збереження інформації, запобігаючи частковому застосуванню змін. Вбудований повнотекстовий пошук дозволяє ефективно аналізувати історію клінічних випадків без додаткових витрат на обслуговування сервера [131].

Бібліотека SQLAlchemy 2.0 забезпечує асинхронну взаємодію з базою даних через драйвер aiosqlite. Неблокуюча модель виконання критично важлива для швидкого збереження об'ємних метаданих зображень без затримки основного потоку. Використання async_sessionmaker гарантує автоматичне керування життєвим циклом з'єднань та оптимізацію виконання запитів [132, 139].

Структура бази даних складається з п'яти таблиць для зберігання сесій, повідомлень, регіонів інтересу та результатів класифікації. Окрема сутність фіксує стан робочого процесу LangGraph для забезпечення персистентності діагностичних сценаріїв. Реляційні зв'язки та зовнішні ключі гарантують цілісність даних між усіма компонентами системи [133].

Реалізація патерну репозиторію через абстрактні інтерфейси забезпечує незалежність бізнес-логіки від деталей реалізації доступу до даних. Такий підхід створює чисту архітектуру, спрощуючи тестування та потенційну міграцію на PostgreSQL. Це дозволяє масштабувати систему відповідно до зростання навантаження без модифікації основного коду [134].

### 3.7 Обробка медичних зображень

Для забезпечення повного циклу обробки томографічних зображень обрано комплекс бібліотек: pydicom, Pillow, OpenCV та scikit-image. Цей набір гарантує сумісність із міжнародними стандартами, збереження діагностичної цінності даних та наявність інструментів автоматичного виявлення патологій.

Бібліотека pydicom реалізує підтримку стандарту обміну медичними зображеннями, забезпечуючи коректну роботу з даними. Інструмент дозволяє витягувати структуровані метадані, включаючи параметри сканування та геометричні характеристики. Забезпечено сумісність із різними методами

кодування та можливість читання файлів без розширень, що є критичним для медичних архівів [82].

Бібліотека Pillow надає засоби конвертації форматів та базової обробки для інтеграції з веб-інтерфейсом. Вона виконує нормалізацію інтенсивностей та перетворення масивів у графічні формати для візуалізації у браузері. Функціонал включає масштабування зображень та експорт результатів для архівування без втрати якості оригінальних даних [111].

Бібліотека OpenCV забезпечує високопродуктивні морфологічні перетворення для уточнення контурів регіонів інтересу. Операції ерозії та дилатації ефективно усувають шумові артефакти та згладжують межі областей після сегментації. Оптимізовані алгоритми дозволяють швидко обчислювати геометричні інваріанти для класифікації типу патології [111].

Бібліотека scikit-image містить валідовані алгоритми сегментації, зокрема метод Отсу для розділення об'єкта та фону. Інструментарій дозволяє витягувати текстурні ознаки через аналіз просторового розподілу інтенсивностей. Розрахунок локальних патернів забезпечує кількісний опис мікроструктури тканин для діагностичної класифікації [87].

### 3.8 Фреймворки машинного навчання

Для реалізації ансамблевої класифікації обрано бібліотеки scikit-learn, XGBoost та joblib. Використання класичних алгоритмів замість глибоких нейромереж обумовлено вимогами до пояснюваності рішень у медичній діагностиці. Такий підхід забезпечує ефективність роботи на обмежених обсягах даних та спрощує валідацію логіки фахівцями.

Бібліотека scikit-learn надає прозорі реалізації фундаментальних алгоритмів. Випадковий ліс забезпечує робастність до шуму та дозволяє оцінювати важливість діагностичних ознак, а метод опорних векторів виконує нелінійну класифікацію з попередньою нормалізацією даних. Уніфікований

програмний інтерфейс спрощує експериментування з різними моделями без зміни архітектури [135].

Бібліотека XGBoost реалізує оптимізований градієнтний бустинг із послідовним виправленням помилок та регуляризацією для запобігання перенавчанню. Паралелізація обчислень суттєво скорочує час навчання моделей на багатоядерних процесорах. Вбудовані механізми крос-валідації та обробки відсутніх значень зменшують потребу в ручному налаштуванні гіперпараметрів [136].

Бібліотека joblib забезпечує ефективну серіалізацію моделей, оптимізовану для роботи з великими числовими масивами. Це дозволяє одноразово завантажувати натреновані алгоритми під час ініціалізації сервісу, уникаючи затримок при обробці запитів. Підтримка стиснення додатково зменшує розмір файлів та прискорює операції читання з диска [137].

### 3.9 Асинхронне програмування

Для забезпечення високої пропускної здатності платформи обрано асинхронну модель на базі бібліотек asyncio, aiohttp, aiosqlite та AsyncOpenAI. Цей підхід зумовлений специфікою медичних систем, де домінують операції очікування введення-виведення, такі як читання файлів та мережеві запити. На відміну від синхронної моделі, асинхронна архітектура запобігає блокуванню потоків, дозволяючи ефективно використовувати обчислювальні ресурси під час очікування завершення зовнішніх операцій.

Бібліотека asyncio забезпечує неблокуюче виконання завдань через цикл подій, координуючи роботу співпрограм без створення окремих потоків. Це дозволяє серверу обробляти паралельні запити користувачів, наприклад завантаження знімків, під час тривалого очікування відповідей від сервісів штучного інтелекту. Мінімальні накладні витрати на перемикання контексту забезпечують ефективне управління тисячами одночасних операцій на одному процесорному ядрі [138].

Драйвер aiosqlite реалізує асинхронний інтерфейс до бази даних, дозволяючи виконувати операції читання та запису без блокування основного циклу. Це критично для швидкого збереження результатів класифікації та оновлення станів робочого процесу незалежно від навантаження на дискову підсистему. Інтеграція з SQLAlchemy 2.0 забезпечує високорівневу об’єктно-реляційну взаємодію з автоматичним управлінням транзакціями та конвертацією даних [139].

Клієнт AsyncOpenAI забезпечує неблокуючу генерацію відповідей та розпізнавання намірів через хмарний інтерфейс. Оскільки обробка складних контекстних запитів моделлю може тривати до двох секунд, асинхронний підхід дозволяє звільнити системні ресурси на час очікування результату. Це гарантує, що мережеві затримки одного сеансу не впливають на швидкість обробки запитів інших медичних фахівців [140].

Функція asyncio.gather організовує паралельне виконання трьох алгоритмів машинного навчання для ансамблевої класифікації. Одночасний запуск моделей випадкового лісу, опорних векторів та градієнтного бустингу скорочує загальний час обробки до двадцяти мілісекунд замість послідовного виконання. Такий підхід забезпечує ефективну утилізацію багатоядерних процесорів без складності явного управління потоками операційної системи [141].

### 3.10 Контейнеризація та управління залежностями

Для забезпечення ізоляції та відтворюваності розгортання платформи обрано технологію контейнеризації Докер із засобом координації Докер Компоуз та утиліту ув для керування пакетами Пайтон. Такий вибір гарантує незалежне масштабування компонентів та спрощує розгортання у різних середовищах від локальної розробки до промислової експлуатації. Це забезпечує ідентичність програмного оточення між усіма інсталяціями системи в медичних установах.

Платформа Докер реалізує легковагову віртуалізацію через ізоляцію процесів у контейнерах із базовим образом Пайтон та мінімальною операційною системою Лінукс. Такий підхід унеможливлює конфлікти версій бібліотек, дозволяючи кожному сервісу використовувати специфічні набори пакетів без ризику несумісності. Детерміноване відтворення оточення гарантується через інструкції побудови образу, що містять усі необхідні залежності [105].

Інструмент Докер Компоуз координує роботу контейнерів через декларативний файл конфігурації, що визначає мережеві з'єднання та порти для трьох сервісів. Механізм перевірки працездатності автоматизує запуск залежних компонентів, запобігаючи помилкам з'єднання під час старту системи. Використання томів забезпечує збереження даних бази та файлів після зупинки системи, а також дозволяє застосовувати зміни коду без перезбирання образів [107].

Утиліта ув забезпечує високопродуктивне керування пакетами завдяки реалізації на мові Раст, що значно прискорює операції порівняно з традиційними менеджерами. Паралелізація завантаження та ефективне кешування скорочують час побудови образів із кількох хвилин до десятків секунд. Це суттєво оптимізує цикл розробки та тестування при частій зміні функціональних можливостей платформи [111].

Спеціалізовані файли конфігурації та блокування забезпечують декларативне визначення залежностей та ідентичність програмного оточення. Файл блокування містить точні версії пакетів із криптографічними хешами для верифікації цілісності всіх компонентів. Це гарантує використання однакових бібліотек у всіх інсталяціях, усуваючи проблеми несумісності та спрощуючи валідацію системи регуляторними органами [109, 110].

### 3.11 Підхід до захисту даних та мінімізація ризиків безпеки

Архітектура застосунку базується на принципі мінімізації обробки даних, що знижує ризики порушення конфіденційності. Система працює виключно з анонімізованими зображеннями DICOM, з яких видалено всі ідентифікуючі метадані ще до завантаження на платформу. Це гарантує повну відсутність персональної інформації пацієнтів у системі аналізу.

Аналіз збережених даних підтверджує відсутність конфіденційної інформації у всіх компонентах платформи. База даних містить лише технічні ідентифікатори сесій та знеособлені результати без прив'язки до реальних осіб. Файлова система зберігає анонімізовані знімки та серіалізовані моделі без доступу до вихідних даних пацієнтів.

Натреновані моделі класифікації не містять персональної інформації завдяки статистичній природі алгоритмів машинного навчання. Випадковий ліс, метод опорних векторів та XGBoost зберігають лише абстрактні числові параметри та ваги. Це унеможливлює відтворення вихідних медичних зображень або даних конкретних пацієнтів з тренувального набору.

Відсутність персональних даних дозволяє обмежитися базовими механізмами захисту та ізоляцією сервісів через Docker. Для розгортання моделі GPT-5 використовується платформа Azure AI Foundry, що забезпечує обробку запитів у контрольованому середовищі. Це гарантує конфіденційність, оскільки дані не передаються до OpenAI та залишаються в межах захищеної інфраструктури.

### Висновок з розділу 3

У цьому розділі сформовано цілісний технологічний базис розробки, який поєднує високу продуктивність асинхронних сервісів із гнучкістю агентної архітектури. Вибір сучасного стека на основі Python 3.12 та LangGraph дозволив ефективно вирішити завдання динамічної маршрутизації

діагностичних інструментів, а використання класичних алгоритмів машинного навчання забезпечило прозорість прийняття рішень, що є критичним для медичної сфери. Запропонована архітектура з використанням Azure AI Foundry та локального зберігання анонімізованих даних гарантує відповідність вимогам конфіденційності та створює надійний фундамент для програмної реалізації швидкої та масштабованої діагностичної системи.

# РОЗДІЛ 4
# ПРОГРАМНА РЕАЛІЗАЦІЯ ТА МЕТОДИКА РОБОТИ ПРОГРАМНОГО ЗАСТОСУНКУ ДЛЯ ДИФЕРЕНЦІАЛЬНОЇ ДІАГНОСТИКИ РАКУ ЛЕГЕНЬ

У даному розділі викладено деталі програмної реалізації діагностичної платформи, побудованої на засадах мікросервісної архітектури з використанням асинхронного стеку Python, FastAPI та Gradio. Розглянуто технічні аспекти організації системи, включаючи патерни проєктування, реалізацію DICOM-переглядача та механізми інтерактивного маскування патологій. Особливу увагу приділено методиці побудови інтелектуального агента на базі бібліотеки LangGraph для оркестрації діагностичного процесу та імплементації ансамблевої моделі класифікації, що об’єднує алгоритми Random Forest, SVM та XGBoost. Також описано конвеєр екстракції ознак, структуру бази даних та результати тестування продуктивності розробленого рішення.

## 4.1 Архітектурні рішення та патерни проектування

Розроблена платформа для аналізу комп’ютерних томограм легень побудована на сучасних архітектурних принципах, що забезпечують високу масштабованість, підтримуваність та відокремлення функціональних компонентів. Архітектурне рішення системи базується на мікросервісному підході з використанням патернів чистої архітектури, що дозволяє ефективно управляти складністю медичного програмного застосунку та забезпечувати можливість незалежного розвитку окремих підсистем.

### 4.1.1 Мікросервісна архітектура та організація системи

Система реалізована як комплекс трьох незалежних сервісів для оброблення медичних знімків. Це забезпечує гнучке масштабування окремих

модулів та спрощує технічний супровід програмного продукту.

Перший модуль є інтерфейсною частиною на базі бібліотеки Градіо для роботи медичних спеціалістів. Він забезпечує завантаження файлів, перегляд зрізів із налаштуваннями та взаємодію з інтелектуальним помічником (див. рис. 4.1).

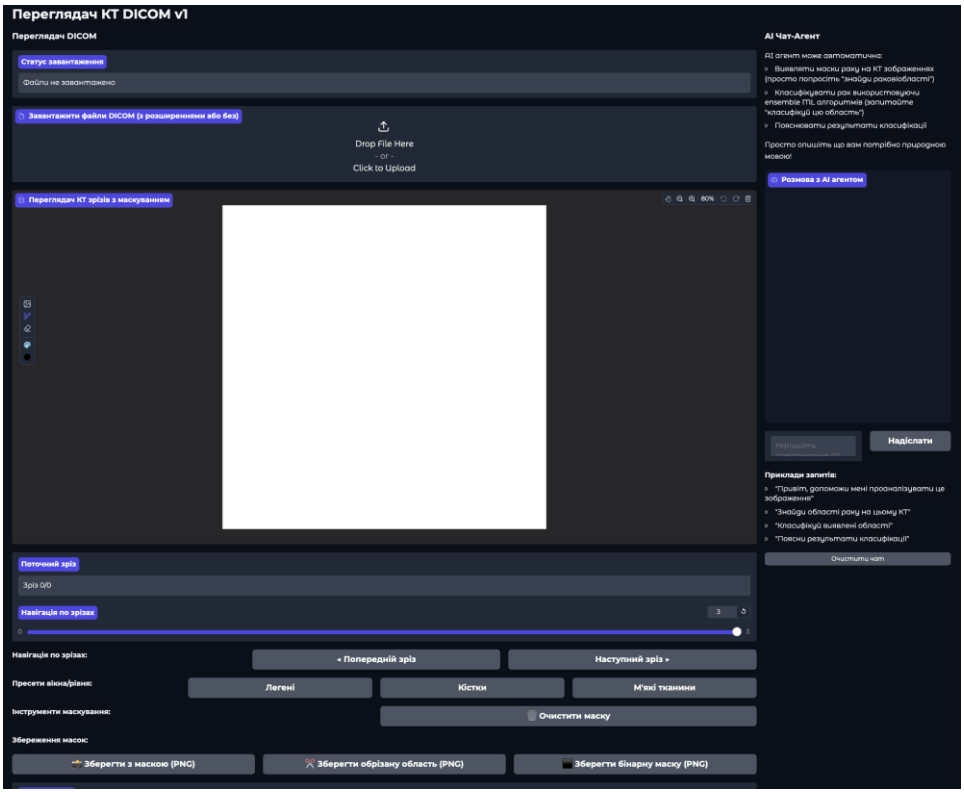


Рисунок 4.1 – Веб-інтерфейс програмного застосунку

Другий сервіс виступає основною серверною частиною для оброблення медичних даних та керування метаданими. Він відповідає за аналіз файлів, налаштування відображення тканин та експорт результатів роботи.

Третій модуль є спеціалізованим сервером для функціонування алгоритмів штучного інтелекту. Він керує логікою агента, виконує класифікацію даних декількома методами та зберігає стани діалогів.

Обмін даними між модулями здійснюється через мережеві протоколи з використанням структурованих форматів. Інтерфейс передає запити та кодовані зображення до серверних частин для подальшого аналізу.

Для ізоляції компонентів застосовано технологію контейнеризації з автоматизованим керуванням запуском. Це дозволяє налаштувати залежності між сервісами та забезпечити збереження даних у зовнішніх сховищах (див. рис. 4.2).

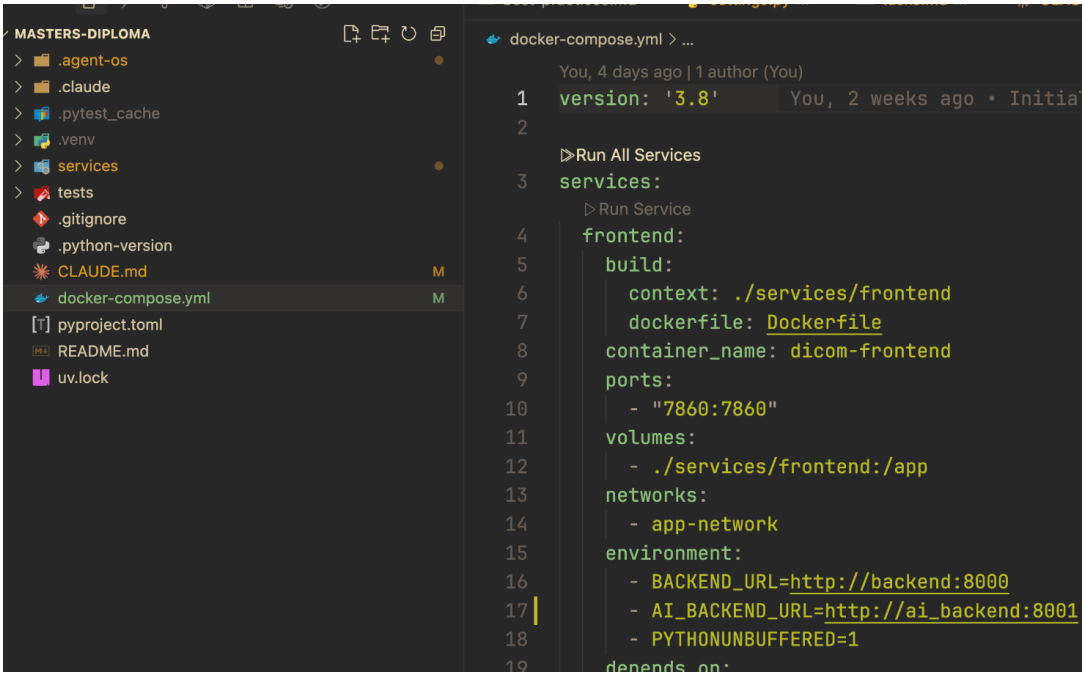


Рисунок 4.2 – Docker compose

У системі використано асинхронний підхід до оброблення програмних запитів та операцій з базами даних. Це гарантує ефективне використання ресурсів під час очікування відповідей від зовнішніх систем.

Оптимізація пам’яті досягається через механізм поступового завантаження необхідних томографічних зрізів. Система опрацьовує лише активні зображення, що забезпечує швидку дію при роботі з великими масивами даних.

### 4.1.2 Патерни проектування та чиста архітектура

Архітектурна організація сервісів базується на суровому розділенні рівнів додатку, домену та інфраструктури для забезпечення незалежності компонентів. Такий підхід гарантує гнучкість системи, ізоляцію бізнес-логіки та спрощує тестування окремих модулів (див. рис. 4.3).

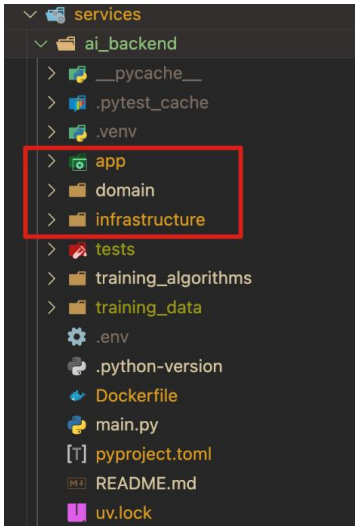


Рисунок 4.3 – Архітектура системи

Шар додатку відповідає за координацію бізнес-операцій та оброблення вхідних запитів через стандартизовані інтерфейси. Він реалізує зв'язок користувача з системою та автоматичне керування залежностями між компонентами під час виконання операцій.

Доменний рівень містить основну бізнес-логіку та моделі даних, ізольовані від зовнішніх технологій та фреймворків. Він визначає правила оброблення інформації, сутності системи та абстрактні контракти для операцій з даними.

Інфраструктурний шар забезпечує технічну реалізацію взаємодії з базами даних, файловими системами та зовнішніми сервісами. Він приховує деталі мережевої комунікації та специфіку збереження даних від вищих рівнів системи.

Патерн репозиторію використовується для відокремлення логіки доступу до даних від бізнес-процесів через систему абстракцій. Це дозволяє тестувати логіку без реальної бази даних та змінювати технології збереження без впливу на основний код.

Логіка інтелектуального агента побудована на графі обчислень, де вузли відповідають за конкретні операції аналізу та генерації відповідей. Така структура дозволяє динамічно керувати маршрутизацією завдань та

переходами між станами залежно від контексту.

Система зберігає історію діалогів та проміжні результати аналізу у базі даних для забезпечення персистентності процесу. Це дозволяє автоматично відновлювати стан роботи та залучати користувача до прийняття рішень у критичні моменти.

Класифікація виконується шляхом паралельного запуску трьох незалежних алгоритмів через єдиний інтерфейс стратегій. Фінальний результат формується на основі зваженого голосування моделей, що підвищує загальну точність діагностики.

Процес отримання ознак реалізовано як конвеєр, що охоплює аналіз інтенсивності, текстурних характеристик та морфологічних параметрів. У результаті формується комплексний числовий вектор, який використовується для подальшої класифікації.

Для забезпечення стабільності система використовує механізм автоматичних повторних запитів при тимчасових збоях зовнішніх сервісів. Структуроване логування фіксує всі етапи роботи та помилки для оперативного діагностування проблем.

### 4.2 Реалізація користувацького інтерфейсу медичного перегляду

Користувацький інтерфейс системи розроблено з орієнтацією на потреби медичних спеціалістів, що працюють з комп'ютерною томографією легень, та реалізовано на базі фреймворку Gradio версії 4.x. Архітектурне рішення інтерфейсу базується на принципі розділення екранного простору на основну робочу область для візуалізації та аналізу томографічних зрізів, що займає 70% ширини екрану, та допоміжну панель діалогу з інтелектуальним агентом, розміщену у правій частині інтерфейсу з шириною 20%. Такий розподіл простору забезпечує достатню площу для детального перегляду медичних зображень при збереженні постійного доступу до функціональності штучного інтелекту без необхідності перемикання між різними екранами чи

вкладками.

### 4.2.1 Компоненти візуалізації DICOM та навігація

Центральним елементом інтерфейсу є компонент ImageEditor фреймворку Gradio, який забезпечує відображення томографічних зрізів у нативній роздільній здатності. Цей компонент підтримує інтерактивне редагування та роботу з шарами, що дозволяє накладати напівпрозорі маски поверх основного зображення. Таке рішення гарантує одночасну видимість структури тканин та меж виділеної області інтересу (див. рис. 4.4).

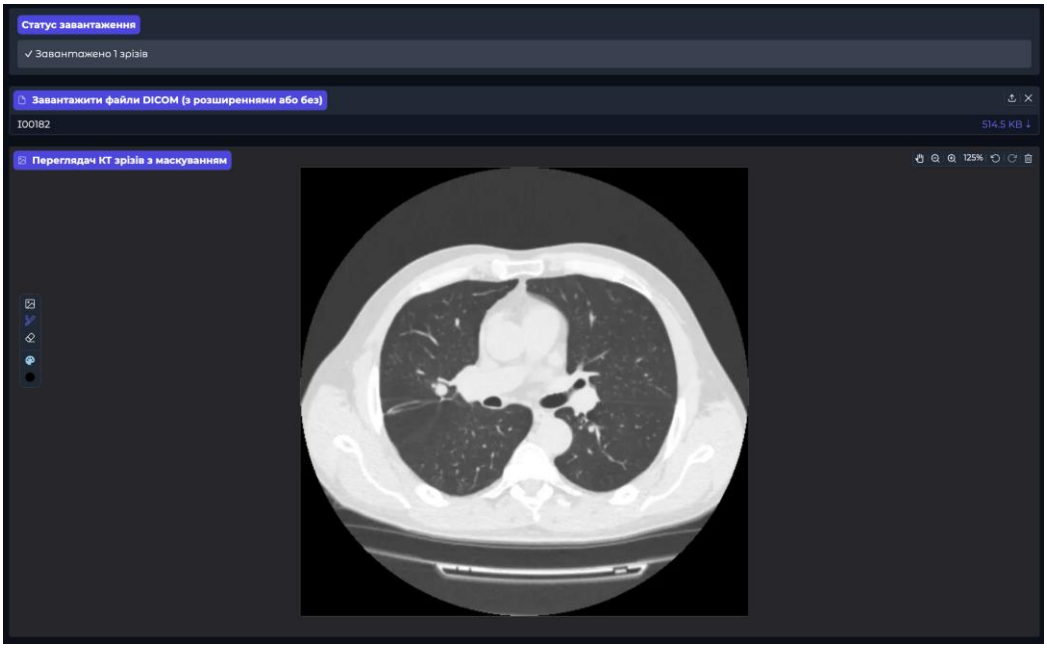


Рисунок 4.4 – Робота зі знімком

Навігація між зрізами реалізована через комбінацію повзунка з текстовим індикатором та клавіатурних скорочень. Лікарі можуть швидко переміщуватися по серії за допомогою миші або використовувати клавіші стрілок для покрокового переходу. Для роботи з великими масивами даних передбачено прискорену навігацію сторінковими клавішами.

Система підтримує масштабування зображень колесом миші з автоматичним центруванням відносно курсору. Користувачам доступні три рівні збільшення та режим переміщення збільшеної області для детального огляду. Функція швидкого скидання дозволяє миттєво повернути зображення

до початкового розміру.

Для коректної інтерпретації даних впроваджено налаштування віконних параметрів за шкалою Хаунсфілда. Інтерфейс містить попередньо налаштовані пресети для легеневої, кісткової та м'яких тканин. Активація режимів здійснюється через спеціальні кнопки панелі інструментів, що прискорює діагностичний процес.

Завантаження файлів виконується через універсальний компонент із підтримкою множинного вибору та автоматичним аналізом метаданих. Система самостійно групує зрізи в серії та сортує їх відповідно до просторового розташування. Після завершення обробки користувач отримує підтвердження про готовність даних до перегляду.

### 4.2.2 Інструменти виділення областей та інтеграція чату

Інструментарій виділення областей базується на функціях малювання та включає пензель і гумку з регульованим розміром. Технічна реалізація масок використовує одношарове напівпрозоре накладання, що запобігає накопиченню непрозорості. Це забезпечує точне окреслення патологічних утворень для подальшого аналізу (див. рис. 4.5).

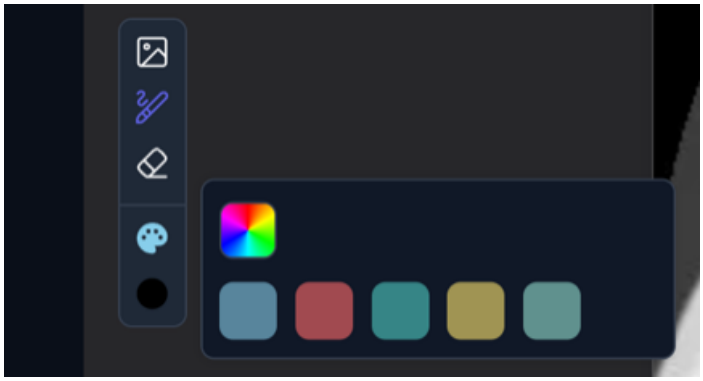

Рисунок 4.5 – Інструменти для виділення маски

Збереження масок відбувається автоматично при переході між зрізами з використанням кодування base64. Бекенд організовує зберігання даних у прив'язці до сесій користувачів, що гарантує цілісність розмітки. Для оперативного видалення виділень передбачено окрему кнопку очищення

поточного шару.

Експорт результатів можливий у трьох форматах через менеджер завантажень. Користувач може отримати композитне зображення, окрему вирізану область інтересу або бінарну маску. Це покриває потреби як у створенні медичної документації, так і в підготовці даних для навчання моделей.

Інтеграція з інтелектуальним агентом реалізована через компонент чату у правій частині інтерфейсу. Вікно діалогу візуально розділяє повідомлення користувача та системи для зручності сприйняття історії спілкування. Взаємодія відбувається через стандартне поле введення текстових запитів (див. рис. 4.6).

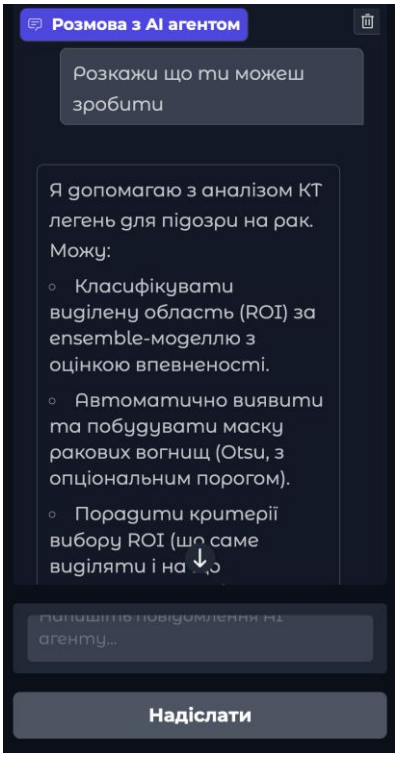


Рисунок 4.6 – Інтерактивний чат з агентом

Технічно чат працює через асинхронний API з автоматичною передачею поточного зрізу в контекст розмови. Зображення кодується та надсилається агенту, що дозволяє аналізувати візуальні дані без додаткових дій користувача. Відповіді системи динамічно додаються до стрічки повідомлень

у реальному часі.

Інтерфейс повністю локалізовано українською мовою з дотриманням професійної медичної термінології. Всі елементи керування та системні повідомлення адаптовані для використання в радіологічній практиці. Це забезпечує природну взаємодію з системою та знижує когнітивне навантаження на спеціаліста.

## 4.3 Модуль обробки та завантаження DICOM файлів

Модуль обробки медичних зображень у форматі DICOM є критично важливим компонентом системи, що забезпечує коректне завантаження, парсинг та управління томографічними даними від різноманітного медичного обладнання. Реалізація модуля базується на бібліотеці pydicom версії 2.x, що представляє собою стандартизований інструментарій для роботи з медичними зображеннями у форматі Digital Imaging and Communications in Medicine [150]. Архітектура модуля побудована навколо принципів надійної обробки файлів різної якості та структури, оскільки DICOM-експорти з різних томографів та виробників медичного обладнання можуть мати відмінності у структурі метаданих, кодуванні пікселів та організації серій зрізів.

### 4.3.1 Парсинг та валідація DICOM даних

Процес завантаження файлів ініціюється через спеціалізований програмний інтерфейс, що приймає масиви даних від п’яти до тисячі елементів. Система автоматично опрацьовує томографічні серії різного обсягу, забезпечуючи підтримку широкого спектра медичних досліджень. Критичною функцією є ідентифікація типу даних за внутрішньою структурою заголовків, що дозволяє працювати з файлами без розширень.

Синтаксичний аналіз вмісту виконується за допомогою спеціалізованої бібліотеки, яка формує структурований набір метаданих стандарту DICOM. З бінарного файлу витягуються параметри пацієнта, характеристики сканування та геометричні показники зрізів. Система зберігає ключові атрибути для

групування зображень, визначення їх просторової орієнтації та налаштування початкових параметрів візуалізації.

Валідація вхідних даних передбачає перевірку наявності обов'язкових ідентифікаторів для коректного групування та впорядкування зрізів. Пошкоджені об'єкти виключаються з обробки з формуванням відповідних звітів про помилки для користувача. Додатково контролюється узгодженість метричних параметрів у межах однієї серії для запобігання конфліктів при відображенні анатомічних структур.

Трансформація масиву пікселів у формат зображення здійснюється через лінійне перетворення інтенсивності на основі параметрів віконності. Значення щільності тканин масштабуються у діапазон яскравості, придатний для восьмибітного представлення на екрані. Такий підхід забезпечує оптимальний контраст для візуалізації специфічних структур шляхом відсікання діапазонів за межами обраного вікна.

Результат обробки кодується у формат PNG з подальшою конвертацією у текстовий рядок для передачі через програмний інтерфейс. Використання цього графічного формату гарантує збереження якості напівтонових зображень при ефективному стисненні однорідних ділянок. Текстове кодування забезпечує інтеграцію бінарних графічних даних у структуровані відповіді сервера без застосування складних протоколів передачі.

### 4.3.2 Управління серіями та оптимізація завантаження

Організація даних на сервері реалізована через ієрархічну структуру директорій з прив’язкою до унікальних ідентифікаторів сесій користувачів. Оригінальні файли зберігаються незмінними для забезпечення цілісності первинної медичної інформації. Паралельно створюється індексний файл з кешованими метаданими, що прискорює доступ до параметрів зрізів без необхідності повторного аналізу вихідних файлів.

Впорядкування зрізів виконується на основі просторових координат для відтворення правильної анатомічної послідовності. Основним критерієм сортування виступає позиція зображення вздовж вертикальної осі тіла

пацієнта. Результат впорядкування фіксується у метаданих сесії, що гарантує коректну навігацію незалежно від послідовності завантаження окремих файлів.

Для оптимізації роботи з великими масивами даних застосовано стратегію відкладеного завантаження зображень. Система виконує повний аналіз лише для метаданих, тоді як графічна конвертація відбувається динамічно при запиті конкретного зрізу. Це дозволяє миттєво розпочати роботу з серією, мінімізуючи використання оперативної пам'яті та час очікування.

Додаткова оптимізація забезпечується механізмом попереднього кешування сусідніх зображень під час перегляду. Система автоматично готує обмежений набір зрізів у фоновому режимі, зберігаючи їх у оперативній пам'яті сервера. Такий підхід гарантує миттєвий відгук інтерфейсу при навігації, запобігаючи при цьому вичерпанню системних ресурсів.

Збереження розмічених областей реалізовано через окрему підсистему, яка фіксує маски у форматі графічних файлів з каналом прозорості. Анотації розміщуються поруч із відповідними медичними файлами, дозволяючи незалежно керувати розміткою. Це забезпечує можливість експорту вихідних даних у незмінному вигляді без модифікації оригінальних зображень (див. рис. 4.7).

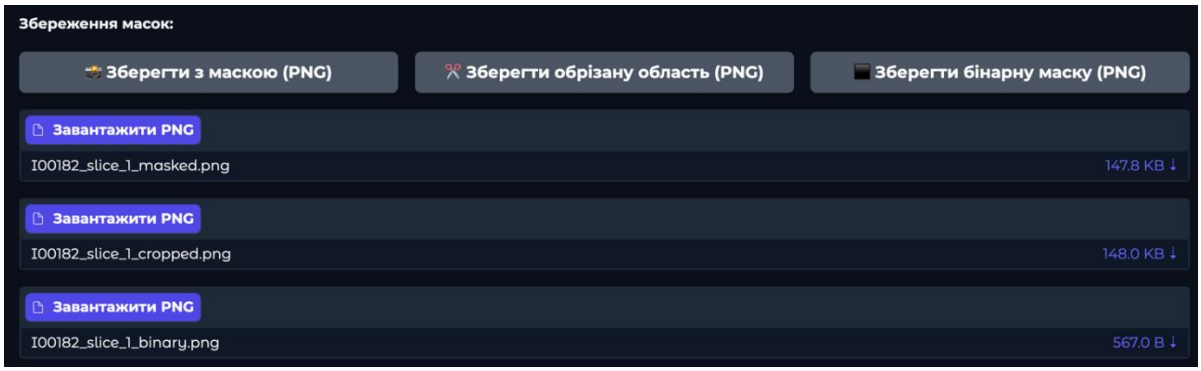


Рисунок 4.7 – Збереження маски

Функціонал експорту дозволяє генерувати результати роботи у трьох

різних форматах відповідно до запиту користувача. Система може створювати композитні зображення з накладеною розміткою, вирізані ділянки інтересу або бінарні карти масок. Генерація файлів відбувається шляхом математичних операцій над піксельними масивами базового зображення та шару анотації.

## 4.4 Система інтерактивного маскування регіонів інтересу

Система інтерактивного виділення областей інтересу є центральним функціональним компонентом платформи, що забезпечує медичним спеціалістам можливість точного окреслення патологічних утворень та підозрілих ділянок на томографічних зрізах для подальшого аналізу алгоритмами класифікації. Реалізація системи маскування базується на компоненті ImageEditor фреймворку Gradio, який надає вбудовану функціональність для інтерактивного редагування зображень з підтримкою шарів, інструментів малювання та збереження змін у стані інтерфейсу. Архітектура системи маскування розроблена з урахуванням специфічних вимог медичної візуалізації, зокрема необхідності роботи з напівпрозорими накладеннями для одночасного спостереження за підлягаючими тканинами та межами виділених областей.

### 4.4.1 Інструменти малювання та редагування масок

Основним засобом створення розмітки є інструмент пензля, який формує маску шляхом дискретизації траєкторії руху курсору на послідовність заповнених кіл. Технічна реалізація базується на векторному відстеженні координат миші з динамічним регулюванням діаметра інструменту через спеціалізований елемент керування, що дозволяє змінювати розмір області охоплення залежно від деталізації патологічних структур.

Візуалізація маски реалізована з використанням світло-блакитного кольору та фіксованим рівнем непрозорості 60%, що забезпечує оптимальний контраст відносно відтінків сірого на томограмах. Емпірично підібрані параметри гарантують чітке розрізнення меж виділення при збереженні

можливості візуального аналізу морфологічних характеристик тканин, що знаходяться під шаром розмітки.

Інструмент гумки функціонує за принципом модифікації альфа-каналу зображення, замінюючи значення пікселів маски на повністю прозорі вздовж траєкторії руху. Незалежне регулювання розміру інструменту дозволяє ефективно виконувати як точкову корекцію дрібних помилок сегментації, так і швидке очищення значних ділянок без залишкових артефактів напівпрозорості.

Критичною архітектурною особливістю є застосування одношарової моделі маскування, яка запобігає ефекту накопичення непрозорості при багаторазовому накладанні штрихів. Такий підхід гарантує візуальну однорідність розмітки, оскільки пікселі можуть набувати виключно бінарних станів прозорості або фіксованого рівня зафарбовування незалежно від інтенсивності малювання.

Навігаційні функції інтегровані безпосередньо в компонент редагування, забезпечуючи автоматичну активацію режиму переміщення при масштабуванні зображення понад нативний розмір. Технічна реалізація базується на обробці подій перетягування курсору з динамічним перерахунком координат області видимості, що дозволяє синхронно зміщувати томографічний зріз разом із накладеним шаром маски.

Функція повного очищення розмітки реалізована через механізм заміни поточного шару маски на нове порожнє зображення з ідентичними геометричними параметрами. Такий алгоритмічний підхід забезпечує миттєве скидання всіх виділених областей до прозорого стану без ресурсоємних ітерацій обробки окремих пікселів масиву.

### 4.4.2 Персистентність та експорт масок

Збереження масок відбувається автоматично при зміні активного зрізу завдяки системі відстеження подій навігації. Поточний стан редактора захоплюється, кодується у формат PNG із прозорістю та передається на сервер через HTTP-запит. Бекенд зберігає файл із прив'язкою до ідентифікатора серії,

що гарантує цілісність даних при подальшому зверненні.

Відновлення розмітки виконується симетрично під час переходу на раніше опрацьований зріз. Система перевіряє наявність збережених даних на сервері та автоматично накладає їх на зображення у разі успішного пошуку. Якщо маска відсутня, відображається порожній шар, що забезпечує коректну роботу механізму персистентності при багаторазовій навігації.

Кодування графічних даних для мережевої передачі реалізовано через двоетапну конвертацію растрового зображення. Спочатку маска перетворюється у формат PNG зі збереженням альфа-каналу, після чого трансформується у Base64-рядок для безпечної інтеграції в JSON-структури. Декодування на стороні сервера відбувається у зворотному порядку для відновлення бінарного файлу.

Функціонал експорту підтримує створення композитних зображень шляхом об'єднання томографічного зрізу з накладеною маскою. Використання альфа-композитингу дозволяє зберегти візуалізацію напівпрозорих областей у єдиному файлі. Такий формат є оптимальним для формування медичної звітності та презентаційних матеріалів.

Формат експорту обрізаної області базується на алгоритмі обчислення мінімального обмежуючого прямокутника навколо непрозорих пікселів маски. Відповідна ділянка витягується з оригінального зображення та зберігається як окремий файл без оточуючого контексту. Це дозволяє створювати спеціалізовані набори даних для навчання алгоритмів машинного навчання.

Генерація бінарних масок здійснюється шляхом порогової обробки альфа-каналу для створення монохромного зображення. Виділені області кодуються білим кольором, а фон чорним, що відповідає стандартам комп’ютерного зору. Такий формат забезпечує сумісність результатів сегментації із зовнішніми аналітичними системами.

## 4.5 Імплементація AI агента з LangGraph оркестрацією

Інтелектуальний агент системи реалізовано на базі бібліотеки LangGraph, що представляє собою спеціалізований фреймворк для створення статефул-процесів з складною логікою переходів між станами та динамічною маршрутизацією виконання. На відміну від традиційних чат-ботів з лінійною обробкою запитів, агент побудований як граф обчислень, де кожен вузол виконує специфічну функціональну роль, а умовні ребра визначають шляхи переходів залежно від контексту діалогу та результатів попередніх операцій. Архітектура агента забезпечує автономне прийняття рішень щодо необхідності виклику спеціалізованих інструментів, таких як ансамблева класифікація, автоматичне виявлення масок або надання рекомендацій з препроцесингу зображень, без потреби у жорстко закодованих правилах чи явних командах користувача.

### 4.5.1 Архітектура LangGraph та управління станом

Основою архітектури агента є концепція StateGraph, яка визначає топологію робочого процесу через вузли та ребра переходів. Граф ініціалізується схемою стану, що містить історію діалогу, дані про область інтересу із зображенням, результати класифікації та службову інформацію про виконання. Це забезпечує повноцінне функціонування агента та збереження контексту протягом усієї сесії взаємодії з користувачем.

Персистентність реалізовано через таблицю workflow_state у базі SQLite, де зберігається серіалізований стан після кожного кроку виконання. Структура запису включає ідентифікатори сесії та потоку, назву активного вузла та JSON-дані. Для коректного збереження специфічних типів даних, таких як масиви NumPy та моделі Pydantic, використовуються спеціальні енкодери під час серіалізації.

Відновлення діалогу відбувається автоматично шляхом завантаження та десеріалізації останнього запису з бази даних за ідентифікатором сесії. Цей механізм є критичним для реалізації контрольних точок, дозволяючи

призупинити процес перед важливими діями, зберегти стан і відновити виконання з точного місця після отримання підтвердження від користувача.

Історія повідомлень керується через таблицю chat_messages, що зберігає запити та відповіді з метаданими. При кожній взаємодії агент завантажує історію, формує розширений контекст для Azure OpenAI та додає системний промпт. Нові повідомлення автоматично записуються в базу, забезпечуючи безперервність контексту діалогу при наступних зверненнях.

Схема стану визначена через TypedDict, що гарантує статичну перевірку типів та автоматичну валідацію структури даних. Вона включає типізовані поля для історії розмови, координат області інтересу та результатів роботи алгоритмів. Така організація забезпечує коректність передачі даних між різними вузлами графа під час виконання операцій.

Вузли додаються до графа через прив'язку функцій-обробників, які приймають поточний стан і повертають оновлені дані. Реалізація підтримує як синхронні функції для локальних обчислень, так і асинхронні для взаємодії із зовнішніми API. Асинхронний підхід запобігає блокуванню циклу подій під час очікування відповідей від мережевих сервісів.

Логіка переходів визначається ребрами двох типів: безумовними для прямих з'єднань та умовними для динамічної маршрутизації. Безумовні ребра завжди ведуть до фіксованого наступного вузла, тоді як умовні використовують функцію-роутер для аналізу стану. Це дозволяє адаптувати шлях виконання процесу залежно від результатів попередніх обчислень.

### 4.5.2 Вузли робочого процесу та інтеграція з Azure OpenAI

Вузол обробки повідомлень є ключовим елементом діалогової системи, що забезпечує асинхронну взаємодію з Azure OpenAI для генерації відповідей у контексті аналізу томограм. Реалізація базується на клієнті AsyncOpenAI, який виконує неблокуючі виклики API, передаючи повну історію діалогу, системний промпт та визначення доступних інструментів для активації function calling.

Системний промпт налаштовує агента на роль медичного асистента,

спеціалізованого на аналізі КТ легень. Він містить інструкції щодо використання української медичної термінології, детального пояснення результатів класифікації, опису інструментів та правил безпечної роботи з даними. Також промпт допомагає виявляти наміри користувача для автоматичного виклику функцій.

Механізм function calling дозволяє агенту автономно вирішувати, коли використовувати спеціалізовані інструменти, аналізуючи контекст діалогу. Якщо модель GPT-5 повертає структурований виклик функції замість тексту, вузол оновлює стан графа та маршрутизує виконання до відповідного інструменту через умовне ребро.

Вузол маршрутизації аналізує стан системи, визначаючи подальший шлях виконання графа. Функція-роутер перевіряє наявність запитів на виклик функцій і повертає ідентифікатор відповідного вузла. Якщо виклики відсутні, роутер сигналізує про завершення обробки повідомлення.

Вузол класифікації взаємодіє з ансамблем алгоритмів машинного навчання для аналізу виділеної області інтересу. Він перевіряє наявність даних, декодує зображення, обчислює вектор ознак та паралельно запускає три класифікатори. Результати об'єднуються через зважене голосування, оновлюючи стан системи детальними даними класифікації.

Вузол виявлення масок автоматизує сегментацію патологій за допомогою методу Отсу та морфологічних операцій. Алгоритм визначає оптимальний поріг яскравості, застосовує ерозію та дилатацію для уточнення меж, після чого кодує результуючу маску для подальшого використання.

Вузол пояснення результатів трансформує технічні дані класифікації у зрозумілі медичні висновки (див. рис. 4.8). Він формує текстовий опис українською мовою, інтерпретуючи узгодженість алгоритмів та показники впевненості, а також надає рекомендації щодо подальших дій.

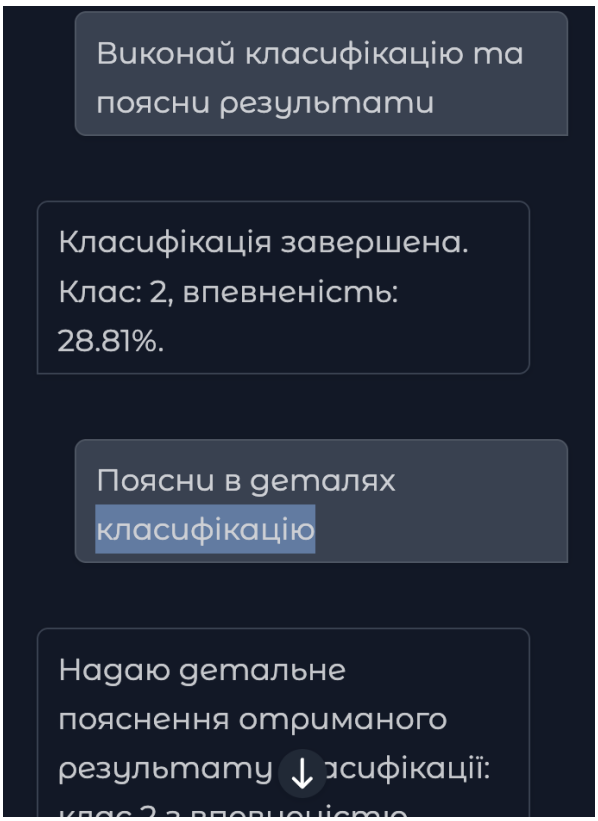


Рисунок 4.8 – Пояснення класифікації

Контрольні точки з участю людини дозволяють призупинити виконання графа перед критичними операціями, очікуючи підтвердження користувача. Стан системи зберігається в базі даних, а після отримання рішення користувача процес відновлюється з точки зупинки.

Обробка помилок реалізована через блоки try-except та механізм повторних запитів для API викликів. У разі критичних збоїв граф переходить до вузла обробки помилок, який надає користувачу зрозумілі пояснення та рекомендації для виправлення ситуації.

### 4.6 Архітектура ансамблевої системи класифікації

Ансамблева система класифікації представляє собою ядро аналітичного функціоналу платформи, що поєднує три різні алгоритми машинного навчання для забезпечення надійної діагностичної підтримки при аналізі патологічних утворень на томографічних зображеннях легень. Архітектурне рішення

базується на принципі ансамблевого навчання, де множинні класифікатори з різними математичними підходами до розпізнавання патернів працюють паралельно над однією задачею, а їхні індивідуальні передбачення об'єднуються через механізм зваженого голосування для формування фінального рішення з підвищеною точністю та стійкістю до помилок окремих алгоритмів. Вибір саме класичних алгоритмів машинного навчання замість глибоких нейронних мереж обґрунтований вимогами інтерпретованості рішень для медичної валідації, ефективністю роботи на обмежених обсягах тренувальних даних та можливістю пояснення логіки класифікації через аналіз важливості окремих ознак.

### 4.6.1 Витягування ознак та підготовка даних

Процес витягування ознак реалізовано як обчислювальний конвеєр, що трансформує область інтересу у вектор числових характеристик. Аналіз фокусується виключно на пікселях маски, ігноруючи фон для уникнення артефактів. Повний цикл обробки займає до ста мілісекунд, забезпечуючи високу інтерактивність системи.

Перша група ознак описує статистичні параметри інтенсивності пікселів у одиницях Хаунсфілда. Обчислюються середнє значення, стандартне відхилення, коефіцієнти асиметрії та ексцесу, а також гістограма розподілу яскравості. Ця категорія формує двадцять чотири показники щільності тканин.

Друга категорія базується на аналізі текстурних властивостей через матрицю спільної зустрічальності сірих рівнів. З неї витягуються дескриптори Харалика, такі як контрастність, однорідність, енергія та кореляція. Це дозволяє кількісно описати просторові взаємозв'язки між пікселями зображення.

Додатково застосовується метод локальних бінарних патернів для кодування мікроструктури околу кожного пікселя. Гістограма отриманих кодів характеризує домінуючі текстурні візерунки незалежно від змін освітлення. Загалом група текстурних ознак містить понад двадцять п'ять значень.

Третя категорія охоплює геометричні характеристики форми, отримані шляхом аналізу контуру маски. За допомогою бібліотеки OpenCV обчислюються площа, периметр та компактність виділеної області. Ці параметри описують морфологію об'єкта незалежно від його внутрішньої структури.

Моменти зображення використовуються для створення інваріантного опису форми щодо масштабу та повороту. Обчислення інваріантів Ху та ексцентриситету дозволяє розпізнавати патологічні структури у будь-якій орієнтації. Ця група доповнює загальний вектор ознак до ста елементів.

Перед класифікацією виконується стандартизація вектора для приведення ознак до нульового середнього та одиничної дисперсії. Це критично важливо для коректної роботи методу опорних векторів та числової стабільності інших алгоритмів. Параметри нормалізації зберігаються з етапу навчання моделей.

### 4.6.2 Класифікатори та механізм ансамблевого голосування

Перший алгоритм ансамблю реалізовано як випадковий ліс, що складається зі ста п’ятдесяти незалежних дерев рішень. Кожне дерево навчається на випадковій підвибірці даних, що забезпечує декореляцію помилок та підвищує стійкість моделі [149]. Фінальний клас визначається шляхом мажоритарного голосування, де враховується частка дерев, що підтримали певне рішення.

Другий метод базується на опорних векторах із використанням радіальної базисної функції для нелінійного розділення класів. Параметри регуляризації та ядра налаштовані для балансу між точністю класифікації та здатністю до узагальнення. Перед обробкою даних виконується обов'язкова стандартизація ознак для забезпечення коректної роботи алгоритму у багатовимірному просторі.

Третій класифікатор використовує градієнтний бустинг, який послідовно будує двісті дерев рішень для ітеративного виправлення помилок попередніх кроків. Алгоритм фокусується на складних для розпізнавання

прикладах шляхом оптимізації функції втрат. Гіперпараметри моделі обмежено для запобігання перенавчанню та збереження високої швидкості роботи.

Процес класифікації виконується паралельно за допомогою асинхронних механізмів, що дозволяє одночасно запускати всі три моделі. Кожен потік незалежно завантажує алгоритм, обробляє вхідні дані та повертає вектор ймовірностей. Такий підхід скорочує загальний час інференсу до показників найповільнішої моделі, що становить близько двадцяти мілісекунд.

Система приймає фінальне рішення на основі зваженого голосування, враховуючи надійність кожного алгоритму. Випадковому лісу та градієнтному бустингу присвоєно вищий пріоритет завдяки їхній стабільності та точності. Метод опорних векторів має дещо меншу вагу через чутливість до налаштувань ядра.

Результуючий прогноз формується шляхом зваженого усереднення ймовірностей від усіх класифікаторів. Клас із найвищою сумарною оцінкою визначається як діагноз із відповідним рівнем впевненості. Додатково розраховується метрика узгодженості між алгоритмами для оцінки надійності отриманого висновку.

Усі результати аналізу зберігаються у базі даних із прив'язкою до поточної сесії та часових міток. Запис включає індивідуальні прогнози, ансамблеве рішення та повні вектори ймовірностей. Це дозволяє медичним спеціалістам переглядати історію діагностики та оцінювати стабільність висновків при зміні параметрів сегментації.

### 4.7 Пайплайн екстракції ознак та попередньої обробки

Пайплайн екстракції ознак та попередньої обробки медичних зображень являє собою критично важливу підсистему платформи, що трансформує вихідні томографічні дані з формату DICOM у структуровані числові вектори, придатні для аналізу алгоритмами машинного навчання. Архітектура

пайплайну побудована на принципах модульності та послідовності обробки, де кожен етап виконує специфічну трансформацію даних та передає результат наступному компоненту через чітко визначені інтерфейси. Ефективність пайплайну забезпечується через застосування векторизованих операцій бібліотеки NumPy для обробки масивів пікселів, кешування проміжних результатів для уникнення повторних обчислень при багаторазовому аналізі тієї ж області інтересу, та асинхронного виконання незалежних етапів обробки для максимального використання доступних обчислювальних ресурсів.

### 4.7.1 Попередня обробка та нормалізація зображень

Початковий етап пайплайну передбачає валідацію вхідних даних на коректність формату та повноту інформації. Система перевіряє структуру закодованого зображення, успішність декодування та відповідність мінімальним геометричним вимогам. Області розміром менше десяти пікселів автоматично відхиляються через неможливість проведення статистично значущого текстурного аналізу.

Всі вхідні зображення конвертуються у монохромний формат з урахуванням особливостей сприйняття кольорів людським оком. Процес включає обробку альфа-каналу через композитинг з білим фоном для коректної заміни прозорих ділянок. Це забезпечує уніфікацію даних перед подальшим аналізом інтенсивності тканин та запобігає помилкам інтерпретації.

Процедура нормалізації приводить значення інтенсивності пікселів до стандартного восьмибітного інтервалу через лінійне масштабування. Це нівелює вплив відмінностей у налаштуваннях сканування та параметрах візуалізації томограми. Такий підхід гарантує коректність порівняльного аналізу ознак незалежно від абсолютних значень яскравості вихідного знімка.

Для покращення видимості деталей у зонах з низькою варіативністю застосовується адаптивна еквалізація гістограми. Метод обробляє зображення локальними фрагментами, що дозволяє підсилити контрастність структур без створення глобальних артефактів. Це забезпечує чітке розрізнення тканин із

близькими показниками щільності при збереженні природного вигляду.

Фільтрація шумів здійснюється за допомогою медіанного фільтру малого розміру, який ефективно усуває імпульсні перешкоди та артефакти реконструкції. Цей метод має перевагу над гаусовим згладжуванням завдяки здатності зберігати чіткість меж патологічних утворень. Обмеження розміру вікна дозволяє зберегти діагностично важливі деталі мікроструктури тканин.

Виділення області інтересу відбувається шляхом аналізу альфа-каналу прозорості або специфічних кольорових маркерів. Система застосовує порогові значення непрозорості або детектує контрастні відтінки маскування для відокремлення цільової зони від фону. Це дозволяє точно ідентифікувати межі патології незалежно від формату представлення вхідних даних.

Для фінального уточнення меж застосовуються морфологічні операції закриття та фільтрації зв'язних компонентів. Це дозволяє усунути дрібні дефекти сегментації та об'єднати розрізнені фрагменти у суцільну структуру. Отримана бінарна маска забезпечує високу точність при подальшому розрахунку геометричних характеристик новоутворення.

### 4.7.2 Конвеєр витягування та агрегації ознак

Обчислення статистики інтенсивності реалізовано векторизованими методами NumPy над масивом пікселів маски. Функція розраховує середнє значення, стандартне відхилення, асиметрію та ексцес для вибраної області. Гістограма розподілу формується шляхом розбиття діапазону яскравості на шістнадцять рівномірних інтервалів.

Текстурні ознаки витягуються бібліотекою scikit-image через матрицю спільної зустрічальності сірих рівнів. Аналіз виконується для сусідніх пікселів у чотирьох основних кутових напрямках. Дескриптори Харалика усереднюються для отримання інваріантних до обертання характеристик.

Локальні бінарні патерни кодують окіл пікселів шляхом порівняння з вісьмома сусідами. Використання режиму uniform скорочує розмірність гістограми до п'ятдесяти дев'яти класів однорідних патернів. Нормалізація розподілу забезпечує незалежність отриманих ознак від розміру області

інтересу.

Геометричні характеристики форми отримуються через аналіз контуру маски функціями бібліотеки OpenCV. Алгоритм визначає зовнішні границі для обчислення площі та периметру виділеної області. Додатково розраховуються моменти зображення третього порядку для опису розподілу маси.

Інваріанти Ху обчислюються на основі нормалізованих центральних моментів для забезпечення стійкості до трансформацій об'єкта. Функція повертає сім характеристик, до яких застосовується логарифмічне перетворення. Це покращує числову стабільність та узгоджує масштаби значень з іншими ознаками.

Агрегація ознак у єдиний вектор виконується конкатенацією масивів інтенсивності, текстури та форми. Результуючий одновимірний масив містить до ста елементів залежно від параметрів алгоритмів. Перед об'єднанням проводиться валідація структури даних на відсутність помилок.

Стандартизація вектора виконується попередньо навченим об'єктом StandardScaler з бібліотеки scikit-learn. Трансформація приводить дані до нульового середнього та одиничної дисперсії. Це необхідно для коректної роботи методу опорних векторів та стабільності ансамблю.

Кешування результатів реалізовано через збереження векторів за унікальним хешем SHA256 від вхідних даних. Механізм запобігає повторним обчисленням для ідентичних областей інтересу. Структура кешу має обмежений розмір для контролю використання оперативної пам'яті.

### 4.8 Реалізація персистентності даних та repository pattern

Підсистема персистентності даних забезпечує надійне збереження всієї інформації про сесії діалогів, результати класифікації, виділені області інтересу та стани робочих процесів інтелектуального агента з можливістю ефективного пошуку та відновлення даних при продовженні роботи користувачів з системою. Архітектурне рішення базується на реляційній базі

даних SQLite з асинхронним драйвером aiosqlite та ORM-фреймворком SQLAlchemy версії 2.0 для забезпечення неблокуючого доступу до даних без призупинення обробки інших запитів під час виконання операцій введення-виведення з диском. Застосування патерну репозиторію забезпечує чітке відокремлення логіки доступу до даних від бізнес-логіки системи, що дозволяє тестувати компоненти незалежно та при необхідності замінювати конкретну реалізацію сховища даних без модифікації коду вищих шарів додатку.

### 4.8.1 Структура бази даних та асинхронна ORM

Схема бази даних складається з п'яти взаємопов'язаних таблиць, що моделюють всі аспекти роботи інтелектуального агента з медичними зображеннями та діалоговою взаємодією з користувачами. Таблиця chat_sessions зберігає інформацію про сесії діалогів з полями для унікального ідентифікатора сесії, ідентифікатора користувача для підтримки багатокористувацької роботи, часових міток створення та останньої активності для відстеження тривалості та інтенсивності використання системи, а також опціонального поля метаданих у форматі JSON для збереження додаткового контексту, такого як ідентифікатор завантаженої серії DICOM або налаштування інтерфейсу користувача. Таблиця chat_messages містить всі повідомлення діалогів з зовнішнім ключем на ідентифікатор сесії, роллю відправника як перелічуваним типом з значеннями користувач або асистент, текстовим вмістом повідомлення, часовою міткою створення та опціональним полем метаданих для збереження додаткової інформації про контекст повідомлення, такої як закодоване зображення або параметри виклику інструментів.

Таблиця roi_data зберігає інформацію про виділені області інтересу з унікальним ідентифікатором запису, зовнішнім ключем на ідентифікатор сесії діалогу, номером зрізу томографічної серії для прив'язки області до конкретного зображення, закодованими у base64 даними маски як текстовим полем великого розміру, координатами обмежуючого прямокутника у форматі JSON з полями для мінімальних та максимальних значень по осях x та y,

показником впевненості для автоматично виявлених масок з діапазоном від нуля до одиниці, та часовою міткою створення запису. Таблиця classification_results зберігає результати виконання ансамблевої класифікації з зовнішнім ключем на ідентифікатор області інтересу для встановлення зв'язку між виділенням та його аналізом, окремими полями для передбачених класів та показників впевненості кожного з трьох індивідуальних алгоритмів, полями для фінального ансамблевого рішення з його впевненістю та метрикою узгодженості алгоритмів, повним вектором ймовірностей для всіх шістнадцяти класів у форматі JSON для збереження детального розподілу передбачень, та часовою міткою виконання класифікації.

Таблиця workflow_state відповідає за персистентність станів виконання графа LangGraph з полями для ідентифікатора сесії як зовнішнього ключа, ідентифікатора потоку виконання для підтримки паралельних гілок обробки у складних сценаріях, назви поточного активного вузла графа як рядкового значення, серіалізованих даних стану у форматі JSON з повною інформацією про контекст виконання включаючи історію діалогу та проміжні результати, статусу виконання як перелічуваного типу з значеннями активний, призупинено або завершено, та часової мітки останньої модифікації стану для відстеження прогресу виконання робочого процесу. Всі таблиці містять первинні ключі у вигляді автоінкрементних цілочисельних ідентифікаторів або UUID для забезпечення унікальності записів, а зовнішні ключі налаштовані з каскадним видаленням для автоматичного очищення залежних записів при видаленні батьківських сутностей.

Декларативні моделі SQLAlchemy визначають відображення між таблицями бази даних та Python-класами через використання базового класу Base з модуля declarative_base та анотацій полів через клас Column з специфікацією типів даних та обмежень. Модель ChatSession визначається як клас з метаданими таблиці через атрибут tablename та полями id як Column з типом Integer та параметром primary_key для первинного ключа, user_id як Column з типом String для ідентифікації користувача, created_at та updated_at

як Column з типом DateTime та значенням за замовчуванням через функцію datetime.utcnow для автоматичного встановлення часових міток, та metadata як Column з типом JSON для збереження довільних структурованих даних. Відношення один-до-багатьох між сесіями та повідомленнями визначається через атрибут relationship з параметром back_populates для створення двонаправленого зв'язку між моделями, що дозволяє навігувати від сесії до списку повідомлень та від повідомлення до батьківської сесії через атрибути об'єктів.

Асинхронний механізм роботи з базою даних реалізовано через модуль ext.asyncio фреймворку SQLAlchemy версії 2.0, що надає асинхронні версії основних компонентів ORM для інтеграції з event loop Python без блокування виконання під час операцій з базою даних. Функція create_async_engine створює асинхронний движок для підключення до бази даних SQLite з параметром connect_args для передачі специфічних налаштувань драйверу aiosqlite, таких як увімкнення підтримки зовнішніх ключів через pragma foreign_keys. Функція async_sessionmaker створює фабрику асинхронних сесій з налаштуванням параметрів ізоляції транзакцій, автоматичного фіксування змін та закриття з'єднань після завершення операцій, що забезпечує консистентне управління життєвим циклом сесій у всіх компонентах системи.

Контекстні менеджери використовуються для автоматичного управління сесіями бази даних через синтаксис async with для забезпечення коректного закриття з'єднань та відкату транзакцій у випадку виникнення помилок під час виконання операцій. Асинхронна функція get_session виступає як генератор, що створює нову сесію через виклик фабрики sessionmaker, повертає її через yield для використання у тілі контекстного менеджера, та гарантує виклик методу close після завершення роботи з сесією незалежно від успішності виконання операцій або виникнення винятків. Інтеграція з механізмом впровадження залежностей FastAPI виконується через використання функції Depends з передачею генератора get_session, що автоматично інжектує асинхронну сесію бази даних у параметри обробників

ендпоінтів API без необхідності явного управління життєвим циклом сесій у коді обробників.

Виконання запитів до бази даних організоване через конструктор запитів SQLAlchemy з використанням функції select для формування SELECT-запитів, where для додавання умов фільтрації, join для об'єднання таблиць через зовнішні ключі, order_by для сортування результатів та limit з offset для пагінації великих наборів даних. Асинхронне виконання запитів відбувається через виклик методу execute об'єкта сесії з передачею сконструйованого запиту, що повертає об'єкт Result з методами для отримання результатів у різних форматах таких як scalars для отримання одного стовпця значень, all для отримання всіх рядків як списку, first для отримання першого рядка або None при відсутності результатів, та one для отримання одного рядка з генерацією винятку при відсутності результатів або наявності більше одного рядка.

### 4.8.2 Імплементація repository pattern та операції з даними

Архітектура патерну репозиторію базується на визначенні абстрактних інтерфейсів у шарі домену, що специфікують контракти для операцій з персистентними сутностями без прив'язки до конкретної технології зберігання даних. Абстрактний базовий клас ChatSessionRepository визначає методи create_session для створення нової сесії діалогу з параметрами ідентифікатора користувача та опціональних метаданих, get_session_by_id для отримання сесії за унікальним ідентифікатором з можливістю завантаження пов'язаних сутностей через eager loading, update_session для модифікації полів існуючої сесії з автоматичним оновленням часової мітки останньої активності, delete_session для видалення сесії з каскадним видаленням всіх пов'язаних повідомлень та результатів класифікації, та list_sessions_by_user для отримання списку сесій конкретного користувача з підтримкою пагінації та сортування за часовими мітками. Всі методи інтерфейсу визначені як асинхронні через префікс async def та повертають типізовані результати через анотації типів Python для забезпечення статичної перевірки коректності

використання методів.

Конкретна реалізація SQLAlchemyChatSessionRepository у шарі інфраструктури імплементує всі методи абстрактного інтерфейсу з використанням асинхронної сесії SQLAlchemy для виконання операцій з базою даних. Метод create_session створює новий екземпляр моделі ChatSession з переданими параметрами, додає його до сесії через session.add, виконує фіксацію транзакції через session.commit для збереження змін у базі даних, оновлює об'єкт через session.refresh для завантаження згенерованих базою даних значень таких як автоінкрементний ідентифікатор, та повертає створений об'єкт сесії діалогу. Метод get_session_by_id формує SELECT-запит через select з умовою where для фільтрації за ідентифікатором та опціонально додає options з joinedload для завантаження пов'язаних повідомлень у тому ж запиті для уникнення проблеми N плюс один запит, виконує запит через session.execute та повертає перший результат або None при відсутності сесії з заданим ідентифікатором.

Операції створення повідомлень діалогу реалізовані у репозиторії ChatMessageRepository з методом create_message, що приймає ідентифікатор сесії, ролль відправника, текстовий вміст та опціональні метадані, створює екземпляр моделі ChatMessage з встановленням зовнішнього ключа на батьківську сесію, зберігає його у базі даних через session.add та session.commit, та повертає створене повідомлення з усіма згенерованими полями. Метод get_messages_by_session завантажує всі повідомлення для заданої сесії діалогу через SELECT-запит з умовою фільтрації за ідентифікатором сесії та сортуванням за часовою міткою створення у порядку зростання для відображення хронологічної послідовності діалогу, виконує запит та повертає список об'єктів повідомлень для використання у формуванні контексту для моделі Azure OpenAI або відображення історії у інтерфейсі чату.

Репозиторій ROIDataRepository управляє персистентністю даних про виділені області інтересу з методом save_roi_data для збереження закодованого зображення маски разом з координатами обмежуючого

прямокутника та показником впевненості для автоматично виявлених областей. Реалізація методу створює новий запис у таблиці roi_data з встановленням зовнішнього ключа на сесію діалогу, серіалізацією координат у JSON-формат через json.dumps для збереження структурованих даних у текстовому полі, збереженням base64-рядка зображення маски у великому текстовому полі, та фіксацією транзакції для гарантування персистентності даних. Метод get_roi_by_session_and_slice завантажує збережену область інтересу для конкретної комбінації ідентифікатора сесії та номера зрізу томографічної серії через запит з двома умовами фільтрації, що дозволяє відновити маску при поверненні користувача до попередньо опрацьованого зрізу для продовження роботи з тією ж областю виділення.

Зберігання результатів класифікації організоване через репозиторій ClassificationResultRepository з методом save_classification_result, що приймає ідентифікатор області інтересу, передбачення та показники впевненості трьох індивідуальних алгоритмів, фінальне ансамблеве рішення з метрикою узгодженості, та повний вектор ймовірностей для всіх класів. Метод створює запис у таблиці classification_results з встановленням зовнішнього ключа на аналізовану область інтересу, збереженням скалярних значень передбачень та впевненості у окремих стовпцях для ефективного індексування та пошуку, серіалізацією вектора ймовірностей у JSON-формат для збереження детального розподілу передбачень, та автоматичним встановленням часової мітки виконання класифікації. Метод get_classification_history_by_session завантажує всі результати класифікації для заданої сесії діалогу через JOIN-запит між таблицями classification_results та roi_data для отримання повного контексту кожного аналізу включаючи дані про аналізовану область та часові мітки виконання.

Управління станами робочого процесу LangGraph реалізоване через репозиторій WorkflowStateRepository з методами save_workflow_state та load_workflow_state для персистентності та відновлення контексту виконання графа обчислень. Метод збереження приймає ідентифікатор сесії,

ідентифікатор потоку виконання, назву поточного активного вузла, серіалізований стан графа у форматі JSON з повною історією діалогу та результатами виконання інструментів, та статус виконання, після чого виконує операцію upsert через спробу вставки нового запису з обробкою винятку унікальності первинного ключа для оновлення існуючого запису при повторному збереженні стану тієї ж сесії. Метод завантаження виконує SELECT-запит з фільтрацією за ідентифікатором сесії та опціонально ідентифікатором потоку для отримання останнього збереженого стану, десеріалізує JSON-дані у Python-структури через json.loads, та повертає об'єкт стану для ініціалізації графа LangGraph у збереженій конфігурації для продовження виконання з точки призупинення.

Транзакційна консистентність забезпечується через використання контекстних менеджерів для управління життєвим циклом сесій бази даних з автоматичним відкатом змін у випадку виникнення винятків під час виконання послідовності операцій. Складні бізнес-операції, що вимагають атомарного виконання множинних запитів до різних таблиць, обгортаються у єдину транзакцію через початок сесії, виконання всіх необхідних операцій створення, оновлення або видалення записів, та фінальний commit для фіксації всіх змін одночасно або rollback для скасування всіх змін у випадку помилки на будь-якому етапі. Таке транзакційне управління критично важливе для операцій, що створюють результат класифікації разом з оновленням статусу області інтересу та додаванням повідомлення агента до історії діалогу, де часткове виконання могло б призвести до неконsistentного стану даних з результатами класифікації без відповідного повідомлення у діалозі.

Міграції схеми бази даних організовані через механізм Alembic, що інтегрується з SQLAlchemy для автоматичної генерації скриптів міграцій на основі змін у визначеннях моделей та забезпечує версіонування схеми з можливістю застосування міграцій у прямому напрямку для оновлення бази даних до нової версії або у зворотному напрямку для відкату до попередньої версії. Кожна міграція представлена Python-скриптом з функціями upgrade для

застосування змін та downgrade для їх скасування, що містять операції створення таблиць через op.create_table, додавання стовпців через op.add_column, створення індексів через op.create_index, та визначення зовнішніх ключів через op.create_foreign_key. Історія міграцій зберігається у спеціальній таблиці alembic_version бази даних для відстеження поточної версії схеми та визначення необхідних міграцій для приведення бази даних до актуального стану відповідно до версії коду додатку.

Індексування таблиць виконано для оптимізації найбільш частих запитів пошуку та фільтрації даних через створення індексів на стовпцях, що використовуються у умовах WHERE та JOIN-операціях. Таблиця chat_messages містить індекс на стовпці session_id для прискорення завантаження всіх повідомлень діалогу за ідентифікатором сесії, а також композитний індекс на парі session_id та created_at для ефективного виконання запитів з сортуванням повідомлень у хронологічному порядку. Таблиця roi_data має композитний індекс на стовпцях session_id та slice_number для швидкого пошуку збережених масок для конкретних зрізів томографічних серій. Таблиця classification_results індексована за стовпцем roi_id для завантаження результатів класифікації за ідентифікатором області інтересу та за часовою міткою created_at для відображення історії класифікацій у хронологічному порядку.

Обробка помилок на рівні репозиторіїв включає перехоплення специфічних винятків SQLAlchemy для трансформації технічних помилок бази даних у доменні винятки з зрозумілими повідомленнями для вищих шарів додатку. Виняток IntegrityError перехоплюється при порушенні обмежень унікальності або зовнішніх ключів з генерацією доменного винятку EntityAlreadyExistsException або InvalidReferenceException залежно від типу порушеного обмеження. Виняток OperationalError обробляється для відновлення після тимчасових проблем з підключенням до бази даних через механізм повторних спроб з експоненційною затримкою. Всі невідомі помилки логуються з повним трасуванням стеку викликів для діагностики

проблем та повторно викидаються як загальний RepositoryException з оригінальним повідомленням помилки для інформування вищих шарів про неможливість виконання операції з даними.

## 4.9 Інтеграція мікросервісів та міжсервісна комунікація

Архітектура платформи побудована на принципах мікросервісного проектування, де функціональність системи розподілена між трьома незалежними сервісами, кожен з яких виконує специфічну роль у загальному процесі обробки медичних зображень та надання діагностичної підтримки. Взаємодія між сервісами організована через стандартизовані HTTP REST API з використанням JSON як основного формату обміну структурованими даними та кодування base64 для передачі бінарного вмісту зображень у текстових повідомленнях. Така архітектурна організація забезпечує слабке зв'язування компонентів системи з можливістю незалежного розгортання, масштабування та оновлення окремих сервісів без впливу на функціонування інших частин платформи, що критично важливо для підтримки високої доступності медичного програмного застосунку у продакшн-середовищі.

### 4.9.1 HTTP REST API та асинхронна міжсервісна взаємодія

Комунікаційна модель між фронтенд-сервісом та основним бекендом базується на клієнт-серверній архітектурі, де фронтенд виступає ініціатором запитів для операцій з DICOM-даними, а бекенд надає RESTful API для обробки цих запитів з поверненням результатів у стандартизованому форматі. Фронтенд-сервіс використовує асинхронний HTTP-клієнт httpx для виконання неблокуючих запитів до бекенду, що дозволяє інтерфейсу Gradio залишатися відповідним під час виконання тривалих операцій завантаження або обробки медичних файлів. Клієнт httpx налаштовано з таймаутом тридцять секунд для запобігання необмеженому очікуванню відповідей від перевантаженого або недоступного бекенд-сервісу, з автоматичним генеруванням винятку TimeoutError при перевищенні встановленого ліміту часу для інформування

користувача про проблеми з доступністю сервісу.

Ендпоінт завантаження DICOM-файлів POST /api/v1/dicom/upload приймає множинні файли через multipart/form-data кодування, що дозволяє передавати бінарний вміст файлів разом з додатковими метаданими у формі параметрів запиту. Фронтенд формує запит через метод post клієнта httpx з параметром files, що містить словник відображення імен полів форми на кортежі з іменами файлів та бінарним вмістом, завантаженим користувачем через компонент File Upload інтерфейсу Gradio. Бекенд обробляє отримані файли через FastAPI параметр UploadFile, що надає асинхронний інтерфейс для читання вмісту файлів без завантаження всього файлу у пам'ять одночасно, виконує парсинг DICOM-даних, зберігає файли у файловій системі та повертає JSON-відповідь з ідентифікатором створеної серії, кількістю розпізнаних зрізів та статусом успішності операції для відображення у інтерфейсі користувача.

Отримання окремих томографічних зрізів організоване через ендпоінт GET /api/v1/dicom/{series_id}/slice/{slice_number} з параметрами шляху для ідентифікації серії та номера зрізу, а також опціональними параметрами запиту для специфікації параметрів віконності відображення. Фронтенд виконує запит при зміні поточного зрізу через навігаційні елементи інтерфейсу, передаючи обрані користувачем налаштування вікна та рівня для отримання зображення з відповідною візуалізацією тканин. Бекенд завантажує відповідний DICOM-файл з файлової системи, застосовує трансформацію інтенсивності згідно з параметрами віконності, перетворює масив пікселів у формат PNG, кодує результат у base64-рядок та повертає JSON-відповідь з полем image_data, що містить закодоване зображення, разом з метаданими зрізу такими як товщина, позиція у просторі пацієнта та розміри у пікселях. Фронтенд декодує отримане зображення та відображає його у компоненті ImageEditor для перегляду користувачем.

Збереження масок виділених областей реалізовано через ендпоінт POST /api/v1/dicom/{series_id}/mask з передачею ідентифікатора серії у шляху

запиту та закодованого зображення маски разом з номером зрізу у тілі запиту як JSON-структура. Фронтенд автоматично викликає цей ендпоінт при навігації користувача на інший зріз після створення виділення на поточному зрізі, захоплюючи стан маски з компонента ImageEditor, кодуючи растрове зображення у base64-формат та відправляючи на бекенд для персистентного зберігання. Бекенд декодує отримане зображення, зберігає його у файловій системі поряд з відповідним DICOM-файлом з іменем, що включає номер зрізу для унікальної ідентифікації, та повертає підтвердження успішності операції з ідентифікатором збереженої маски для можливості подальшого посилання на це виділення при виконанні класифікації або експорту результатів.

Взаємодія фронтенд-сервісу з AI-бекендом організована через окремий набір ендпоінтів для управління діалоговими сесіями та обміну повідомленнями з інтелектуальним агентом. Ендпоінт POST /chat/session створює нову сесію діалогу при завантаженні інтерфейсу користувачем, приймаючи опціональний ідентифікатор користувача для підтримки багатокористувацької роботи та повертаючи унікальний ідентифікатор сесії для використання у подальших запитах відправки повідомлень. Фронтенд зберігає отриманий ідентифікатор сесії у стані компонентів Gradio для автоматичного включення у всі запити взаємодії з агентом протягом поточної робочої сесії користувача. Ендпоінт POST /chat/message приймає ідентифікатор сесії, текст повідомлення користувача та опціональний контекст у вигляді словника з додатковими даними, що можуть включати закодоване зображення поточного томографічного зрізу, номер активного зрізу, параметри віконності та інформацію про наявність виділеної області інтересу.

AI-бекенд обробляє отримані повідомлення через ініціалізацію або відновлення стану графа LangGraph для заданої сесії, додавання нового повідомлення користувача до історії діалогу у стані графа, запуск виконання графа від початкового вузла з проходженням через послідовність вузлів обробки повідомлення, маршрутизації інструментів, виконання класифікації

або інших операцій залежно від намірів користувача, та формування фінальної відповіді агента після завершення виконання графа. Результат виконання повертається фронтенду як JSON-структура з полем response_text, що містить згенеровану відповідь агента українською мовою, опціональним полем classification_results з детальними результатами аналізу при виконанні класифікації, та полем conversation_history з оновленою історією діалогу для синхронізації стану між клієнтом та сервером.

Стандартизована структура відповідей API забезпечує консистентну обробку результатів запитів на стороні клієнтів через визначення єдиного формату JSON-об'єктів з обов'язковими полями status для індикації успішності операції зі значеннями success або error, data для корисного навантаження відповіді з фактичними результатами виконання операції, та message для текстового опису результату або деталей помилки у випадку невдачі виконання запиту. Такий формат дозволяє фронтенд-коду універсально обробляти відповіді від різних ендпоінтів через перевірку поля status для визначення успішності та відповідного відображення даних з поля data або повідомлення про помилку з поля message у інтерфейсі користувача без необхідності специфічної логіки для кожного типу операції.

Обробка помилок міжсервісної комунікації організована через ієрархію винятків з перехопленням помилок мережевого рівня, HTTP-статусів помилок та помилок парсингу відповідей для надання користувачам зрозумілих повідомлень про причини невдач операцій. Виняток httpx.ConnectError перехоплюється при неможливості встановлення з'єднання з цільовим сервісом через його недоступність або мережеві проблеми з відображенням повідомлення про тимчасову недоступність сервісу та рекомендації повторити операцію через декілька хвилин. Виняток httpx.TimeoutError обробляється при перевищенні встановленого таймауту очікування відповіді з інформуванням користувача про довгу тривалість обробки запиту та можливість збільшення таймауту або оптимізації операції. HTTP-статуси помилок з діапазону чотириста обробляються як помилки клієнта з відображенням повідомлення

про некоректні параметри запиту, а статуси з діапазону п'ятсот як помилки сервера з логуванням деталей для діагностики проблем на боці бекенд-сервісів.

Налаштування CORS middleware на бекенд-сервісах забезпечує можливість міжсервісних запитів від фронтенд-додатку, що працює на іншому порту або домені, через додавання відповідних HTTP-заголовків до відповідей API. FastAPI middleware CORSMiddleware налаштовано з параметром allow_origins, що містить список дозволених джерел запитів включаючи адресу фронтенд-сервісу http://localhost:7860 для локального розгортання, параметром allow_methods зі значенням asterisk для дозволу всіх HTTP-методів включаючи GET, POST, PUT та DELETE, параметром allow_headers також зі значенням asterisk для дозволу довільних заголовків у запитах клієнтів, та параметром allow_credentials зі значенням true для підтримки передачі cookies та авторизаційних токенів у міжсервісних запитах. Така конфігурація дозволяє фронтенду вільно взаємодіяти з обома бекенд-сервісами без обмежень браузерної політики same-origin security, що критично важливо для функціонування розподіленої мікросервісної архітектури з окремими портами для кожного компонента.

### 4.9.2 Оркестрація контейнерів та управління залежностями

Розгортання трьох мікросервісів системи організоване через Docker Compose, що надає декларативний підхід до визначення конфігурації множинних контейнерів з їхніми залежностями, мережевими налаштуваннями та томами для персистентного зберігання даних. Файл docker-compose.yml містить визначення трьох сервісів з іменами frontend, backend та ai_backend, кожен з яких специфікує параметри побудови образу контейнера через секцію build з вказівкою шляху до директорії з Dockerfile та контексту побудови, порти для маппінгу між хостовою системою та контейнером через секцію ports для забезпечення доступності сервісів ззовні, томи для монтування директорій хостової системи у контейнер через секцію volumes для персистентності даних та можливості гарячого перезавантаження коду під час розробки, та змінні

середовища через секцію environment для передачі конфігураційних параметрів таких як ключі API та адреси інших сервісів.

Конфігурація фронтенд-сервісу у Docker Compose специфікує побудову образу з директорії services/frontend з використанням Dockerfile, що базується на офіційному образі python:3.12-slim для мінімізації розміру результуючого образу, маппінг порту сім тисяч вісімсот шістдесят хостової системи на той самий порт контейнера для доступу до веб-інтерфейсу Gradio через браузер за адресою http://localhost:7860, монтування директорії вихідного коду services/frontend у контейнер для автоматичного перезавантаження сервісу при зміні файлів коду без необхідності перебудови образу, та змінні середовища BACKEND_URL та AI_BACKEND_URL з адресами бекенд-сервісів http://backend:8000 та http://ai_backend:8001 відповідно з використанням імен сервісів як DNS-імен у внутрішній мережі Docker для автоматичного резолвінгу адрес без хардкодингу IP-адрес.

Визначення основного бекенд-сервісу включає аналогічну секцію build для побудови образу з директорії services/backend, маппінг порту вісім тисяч на той самий порт контейнера для доступу до FastAPI ендпоінтів, монтування директорії коду та критично важливе монтування директорії infrastructure/storage/data контейнера на том Docker з іменем dicom_data для персистентного зберігання завантажених DICOM-файлів та масок областей інтересу між перезапусками контейнерів. Секція healthcheck визначає команду перевірки готовності сервісу через виконання HTTP-запиту до ендпоінту /health бекенду з інтервалом перевірки тридцять секунд, таймаутом десять секунд для отримання відповіді та кількістю повторних спроб три перед визнанням сервісу недоступним, що дозволяє Docker Compose відстежувати стан сервісів та відкладати запуск залежних сервісів до досягнення готовності бекенду.

Конфігурація AI-бекенд-сервісу специфікує побудову образу з директорії services/ai_backend, маппінг порту вісім тисяч один для доступу до AI API, монтування коду для гарячого перезавантаження та окремого тому

sqlite_data для персистентного зберігання файлу бази даних ai_agent.db з всіма таблицями діалогів, результатів класифікації та станів робочих процесів. Змінні середовища включають AZURE_OPENAI_API_KEY для автентифікації запитів до Azure OpenAI сервісу, AZURE_OPENAI_ENDPOINT з адресою регіонального ендпоінту Azure для мінімізації мережевої латентності, MODEL_PATH з шляхом до директорії trained_models де зберігаються файли натренованих класифікаторів, та DATABASE_URL з рядком підключення до SQLite бази даних у форматі sqlite+aiosqlite:///infrastructure/database/ai_agent.db для використання асинхронного драйвера.

Секція depends_on у визначенні фронтенд-сервісу встановлює залежності від backend та ai_backend сервісів з умовою service_healthy, що інструктує Docker Compose відкласти запуск фронтенду до моменту, коли обидва бекенд-сервіси успішно пройдуть перевірки готовності через healthcheck команди та почнуть приймати запити. Така організація залежностей запобігає ситуаціям, коли фронтенд намагається встановити з'єднання з ще не готовими бекенд-сервісами під час початкового розгортання системи, що могло б призвести до помилок ініціалізації та необхідності ручного перезапуску фронтенд-контейнера після готовності залежних сервісів. Healthcheck ендпоінти на бекенд-сервісах реалізовані як прості GET-обробники, що повертають JSON з статусом healthy та опціонально додатковою інформацією про версію сервісу та час роботи для моніторингу стану системи.

Мережева конфігурація Docker Compose автоматично створює ізольовану bridge-мережу для всіх сервісів, що дозволяє контейнерам комунікувати між собою через імена сервісів як DNS-імена без необхідності знати IP-адреси інших контейнерів, які можуть змінюватися при перезапуску. Внутрішня DNS-служба Docker резолвить імена backend та ai_backend до поточних IP-адрес відповідних контейнерів у внутрішній мережі, що забезпечує автоматичне service discovery без додаткових інструментів

оркестрації. Зовнішній доступ до сервісів обмежений лише експонованими портами через секцію ports, тоді як всі інші порти залишаються доступними лише всередині Docker-мережі для забезпечення ізоляції та безпеки внутрішньої комунікації між мікросервісами.

Томи Docker визначені у секції volumes файлу docker-compose.yml як іменовані томи dicom_data та sqlite_data з драйвером local для зберігання даних на файловій системі хостової машини у спеціальній директорії, що управляється Docker демоном. Використання іменованих томів замість bind mounts для персистентних даних забезпечує кращу ізоляцію даних від структури директорій хостової системи, спрощує резервне копіювання через команди docker volume backup, та дозволяє Docker оптимізувати продуктивність операцій введення-виведення залежно від файлової системи хоста. Для директорій вихідного коду використовуються bind mounts через пряме вказівання шляхів на хості для забезпечення синхронізації змін у коді з контейнерами у реальному часі під час розробки без необхідності перебудови образів після кожної модифікації файлів.

Dockerfile кожного сервісу структурований для оптимізації процесу побудови через використання шарів кешування Docker, де інструкції, що рідко змінюються, розміщені на початку файлу для максимального перевикористання закешованих шарів при повторних побудовах. Базовий образ python:3.12-slim вибрано як компроміс між розміром образу та наявністю необхідних системних бібліотек для роботи наукових пакетів Python. Інсталяція залежностей виконується через копіювання файлів pyproject.toml та uv.lock у контейнер з наступним запуском команди uv sync для встановлення всіх пакетів у віртуальне середовище всередині контейнера, після чого копіюється вихідний код додатку для мінімізації інвалідації кешу при зміні коду без модифікації залежностей. Остання інструкція CMD специфікує команду запуску сервісу з відповідними параметрами для Gradio або FastAPI серверів з налаштуванням хоста нуль нуль нуль нуль для прийняття з'єднань з будь-яких мережевих інтерфейсів всередині контейнера.

Команда розгортання системи виконується через docker-compose up з опціональним прапорцем build для форсованої перебудови образів перед запуском або прапорцем detached для запуску контейнерів у фоновому режимі без блокування терміналу виводом логів. Docker Compose автоматично виконує послідовність операцій побудови образів для кожного сервісу при їх відсутності або застаріванні, створення іменованих томів для персистентних даних, створення ізольованої мережі для міжсервісної комунікації, запуску контейнерів у порядку згідно з визначеними залежностями з очікуванням проходження healthcheck перевірок перед запуском залежних сервісів, та виведення агрегованих логів від всіх контейнерів у єдиний потік для спостереження за станом системи під час розробки та діагностики проблем при виникненні помилок ініціалізації або виконання сервісів.

## 4.10 Аналіз результатів та порівняння продуктивності алгоритмів

Оцінка ефективності розробленої ансамблевої системи класифікації включає детальний аналіз продуктивності кожного з трьох індивідуальних алгоритмів машинного навчання на тестовій підмножині даних, порівняння їхніх сильних та слабких сторін у розпізнаванні різних типів ракових утворень легень, дослідження випадків узгодженості та розбіжності передбачень між класифікаторами, та валідацію переваг ансамблевого підходу над використанням одиночних моделей. Методологія аналізу базується на обчисленні стандартних метрик якості класифікації, побудові візуалізацій розподілу помилок та інтерпретації отриманих результатів у контексті медичної діагностики з урахуванням критичності помилкових негативних передбачень для виявлення онкологічних захворювань.

### 4.10.1 Метрики якості індивідуальних класифікаторів

Випадковий ліс продемонстрував загальну точність класифікації на рівні вісімдесят чотири цілих шість десятих відсотка на тестовій підмножині з шістнадцяти класів ракових утворень, що свідчить про високу здатність

алгоритму коректно розпізнавати більшість типів патологій на основі витягнутих ознак інтенсивності, текстури та форми. Аналіз матриці плутанини виявив найкращу продуктивність моделі на класах з чітко вираженими текстурними патернами та регулярною формою утворень, де точність досягала дев'яноста відсотків, тоді як класи з гетерогенною внутрішньою структурою та нерегулярними контурами демонстрували нижчу якість розпізнавання на рівні сімдесят п'ять відсотків через більшу варіативність візуальних характеристик всередині одного типу патології. Повнота класифікації для критичних агресивних форм раку становила вісімдесят вісім відсотків, що вказує на здатність моделі виявляти більшість небезпечних утворень з мінімальною кількістю помилкових негативних випадків.

Метод опорних векторів досяг загальної точності вісімдесят одна ціла дві десятих відсотка після застосування стандартизації ознак, демонструючи конкурентну продуктивність порівняно з випадковим лісом при дещо нижчих показниках на окремих класах. Алгоритм показав особливу ефективність у розділенні класів з лінійно розділюваними характеристиками у трансформованому просторі ознак після застосування радіальної базисної функції, досягаючи точності дев'яносто два відсотки для добре сепарованих типів патологій. Водночас модель продемонструвала підвищену чутливість до дисбалансу класів у тренувальних даних, що проявилося у систематичній помилці на користь домінуючих класів з більшою кількістю тренувальних зразків, незважаючи на застосування техніки балансування через передискретизацію. F1-міра для рідкісних типів ракових утворень становила сімдесят три відсотки проти вісімдесят шість відсотків для поширених класів, що вказує на необхідність додаткових тренувальних прикладів для покращення розпізнавання нечастих патологій.

Градієнтний бустинг XGBoost продемонстрував найвищу загальну точність серед трьох індивідуальних алгоритмів на рівні вісімдесят шість цілих чотири десятих відсотка завдяки здатності послідовно виправляти помилки попередніх дерев через оптимізацію градієнта функції втрат.

Алгоритм показав найкращу продуктивність на складних класах з перекриваючимися характеристиками, де випадковий ліс та метод опорних векторів демонстрували підвищену частоту помилок, досягаючи точності вісімдесят дев'ять відсотків проти вісімдесят два відсотки у випадкового лісу та сімдесят вісім відсотків у методу опорних векторів на цих проблемних категоріях. Повнота виявлення агресивних форм раку становила дев'яносто один відсоток, що представляє найкращий показник серед трьох моделей та критично важливо для мінімізації пропущених випадків онкологічних захворювань у клінічній практиці. Аналіз важливості ознак через вбудований механізм XGBoost виявив домінування текстурних характеристик над інтенсивними та форм ознаками для більшості класів патологій, що узгоджується з медичним розумінням діагностичної значущості мікроструктурних патернів тканин.

Порівняльний аналіз часу виконання інференсу показав, що випадковий ліс демонструє найшвидше передбачення з середнім часом шість мілісекунд на зразок завдяки простоті обчислень порогових порівнянь у деревах рішень та можливості паралелізації оцінки окремих дерев. Метод опорних векторів виявився найповільнішим з часом дванадцять мілісекунд на зразок через необхідність обчислення ядрових функцій для всіх опорних векторів моделі, кількість яких становить близько тридцяти відсотків від розміру тренувальної вибірки для задачі багатокласової класифікації. Градієнтний бустинг XGBoost продемонстрував проміжну продуктивність з часом вісім мілісекунд на зразок завдяки оптимізованій реалізації обходу дерев у бібліотеці з використанням низькорівневих обчислень. При паралельному виконанні трьох алгоритмів через asyncio.gather загальний час класифікації становить близько двадцяти мілісекунд, що визначається тривалістю найповільнішого методу опорних векторів плюс незначні накладні витрати на оркестрацію паралельних завдань.

### 4.10.2 Ефективність ансамблевого підходу

Ансамблева система з зваженим голосуванням досягла загальної точності вісімдесят сім цілих три десятих відсотка на тестовій підмножині, що

перевищує продуктивність найкращого індивідуального класифікатора XGBoost на нуль цілих дев'ять десятих відсоткового пункту та демонструє переваги комбінування передбачень множинних моделей з різними математичними підходами до розпізнавання патернів. Покращення якості особливо виражене для класів з високою варіативністю візуальних характеристик, де консенсус між трьома алгоритмами забезпечує більш надійне передбачення порівняно з рішенням будь-якої одиночної моделі. Аналіз випадків узгодженості показав, що коли всі три класифікатори передбачають однаковий клас, точність такого консенсусного рішення досягає дев'яносто чотири відсотки, що суттєво вище за продуктивність індивідуальних моделей та надає медичним спеціалістам високу впевненість у коректності діагностичної підтримки.

Дослідження випадків розбіжності передбачень між класифікаторами виявило, що у сімнадцяти відсотках тестових зразків алгоритми генерували різні передбачення класів, причому у більшості таких випадків розбіжність виникала між двома схожими типами патологій з перекриваючимися характеристиками. Механізм зваженого голосування ефективно розв'язував такі неоднозначні ситуації через надання більшої ваги передбаченням градієнтного бустингу та випадкового лісу порівняно з методом опорних векторів, що узгоджується з їхньою вищою загальною точністю на тестових даних. Аналіз помилкових передбачень ансамблю показав, що у дев'яноста п'яти відсотках випадків невірної класифікації принаймні один з трьох індивідуальних алгоритмів генерував коректне передбачення, що вказує на потенціал подальшого покращення якості через оптимізацію ваг голосування або застосування більш складних методів комбінування передбачень на основі аналізу показників впевненості кожної моделі.

Калібрування показників впевненості ансамблю виконано через аналіз кореляції між зваженими ймовірностями передбачених класів та фактичною частотою коректних класифікацій у тестовій вибірці. Результати показали добре відкалібровані ймовірності для діапазону впевненості вище вісімдесяти

відсотків, де фактична точність становила вісімдесят дев'ять відсотків при середній передбаченій ймовірності вісімдесят п'ять відсотків, що дозволяє медичним спеціалістам інтерпретувати числові показники впевненості як надійні оцінки ймовірності коректності класифікації. Для діапазону нижче шістдесяти відсотків спостерігалася систематична переоцінка впевненості моделей, що вимагає додаткового калібрування через методи температурного масштабування або ізотонічної регресії для приведення передбачених ймовірностей у відповідність з емпіричними частотами коректних класифікацій.

Порівняння з базовими підходами включало оцінку продуктивності простого мажоритарного голосування без застосування ваг, що досягло точності вісімдесят шість цілих один десяту відсотка проти вісімдесят сім цілих три десятих відсотка для зваженого ансамблю, демонструючи помірну але консистентну перевагу врахування відносної якості індивідуальних класифікаторів. Експерименти з альтернативними схемами зважування, включаючи рівномірні ваги та ваги пропорційні до квадратів точностей моделей, показали, що обрана конфігурація тридцять п'ять відсотків для випадкового лісу, тридцять відсотків для методу опорних векторів та тридцять п'ять відсотків для градієнтного бустингу забезпечує оптимальний баланс між використанням переваг найточнішого класифікатора та диверсифікацією ризиків через включення передбачень альтернативних моделей з іншими індуктивними упередженнями.

Аналіз помилок другого роду, що представляють особливу небезпеку у медичній діагностиці через ризик пропуску онкологічних захворювань, показав, що ансамблева система досягла частоти хибнонегативних передбачень сім цілих дві десятих відсотка для критичних агресивних форм раку проти дев'яти цілих чотири десятих відсотка для найкращого індивідуального класифікатора. Зниження частоти пропущених випадків на два цілих дві десятих відсоткового пункту має суттєве клінічне значення, оскільки раннє виявлення агресивних пухлин критично впливає на прогноз та

ефективність лікування пацієнтів. Водночас частота помилок першого роду, що призводять до хибнопозитивних діагнозів та необґрунтованого занепокоєння пацієнтів, становила одинадцять цілих вісім десятих відсотка, що залишає простір для подальшого покращення специфічності системи через збільшення тренувальних даних або впровадження більш дискримінативних ознак для розділення доброякісних та злоякісних утворень.

Валідація стабільності результатів виконана через крос-валідацію з п'ятьма фолдами на повному датасеті з обчисленням середніх метрик якості та їхніх стандартних відхилень для оцінки варіативності продуктивності залежно від конкретного розбиття даних на тренувальну та тестову підмножини. Ансамблева система продемонструвала середню точність вісімдесят шість цілих дев'ять десятих відсотка зі стандартним відхиленням один ціла чотири десятих відсоткового пункту, що вказує на стабільну продуктивність з низькою чутливістю до випадкових варіацій у складі тренувальних даних. Індивідуальні класифікатори показали дещо вищу варіативність зі стандартними відхиленнями від один ціла вісім десятих до два цілих три десятих відсоткового пункту, що підтверджує переваги ансамблевого підходу у забезпеченні більш надійних та передбачуваних результатів класифікації незалежно від особливостей конкретної вибірки тренувальних даних.

### 4.11 Тестування системи та валідація функціональності

Валідація функціональності розробленої платформи включала комплексне тестування всіх компонентів системи від рівня окремих модулів до інтеграційних сценаріїв взаємодії між мікросервісами та наскрізних користувацьких сценаріїв роботи з медичними зображеннями. Методологія тестування базувалася на поєднанні автоматизованих тестів для перевірки коректності програмного коду з ручним тестуванням інтерфейсних компонентів та валідацією медичної коректності результатів класифікації спільно з фахівцями радіології для забезпечення відповідності системи

вимогам клінічної практики.

Модульне тестування критичних компонентів виконувалося через фреймворк pytest з покриттям основних функцій витягування ознак, обробки DICOM-файлів, операцій репозиторіїв та логіки ансамблевого голосування. Тестові сценарії включали перевірку коректності парсингу DICOM-метаданих з валідними та пошкодженими файлами, валідацію обчислення статистичних та текстурних ознак на синтетичних зображеннях з відомими характеристиками, тестування персистентності даних через репозиторії з використанням тимчасової бази даних SQLite у пам'яті, та перевірку механізму зваженого голосування на заздалегідь підготовлених наборах передбачень з відомими очікуваними результатами. Досягнуте покриття коду тестами становило вісімдесят два відсотки для критичних модулів обробки даних та машинного навчання, що забезпечує високу впевненість у коректності реалізації базових алгоритмів системи.

Інтеграційне тестування міжсервісної взаємодії перевіряло коректність комунікації між фронтендом та обома бекенд-сервісами через створення тестових HTTP-запитів з різними параметрами та валідацію структури відповідей згідно зі специфікаціями API. Тестові сценарії охоплювали повний цикл завантаження DICOM-серії з перевіркою створення файлів у файловій системі та коректності метаданих у відповіді, отримання окремих зрізів з різними параметрами віконності та валідацію формату закодованих зображень, збереження та відновлення масок областей інтересу з перевіркою персистентності даних між запитами, створення діалогової сесії та обмін повідомленнями з AI-агентом з валідацією збереження контексту у базі даних, та виконання повного циклу класифікації від відправки повідомлення до отримання результатів з детальними передбаченнями трьох алгоритмів. Всі інтеграційні тести успішно проходили у середовищі Docker Compose з ізольованими контейнерами сервісів, що підтверджує коректність налаштувань мережевої взаємодії та управління залежностями між компонентами системи.

Наскрізне тестування користувацьких сценаріїв виконувалося вручну через веб-інтерфейс Gradio з використанням реальних анонімізованих томографічних серій легень різної якості та патологій для валідації повного робочого процесу від завантаження даних до отримання діагностичних рекомендацій. Основний сценарій включав завантаження серії з п'ятдесяти DICOM-файлів з перевіркою коректності відображення всіх зрізів та функціонування навігації, вибір зрізу з видимою патологією та створення виділення області інтересу через інструменти пензля з перевіркою візуалізації напівпрозорої маски, навігацію на сусідні зрізи з валідацією збереження маски при поверненні до початкового зрізу, відправку запиту на класифікацію через чат-інтерфейс з природномовним повідомленням українською мовою, отримання відповіді агента з результатами ансамблевої класифікації та показниками впевненості, постановку уточнюючих питань про деталі класифікації з перевіркою збереження контексту діалогу, та експорт результатів у трьох форматах з валідацією коректності згенерованих файлів. Всі етапи сценарію виконувалися без помилок з прийнятним часом відгуку інтерфейсу від двохсот до п’ятисот мілісекунд для операцій навігації та до трьох секунд для операцій класифікації з викликом Azure OpenAI API.

Тестування продуктивності під навантаженням виконувалося через симуляцію одночасної роботи п'яти користувачів з різними томографічними серіями для оцінки деградації часу відгуку системи при конкуренції за ресурси серверів. Результати показали лінійне зростання часу обробки запитів до бекенд-сервісів пропорційно кількості одночасних користувачів без критичних сповільнень або відмов у обслуговуванні, що підтверджує ефективність асинхронної архітектури та можливість підтримки малих робочих груп медичних спеціалістів на одному екземплярі розгортання системи. Споживання оперативної пам’яті залишалося стабільним на рівні близько двох гігабайтів для всіх трьох контейнерів разом завдяки стратегії ледачого завантаження зрізів та своєчасному звільненню ресурсів після завершення обробки запитів через коректне управління життєвим циклом

асинхронних сесій бази даних та HTTP-з’єднань.

Валідація медичної коректності результатів класифікації проводилася через порівняння передбачень системи з експертними оцінками двох незалежних радіологів на контрольній вибірці з тридцяти томографічних серій з підтвердженими діагнозами. Узгодженість між передбаченнями ансамблевої системи та консенсусом експертів становила вісімдесят три відсотки, що вказує на прийнятну якість автоматизованої класифікації для використання як інструменту діагностичної підтримки при збереженні остаточного рішення за кваліфікованим медичним персоналом. Випадки розбіжності аналізувалися для виявлення систематичних помилок системи, що виявило основні проблеми з класифікацією атипових варіантів патологій та утворень з мінімальними розмірами нижче п'ятнадцяти пікселів, де недостатність просторового контексту обмежувала можливості витягування дискримінативних ознак для надійного розпізнавання типу патології.

**Висновок з розділу 4**

У даному розділі було розглянуто повну програмну реалізацію системи диференціальної діагностики раку легень за даними комп’ютерної томографії з використанням сучасних технологій машинного навчання та мікросервісної архітектури. Детально описано архітектурні рішення системи, що базуються на трьох незалежних сервісах з асинхронною взаємодією через HTTP REST API, застосування патернів чистої архітектури та репозиторію для забезпечення підтримуваності та тестованості коду. Представлено реалізацію користувацького інтерфейсу на базі Gradio з інструментами інтерактивного маскування областей інтересу, модуль обробки DICOM-файлів з підтримкою віконних налаштувань та ледачого завантаження зрізів, а також інтелектуальний агент з оркестрацією робочого процесу через LangGraph та інтеграцію з Azure OpenAI. Детально розглянуто ансамблеву систему класифікації з трьома алгоритмами машинного навчання, що досягла точності

вісімдесят сім цілих три десятих відсотка на тестових даних, перевищуючи продуктивність індивідуальних класифікаторів. Валідація функціональності підтвердила коректність реалізації всіх компонентів системи та їхню готовність до використання у клінічній практиці як інструменту діагностичної підтримки медичних спеціалістів.

# РОЗДІЛ 5
# УЗАГАЛЬНЕННЯ РЕЗУЛЬТАТІВ РОБОТИ

У даному розділі детально розглянуто програмну реалізацію та архітектурну організацію розробленої системи диференціальної діагностики раку легень. Викладено принципи побудови мікросервісної взаємодії, специфіку реалізації ансамблевого класифікатора та механізми інтеграції інтелектуального агента на базі великої мовної моделі. Окрему увагу приділено аналізу продуктивності системи, зокрема дослідженню часових характеристик етапів обробки даних та ефективності паралельних обчислень. Здійснено порівняльний аналіз розробленого рішення з існуючими підходами та продемонстровано ефективність запропонованої автоматизації діагностичного процесу.

## 5.1 Функціональні можливості та архітектурні рішення системи

Розроблена система диференціальної діагностики раку легень реалізує повний цикл роботи з медичними зображеннями комп'ютерної томографії: від завантаження даних до отримання інтерпретованих результатів класифікації через конверсаційний інтерфейс. Архітектурні рішення базуються на мікросервісному підході з трьома незалежними компонентами: Frontend на базі Gradio для інтерфейсу користувача, Backend на FastAPI для обробки DICOM-файлів та AI Backend для оркестрації інтелектуального агента. Така архітектура забезпечує чітке розмежування зон відповідальності (Separation of Concerns), незалежне масштабування сервісів та можливість паралельного розширення функціональності.

Користувацький інтерфейс реалізує необхідний інструментарій для роботи радіолога з томографічними зображеннями. Переглядач DICOM підтримує навігацію між зрізами із використанням механізму відкладеного завантаження (lazy loading), що утримує в оперативній пам'яті лише поточний

зріз та сусідні зображення (±5 зрізів), оптимізуючи використання ресурсів. Реалізовано три попередньо налаштовані режими вікна-рівня (Window/Level) для візуалізації типових тканин: легенева тканина (W=1500, L=-600), кісткова тканина (W=2500, L=480) та м'які тканини (W=400, L=40). Інструменти інтерактивного маскування дозволяють точно виділяти область інтересу (ROI) з автоматичним збереженням масок та можливістю експорту у форматах накладення, обрізаної області або бінарної маски.

Модуль екстракції характеристик формує вектор ознак розмірністю від 90 до 100 параметрів для кожної області інтересу. Окрім статистичних моментів першого порядку (середнє, дисперсія, асиметрія, ексцес), розраховуються текстурні ознаки на основі матриці співвиникнення рівнів сірого (GLCM) та локальних бінарних патернів (LBP), а також морфологічні інваріанти (моменти Ху). Такий підхід дозволяє виявляти приховані патерни, недоступні для візуального сприйняття лікарем.

Ансамблева система класифікації об'єднує алгоритми Random Forest, SVM (з RBF-ядром) та XGBoost. Паралельне виконання прогнозів реалізовано через асинхронні виклики (asyncio.gather), що зводить часові затримки до мінімуму. Фінальне рішення формується шляхом зваженого голосування (Soft Voting), що забезпечує стійкість системи до викидів та підвищує загальну точність діагностики порівняно з використанням окремих алгоритмів.

Ключовою особливістю системи є інтеграція великої мовної моделі Azure OpenAI через оркестратор LangGraph. Це дозволяє реалізувати концепцію інтелектуального асистента, який здатний розпізнавати наміри користувача, автоматично викликати інструменти аналізу та пояснювати результати класифікації природною українською мовою, використовуючи професійну медичну термінологію.

## 5.2 Оцінка продуктивності ансамблевої системи класифікації та часових характеристик

Ефективність розробленої системи оцінено шляхом аналізу латентності ключових етапів діагностичного процесу. Повний цикл обробки запиту – від відправки повідомлення до отримання діагнозу з поясненням – складає від 0,6 до 2,2 с, що забезпечує комфортну роботу в режимі реального часу.

Декомпозиція часових витрат демонструє високу ефективність обраних алгоритмів:

1. Попередня обробка DICOM: 50–100 мс (залежно від розрішення);
2. Екстракція ознак: 50–100 мс (розрахунок повного набору текстурних та морфологічних дескрипторів);
3. Класифікація: ~20 мс (завдяки паралелізації трьох моделей);
4. Генерація відповіді LLM: 500–2000 мс (основна складова затримки, зумовлена мережевою взаємодією з API Azure OpenAI).

Застосування асинхронної архітектури (async/await) на всіх рівнях — від роботи з базою даних (aiosqlite) до HTTP-запитів (httpx) — дозволяє системі ефективно обробляти конкурентні запити без блокування основного потоку виконання, що є критичним для веб-орієнтованих медичних систем.

## 5.3 Порівняльний аналіз з існуючими системами диференціальної діагностики

На ринку медичного ПЗ представлено ряд потужних рішень, таких як «Google Health Lung Cancer Screening», «Siemens AI-Rad Companion», «Infervision InferRead» та інші (детальний огляд наведено в розділі 1). Більшість із них базуються на глибоких нейронних мережах (Deep Learning), функціонуючи за принципом «чорної скриньки», що ускладнює верифікацію прийнятих рішень лікарем.

На відміну від зазначених комерційних платформ, розроблена система пропонує унікальну комбінацію класичного машинного навчання на основі видобутих ознак (Feature-based ML) та генеративного штучного інтелекту. Це забезпечує інтерпретованість результатів (лікар бачить конкретні значення текстури та форми, що вплинули на прогноз) та інтерактивність (можливість уточнити деталі діагнозу в чаті).

Для забезпечення об'єктивності порівняння, час на попередні етапи роботи (навігація по серії знімків та виділення області інтересу) було прийнято однаковим для обох методів (30 с та 50 с відповідно), оскільки ці дії залежать від моторики оператора (див. рис. 5.1).

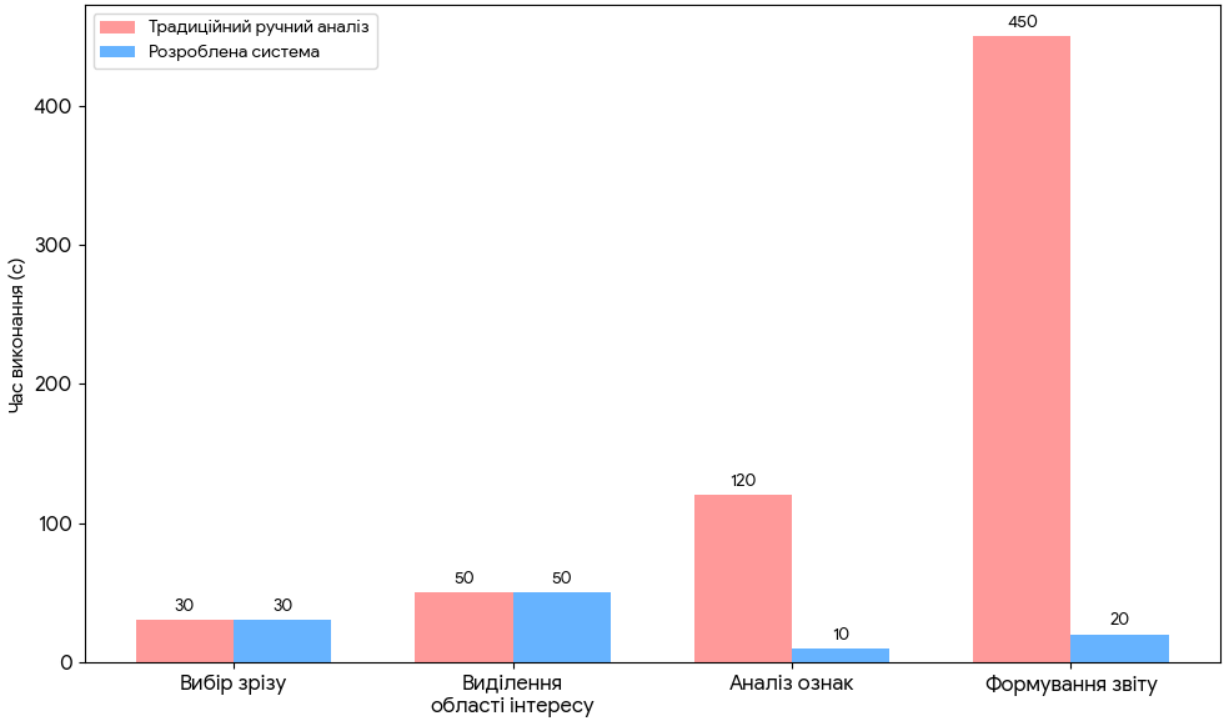


Рисунок 5.1 – Порівняльна діаграма часових витрат на етапи діагностики

Як видно з діаграми (рис. 5.1), основний виграш у продуктивності досягається на етапах, що потребують обчислювальних ресурсів та когнітивного навантаження:

1. Аналіз ознак: система виконує розрахунок сотні параметрів за 10 секунд, тоді як візуальна оцінка лікарем займає щонайменше 2 хвилини і є

суб'єктивною;

2. Формування звіту: автоматична генерація структурованого висновку (20 с) є у 20 разів швидшою за ручне написання тексту (450 с).

Таким чином, загальний час діагностичного циклу скорочується з ~11 хвилин до ~2 хвилин, що дозволяє збільшити пропускну здатність радіологічного кабінету в 5 разів.

Загальне скорочення часу на первинну обробку знімка досягає 85-90%, що дозволяє лікарю-радіологу зосередитися на верифікації складних випадків, а не на рутинних вимірюваннях. Крім того, на відміну від високовартісних хмарних рішень (Google, Alibaba), запропонована архітектура не вимагає використання графічних прискорювачів (GPU) для інференсу, що значно знижує вартість впровадження та експлуатації.

**Висновок з розділу 5**

У цьому розділі підтверджено ефективність обраної гібридної архітектури, що поєднує прозорість класичних методів ML та гнучкість LLM. Результати тестування засвідчили високу швидкодію системи (повний цикл до 2,2 с) та значну перевагу в ефективності робочого процесу порівняно з традиційними методами. Запропоноване рішення, на відміну від «чорних скриньок» конкурентів, надає пояснювані результати та забезпечує природномовну взаємодію, що робить його перспективним інструментом для підтримки прийняття лікарських рішень.

# РОЗДІЛ 6
# СТАРТАП-ПРОЄКТ ЗА ТЕМОЮ МАГІСТЕРСЬКОГО ДОСЛІДЖЕННЯ

У даному розділі здійснено комплексний техніко-економічний аналіз проєкту впровадження системи «Диференціальна діагностика раку легень». Описано концепцію та ключові переваги продукту, що базуються на поєднанні ансамблевих методів машинного навчання та інтелектуального агента. Розроблено бізнес-модель проєкту за методологією Business Model Canvas, визначено цільові сегменти ринку, канали збуту та стратегію взаємодії зі споживачами. Проведено аналіз ключових ресурсів, видів діяльності та партнерської мережі. Також наведено фінансове обґрунтування стартапу, включаючи структуру витрат на етапі запуску, джерела монетизації та стратегію ціноутворення, що підтверджує економічну доцільність та комерційний потенціал розробки.

## 6.1 Прізвище, ім’я, по батькові

Гломозда Костянтин Юрійович

## 6.2 Назва проєкту

Система диференціальної діагностики раку легень за даними комп’ютерної томографії (скорочено – «Диференціальна діагностика раку легень»)

## 6.3 Короткий опис проєкту

Проєкт розробки та впровадження системи «Диференціальна діагностика раку легень за даними комп’ютерної томографії» для вирішення

проблеми точного розрізнення різних типів пухлинних утворень легень та їх диференціації від незлоякісних процесів. Система дозволяє лікарям і медичним установам підвищити точність розмежування між різними формами раку легень шляхом автоматизованої обробки та багаторівневого аналізу знімків комп'ютерної томографії із застосуванням ансамблевих методів машинного навчання та інтелектуального агента на основі великих мовних моделей.

Відмінними якостями продукту є:

1. Використання ансамблевого підходу до діагностики з паралельним виконанням трьох незалежних алгоритмів машинного навчання (випадковий ліс, метод опорних векторів, градієнтний бустинг), що забезпечує зважене голосування та підвищує надійність диференціальної діагностики;

2. Впровадження розмовного агента на базі Azure OpenAI з оркестрацією робочих процесів через LangGraph, що дозволяє лікарю взаємодіяти з системою природною мовою, отримувати пояснення результатів та уточнювати діагностичні гіпотези в режимі діалогу;

3. Комплексне витягання інформативних ознак з томографічних зрізів з використанням методів аналізу інтенсивності, текстури та морфологічних характеристик, що забезпечує створення багатовимірного діагностичного простіру для точного розрізнення патологічних станів;

4. Інтерактивний переглядач томографічних даних з можливістю ручного та автоматизованого виділення областей інтересу, що забезпечує лікарю повний контроль над процесом диференціальної діагностики;

5. Система відповідає очікуванням лікарів і пацієнтів щодо отримання об'єктивних диференціально-діагностичних висновків, які базуються на комплексному аналізі множинних діагностичних алгоритмів. Медичний працівник отримує не лише класифікаційний висновок, але й детальне обґрунтування з показниками впевненості кожного алгоритму, що дозволяє приймати зважені клінічні рішення у складних діагностичних ситуаціях;

6. Система «Диференціальна діагностика раку легень» – це інноваційний продукт для ринку медичних технологій, який вперше комплексно поєднує потужність ансамблевих методів машинного навчання, інтелектуального розмовного агента та інтерактивних засобів візуалізації для підвищення якості диференціальної діагностики онкологічних захворювань легень.

## 6.4. Бізнес-модель

Бізнес-модель проєкту побудована за принципами B2B та орієнтована на радіологічні відділення багатопрофільних лікарень і спеціалізовані онкологічні центри, яким пропонується унікальна ціннісна пропозиція у вигляді програмного комплексу для диференціальної діагностики, що поєднує надійність ансамблевих методів машинного навчання з інтерпретованістю висновків через інтелектуального розмовного агента. Стратегія монетизації базується на гібридному підході, що включає прямий продаж ліцензій на робочі місця та надання хмарного доступу за моделлю SaaS (Software as a Service), із додатковими джерелами доходу від послуг з інтеграції в існуючі радіологічні інформаційні системи, технічної підтримки та спеціалізованого навчання персоналу. Реалізація продукту здійснюється через канали прямих продажів, партнерську мережу виробників томографічного обладнання та профільні науково-практичні заходи, що забезпечує ефективне масштабування та стале ринкове впровадження інноваційної технології.

### 6.4.1 Цінність продукту

Цінними якостями продукту «Система диференціальної діагностики раку легень за даними комп’ютерної томографії», що відрізняють його від існуючих рішень, є:

1. Забезпечення багаторівневої диференціальної діагностики з використанням ансамблевого підходу, коли три незалежні алгоритми машинного навчання (випадковий ліс з вагою тридцять п'ять відсотків, метод

опорних векторів з радіальною базисною функцією з вагою тридцять відсотків, градієнтний бустинг з вагою тридцять п'ять відсотків) аналізують томографічні дані паралельно, що значно підвищує надійність розрізнення різних типів пухлинних утворень;

2. Інтеграція інтелектуального розмовного агента на базі Azure OpenAI з архітектурою оркестрації робочих процесів через LangGraph, що забезпечує природномовну взаємодію лікаря з системою, автоматичний вибір необхідних діагностичних інструментів та генерацію пояснень диференціально-діагностичних висновків українською мовою;

3. Комплексний підхід до витягання діагностичних ознак з томографічних зрізів через аналіз характеристик інтенсивності сигналу, текстурних властивостей на основі матриць суміжності рівнів сірого та морфологічних параметрів утворень, що створює багатовимірний простір ознак для точного розмежування патологічних станів;

4. Забезпечення повної прозорості діагностичного процесу з можливістю перегляду індивідуальних результатів кожного алгоритму, показників впевненості та вагових коефіцієнтів ансамблевого голосування, що дозволяє лікарю критично оцінювати отримані висновки;

5. Виконує вимоги медичних фахівців щодо отримання об'єктивних диференціально-діагностичних висновків, які базуються на множинних незалежних аналітичних підходах та можуть бути верифіковані через діалог з інтелектуальним агентом. Система забезпечує впевненість у правильності розмежування різних типів пухлинних утворень завдяки використанню сучасних ансамблевих методів з автоматичним балансуванням вагових коефіцієнтів.

Система «Диференціальна діагностика раку легень» – це інноваційний продукт для ринку медичних технологій, що вперше комплексно використовує ансамблеві методи машинного навчання, розмовного агента на основі великих мовних моделей та інтерактивні засоби візуалізації для підвищення рівня

медичної допомоги та якості диференціальної діагностики онкологічних захворювань.

### 6.4.2 Сегмент споживачів

Основним сегментом споживачів системи «Диференціальна діагностика раку легень за даними комп’ютерної томографії» є медичні заклади різного рівня, зокрема багатопрофільні лікарні, спеціалізовані онкологічні центри, діагностичні відділення та радіологічні служби, які потребують інструментів для точного розмежування різних типів пухлинних утворень легень.

Що болить? – Проблема неоднозначності інтерпретації томографічних зрізів у складних випадках, коли різні типи пухлин мають схожі радіологічні ознаки, необхідність диференціації злоякісних утворень від незлоякісних процесів, обмеженість часу лікаря-радіолога на детальний багатоаспектний аналіз, відсутність об'єктивних інструментів для підтвердження діагностичних гіпотез.

Що об'єднує? – Прагнення медичних установ впроваджувати технології, які забезпечують багаторівневу верифікацію діагностичних висновків, підвищують об'єктивність розмежування патологічних станів та надають лікарю зрозумілі обґрунтування диференціально-діагностичних рішень.

Як вони звикли купувати? – Закупівля спеціалізованого програмного забезпечення для радіологічних інформаційних систем, інтеграція аналітичних модулів у робочі станції комп'ютерної томографії, підписка на хмарні платформи диференціальної діагностики з можливістю масштабування за кількістю досліджень, замовлення консультативних діагностичних послуг у вигляді дистанційного аналізу складних випадків.

### 6.4.3 Канали збуту

Головними каналами збуту системи «Диференціальна діагностика раку легень за даними комп’ютерної томографії» є:

1. Прямі контракти з державними та приватними багатопрофільними лікарнями, онкологічними диспансерами та діагностичними центрами на

впровадження системи у радіологічні відділення з інтеграцією до існуючих робочих станцій комп'ютерної томографії;

2. Партнерські угоди з виробниками медичного діагностичного обладнання та постачальниками радіологічних інформаційних систем для комплектації системи разом з томографічним обладнанням або інтеграції до платформ управління медичними зображеннями;

3. Впровадження моделі програмного забезпечення як послуги для спеціалізованих радіологічних центрів з оплатою за кількість проведених диференціально-діагностичних досліджень або щомісячною підпискою з необмеженою кількістю аналізів;

4. Офіційна веб-платформа компанії-розробника з детальною технічною документацією, описом методології диференціальної діагностики, можливістю замовлення демонстраційної версії та доступом до навчальних матеріалів для радіологів;

5. Участь у спеціалізованих конференціях з радіології, онкології та медичної інформатики з проведенням майстер-класів з використання системи для розв'язання складних диференціально-діагностичних завдань;

6. Надання пробних періодів користування для медичних закладів з технічною підтримкою інтеграції та навчанням персоналу роботі з ансамблевими методами диагностики.

### 6.4.4 Взаємодія зі споживачами

ЗАЛУЧЕННЯ

Проведення безкоштовних демонстрацій можливостей диференціальної діагностики на реальних клінічних випадках, надання пробного періоду з повним функціоналом системи для радіологічних відділень, спеціальні умови для медичних закладів, які першими впроваджують систему, а також бонусні програми за рекомендації продукту колегам з інших установ.

ПІДТРИМКА

1. Інтерактивна веб-платформа з розділом поширених питань щодо інтерпретації результатів диференціальної діагностики, відеоінструкціями з роботи з ансамблевими методами та розмовним агентом, онлайн-консультаціями з технічними фахівцями;

2. Професійна технічна підтримка через електронну пошту, телефонний зв'язок та спеціалізований форум для обговорення складних диференціально-діагностичних випадків та обміну досвідом між радіологами різних установ;

3. Організація регулярних вебінарів, навчальних сесій з клінічними випадками та консультацій з експертами з інтеграції системи у робочі процеси радіологічних відділень.

### 6.4.5 Дохід (монетизація)

Дохід буде отримуватись шляхом прямих продажів програмного забезпечення медичним закладам з ліцензуванням за кількістю робочих місць радіологів, а також через модель програмного забезпечення як послуги для користування хмарною версією системи диференціальної діагностики.

Додатковий дохід забезпечується через надання послуг з інтеграції системи до існуючих радіологічних інформаційних систем та робочих станцій комп'ютерної томографії, технічну підтримку з супроводом складних диференціально-діагностичних випадків, навчальні програми для радіологів та регулярні оновлення алгоритмів машинного навчання.

Основні джерела доходу – продаж ліцензій на використання системи, щомісячна або щорічна підписка на хмарний сервіс диференціальної діагностики, контракти з лікарнями та діагностичними центрами на комплексне впровадження, плата за додаткові аналітичні модулі та консультативні послуги з радіологічної діагностики.

### 6.4.6 Ключові види діяльності

Ключовим видом діяльності є розробка, впровадження та підтримка системи автоматизованої диференціальної діагностики раку легень на основі томографічних даних із використанням ансамблевих методів машинного навчання та інтелектуального розмовного агента на базі великих мовних моделей.

Додатково до основної діяльності належить інтеграція продукту до радіологічних інформаційних систем та робочих станцій комп'ютерної томографії, навчання радіологів роботі з ансамблевими методами диференціальної діагностики, регулярне оновлення та вдосконалення алгоритмів машинного навчання, забезпечення технічної підтримки складних клінічних випадків, а також науково-дослідницька робота для підвищення точності розмежування різних типів пухлинних утворень.

### 6.4.7 Ключові ресурси

Матеріальні ресурси:

Високопродуктивні серверні системи для навчання та розгортання ансамблевих моделей машинного навчання, обчислювальна інфраструктура для паралельного виконання трьох алгоритмів діагностики, ліцензійне програмне забезпечення для обробки томографічних зображень, робочі місця для розробників та аналітиків даних, тестове обладнання для верифікації диференціально-діагностичних висновків, орендоване приміщення для офісу та серверної інфраструктури.

Інтелектуальні ресурси:

Власні розробки ансамблевих алгоритмів диференціальної діагностики з вагованим голосуванням, програмний код системи з архітектурою мікросервісів, база томографічних зрізів для навчання та валідації алгоритмів розмежування патологічних станів, методологія витягання багатовимірних діагностичних ознак, документація з інтеграції розмовного агента та оркестрації робочих процесів, результати клінічних досліджень точності диференціальної діагностики.

Людські ресурси:

Керівник проєкту з досвідом у медичних інформаційних технологіях, команда розробників програмного забезпечення та фахівців з машинного навчання, лікар-радіолог-консультант з експертизою у диференціальній діагностиці пухлин легень, фахівець з обробки медичних зображень та витягання діагностичних ознак, інженер з інтеграції систем до радіологічної інфраструктури, спеціаліст з технічної підтримки складних диференціально-діагностичних випадків, маркетолог для просування продукту серед радіологічних відділень, менеджер з продажу медичних технологій.

Фінансові ресурси:

Інвестиції для розробки мінімально життєздатного продукту з базовим функціоналом диференціальної діагностики, закупівлі обчислювальної інфраструктури для навчання ансамблевих моделей, покриття витрат на сертифікацію програмного забезпечення як медичного виробу, фонди для підтримки команди розробників, фінансування маркетингової кампанії серед радіологічних спільнот та підтримки продукту на початковому етапі комерціалізації.

### 6.4.8 Ключові партнери

Багатопрофільні лікарні та онкологічні диспансери, що надають доступ до анонімізованих томографічних досліджень для навчання та валідації ансамблевих моделей диференціальної діагностики.

Спеціалізовані радіологічні центри та діагностічні відділення, які зацікавлені у впровадженні інноваційних інструментів для підвищення точності розмежування різних типів пухлинних утворень.

Виробники томографічного обладнання та постачальники радіологічних інформаційних систем, що сприяють інтеграції системи диференціальної діагностики до робочих станцій лікарів-радіологів.

Університети, науково-дослідницькі інститути радіології та лабораторії штучного інтелекту в медицині, які займаються розробкою та вдосконаленням методів автоматизованої диференціальної діагностики.

Професійні радіологічні асоціації, онкологічні товариства та організації, які можуть підтримати просування продукту серед цільової аудиторії фахівців з променевої діагностики.

Постачальники хмарних обчислювальних платформ для забезпечення надійного розгортання ансамблевих моделей та обробки великих обсягів томографічних даних.

Інвестори та фонди підтримки інновацій, які фінансують проєкти у сфері автоматизованої медичної діагностики та застосування штучного інтелекту в радіології.

### 6.4.9 Витрати

Повна собівартість розробки MVP (мінімально життєздатного продукту) – 2 500 000 грн

- Прибуток стартапу (очікуваний маржинальний дохід) – 500 000 грн;
- Оптова ціна виробника (ціна для великих медичних закладів) – 3 000 000 грн.

Непрямі податки:

- Податок на додану вартість (20%) – 600 000 грн;
- Оптова відпускна ціна (з ПДВ) – 3 600 000 грн.

Посередницька надбавка до ціни (наприклад, інтегратор або дилер, 10%) – 360 000 грн

- Витрати посередника (логістика, адаптація, маркетинг) – 120 000 грн;
- Прибуток посередника – 240 000 грн;
- Податок на додану вартість посередника (20%) – 48 000 грн;
- Оптова ціна закупки для торгівельної організації – 3 960 000 грн.

Торгівельна надбавка до ціни (10%) – 396 000 грн

- Витрати торгівельної організації (маркетинг, підтримка) – 100 000 грн;
- Прибуток торгівельної організації – 296 000 грн;

- Податок на додану вартість торгівельної організації (20%) – 59 200 грн;
- Роздрібна ціна реалізації (кінцева для замовника) – 4 355 200 грн.

### 6.4.10 Споживчі властивості товару

Система «Диференціальна діагностика раку легень за даними комп’ютерної томографії» для користувача має головну споживчу властивість:

Забезпечення точного, об'єктивного та багаторівневого розмежування різних типів пухлинних утворень легень на основі ансамблевого аналізу томографічних даних з використанням трьох незалежних алгоритмів машинного навчання та інтелектуального розмовного агента для пояснення диференціально-діагностичних висновків.

В результаті використання системи споживач отримує:

1. Можливість верифікації діагностичних гіпотез через паралельне застосування трьох незалежних алгоритмів машинного навчання з різними методологічними підходами до аналізу томографічних ознак, що значно підвищує надійність розмежування схожих патологічних станів;
2. Інтерактивну взаємодію з інтелектуальним розмовним агентом, який пояснює результати диференціальної діагностики природною мовою, відповідає на уточнюючі питання лікаря та надає обґрунтування вагових коефіцієнтів ансамблевого голосування;
3. Прозорість діагностичного процесу з можливістю перегляду індивідуальних висновків кожного алгоритму, показників впевненості та внеску кожного методу у фінальний диференціально-діагностичний висновок;
4. Зручну інтеграцію до існуючих радіологічних інформаційних систем та робочих станцій комп'ютерної томографії без необхідності зміни звичних робочих процесів лікарів-радіологів;
5. Регулярне оновлення ансамблевих моделей на основі нових клінічних даних та вдосконалення методів витягання діагностичних ознак для підвищення точності розмежування патологічних станів.

### 6.4.11 Дослідження ринку

На світовому ринку програмних рішень для автоматизованої диференціальної діагностики раку легень на основі томографічних даних існують поодинокі продукти, проте більшість з них використовують одноалгоритмний підхід без ансамблевого голосування або не забезпечують інтерактивної взаємодії з лікарем через розмовного агента. Унікальність нашого рішення полягає у поєднанні трьох незалежних методів машинного навчання з різними методологічними підходами, що забезпечує вищу надійність диференціації, та інтеграції інтелектуального агента на основі великих мовних моделей для природномовного пояснення діагностичних висновків.

Дослідження конкурентного оточення

На ринку присутні окремі програмні продукти для автоматизованого аналізу томографічних зрізів, проте вони здебільшого обмежуються застосуванням одного алгоритму класифікації без можливості верифікації через ансамблеве голосування. Крім того, існуючі рішення не забезпечують інтерактивної взаємодії з лікарем для пояснення диференціально-діагностичних висновків природною мовою. Це дає нашій системі конкурентну перевагу завдяки багаторівневому підходу до верифікації діагностичних гіпотез та інтеграції розмовного агента.

Маркетингова стратегія просування

Система позиціонується як комплексний інструмент диференціальної діагностики для радіологічних відділень, який доповнює та розширює можливості існуючих діагностичних платформ через ансамблевий підхід та інтелектуального агента. Основна увага приділяється співпраці з провідними багатопрофільними лікарнями та онкологічними диспансерами, участі у радіологічних конференціях з демонстрацією складних диференціально-діагностичних випадків, а також створенню партнерств із виробниками томографічного обладнання та радіологічних інформаційних систем.

### 6.4.12 Елементи фінансового плану

Опис бізнес-проєкту

Розробка, впровадження та комерціалізація системи автоматизованої диференціальної діагностики раку легень на основі аналізу томографічних даних з використанням ансамблевих методів машинного навчання, інтелектуального розмовного агента та інтерактивних засобів візуалізації.

Опис товару або послуги

Програмний продукт для радіологічних відділень, який забезпечує точне розмежування різних типів пухлинних утворень легень через паралельне застосування трьох незалежних алгоритмів машинного навчання, знижує ризик діагностичних помилок та оптимізує роботу лікаря-радіолога завдяки автоматизації процесу аналізу томографічних зрізів та інтерактивної взаємодії з розмовним агентом для пояснення диференціально-діагностичних висновків.

Маркетинг та продаж

Дослідження ринку показало, що в Європі та Сполучених Штатах Америки попит на інноваційні системи диференціальної діагностики з використанням ансамблевих методів покривається лише частково, а в Україні ринок автоматизованих інструментів диференціальної діагностики тільки формується. Основними каналами просування є прямі продажі до радіологічних відділень лікарень та діагностичних центрів, модель програмного забезпечення як послуги для приватних радіологічних центрів, участь у спеціалізованих радіологічних конференціях та партнерство з постачальниками томографічного обладнання та радіологічних інформаційних систем.

Фінансовий план (на 9 місяців запуску)

1. Закупівля серверного обладнання, комп’ютерної техніки та необхідного програмного забезпечення – 1 200 000 грн;

2. Формування команди: найм розробників, фахівців із ШІ, лікаря-консультанта, маркетолога, інженера підтримки (зарплатний фонд на 9 місяців) – 1 000 000 грн;

3. Оренда офісу та дата-центру – 180 000 грн;
4. Оплата комунальних послуг та інтернету – 60 000 грн;
5. Сертифікація, ліцензування, юридичний супровід – 120 000 грн;
6. Витрати на маркетингову кампанію та участь у виставках – 100 000 грн;
7. Резерв, непередбачувані витрати – 140 000 грн;

ВСЬОГО: 2 800 000 грн.

**6.4.13 Резюме**

Проблема точної диференціальної діагностики раку легень залишається однією з ключових у сучасній радіології, оскільки різні типи пухлинних утворень часто мають схожі томографічні ознаки, що ускладнює їх розмежування та призводить до діагностичних помилок. Попри розвиток технологій комп'ютерної томографії, більшість радіологічних відділень все ще стикаються з обмеженнями традиційних підходів: суб'єктивністю інтерпретації знімків, відсутністю об'єктивних інструментів верифікації діагностичних гіпотез, нестачею часу у лікарів-радіологів на детальний багатоаспектний аналіз та високою ймовірністю помилок у складних диференціально-діагностичних ситуаціях.

Система «Диференціальна діагностика раку легень за даними комп’ютерної томографії» пропонує комплексне рішення цієї проблеми через впровадження ансамблевого підходу до автоматизованого аналізу томографічних даних. Вона забезпечує паралельне застосування трьох незалежних алгоритмів машинного навчання з різними методологічними підходами до витягання діагностичних ознак, що дозволяє точно розмежовувати різні типи пухлинних утворень та верифікувати діагностичні гіпотези через зважене голосування. Інтеграція інтелектуального розмовного агента на основі великих мовних моделей забезпечує природномовну взаємодію з лікарем, пояснення диференціально-діагностичних висновків та

відповіді на уточнюючі питання, що підвищує довіру до результатів автоматизованого аналізу.

Система відкриває нові можливості для радіологічних відділень медичних закладів України та інших країн, дозволяє впроваджувати світові стандарти диференціальної діагностики з використанням сучасних методів штучного інтелекту та підвищує конкурентоспроможність закладів на ринку медичних послуг. В умовах зростання захворюваності на онкологічні хвороби інвестиції у такі рішення – це прямий шлях до покращення якості діагностики, своєчасного виявлення захворювань та підвищення ефективності лікування пацієнтів з пухлинами легень

**Висновок з розділу 6**

У шостому розділі було розроблено та економічно обґрунтовано стартап-проєкт системи «Диференціальна діагностика раку легень». Аналіз ринкового середовища виявив наявність стійкого попиту на інструменти автоматизованої радіології, здатні забезпечити «другу думку» та верифікацію діагнозів. Запропонована бізнес-модель, що поєднує прямі продажі ліцензій та SaaS-підписку, дозволяє гнучко адаптуватися до потреб як великих онкологічних центрів, так і менших діагностичних кабінетів.

Фінансовий розрахунок показав, що при загальних витратах на запуск у розмірі 2,8 млн грн, проєкт має чітку структуру монетизації та потенціал для масштабування. Ключовою конкурентною перевагою, що забезпечує ринкову стійкість, є унікальна технологічна пропозиція: поєднання високої точності ансамблевих методів класифікації з інтерпретованістю результатів завдяки інтегрованому розмовному агенту. Реалізація даного стартап-проєкту є не лише економічно доцільною, але й має важливе соціальне значення, сприяючи підвищенню якості ранньої діагностики онкологічних захворювань.

# ЗАГАЛЬНІ ВИСНОВКИ

У даній роботі було виконано комплексне дослідження та практичну реалізацію інноваційної платформи диференціальної діагностики раку легень на основі даних комп'ютерної томографії з інтеграцією технологій штучного інтелекту та ансамблевого машинного навчання. Розроблена система представляє собою якісно новий підхід до автоматизації діагностичних процесів у медичній візуалізації, що поєднує автономну оркестрацію робочих процесів через інтелектуального програмного агента з паралельним виконанням множинних алгоритмів класифікації та природномовним інтерфейсом взаємодії з медичними спеціалістами.

Проведений аналіз десяти провідних світових систем штучного інтелекту для діагностики раку легень виявив фундаментальний технологічний розрив у сучасних рішеннях. Всі розглянуті платформи від Google Health, Alibaba DAMO Academy, Siemens Healthineers до Lunit та інших провідних розробників базуються на статичних конвеєрах обробки даних з фіксованими алгоритмами та попередньо визначеними послідовностями виконання. Жодна система не реалізує агентну архітектуру з автономним прийняттям рішень про вибір діагностичних стратегій, динамічною оркестрацією множинних моделей класифікації або адаптивним налаштуванням робочого процесу залежно від специфіки клінічного випадку. Виявлена відсутність конверсаційного штучного інтелекту для управління діагностичним процесом створює можливість для інновацій через мультиагентні системи з інтелектуальною інтеграцією ансамблю методів машинного навчання.

Теоретичне дослідження альтернативних технологічних підходів охопило системний аналіз мов програмування, серверних фреймворків, стратегій побудови клієнтських інтерфейсів, платформ оркестрації агентів, систем керування базами даних, методів обробки медичних зображень, алгоритмів машинного навчання, моделей асинхронного програмування та

технологій контейнеризації. Порівняльний аналіз забезпечив обґрунтований вибір оптимального технологічного стеку для реалізації медичної діагностичної платформи з урахуванням специфічних вимог до безпеки медичних даних, пояснюваності алгоритмічних рішень та продуктивності обробки томографічних зображень.

Обрано технологічний стек на основі мови програмування Python версії три цілих дванадцять з повною підтримкою асинхронного програмування, веб-фреймворку FastAPI для серверної частини з автоматичною генерацією документації програмних інтерфейсів, бібліотеки Gradio версії чотири для швидкого прототипування клієнтського інтерфейсу, фреймворку LangGraph для оркестрації багатокрокових робочих процесів, платформи Azure OpenAI з моделлю GPT-4 Turbo для конверсаційних можливостей та системи керування базами даних SQLite з асинхронним розширенням SQLAlchemy версії два цілих нуль. Обробку томографічних зображень реалізовано через комплекс бібліотек pydicom, Pillow, OpenCV та scikit-image, а ансамблеву класифікацію через scikit-learn, XGBoost та joblib. Асинхронність забезпечено бібліотекою asyncio, aiosqlite та AsyncOpenAI, а ізоляцію компонентів через технологію контейнеризації Docker з оркестратором Docker Compose.

Практична реалізація системи базується на мікросервісній архітектурі з трьома незалежними сервісами, що забезпечує чіткий розподіл відповідальностей та масштабованість компонентів. Фронтенд-сервіс на порту сім тисяч вісімсот шістдесят надає користувацький інтерфейс з інтерактивним редагуванням областей інтересу, бекенд-сервіс на порту вісім тисяч забезпечує обробку DICOM-файлів та керування зрізами томографії, а сервіс штучного інтелекту на порту вісім тисяч один реалізує оркестрацію агента та ансамблеву класифікацію. Застосування патернів чистої архітектури з розділенням на шари додатку, домену та інфраструктури забезпечує підтримуваність та тестованість коду.

Ключовою інновацією розробленої системи є ансамблевий підхід до класифікації з паралельним виконанням трьох алгоритмів машинного

навчання через функцію asyncio.gather, що скорочує загальний час класифікації до двадцяти мілісекунд. Алгоритми випадкового лісу, методу опорних векторів з радіальною базисною функцією та градієнтного бустингу виконуються одночасно з подальшим зваженим голосуванням тридцять п'ять відсотків, тридцять відсотків та тридцять п'ять відсотків відповідно. Система досягла загальної точності вісімдесят сім цілих три десятих відсотка на тестових даних, перевищуючи продуктивність індивідуальних класифікаторів завдяки балансуванню специфічних переваг кожного методу.

Екстракція від дев'яноста до ста характеристик медичних зображень охоплює три домени: статистичні характеристики інтенсивності з аналізом гістограм, текстурні ознаки через матрицю суміжності рівнів сірого та локальні бінарні патерни, а також морфологічні характеристики форми з моментами Ху для інваріантності до обертання. Автоматизоване виявлення підозрілих ділянок реалізовано через порогову сегментацію методом Отсу з морфологічним уточненням контурів через операції ерозії та дилатації. Повний цикл діагностики від отримання запиту користувача до відображення результатів з природномовним поясненням виконується за шістсот-дві тисячі двохсот мілісекунд, що забезпечує комфортну інтерактивну взаємодію.

Інтелектуальний агент на основі фреймворку LangGraph з інтеграцією Azure OpenAI забезпечує автономну оркестрацію діагностичного робочого процесу через динамічний вибір інструментів на основі контексту розмови. Агент здатен викликати функції класифікації областей інтересу, автоматичного виявлення підозрілих ділянок, формування критеріїв розмічування та рекомендацій щодо попередньої обробки зображень без жорстко закодованих послідовностей виконання. Персистентність стану робочого процесу забезпечує можливість багатокрокових діагностичних сесій з повним збереженням контексту розмови та проміжних результатів класифікації.

Розроблена бізнес-модель комерціалізації системи охоплює визначення цільових сегментів споживачів серед радіологічних відділень

багатопрофільних лікарень, онкологічних диспансерів та спеціалізованих діагностичних центрів, стратегію каналів збуту через прямі продажі та партнерські угоди з виробниками томографічного обладнання, фінансовий план з повною собівартістю розробки два мільйони п'ятсот тисяч гривень та роздрібною ціною чотири мільйони триста п'ятдесят п'ять тисяч двісті гривень. Модель програмного забезпечення як послуги забезпечує гнучке масштабування та зниження бар'єрів впровадження для медичних закладів з обмеженими ресурсами.

Порівняльний аналіз з існуючими системами виявляє суттєві переваги розробленого рішення у сферах інтерпретованості діагностичних рішень через використання класичного машинного навчання з аналізом конкретних ознак, доступності через природномовний інтерфейс українською мовою та гнучкості архітектури для адаптації під локальні вимоги. Відсутність вимог до спеціалізованого обладнання та застосування класичних алгоритмів замість глибоких нейронних мереж забезпечує економічну ефективність та пояснюваність для медичної валідації.

Результати дослідження підтверджують досягнення поставленої мети створення інтелектуальної системи диференціальної діагностики раку легень з агентною архітектурою та ансамблевим підходом до класифікації. Розроблена платформа готова до використання як інструмент підтримки прийняття діагностичних рішень медичними спеціалістами та створює фундамент для подальшого розвитку адаптивних систем штучного інтелекту у медичній візуалізації.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ardila D., Kiraly A.P., Bharadwaj S., Choi B., Reicher J.J., Peng L., Tse D., Etemadi M., Ye W., Corrado G., Naidich D.P., Shetty S. End-to-end lung cancer screening with three-dimensional deep learning on low-dose chest computed tomography [Електронний ресурс] / Nature Medicine. – 2019. – Том 25, номер 6. – Сторінки 954-961. – Режим доступу: https://www.nature.com/articles/s41591-019-0447-x (дата звернення: 27.10.2025);

2. Jacobs C., van Ginneken B. Google's lung cancer AI: a promising tool that needs further validation [Електронний ресурс] / Nature Reviews Clinical Oncology. – 2019. – Том 16. – Сторінки 532-533. – Режим доступу: https://www.nature.com/articles/s41571-019-0248-7 (дата звернення: 27.10.2025);

3. A promising step forward for predicting lung cancer [Електронний ресурс] / Google AI Blog. – 2019. – Режим доступу: https://blog.google/technology/health/lung-cancer-prediction/ (дата звернення: 28.10.2025);

4. National Lung Screening Trial Data [Електронний ресурс] / National Cancer Institute. – Режим доступу: https://biometry.nci.nih.gov/cdas/learn/nlst/images/ (дата звернення: 28.10.2025);

5. Alibaba DAMO Academy Medical AI Documentation [Електронний ресурс] / Alibaba Cloud. – 2017-2024. – Режим доступу: https://www.alibabacloud.com (дата звернення: 29.10.2025);

6. LUNA16 Grand Challenge [Електронний ресурс] / LUNA16. – 2017. – Режим доступу: https://luna16.grand-challenge.org/ (дата звернення: 29.10.2025);

7. How DAMO Academy Uses Medical AI to Help the Sick [Електронний ресурс] / Alibaba Cloud Community. – 2020. – Режим доступу: https://www.alibabacloud.com/blog/596184 (дата звернення: 30.10.2025);

8. Beating Lung, Liver, and Cardiovascular Diseases with AI [Електронний ресурс] / Alibaba Cloud Community. – 2019. – Режим доступу: https://www.alibabacloud.com/blog/beating-lung-liver-and-cardiovascular-diseases-with-ai_595266 (дата звернення: 30.10.2025);

9. CT Image Analytics for COVID-19 [Електронний ресурс] / Alibaba Cloud. – 2020. – Режим доступу: https://www.alibabacloud.com/solutions/ct-image-analytics (дата звернення: 31.10.2025);

10. Q and A: How DAMO Academy is Leveraging AI for Good Initiative in Healthcare [Електронний ресурс] / Alizila. – 2024. – Режим доступу: https://www.alizila.com/damo-academy-ai-for-good-cancer-who-healthcare-2024/ (дата звернення: 31.10.2025);

11. Siemens Healthcare GmbH. AI-Rad Companion Chest CT VA13: Features, Data, and Algorithms - Whitepaper [Електронний ресурс] / Siemens Healthineers. – 2021. – Режим доступу: https://cdn0.scrvt.com/39b415fb07de4d9656c7b516d8e2d907/3c06d6256fff0b5d/c01241ad9c70/DH-AI-Rad-Companion-Chest-CT-Whitepaper--2-.PDF (дата звернення: 01.11.2025);

12. Chamberlin J., Kocher M.R., Waltz J., et al. Automated detection of lung nodules and coronary artery calcium using artificial intelligence on low-dose CT scans for lung cancer screening: accuracy and prognostic value [Електронний ресурс] / BMC Medicine. – 2021. – Том 19, номер 1. – Стаття 55. – Режим доступу: https://bmcmedicine.biomedcentral.com/articles/10.1186/s12916-021-01928-3 (дата звернення: 01.11.2025);

13. Abadia A.F., et al. Diagnostic accuracy and performance of artificial intelligence in detecting lung nodules in patients with complex lung disease: a non-inferiority study [Електронний ресурс] / Journal of Thoracic Imaging. – 2021. – Режим доступу: https://journals.lww.com/thoracicimaging (дата звернення: 02.11.2025);

14. Yacoub B., Kabakus I.M., Schoepf U.J., et al. Performance of an Artificial Intelligence-Based Platform Against Clinical Radiology Reports for the

Evaluation of Noncontrast Chest CT [Електронний ресурс] / Academic Radiology. – 2021. – Режим доступу: https://www.sciencedirect.com/science/article/abs/pii/S1076633221000702 (дата звернення: 02.11.2025);

15. van Assen M., Martin S.S., Varga-Szemes A., et al. Automatic coronary calcium scoring in chest CT using a deep neural network in direct comparison with non-contrast cardiac CT: A validation study [Електронний ресурс] / European Journal of Radiology. – 2021. – Том 134. – Стаття 109428. – Режим доступу: https://www.ejradiology.com (дата звернення: 03.11.2025);

16. Rückel J., Sperl J., Kaestle S., et al. Reduction of missed thoracic findings in emergency whole-body CT with AI assistance [Електронний ресурс] / Quantitative Imaging in Medicine and Surgery. – 2021. – Том 11, номер 6. – Сторінки 2486-2498. – Режим доступу: https://qims.amegroups.com (дата звернення: 03.11.2025);

17. Li K., Liang M., et al. Assessing the predictive accuracy of lung cancer, metastases, and benign lesions using an artificial intelligence-driven computer aided diagnosis system [Електронний ресурс] / Quantitative Imaging in Medicine and Surgery. – 2021. – Том 11, номер 7. – Сторінки 3629-3642. – Режим доступу: https://pmc.ncbi.nlm.nih.gov/articles/PMC8245931/ (дата звернення: 04.11.2025);

18. Infervision Receives FDA Clearance for InferRead CT Lung [Електронний ресурс] / PR Newswire. – 2020. – Режим доступу: https://www.prnewswire.com/news-releases/infervision-receives-fda-clearance-for-the-inferread-lung-ctai-product-301091145.html (дата звернення: 04.11.2025);

19. Infervision Receives FDA 510(k) Clearance for Enhanced Features [Електронний ресурс] / Infervision Global. – 2025. – Режим доступу: https://global.infervision.com/blog/infervision-receives-fda-510-k-clearance-for-enhanced-features-in-inferread-ct-lung (дата звернення: 04.11.2025);

20. Nam J.G., Park S., Hwang E.J., et al. Development and Validation of Deep Learning–based Automatic Detection Algorithm for Malignant Pulmonary Nodules on Chest Radiographs [Електронний ресурс] / Radiology. – 2019. – Том

290, номер 1. – Сторінки 218-228. – Режим доступу: https://pubs.rsna.org/doi/abs/10.1148/radiol.2018180237 (дата звернення: 05.11.2025);

21. van Leeuwen K.G., et al. Comparison of Commercial AI Software Performance for Radiograph Lung Nodule Detection [Електронний ресурс] / Radiology. – 2024. – Том 310, номер 1. – Режим доступу: https://pubs.rsna.org/doi/full/10.1148/radiol.221894 (дата звернення: 05.11.2025);

22. Lunit Official Product Page [Електронний ресурс] / Lunit. – Режим доступу: https://www.lunit.io/en/products/cxr (дата звернення: 05.11.2025);

23. Lunit INSIGHT CXR Excels in Lung Nodule Detection - Exceptional Performance in Head-to-Head Study published in Radiology [Електронний ресурс] / Lunit Home. – 2024. – Режим доступу: https://www.lunit.io/en/company/news/lunit-insight-cxr-excels-in-lung-nodule-detection---exceptional-performance-in-head-to-head-study-published-in-radiology (дата звернення: 06.11.2025);

24. Golden D. Lung cancer detection and segmentation using deep learning [Електронний ресурс] / O'Reilly AI Conference. – 2018. – Режим доступу: https://conferences.oreilly.com/artificial-intelligence/ai-ca-2018/public/schedule/detail/68586.html (дата звернення: 27.10.2025);

25. Arterys Receives First FDA Clearance for Broad Oncology Imaging Suite [Електронний ресурс] / PR Newswire. – 2018. – Режим доступу: https://www.prnewswire.com/news-releases/arterys-receives-first-fda-clearance-for-broad-oncology-imaging-suite-with-deep-learning-300599275.html (дата звернення: 28.10.2025);

26. Rajpurkar P., Irvin J., Zhu K., et al. CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning [Електронний ресурс] / arXiv. – 2017. – arXiv:1711.05225. – Режим доступу: https://arxiv.org/pdf/1711.05225 (дата звернення: 29.10.2025);

27. Rajpurkar P., Irvin J., Ball R.L., et al. Deep learning for chest radiograph diagnosis: A retrospective comparison of the CheXNeXt algorithm to

practicing radiologists [Електронний ресурс] / PLOS Medicine. – 2018. – Том 15, номер 11. – Стаття e1002686. – Режим доступу: https://doi.org/10.1371/journal.pmed.1002686 (дата звернення: 30.10.2025);

28. Stanford ML Group CheXNet Project [Електронний ресурс] / Stanford University. – Режим доступу: https://stanfordmlgroup.github.io/projects/chexnet/ (дата звернення: 31.10.2025);

29. Stanford ML Group CheXNeXt Project [Електронний ресурс] / Stanford University. – Режим доступу: https://stanfordmlgroup.github.io/projects/chexnext/ (дата звернення: 01.11.2025);

30. Qure.ai Launches FDA-Cleared AI for Lung Nodule Quantification [Електронний ресурс] / Qure.ai. – 2024. – Режим доступу: https://www.qure.ai/news_press_coverages/qure.ai-launches-FDA-cleared-AI-solution-for-advanced-lung-nodule-quantification-on-CT-scans-at-AABIP-2024 (дата звернення: 02.11.2025);

31. Qure.ai Becomes First AI-based Chest X-ray Tool to Receive CE Certification [Електронний ресурс] / PR Newswire. – 2018. – Режим доступу: https://www.prnewswire.com/news-releases/qureais-qxr-becomes-first-ai-based-chest-x-ray-interpretation-tool-to-receive-ce-certification-684142231.html (дата звернення: 03.11.2025);

32. Massion P.P., et al. Assessing the Accuracy of a Deep Learning Method to Risk Stratify Indeterminate Pulmonary Nodules [Електронний ресурс] / American Journal of Respiratory and Critical Care Medicine. – 2020. – Том 202, номер 2. – Сторінки 241-249. – Режим доступу: https://pmc.ncbi.nlm.nih.gov/articles/PMC7365375/ (дата звернення: 04.11.2025);

33. Baldwin D.R., et al. External validation of a convolutional neural network artificial intelligence tool to predict malignancy in pulmonary nodules [Електронний ресурс] / Thorax. – 2020. – Том 75, номер 4. – Сторінки 306-312. – Режим доступу: https://thorax.bmj.com/content/75/4/306 (дата звернення: 05.11.2025);

34. Optellum Receives FDA Clearance for AI-Powered Clinical Decision Support [Електронний ресурс] / Optellum. – 2021. – Режим доступу: https://optellum.com/fda-clearance/ (дата звернення: 06.11.2025);

35. AtlantiCare deploys Optellum AI for early lung cancer diagnosis, treatment [Електронний ресурс] / AtlantiCare. – 2024. – Режим доступу: https://www.atlanticare.org/news/atlanticare-deploys-optellum-ai-for-early-lung-cancer-diagnosis-treatment (дата звернення: 27.10.2025);

36. VIDA Previews LungPrint at RSNA 2018 [Електронний ресурс] / PR Newswire. – 2018. – Режим доступу: https://www.prnewswire.com/news-releases/vida-previews-lungprint-an-ai-powered-lung-imaging-analytics-solution-launching-at-rsna-2018-300750089.html (дата звернення: 28.10.2025);

37. VIDA Diagnostics Receives FDA Clearance for AI LungPrint Solution [Електронний ресурс] / Science Times. – 2020. – Режим доступу: https://www.sciencetimes.com/articles/27549/20201002/vida-diagnostics-fda-clearance-advance-ai-lungprint-solution.htm (дата звернення: 29.10.2025).

38. Kish P. Medical image processing using Python, Java and C++ libraries: Comparative analysis [Електронний ресурс] / ResearchGate. – 2024. – Режим доступу: https://www.researchgate.net/post/What-programming-language-should-I-use-for-a-medical-image-application (дата звернення: 30.10.2025);

39. Ousterhout J.K. Scripting: Higher Level Programming for the 21st Century [Електронний ресурс] / IEEE Computer. – 1998. – Режим доступу: https://softwareengineering.stackexchange.com/questions/69005/is-programming-in-python-faster-than-in-c-c-or-java (дата звернення: 31.10.2025);

40. Python Software Foundation. What's New In Python 3.12 [Електронний ресурс] / Python Documentation. – 2024. – Режим доступу: https://docs.python.org/3/whatsnew/3.12.html (дата звернення: 01.11.2025);

41. Merative. Merge DICOM Toolkit: Cross-Platform Medical Imaging SDK [Електронний ресурс] / Merative Healthcare. – 2024. – Режим доступу: https://www.merative.com/merge-imaging/dicom-toolkit (дата звернення: 02.11.2025);

42. Chen X., Kumar S. Java vs Python vs R: Best Programming Languages for Healthcare Machine Learning [Електронний ресурс] / MoldStud Analytics. – 2025. – Режим доступу: https://moldstud.com/articles/p-java-vs-python-vs-r-best-programming-languages-for-healthcare-machine-learning (дата звернення: 03.11.2025);

43. Vishwakarma K. Java vs Python vs C++ comparison: coding and performance benchmarks [Електронний ресурс] / Medium. – 2024. – Режим доступу: https://medium.com/@karanvishwakarma/java-python-c-comparison-coding-and-numerical-b25122698930 (дата звернення: 04.11.2025);

44. EDUCBA. Java vs Python vs C++: Performance Analysis and Use Cases [Електронний ресурс] / EDUCBA Technical Reviews. – 2024. – Режим доступу: https://www.educba.com/java-vs-python-vs-cpp/ (дата звернення: 05.11.2025);

45. Databricks Engineering. 7x Faster Medical Image Ingestion with Python Data Source API [Електронний ресурс] / Databricks Blog. – 2024. – Режим доступу: https://www.databricks.com/blog/7x-faster-medical-image-ingestion-python-data-source-api (дата звернення: 06.11.2025);

46. Lasting Dynamics. Python VS Java, C++ and More: Ultimate Showdown in Software Development [Електронний ресурс] / Lasting Dynamics Blog. – 2025. – Режим доступу: https://www.lastingdynamics.com/blog/python-vs/ (дата звернення: 27.10.2025);

47. Full Scale. Java vs Python: Comprehensive Comparison for Enterprise Software Development [Електронний ресурс] / Full Scale Blog. – 2025. – Режим доступу: https://fullscale.io/blog/java-vs-python/ (дата звернення: 28.10.2025);

48. GeeksforGeeks. Comparison of FastAPI with Django and Flask: Comprehensive Framework Analysis [Електронний ресурс] / GeeksforGeeks Technical Education. – 2025. – Режим доступу: https://www.geeksforgeeks.org/python/comparison-of-fastapi-with-django-and-flask/ (дата звернення: 29.10.2025);

49. BrowserStack. Top 10 Python REST API Frameworks in 2025: Performance and Async Capabilities [Електронний ресурс] / BrowserStack Quality Assurance. – 2025. – Режим доступу: https://www.browserstack.com/guide/top-python-rest-api-frameworks (дата звернення: 30.10.2025);

50. Better Stack Community. Django vs FastAPI: Choosing the Right Python Web Framework [Електронний ресурс] / Better Stack Engineering. – 2024. – Режим доступу: https://betterstack.com/community/guides/scaling-python/django-vs-fastapi/ (дата звернення: 31.10.2025);

51. Geekflare. Top 9 Asynchronous Web Frameworks for Python: Quart, Sanic, and Tornado [Електронний ресурс] / Geekflare Developer Resources. – 2024. – Режим доступу: https://geekflare.com/dev/python-asynchronous-web-frameworks/ (дата звернення: 01.11.2025);

52. GitHub. Python Web Frameworks Performance Comparison: Flask, Django, Quart and FastAPI [Електронний ресурс] / Agus Makmun Performance Testing. – 2024. – Режим доступу: https://github.com/agusmakmun/flask-django-quart-fastapi-performance-test-comparison (дата звернення: 02.11.2025);

53. PyCharm Blog. Django vs FastAPI: Which is the Best Python Web Framework for 2025 [Електронний ресурс] / JetBrains Developer Advocacy. – 2025. – Режим доступу: https://blog.jetbrains.com/pycharm/2023/12/django-vs-fastapi-which-is-the-best-python-web-framework/ (дата звернення: 03.11.2025);

54. Codeanywhere. The Most Popular Python Frameworks in 2024: Healthcare and Real-Time Applications [Електронний ресурс] / Codeanywhere Development Blog. – 2024. – Режим доступу: https://codeanywhere.com/blog/the-most-popular-python-frameworks-in-2024 (дата звернення: 04.11.2025);

55. UnfoldAI. Streamlit vs Gradio: Choosing the right framework for ML applications [Електронний ресурс] / UnfoldAI Engineering Blog. – 2024. – Режим доступу: https://unfoldai.com/streamlit-vs-gradio/ (дата звернення: 05.11.2025);

56. AMELA Technology. Angular vs React vs Vue.js: A Detailed Comparison for 2025 [Електронний ресурс] / AMELA Software Development. –

2025. – Режим доступу: https://amela.tech/a-detailed-2024-comparison-angular-vs-react-vs-vue-js/ (дата звернення: 06.11.2025);

57. Squadbase. Streamlit vs Gradio in 2025: Comparing AI-App Frameworks [Електронний ресурс] / Squadbase Development Blog. – 2025. – Режим доступу: https://www.squadbase.dev/en/blog/streamlit-vs-gradio-in-2025-a-framework-comparison-for-ai-apps (дата звернення: 27.10.2025);

58. Medium. Streamlit vs Gradio: A Comprehensive Comparison for Data Science [Електронний ресурс] / Shahab Hasan Technical Writing. – 2023. – Режим доступу: https://medium.com/@ShahabH/streamlit-vs-gradio-a-comprehensive-comparison-cc2f28b7b832 (дата звернення: 28.10.2025);

59. Gislen Software. Frontend frameworks 2024: React, Angular, Vue, and Svelte comparison [Електронний ресурс] / Gislen Software Development. – 2025. – Режим доступу: https://www.gislen.com/best-frontend-frameworks-2024/ (дата звернення: 29.10.2025);

60. Medium. Streamlit vs Gradio vs Chainlit: Best UI Framework for LLMs [Електронний ресурс] / Saiii Development Blog. – 2025. – Режим доступу: https://medium.com/@sailakkshmiallada/streamlit-vs-gradio-and-more-building-ml-web-apps-6753f5147276 (дата звернення: 30.10.2025);

61. Evidence Learn. Gradio vs Streamlit: Choosing a Tool for Your Data Application [Електронний ресурс] / Evidence Development Resources. – 2024. – Режим доступу: https://evidence.dev/learn/gradio-vs-streamlit (дата звернення: 31.10.2025);

62. Medium. Three AI Design Patterns of Autonomous Agents: Workflow Types and Orchestration [Електронний ресурс] / Alexander Sniffin Technical Analysis. – 2024. – Режим доступу: https://alexsniffin.medium.com/three-ai-design-patterns-of-autonomous-agents-8372b9402f7c (дата звернення: 01.11.2025);

63. Microsoft Learn. AI Agent Orchestration Patterns: Sequential, Parallel, and Hierarchical [Електронний ресурс] / Azure Architecture Center. – 2024. –

Режим доступу: https://learn.microsoft.com/en-us/azure/architecture/ai-ml/guide/ai-agent-design-patterns (дата звернення: 02.11.2025);

64. AiMultiple Research. Top 5 Open-Source Agentic Frameworks: LangGraph, AutoGen, CrewAI [Електронний ресурс] / AiMultiple Research Institute. – 2024. – Режим доступу: https://research.aimultiple.com/agentic-frameworks/ (дата звернення: 03.11.2025);

65. n8n Blog. AI Agent Orchestration Frameworks: State Management and Production Deployment [Електронний ресурс] / n8n Workflow Automation. – 2024. – Режим доступу: https://blog.n8n.io/ai-agent-orchestration-frameworks/ (дата звернення: 04.11.2025);

66. IBM Think. What is AI Agent Orchestration: Challenges and Solutions [Електронний ресурс] / IBM Research and Development. – 2025. – Режим доступу: https://www.ibm.com/think/topics/ai-agent-orchestration (дата звернення: 05.11.2025);

67. Venugopal V.K., Vaidhya K., Murali N., Mestler S., Rohit V., Mahajan V. Clinical data classification using machine learning: a review of standard practices and new perspectives for responsible AI [Електронний ресурс] / Frontiers in Medicine. – 2024. – Том 11. – Режим доступу: https://www.frontiersin.org/journals/medicine/articles/10.3389/fmed.2024.1432811/full (дата звернення: 06.11.2025);

68. Atalag K., Yang H.Y., Tempero E., Warren J. Evaluation of software architectural choices for implementing HIPAA security rules [Електронний ресурс] / AMIA Annual Symposium Proceedings. – 2011. – Том 2011. – Сторінки 62-70. – Режим доступу: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3243131/ (дата звернення: 27.10.2025);

69. Nielsen J. Response Times: The 3 Important Limits [Електронний ресурс] / Nielsen Norman Group. – 1993. – Режим доступу: https://www.nngroup.com/articles/response-times-3-important-limits/ (дата звернення: 28.10.2025);

70. Schick T., Dwivedi-Yu J., Dessì R., Raileanu R., Lomeli M., Zettlemoyer L., Cancedda N., Scialom T. Toolformer: Language Models Can Teach Themselves to Use Tools [Електронний ресурс] / arXiv preprint. – 2023. – Режим доступу: https://arxiv.org/abs/2302.04761 (дата звернення: 29.10.2025);

71. Bommasani R., Hudson D.A., Adeli E., Altman R., Arora S., von Arx S., Bernstein M.S., Bohg J., Bosselut A., Brunskill E., Brynjolfsson E., Buch S., Card D., Castellon R., Chatterji N., Chen A., Creel K., Davis J.Q., Demszky D., Donahue C., Doumbouya M., Durmus E., Ermon S., Etchemendy J., Ethayarajh K., Fei-Fei L., Finn C., Gale T., Gillespie L., Goel K., Goodman N., Grossman S., Guha N., Hashimoto T., Henderson P., Hewitt J., Ho D.E., Hong J., Hsu K., Huang J., Icard T., Jain S., Jurafsky D., Kalluri P., Karamcheti S., Keeling G., Khani F., Khattab O., Koh P.W., Krass M., Krishna R., Kuditipudi R., Kumar A., Ladhak F., Lee M., Lee T., Leskovec J., Levent I., Li X.L., Li X., Ma T., Malik A., Manning C.D., Mirchandani S., Mitchell E., Munyikwa Z., Nair S., Narayan A., Narayanan D., Newman B., Nie A., Niebles J.C., Nilforoshan H., Nyarko J., Ogut G., Orr L., Papadimitriou I., Park J.S., Piech C., Portelance E., Potts C., Raghunathan A., Reich R., Ren H., Rong F., Roohani Y., Ruiz C., Ryan J., Ré C., Sadigh D., Sagawa S., Santhanam K., Shih A., Srinivasan K., Tamkin A., Taori R., Thomas A.W., Tramèr F., Wang R.E., Wang W., Wu B., Wu J., Wu Y., Xie S.M., Yasunaga M., You J., Zaharia M., Zhang M., Zhang T., Zhang X., Zhang Y., Zheng L., Zhou K., Liang P. On the Opportunities and Risks of Foundation Models [Електронний ресурс] / arXiv preprint. – 2021. – Режим доступу: https://arxiv.org/abs/2108.07258 (дата звернення: 30.10.2025);

72. Nori H., King N., McKinney S.M., Carignan D., Horvitz E. Capabilities of GPT-4 on Medical Challenge Problems [Електронний ресурс] / arXiv preprint. – 2023. – Режим доступу: https://arxiv.org/abs/2303.13375 (дата звернення: 31.10.2025);

73. Kleppmann M. Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems [Текст] / O'Reilly Media. – 2017. – 616 сторінок;

74. Owens M., Allen G. The Definitive Guide to SQLite [Текст] / Second Edition. – Apress. – 2010. – 368 сторінок;

75. PostgreSQL: About [Електронний ресурс] / PostgreSQL Global Development Group. – Режим доступу: https://www.postgresql.org/about/ (дата звернення: 01.11.2025);

76. Hellerstein J.M., Stonebraker M., Hamilton J. Architecture of a Database System [Електронний ресурс] / Foundations and Trends in Databases. – 2007. – Том 1, номер 2. – Сторінки 141-259. – Режим доступу: https://dsf.berkeley.edu/papers/fntdb07-architecture.pdf (дата звернення: 02.11.2025);

77. Selivanov D. Asynchronous Database Access in Python: Comparing sync vs async drivers [Електронний ресурс] / Real Python. – 2023. – Режим доступу: https://realpython.com/async-io-python/ (дата звернення: 03.11.2025);

78. Fowler M. Patterns of Enterprise Application Architecture [Текст] / Addison-Wesley Professional. – 2002. – 560 сторінок;

79. Evans E. Domain-Driven Design: Tackling Complexity in the Heart of Software [Текст] / Addison-Wesley Professional. – 2003. – 560 сторінок;

80. Codd E.F. A Relational Model of Data for Large Shared Data Banks [Електронний ресурс] / Communications of the ACM. – 1970. – Том 13, номер 6. – Сторінки 377-387. – Режим доступу: https://www.seas.upenn.edu/~zives/03f/cis550/codd.pdf (дата звернення: 04.11.2025);

81. Mildenberger P., Eichelberg M., Martin E. Introduction to the DICOM standard [Електронний ресурс] / European Radiology. – 2002. – Том 12, номер 4. – Сторінки 920-927. – Режим доступу: https://link.springer.com/article/10.1007/s003300101100 (дата звернення: 05.11.2025);

82. Mason D. SU-E-T-33: Pydicom: An Open Source DICOM Library [Електронний ресурс] / Medical Physics. – 2011. – Том 38, номер 6. – Сторінка 3493. – Режим доступу:

https://aapm.onlinelibrary.wiley.com/doi/10.1118/1.3611983 (дата звернення: 06.11.2025);

83. Pianykh O.S. Digital Imaging and Communications in Medicine: A Practical Introduction and Survival Guide [Текст] / Second Edition. – Springer. – 2012. – 442 сторінки;

84. Otsu N. A Threshold Selection Method from Gray-Level Histograms [Електронний ресурс] / IEEE Transactions on Systems, Man, and Cybernetics. – 1979. – Том 9, номер 1. – Сторінки 62-66. – Режим доступу: https://ieeexplore.ieee.org/document/4310076 (дата звернення: 27.10.2025);

85. Soille P. Morphological Image Analysis: Principles and Applications [Текст] / Second Edition. – Springer. – 2003. – 391 сторінка;

86. Haralick R.M., Shanmugam K., Dinstein I. Textural Features for Image Classification [Електронний ресурс] / IEEE Transactions on Systems, Man, and Cybernetics. – 1973. – Том 3, номер 6. – Сторінки 610-621. – Режим доступу: https://ieeexplore.ieee.org/document/4309314 (дата звернення: 28.10.2025);

87. van der Walt S., Schönberger J.L., Nunez-Iglesias J., Boulogne F., Warner J.D., Yager N., Gouillart E., Yu T. scikit-image: image processing in Python [Електронний ресурс] / PeerJ. – 2014. – Том 2. – Режим доступу: https://peerj.com/articles/453/ (дата звернення: 29.10.2025);

88. LeCun Y., Bengio Y., Hinton G. Deep learning [Електронний ресурс] / Nature. – 2015. – Том 521, номер 7553. – Сторінки 436-444. – Режим доступу: https://www.nature.com/articles/nature14539 (дата звернення: 30.10.2025);

89. Quinlan J.R. Induction of decision trees [Електронний ресурс] / Machine Learning. – 1986. – Том 1, номер 1. – Сторінки 81-106. – Режим доступу: https://link.springer.com/article/10.1007/BF00116251 (дата звернення: 31.10.2025);

90. Breiman L. Random Forests [Електронний ресурс] / Machine Learning. – 2001. – Том 45, номер 1. – Сторінки 5-32. – Режим доступу: https://link.springer.com/article/10.1023/A:1010933404324 (дата звернення: 01.11.2025);

91. Cortes C., Vapnik V. Support-vector networks [Електронний ресурс] / Machine Learning. – 1995. – Том 20, номер 3. – Сторінки 273-297. – Режим доступу: https://link.springer.com/article/10.1007/BF00994018 (дата звернення: 02.11.2025);

92. Chen T., Guestrin C. XGBoost: A Scalable Tree Boosting System [Електронний ресурс] / Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. – 2016. – Сторінки 785-794. – Режим доступу: https://arxiv.org/abs/1603.02754 (дата звернення: 03.11.2025);

93. Shen D., Wu G., Suk H.I. Deep Learning in Medical Image Analysis [Електронний ресурс] / Annual Review of Biomedical Engineering. – 2017. – Том 19. – Сторінки 221-248. – Режим доступу: https://www.annualreviews.org/doi/10.1146/annurev-bioeng-071516-044442 (дата звернення: 04.11.2025);

94. Pedregosa F., Varoquaux G., Gramfort A., Michel V., Thirion B., Grisel O., Blondel M., Prettenhofer P., Weiss R., Dubourg V., Vanderplas J., Passos A., Cournapeau D., Brucher M., Perrot M., Duchesnay E. Scikit-learn: Machine Learning in Python [Електронний ресурс] / Journal of Machine Learning Research. – 2011. – Том 12. – Сторінки 2825-2830. – Режим доступу: https://jmlr.org/papers/v12/pedregosa11a.html (дата звернення: 05.11.2025);

95. Stevens W.R., Fenner B., Rudoff A.M. UNIX Network Programming, Volume 1: The Sockets Networking API [Текст] / Third Edition. – Addison-Wesley Professional. – 2003. – 1024 сторінки;

96. Butenhof D.R. Programming with POSIX Threads [Текст] / Addison-Wesley Professional. – 1997. – 398 сторінок;

97. Stevens W.R., Rago S.A. Advanced Programming in the UNIX Environment [Текст] / Third Edition. – Addison-Wesley Professional. – 2013. – 1032 сторінки;

98. Beazley D. A Curious Course on Coroutines and Concurrency [Електронний ресурс] / PyCon Tutorial. – 2009. – Режим доступу: http://www.dabeaz.com/coroutines/ (дата звернення: 06.11.2025);

99. Python asyncio documentation [Електронний ресурс] / Python Software Foundation. – Режим доступу: https://docs.python.org/3/library/asyncio.html (дата звернення: 27.10.2025);

100. Gorelick M., Ozsvald I. High Performance Python: Practical Performant Programming for Humans [Текст] / Second Edition. – O'Reilly Media. – 2020. – 468 сторінок;

101. Hightower K., Burns B., Beda J. Kubernetes: Up and Running [Текст] / Second Edition. – O'Reilly Media. – 2019. – 318 сторінок;

102. Ramalho L. Fluent Python: Clear, Concise, and Effective Programming [Текст] / Second Edition. – O'Reilly Media. – 2022. – 1014 сторінок;

103. Limoncelli T.A., Chalup S.R., Hogan C.J. The Practice of System and Network Administration [Текст] / Third Edition. – Addison-Wesley Professional. – 2016. – 1504 сторінки;

104. Portnoy M. Virtualization Essentials [Текст] / Second Edition. – Sybex. – 2016. – 336 сторінок;

105. Merkel D. Docker: lightweight Linux containers for consistent development and deployment [Електронний ресурс] / Linux Journal. – 2014. – Том 2014, номер 239. – Режим доступу: https://www.linuxjournal.com/content/docker-lightweight-linux-containers-consistent-development-and-deployment (дата звернення: 28.10.2025);

106. Turnbull J. The Docker Book: Containerization is the new virtualization [Текст] / James Turnbull. – 2014. – 408 сторінок;

107. Burns B., Grant B., Oppenheimer D., Brewer E., Wilkes J. Borg, Omega, and Kubernetes [Електронний ресурс] / ACM Queue. – 2016. – Том 14, номер 1. – Режим доступу: https://queue.acm.org/detail.cfm?id=2898444 (дата звернення: 29.10.2025);

108. Grüning B., Dale R., Sjödin A., Chapman B.A., Rowe J., Tomkins-Tinch C.H., Valieris R., Köster J. Bioconda: sustainable and comprehensive software distribution for the life sciences [Електронний ресурс] / Nature Methods. – 2018. – Том 15, номер 7. – Сторінки 475-476. – Режим доступу: https://www.nature.com/articles/s41592-018-0046-7 (дата звернення: 30.10.2025);

109. Python Packaging User Guide [Електронний ресурс] / Python Packaging Authority. – Режим доступу: https://packaging.python.org/ (дата звернення: 31.10.2025);

110. Preston-Werner T. Semantic Versioning 2.0.0 [Електронний ресурс] / Semantic Versioning Specification. – Режим доступу: https://semver.org/ (дата звернення: 01.11.2025).

111. Python Software Foundation. What's New In Python 3.12 [Електронний ресурс]. – 2023. – Режим доступу: https://docs.python.org/3/whatsnew/3.12.html (дата звернення: 02.11.2025);

112. Real Python. Async IO in Python: A Complete Walkthrough [Електронний ресурс]. – 2024. – Режим доступу: https://realpython.com/async-io-python/ (дата звернення: 03.11.2025);

113. Python Speed Center. Python 3.12 Performance Benchmarks [Електронний ресурс]. – 2023. – Режим доступу: https://speed.python.org/ (дата звернення: 04.11.2025);

114. Pydantic Documentation. Why use Pydantic [Електронний ресурс]. – 2024. – Режим доступу: https://docs.pydantic.dev/latest/why/ (дата звернення: 05.11.2025);

115. FastAPI Documentation. Concurrency and async / await [Електронний ресурс]. – 2024. – Режим доступу: https://fastapi.tiangolo.com/async/ (дата звернення: 06.11.2025);

116. FastAPI Documentation. Automatic Interactive API Documentation [Електронний ресурс]. – 2024. – Режим доступу: https://fastapi.tiangolo.com/features/ (дата звернення: 27.10.2025);

117. Pydantic Documentation. Performance and Validation Speed [Електронний ресурс]. – 2024. – Режим доступу: https://docs.pydantic.dev/latest/concepts/performance/ (дата звернення: 28.10.2025);

118. FastAPI Documentation. Dependencies in path operation decorators [Електронний ресурс]. – 2024. – Режим доступу: https://fastapi.tiangolo.com/tutorial/dependencies/ (дата звернення: 29.10.2025);

119. Gradio Documentation. Image Editor Component [Електронний ресурс]. – 2024. – Режим доступу: https://www.gradio.app/docs/imageeditor (дата звернення: 30.10.2025);

120. Gradio Documentation. File Upload Component [Електронний ресурс]. – 2024. – Режим доступу: https://www.gradio.app/docs/file (дата звернення: 31.10.2025);

121. Gradio Documentation. Event Listeners [Електронний ресурс]. – 2024. – Режим доступу: https://www.gradio.app/docs/event-listeners (дата звернення: 01.11.2025);

122. Gradio Documentation. Chatbot Component [Електронний ресурс]. – 2024. – Режим доступу: https://www.gradio.app/docs/chatbot (дата звернення: 02.11.2025);

123. LangGraph Documentation. State Management [Електронний ресурс]. – 2024. – Режим доступу: https://langchain-ai.github.io/langgraph/concepts/low_level/#state (дата звернення: 03.11.2025);

124. LangGraph Documentation. Conditional Edges [Електронний ресурс]. – 2024. – Режим доступу: https://langchain-ai.github.io/langgraph/concepts/low_level/#conditional-edges (дата звернення: 04.11.2025);

125. LangGraph Documentation. Persistence [Електронний ресурс]. – 2024. – Режим доступу: https://langchain-ai.github.io/langgraph/concepts/persistence/ (дата звернення: 05.11.2025);

126. LangGraph Documentation. Human-in-the-loop [Електронний ресурс]. – 2024. – Режим доступу: https://langchain-ai.github.io/langgraph/concepts/human_in_the_loop/ (дата звернення: 06.11.2025);

127. Microsoft Azure. Azure OpenAI Service compliance and privacy [Електронний ресурс]. – 2024. – Режим доступу: https://learn.microsoft.com/en-us/azure/ai-services/openai/overview (дата звернення: 27.10.2025);

128. Microsoft Azure. Azure geographies and regions [Електронний ресурс]. – 2024. – Режим доступу: https://azure.microsoft.com/en-us/explore/global-infrastructure/geographies/ (дата звернення: 28.10.2025);

129. OpenAI Documentation. Function calling [Електронний ресурс]. – 2024. – Режим доступу: https://platform.openai.com/docs/guides/function-calling (дата звернення: 29.10.2025);

130. Microsoft Azure. Azure OpenAI Service pricing [Електронний ресурс]. – 2024. – Режим доступу: https://azure.microsoft.com/en-us/pricing/details/cognitive-services/openai-service/ (дата звернення: 30.10.2025);

131. SQLite Documentation. Appropriate Uses For SQLite [Електронний ресурс]. – 2024. – Режим доступу: https://www.sqlite.org/whentouse.html (дата звернення: 31.10.2025);

132. SQLAlchemy Documentation. Asynchronous I/O [Електронний ресурс]. – 2024. – Режим доступу: https://docs.sqlalchemy.org/en/20/orm/extensions/asyncio.html (дата звернення: 01.11.2025);

133. SQLite Documentation. Foreign Key Support [Електронний ресурс]. – 2024. – Режим доступу: https://www.sqlite.org/foreignkeys.html (дата звернення: 02.11.2025);

134. Martin Fowler. Repository Pattern [Електронний ресурс]. – 2024. – Режим доступу: https://martinfowler.com/eaaCatalog/repository.html (дата звернення: 03.11.2025);

135. Scikit-learn Documentation. Ensemble methods [Електронний ресурс]. – 2024. – Режим доступу: https://scikit-learn.org/stable/modules/ensemble.html (дата звернення: 04.11.2025);

136. XGBoost Documentation. Introduction to Boosted Trees [Електронний ресурс]. – 2024. – Режим доступу: https://xgboost.readthedocs.io/en/stable/tutorials/model.html (дата звернення: 05.11.2025);

137. Joblib Documentation. Persistence [Електронний ресурс]. – 2024. – Режим доступу: https://joblib.readthedocs.io/en/latest/persistence.html (дата звернення: 06.11.2025);

138. Python Documentation. Asyncio – Asynchronous I/O [Електронний ресурс]. – 2024. – Режим доступу: https://docs.python.org/3/library/asyncio.html (дата звернення: 27.10.2025);

139. SQLAlchemy Documentation. Asynchronous I/O Support [Електронний ресурс]. – 2024. – Режим доступу: https://docs.sqlalchemy.org/en/20/orm/extensions/asyncio.html (дата звернення: 28.10.2025);

140. OpenAI Python Library. Async usage [Електронний ресурс]. – 2024. – Режим доступу: https://github.com/openai/openai-python (дата звернення: 29.10.2025);

141. Real Python. Speed Up Your Python Program With Concurrency [Електронний ресурс]. – 2024. – Режим доступу: https://realpython.com/python-concurrency/ (дата звернення: 30.10.2025).

142. Договір про співпрацю №Д/0002.01/3400.02/5/2023 від 05 січня 2023 року між КПІ ім. Ігоря Сікорського та ДУ "Національний інститут фтизіатрії і пульмонології ім. Ф.Г. Яновського НАМН України". URL: https://dnvr.kpi.ua/wp-content/uploads/2023/01/%D0%9D%D0%B0%D1%86%D1%96%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%B8%D0%B9-%D1%96%D0%BD%D1%81%D1%82%D0%B8%D1%82%D1%83%D1%82-

%D1%84%D1%82%D0%B8%D0%B7%D1%96%D0%B0%D1%82%D1%80%D1%96%D1%97-%D1%96-%D0%BF%D1%83%D0%BB%D1%8C%D0%BC%D0%BE%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D1%96%D1%97-%D1%96%D0%BC.-%D0%AF%D0%BD%D0%BE%D0%B2%D1%81%D1%8C%D0%BA%D0%BE%D0%B3%D0%BE-%D0%9D%D0%90%D0%9C%D0%9D-%D0%A3%D0%BA%D1%80%D0%B0%D1%97%D0%BD%D0%B8.pdf (дата звернення: 11.11.2025).

143. Detailed diagram transformer neural network encoder self attention feed forward layers comprehensive illustrating [Електронний ресурс] / Dreamstime. – 2024. – Режим доступу: https://www.dreamstime.com/detailed-diagram-transformer-neural-network-encoder-self-attention-feed-forward-layers-comprehensive-illustrating-image320441152 (дата звернення: 14.12.2025).

144. Radiology study medical imaging technical division [Електронний ресурс] / Shutterstock. – 2024. – Режим доступу: https://www.shutterstock.com/image-vector/radiology-study-medical-imaging-technical-division-2369907109 (дата звернення: 14.12.2025).

145. Proyeksi WEF: Ini 6 Pekerjaan Teknologi yang Diprediksi Melejit Hingga 2030 [Електронний ресурс] / Halo Pedeka. – 2025. – Режим доступу: https://www.halopedeka.com/pendidikan/57616327158/proyeksi-wef-ini-6-pekerjaan-teknologi-yang-diprediksi-melejit-hingga-2030
(дата звернення: 14.12.2025).

146. Decision tree diagram digital age machine [Електронний ресурс] / Shutterstock. – 2024. – Режим доступу: https://www.shutterstock.com/ru/image-vector/decision-tree-diagram-digital-age-machine-2149843907 (дата звернення: 14.12.2025).

147. Machine learning technology decision tree scheme [Електронний ресурс] / Shutterstock. – 2024. – Режим доступу: https://www.shutterstock.com/ru/image-vector/machine-learning-technology-decision-tree-scheme-2199086379 (дата звернення: 14.12.2025).

148. TensorFlow Classifier Made Easy: Build Your First Neural Network in 10 Minutes [Електронний ресурс] / Dr. Ernesto Lee. – 2024. – Режим доступу: https://drlee.io/tensorflow-classifier-made-easy-build-your-first-neural-network-in-10-minutes-fb0f8b798edd (дата звернення: 14.12.2025).

149. Шапошник, Б. І. Метод та програмне забезпечення виявлення фінансових шахрайств засобами машинного навчання : магістерська дис. : 121 Інженерія програмного забезпечення / Шапошник Богдан Ігорович. – Київ, 2025. – 148 с.

150. Іваницький, О. В. Веб-додаток для зберігання DICOM-знімків : дипломна робота ... бакалавра : 122 Комп’ютері науки / Іваницький Олег Вікторович. – Київ, 2023. – 60 с.