НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

«На правах рукопису»
УДК _004. 9+004.8_

До захисту допущено:
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____2025 р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-професійною програмою
«Інтегровані інформаційні системи»

зі спеціальності 126 «Інформаційні системи та технології»

на тему: «Інтелектуальна система діагностики та рекомендацій щодо хвороб сільськогосподарських культур»

Виконав:
студент 2 курсу, групи ІА-41мп
Кальченко Єгор Олександрович _____

Керівник:
доцент каф. ІСТ, к.т.н., доц.
Богданова Наталія Володимирівна _____

Рецензент:
професор каф. ЕПС, ФЕЛ, д.т.н., проф.
Мельник Ігор Віталійович _____

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

Київ — 2025 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти — другий (магістерський)

Спеціальність — 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інтегровані інформаційні системи»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____2025 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Кальченку Єгору Олександровичу

1. Тема дисертації «Інтелектуальна система діагностики та рекомендацій щодо хвороб сільськогосподарських культур», науковий керівник дисертації Богданова Наталія Володимирівна, к.т.н., доц., затверджені наказом по університету від «06» 11 2025 р. № 4841-с

2. Термін подання студентом дисертації «15» 12 2025 р.

3. Об’єкт дослідження: процес інформаційної підтримки прийняття рішень при діагностуванні захворювань сільськогосподарських культур.

4. Вихідні дані: архітектура клієнт-сервер; підтримка мультимодальних вхідних даних; час обробки діагностичного запиту не більше 5 секунд; точність класифікації зображень не нижче 85%; забезпечення збереження історії запитів у базі даних; відсутність необхідності встановлення додаткового ПЗ на клієнтській стороні; інтеграція з хмарними сервісами AWS.

5. Перелік завдань, які потрібно розробити: провести аналіз існуючих рішень, спроєктувати архітектуру системи, реалізувати бек-енд та фронт-енд, провести тестування, підготувати текстову та графічну частину пояснювальної записки.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу: діаграма діяльності, функціональна структура системи, діаграма варіантів використання, діаграма послідовності, діаграма класів системи, діаграма компонентів системи, діаграма потоків даних, структурна схема системи в середовищі AWS.

7. Орієнтовний перелік публікацій: -

8. Дата видачі завдання 01.09.2025 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1.	Опис предметної області	2 вересня 2025	
2.	Аналіз готових рішень	4 вересня 2025	
3.	Формування вимог до системи	8 вересня 2025	
4.	Вибір технологій розробки	20 вересня 2025	
5.	Проєктування архітектури інформаційної системи	24 вересня 2025	
6.	Розробка інформаційної системи	24 жовтня 2025	
7.	Тестування і налагодження програми	10 листопада 2025	
8.	Виконання графічних документів	20 листопада 2025	

Студент Єгор КАЛЬЧЕНКО

Науковий керівник Наталія БОГДАНОВА

РЕФЕРАТ

Інформаційна система підтримки прийняття рішень в агротехнічній галузі: 142 с., 21 табл., 3 рис., 9 дод., 21 джерел.

ІНФОРМАЦІЙНА СИСТЕМА, ПІДТРИМКА ПРИЙНЯТТЯ РІШЕНЬ, АГРОТЕХНІЧНА ГАЛУЗЬ, ДІАГНОСТИКА ХВОРОБ РОСЛИН, КОМП'ЮТЕРНИЙ ЗІР, RAG, AWS, ВЕБЗАСТОСУНОК.

Актуальність теми зумовлена необхідністю забезпечення продовольчої безпеки та мінімізації втрат врожаю, що спричиняються хворобами сільськогосподарських культур. В умовах сучасного агровиробництва, особливо за дефіциту кваліфікованих кадрів, існує критична потреба в інструментах, що дозволяють швидко та точно ідентифікувати патогени безпосередньо в полі. Існуючі рішення часто працюють як «чорна скринька» або не враховують локальний контекст, що робить розробку прозорої, мультимодальної діагностичної системи своєчасною та доцільною.

Робота була виконана в рамках НДР "Інформаційні технології і системи підтримки прийняття рішень. Високопродуктивні комп'ютерні системи та мережі" (держ. реєстраційний номер 0124U002078).

Метою роботи є підвищення ефективності та точності діагностики хвороб сільськогосподарських культур шляхом створення прототипу інформаційної системи, що поєднує візуальний аналіз зображень із пошуком у базі знань. Для досягнення мети вирішено такі задачі: проведено аналіз предметної області та існуючих підходів до діагностики хвороб рослин; сформовано функціональні та нефункціональні вимоги до системи AgroDiag; спроєктовано архітектуру системи на основі клієнт-серверних принципів та хмарних технологій AWS; реалізовано 6-етапний діагностичний конвеєр, що включає комп'ютерний зір (CV) та RAG (Retrieval-Augmented Generation); розроблено програмний прототип та проведено його тестування на реальних даних.

Об'єкт дослідження — процес діагностики та прийняття рішень щодо захисту рослин в агротехнічній галузі.

Предмет дослідження — методи та засоби побудови інтелектуальних інформаційних систем для ідентифікації хвороб культур на основі мультимодальних даних (зображення та текст).

У роботі використано методи системного аналізу для формування вимог; методи об’єктно-орієнтованого проєктування (UML) для побудови архітектури; методи комп’ютерного зору (AWS Rekognition, Pillow) для аналізу зображень; метод RAG (Retrieval-Augmented Generation) та TF-IDF для пошуку в базі знань; методи інженерії програмного забезпечення для реалізації вебзастосунку.

Практичне значення одержаних результатів. Розроблено та впроваджено прототип системи AgroDiag, що підтримує діагностику 50 хвороб для 10 видів культур. Система дозволяє агрономам та фермерам отримувати оперативні рекомендації щодо лікування рослин, що сприяє зменшенню використання агрохімікатів та збереженню врожаю. Реалізовано гібридну архітектуру з можливістю роботи в локальному та хмарному режимах.

ABSTRACT

Information system for decision support in the agricultural sector: 142 p., 18 tab., 3 draw., 9 app., 21 sources.

INFORMATION SYSTEM, DECISION SUPPORT, AGRICULTURAL SECTOR, PLANT DISEASE DIAGNOSIS, COMPUTER VISION, RAG, AWS, WEB APPLICATION.

The relevance of the topic is determined by the need to ensure food security and minimize crop losses caused by diseases. In modern agricultural production, especially under conditions of limited qualified personnel, there is a critical need for tools that allow quick and accurate identification of pathogens directly in the field. Existing solutions often operate as a "black box" or lack local context, making the development of a transparent, multi-modal diagnostic system timely and appropriate.

The work was carried out within the framework of the research project "Information technologies and decision support systems". High-performance computer systems and networks" (state registration number 0124U002078).

The aim and tasks of the research. The aim of the work is to increase the efficiency and accuracy of crop disease diagnosis by creating an information system that combines visual image analysis with knowledge base retrieval. To achieve this aim, the following tasks were solved: analysis of the subject area and existing approaches to plant disease diagnosis was conducted; functional and non-functional requirements for the AgroDiag system were formulated; the system architecture was designed based on client-server principles and AWS cloud technologies; a 6-stage diagnostic pipeline including Computer Vision (CV) and RAG (Retrieval-Augmented Generation) was implemented; a software prototype was developed and tested on real data.

The object of research is the process of diagnosis and decision-making regarding plant protection in the agricultural sector.

The subject of research includes methods and tools for building intelligent information systems for crop disease identification based on multi-modal data (images and text).

The work uses system analysis methods for requirements formulation; object-oriented design methods (UML) for architecture construction; computer vision methods (AWS Rekognition, Pillow) for image analysis; RAG (Retrieval-Augmented Generation) and TF-IDF methods for knowledge base search; software engineering methods for web application implementation.

The practical value of the obtained results. The AgroDiag system prototype has been developed and implemented, supporting the diagnosis of 50 diseases for 10 crop types. The system allows agronomists and farmers to receive prompt recommendations for plant treatment, contributing to the reduction of agrochemical use and crop preservation. A hybrid architecture capable of operating in local and cloud modes has been implemented.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

БД — база даних

ЗЗР — засоби захисту рослин

ЗІЗ — засоби індивідуального захисту

ІС — інформаційна система

ІТ — інформаційні технології

ПЗ — програмне забезпечення

СУБД — система управління базами даних

API — Application Programming Interface — набір визначень, протоколів та інструментів для створення програмного забезпечення і взаємодії між різними компонентами системи

AWS — Amazon Web Services — дочірня компанія Amazon, що надає платформи хмарних обчислень та API на вимогу

CI/CD — Continuous Integration / Continuous Delivery — набір практик у розробці програмного забезпечення, що полягає у частій інтеграції змін коду та автоматизованому розгортанні

CNN — Convolutional Neural Network — клас нейронних мереж, що найчастіше використовується для аналізу візуальних зображень

CV — Computer Vision — галузь штучного інтелекту, що займається методами, які дозволяють комп'ютерам отримувати високорівневе розуміння з цифрових зображень або відео

ECS — Elastic Container Service — високопродуктивний сервіс оркестрації контейнерів, що підтримує Docker-контейнери

GPS — Global Positioning System — глобальна супутникова радіонавігаційна система, що дозволяє визначати місцезнаходження та час у будь-якій точці планети

HTTP — HyperText Transfer Protocol — протокол прикладного рівня для передачі даних, який є основою обміну інформацією в Інтернеті

JSON — JavaScript Object Notation — текстовий формат обміну даними, що базується на підмножині мови програмування JavaScript, але є незалежним від мови

LLM — Large Language Model — тип моделі штучного інтелекту, навченої на величезних масивах текстових даних для розуміння та генерації людської мови

MVP — Minimum Viable Product — версія продукту з мінімальним набором функцій, достатнім для задоволення перших споживачів та збору зворотного зв'язку

ORM — Object-Relational Mapping — технологія програмування, яка дозволяє перетворювати дані між несумісними системами типів у об'єктно-орієнтованих мовах програмування

RAG — Retrieval-Augmented Generation — техніка в обробці природної мови, яка поєднує генеративні моделі з пошуком інформації в зовнішніх джерелах для підвищення точності відповідей

RDS — Relational Database Service — вебсервіс, що спрощує налаштування, роботу та масштабування реляційної бази даних у хмарі

REST — Representational State Transfer — архітектурний стиль взаємодії компонентів розподіленого додатка в мережі, що використовує стандартні методи протоколу HTTP

S3 — Simple Storage Service — сервіс хмарного об'єктного сховища, що надається Amazon Web Services

SaaS — Software as a Service — модель надання програмного забезпечення, при якій постачальник розробляє веб-застосунок і надає замовнику доступ до нього через Інтернет

SDK — Software Development Kit — комплект засобів розробки, який дозволяє фахівцям створювати додатки для певного пакета програмного забезпечення або платформи

SQL — Structured Query Language — декларативна мова програмування для взаємодії з реляційними базами даних

UI — User Interface — сукупність засобів, за допомогою яких користувач взаємодіє з програмою або пристроєм

UML — Unified Modeling Language — уніфікована мова моделювання, що використовується для опису, візуалізації та документування об'єктно-орієнтованих систем

YAML — YAML Ain't Markup Language — зручний для читання людиною формат серіалізації даних, що часто використовується для файлів конфігурації

ВСТУП

Сільське господарство України функціонує в умовах високої невизначеності, спричиненої кліматичними змінами, економічним тиском та війною, де кожна помилка в діагностиці хвороб рослин загрожує втратою врожаю та зайвими витратами. Польова ідентифікація патогенів є складним процесом через неоднозначність симптомів, вплив супутніх стресових факторів та часто низьку якість вихідних даних, таких як аматорські фотографії чи неповні описи. Існуючі інструменти — від повільних паперових довідників до сучасних застосунків на базі комп'ютерного зору — не повністю закривають потребу аграріїв: перші є застарілими, а другі часто працюють як «чорна скринька», надаючи результат без пояснень та без урахування локальної специфіки чи історії поля.

Вирішенням проблеми може стати впровадження інтелектуальних систем підтримки прийняття рішень, які аналізують не лише зображення, а й текстові описи симптомів та контекст конкретного господарства. Мета таких систем — не повна автоматизація, а допомога агроному у структуруванні інформації, формуванні обґрунтованого переліку ймовірних загроз та поясненні логіки діагнозу. Такий підхід дозволить поєднати переваги цифрових технологій з реальними потребами виробництва, забезпечуючи фахівців інструментом для швидкого, але зваженого реагування на біотичні та абіотичні виклики.

У цій магістерській роботі розглядається підхід до побудови інтелектуальної системи діагностики та рекомендацій щодо хвороб сільськогосподарських культур, орієнтованої на реалістичний, прототипний рівень реалізації. Система приймає від користувача інформацію про культуру та текстовий опис симптомів, за потреби доповнений фотографією, обробляє ці дані в межах послідовного програмного конвеєра і повертає користувачеві ранжований перелік гіпотез щодо можливих хвороб з короткими поясненнями. Центральним елементом запропонованого підходу є формування для кожного звернення окремої «папки кейсу», де фіксуються вхідні дані, проміжні результати, часові мітки та остаточна відповідь. Навіть за обмеженої функціональності прототипу це дає можливість аналізувати,

як саме система прийшла до певного висновку, переглядати історію звернень і в подальшому розширювати базу знань реальними кейсами.

Актуальність теми магістерської роботи визначається поєднанням трьох факторів. По-перше, зростає потреба аграрних виробників у швидких, але водночас контрольованих інструментах попередньої діагностики, які можна використовувати без глибоких спеціальних знань. По-друге, наявні рішення не завжди придатні для безпосереднього застосування в українських умовах та не орієнтовані на прозорість і відтворюваність діагностичного процесу. По-третє, розвиток технологій веб-сервісів, обробки текстових даних і баз знань дозволяє реалізувати прототипи таких систем у доступних технологічних стосах, що робить дослідження не лише теоретично, а й практично значущим для майбутнього впровадження.

Метою магістерської дисертації є підвищення ефективності та точності діагностики хвороб сільськогосподарських культур шляхом створення прототипу інформаційної системи, що поєднує візуальний аналіз зображень із пошуком у базі знань, який підтримує інтерпретацію симптомів хвороб сільськогосподарських культур на основі текстових описів і простих візуальних ознак, взаємодії з внутрішньою базою знань та надання користувачеві пояснюваних рекомендацій щодо можливих подальших дій.

Для досягнення цієї мети в роботі потрібно виконати наступні завдання:

— виконати предметний аналіз предметної області та особливостей польової діагностики хвороб культур, наявних цифрових рішень і сервісів діагностики з точки зору їх можливостей і обмежень;

— сформувати вимоги до системи, враховуючи потреби кінцевих користувачів та технічні обмеження;

— розробити концепцію бази знань і конвеєра оброблення запиту, включно з етапами валідації, нормалізації й пошуку релевантних записів;

— реалізувати серверну частину прототипу, яка підтримує базові сценарії роботи системи;

— виконати початкову перевірку працездатності прототипу на тестових прикладах та окреслення напрямів подальшого розвитку і можливостей комерційного використання результатів у форматі стартап-проєкту.

Об’єктом дослідження є процеси комп’ютеризованої підтримки діагностики хвороб сільськогосподарських культур у виробничих умовах.

Предметом дослідження є архітектурні та алгоритмічні рішення, пов’язані з організацією бази знань про хвороби культур, обробленням текстових описів і простих візуальних ознак, формуванням гіпотез та поданням рекомендацій користувачеві таким чином, щоб забезпечити зрозумілість, відтворюваність і можливість аудиту.

Методологічну основу роботи становлять аналіз і синтез наукових та прикладних джерел з фітопатології, інформаційних систем та інтелектуальних технологій, порівняльний аналіз існуючих програмних продуктів для діагностики хвороб рослин, методи структурного та об’єктно-орієнтованого проєктування програмних систем, а також практичні підходи до побудови веб-сервісів і роботи з базами знань. Для оцінки працездатності запропонованих рішень використовується підхід прототипування, коли реалізується мінімально необхідний набір функцій системи та проводиться перевірка на обмеженій множині сценаріїв, характерних для реальних звернень користувачів.

Наукова новизна роботи полягає в удосконаленні методу первинної діагностики хвороб рослин. Запропоновано комплексний алгоритм, який поєднує аналіз текстових скарг агронома, виявлення ключових візуальних ознак та перевірку за спеціалізованою базою знань. Також розроблено структуру уніфікованої картки запиту, яка зберігає не лише результат діагностики, а й увесь шлях пошуку рішення. Це дозволяє зробити процес прозорим, аналізувати логіку системи та накопичувати дані для її навчання. Результати роботи орієнтовані на практичне застосування та створення робочого прототипу.

Практичне значення одержаних результатів полягає в можливості використання розробленого прототипу системи як допоміжного інструмента для агрономів, консультантів і фахівців з інформаційних технологій в аграрній сфері.

Навіть обмежена версія системи, що працює з фіксованою кількістю культур і хвороб, дає змогу структурувати звернення, зберігати історію випадків, зменшувати кількість хаотичних або надмірних обробок і формувати у користувачів звичку до більш системного підходу в описі симптомів. Крім того, розроблений прототип може бути використаний як навчальний приклад під час підготовки студентів за спеціальністю «Інформаційні системи та технології» при вивченні дисциплін, пов’язаних із проєктуванням та впровадженням галузевих інформаційних систем.

1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Загальний опис предметної області

Рослинництво залишається фундаментом національної економіки України, хоча й функціонує як система з високою вразливістю до зовнішнього тиску. На результативність виробництва безпосередньо впливають кліматичні зміни, коливання режимів зволоження, поширення нових патогенів та різноманітні організаційні перешкоди. До останніх належать логістичні збої, дефіцит ресурсів та кадрові складнощі, які загострилися в умовах воєнного стану. Така сукупність обставин створює середовище значної невизначеності, де агрономічні рішення мусять бути водночас оперативними та обґрунтованими. Саме тому своєчасна і точна діагностика захворювань сільськогосподарських культур перетворюється на критичний елемент управління врожайністю, економічною стабільністю підприємств та безпекою виробленої продукції [1].

Предметна область цієї роботи охоплює весь шлях від виявлення перших ознак ураження на полі до затвердження плану захисних дій. У виробничих умовах агроном чи інший працівник спершу помічає візуальні відхилення у розвитку рослин, як-от зміни забарвлення, плями, наліт, деформації або відмирання тканин. Наступним кроком стає висунення попереднього припущення щодо походження проблеми. Фахівець визначає природу явища та з'ясовує, чи викликана вона живими організмами на кшталт грибків і шкідників, чи стала наслідком неживих факторів на зразок температурних стресів або хімічних опіків. Для фіналізації висновку спостережувані ознаки необхідно порівняти з авторитетними джерелами інформації, що містять описи хвороб, специфіку їхнього прояву та рекомендації щодо лікування [2].

Ключовими дійовими особами в цьому процесі виступають агрономи, зовнішні консультанти, керівники підприємств та працівники, відповідальні за моніторинг полів. Для них визначення хвороби не є самоціллю, а слугує складовою загальної стратегії управління посівами, яка включає планування обробок та оцінку фінансових ризиків. Інформаційна підтримка в такому випадку повинна виходити

за межі простої ідентифікації патогена. Вона має допомагати інтегрувати отриманий діагноз у специфіку конкретного поля, враховуючи технологічну карту та наявні ресурсні можливості господарства [2].

Характерною особливістю цієї сфери є значне розмаїття культур, сортів та застосовуваних технологій вирощування. Прояви однієї й тієї ж хвороби можуть суттєво відрізнятися на різних рослинах або змінюватися залежно від фази розвитку, типу ґрунту та погодних умов попереднього періоду. Часто ідентичні візуальні ознаки в різних ситуаціях свідчать про абсолютно різні проблеми або мають неоднакову діагностичну вагу. Це зумовлює сильну залежність діагностики від контексту, адже ігнорування відомостей про культуру, стадію вегетації та історію поля значно підвищує ймовірність помилкових висновків [2].

Інформаційна основа діагностики традиційно формується з друкованих довідників, методичних розробок, результатів наукових дослідів та експертного досвіду. Нині цей перелік доповнюється цифровими ресурсами, мобільними додатками та корпоративними базами даних агропідприємств. Ефективність застосування цих знань залежить не лише від їхньої наукової достовірності, а й від швидкості доступу до необхідної інформації безпосередньо в полі. В умовах дефіциту часу та розсіяної уваги саме оперативність отримання даних стає вирішальним фактором [2].

Окремої уваги потребує питання якості вхідних даних, які зазвичай фіксуються через короткі текстові повідомлення, фотографії зі смартфонів або голосові нотатки. Така інформація часто страждає від неповноти, коли відсутні дані про сорт, фазу росту чи попередні агротехнічні заходи. Фотографії нерідко виконуються при поганому освітленні, без дотримання масштабу або з фокусуванням на сторонніх об'єктах. Ці фактори посилюють невизначеність і вимагають використання інструментів, які б стимулювали користувача надавати повний набір ознак та забезпечували їх правильне тлумачення[3, 5, 6].

Задачі діагностики нерозривно пов'язані з плануванням захисту рослин, управлінням ризиками та контролем економічної ефективності виробництва. Помилковий або запізнілий висновок загрожує прямими втратами врожаю та

непрямими збитками через марне витрачання препаратів чи порушення екологічних стандартів. Водночас тактика надмірного перестрахування без належних підстав також вважається шкідливою. Вона призводить до невиправданого зростання витрат і сприяє виникненню стійкості у збудників хвороб до хімічних засобів [3, 6].

Зазначені особливості дозволяють визначити предметну область як комплексне поєднання агротехнологій, фітопатології та процедур прийняття управлінських рішень. Саме на перетині цих напрямів виникає нагальна потреба в інтелектуальних інформаційних системах. Такі рішення повинні допомагати в інтерпретації симптомів, структурувати знання про хвороби та гарантувати прозорість діагностичного процесу. У наступних частинах роботи цей загальний опис буде розширено через деталізацію етапів діагностики та формулювання вимог до програмного забезпечення.

1.2 Процес діагностики хвороб сільськогосподарських культур

Процес діагностики хвороб сільськогосподарських культур у реальних виробничих умовах являє собою послідовність етапів, що охоплюють спостереження, формулювання гіпотез, перевірку цих гіпотез на основі наявних знань, вибір заходів захисту та подальший моніторинг результатів. На відміну від лабораторної діагностики, де доступні розширені методи аналізу (мікроскопія, ПЛР, серологічні тести тощо), польова діагностика спирається переважно на візуальні ознаки, контекстні фактори та досвід фахівця. Це зумовлює високу роль суб’єктивних оцінок і підвищує значення будь-яких інструментів, які допомагають структуровано організувати процес прийняття рішень [3, 4].

Вихідною точкою діагностичного процесу є спостереження симптомів ураження на рослинах. Симптоми можуть бути помічені під час планових оглядів посівів, при вибірковому огляді проблемних ділянок або за зверненням персоналу, який працює безпосередньо в полі. Найчастіше фіксуються зміни забарвлення листків, поява плям різної форми й інтенсивності, наліт, некрози, деформації

органів, відставання в рості, в’янення окремих рослин або сегментів поля. Додатково враховується просторовий характер ураження (поодинокі рослини, смуги, плями, суцільні ділянки), що може вказувати на певні групи причин. На цьому етапі агроном проводить первинне групування ознак: відносить їх до ймовірних біотичних або абіотичних факторів, відзначає, які симптоми є основними, а які — супутні [2]. На рисунку 1.1 представлено приклад польового огляду посівів з фіксацією симптомів у фоторежимі.

Рисунок 1.1 — Приклад польового огляду посівів з фіксацією симптомів на фото

Паралельно з візуальним оглядом формується контекст діагностичного випадку. До нього належать відомості про культуру та сорт, фазу вегетації, попередні культури у сівозміні, історію поля, останні агротехнічні операції (посів, обробіток ґрунту, підживлення, обприскування), погодні умови останніх днів (опади, температурні коливання, приморозки, тривалі посухи). Частина цих даних може бути відразу доступною агроному, частина — витребуваною у відповідальних за виконання робіт чи з виробничих журналів [2, 4, 5]. По суті, на цьому етапі формується «каркас» майбутнього діагностичного кейсу: від того, наскільки повно та структуровано зібрані контекстні дані, залежить якість подальшої інтерпретації.

Наступний крок — формулювання попередніх діагностичних гіпотез. Спираючись на власний досвід, відомості з довідників та методичних матеріалів, а також на результати консультацій, агроном співвідносить спостережувані симптоми з типовими проявами відомих хвороб, шкідників або абіотичних стресів [2, 5]. Зазвичай формується не одна, а декілька альтернативних гіпотез; при цьому одразу оцінюється їх правдоподібність з урахуванням контексту. Наприклад, певна хвороба може бути характерною для іншої фази розвитку культури або не відповідати спостережуваним погодним умовам; у такому разі її гіпотетична ймовірність знижується. У традиційному підході цей етап зазвичай відбувається в голові фахівця і рідко формалізується у вигляді явного переліку варіантів, що ускладнює подальший аналіз якості рішень.

Після формування початкового набору гіпотез відбувається їх перевірка й уточнення на основі наявних джерел знань. На практиці це означає роботу з друкованими довідниками, електронними каталогами, внутрішніми інструкціями господарства, а також звернення по консультацію до фітопатологів чи більш досвідчених колег. На цьому етапі відбувається звіряння переліку симптомів, характеру їх розвитку, просторового розподілу, швидкості прогресування хвороби з описами в джерелах. Важливу роль відіграють так звані «диференційні ознаки» — деталі, які дають змогу відрізнити схожі за проявами хвороби чи стресові фактори. Так, наприклад, характер краю плями, наявність ореолу, колір і структура нальоту, послідовність зміни забарвлення тканин можуть слугувати підставою для відмежування бактеріальних та грибних хвороб або відокремлення інфекційного процесу від хімічного опіку [2, 5].

Одночасно з уточненням діагнозу проводиться оцінка масштабу та динаміки проблеми. Агроном визначає, чи є ураження локальним чи загрозливим для значної частини посіву, чи прогресують симптоми з часу їх першого виявлення, чи впливають вони на ключові органи (листковий апарат, генеративні органи, стебло). Може бути прийнято рішення про додаткові обстеження ділянки, відбір зразків для лабораторної діагностики, організацію повторного огляду через певний час. Таким чином, процес діагностики набуває ітеративного характеру: на основі нових

спостережень гіпотези уточнюються, частина з них відкидається, частина — підтверджується як найбільш імовірна [2].

Ключовим завершальним етапом є трансформація діагностичного висновку у конкретне управлінське рішення: вибір заходів захисту або відмову від втручання з відповідним обґрунтуванням. На основі найбільш правдоподібної гіпотези (або декількох гіпотез) підбираються схеми обробок, які враховують зареєстровані на ринку препарати, їхню ефективність проти конкретних патогенів, спектр дії, строки очікування, обмеження щодо культури, фазу розвитку та регуляторні вимоги щодо безпеки. Важливо при цьому зважати не лише на біологічні аспекти, а й на економічну доцільність: прогнозовану шкоду від невтручання, вартість препаратів, витрати на виконання робіт. Нерідко ухвалюється рішення про «спостережне очікування», якщо рівень загрози оцінюється як низький або невизначений [4].

У традиційній практиці значна частина цих кроків залишається неформалізованою. Часто фіксується лише кінцевий факт обробки (дата, препарат, норма внесення) в журналах чи електронних системах, тоді як історія формування діагнозу, перелік розглянутих гіпотез, мотиви вибору саме цієї схеми захисту не документуються. Це ускладнює подальший аналіз ефективності рішень, розбір помилок, навчання нових працівників і побудову внутрішньої системи управління якістю. Фактично втрачається «цифровий слід» діагностичного процесу, який є критично важливим для побудови інтелектуальних інформаційних систем, що орієнтуються на пояснюваність і відтворюваність [3].

При цьому, якщо розглядати процес діагностики з точки зору інженерії інформаційних систем, він природно розкладається на низку логічних етапів, які можуть бути відображені у вигляді конвеєра. Першим етапом є валідація вхідних даних: перевірка того, чи задано культуру, чи достатньо описано симптоми, чи є, за потреби, фотографії прийнятної якості. Наступним кроком є нормалізація опису — перетворення неформальних записів у більш структурований вигляд, де симптоми, культури, фаза розвитку, погодні явища представлені у погодженій системі термінів і позначень. Далі здійснюється зіставлення отриманого опису з наявною базою знань: пошук карток хвороб і стресових факторів, які відповідають

набору ознак, контексту культури та умовам розвитку. Паралельно або на наступному етапі застосовуються доменні правила та евристики, які дозволяють звузити перелік гіпотез, посилити або послабити певні варіанти, врахувати локальні обмеження господарства та регуляторні вимоги [3, 4].

Результатом такого «алгоритмічного бачення» процесу діагностики є ранжований перелік гіпотез щодо можливих хвороб чи стресів, доповнений поясненнями: які симптоми й ознаки зробили ту чи іншу гіпотезу ймовірною, які умови є критичними для застосування рекомендованих заходів, які додаткові спостереження бажано зробити для зменшення невизначеності. На практиці така подача результату ближча до того, як мислить сам агроном: він не очікує «абсолютно правильного» діагнозу, а потребує структурованого набору варіантів із зрозумілими аргументами «за» і «проти».

Важливою складовою процесу діагностики є робота з невизначеністю. У польових умовах рідко вдається зібрати повний набір ознак, який однозначно відповідав би певній хворобі. Часто маємо справу з частковою інформацією, афтерами після попередніх обробок, ситуаціями, коли частина симптомів уже зникла або трансформувалася. Тому діагностичний процес має враховувати не лише наявність певних ознак, а й невизначеність щодо їх достовірності та повноти. З погляду інформаційної підтримки це означає необхідність працювати з імовірнісними оцінками, інтервалами впевненості, чітко фіксувати рівень припущення (наприклад, «попередня гіпотеза», «найбільш ймовірна», «можливий альтернативний варіант») і не подавати результат як безумовний факт [5].

Ще один вимір процесу діагностики — його інтегрованість у ширший контур управління господарством. Діагностичні висновки взаємодіють з операційними планами (графіки обробок, наявність техніки, завантаженість персоналу), фінансовими обмеженнями, контрактними зобов’язаннями щодо якості та строків постачання продукції. Тому реальні рішення іноді відхиляються від «ідеальної» агротехнічної схеми: наприклад, обирається менш оптимальний із точки зору біології препарат, але доступний і економічно прийнятний. З погляду інформаційної системи важливо не підміняти собою управлінські рішення, а

прозоро показувати діагностичні аргументи, можливі наслідки різних сценаріїв і тим самим підтримувати відповідальних осіб у прийнятті зважених компромісів [1, 4].

Нарешті, процес діагностики не завершується на етапі виконання обробки. Важливою його частиною є постфактум-аналіз: спостереження за динамікою симптомів після втручання, оцінка ефективності заходів, фіксація уроків, які можуть бути корисними в наступних сезонах. У ідеалі кожен діагностичний випадок перетворюється на «кейс» із чітко описаними вхідними даними, гіпотезами, прийнятими рішеннями та результатами. Накопичення таких кейсів дозволяє господарству формувати власну базу знань, адаптовану до локальних умов, і підвищувати якість рішень з року в рік. Саме така логіка роботи з кейсами відповідає підходу, покладеному в основу магістерської роботи: кожне звернення до системи не лише дає відповідь користувачу, а й збагачує інформаційну базу для подальшого вдосконалення діагностичного процесу [4].

1.3 Джерела та якість вхідних даних

У контексті діагностики хвороб рослин «вхідні дані» являють собою мультимодальну сукупність: текстові описи, зображення та метадані. На практиці ключовими є текстові повідомлення та фотографії. Текстові описи зазвичай лаконічні й неформальні, але саме вони відображають реальну комунікацію в господарстві. Фотографії, зроблені смартфонами у польових умовах, часто мають технічні недоліки: погане освітлення, відсутність фокусу чи масштабу. Метадані ж нерідко надходять неповними через недбалість або вимоги конфіденційності (відсутність геолокації) [4, 6].

Якість вхідних даних визначається їх повнотою, точністю фактів, узгодженістю термінології та технічними параметрами знімків. Оскільки система працюватиме з реальними, а не ідеальними описами, вона повинна містити механізми «м’якого дисциплінування» користувача: підказки щодо необхідного

мінімуму інформації та інструкції з фотографування. Це дозволяє стандартизувати вхідний потік без надмірного ускладнення роботи агронома.

У межах розробки системи пропонується зберігати всі дані у структурі «папки кейсу», що включає запит, оригінальні фото, витягнуті ознаки та результати діагностики. Це дозволяє формувати базу для подальшого аналізу та вдосконалення алгоритмів. Водночас враховуються етичні обмеження, а саме чутливі дані підлягають мінімізації та захисту. Таким чином, система має бути адаптованою до обробки неповних і різнорідних даних, забезпечуючи при цьому надійність і безпеку [4, 6].

Висновки до розділу 1

У першому розділі проаналізовано специфіку діагностики хвороб рослин в Україні, яка здійснюється в умовах високої невизначеності та ресурсних обмежень. Встановлено, що нинішній процес прийняття рішень є переважно неформалізованим: фіксуються лише кінцеві заходи захисту, тоді як логіка діагностики та альтернативні гіпотези не документуються [3]. Це унеможливлює аналіз ефективності рішень і формування «організаційної пам’яті». Ситуацію ускладнює природа вхідних даних — це мультимодальна суміш суб’єктивних описів, неякісних фото та фрагментарних метаданих, які важко інтерпретувати без врахування контексту конкретного поля [6].

Виявлені проблеми — зокрема контекстна залежність симптомів, неоднорідність даних та відсутність єдиного стандарту опису кейсів — обґрунтовують потребу в нових інформаційних інструментах. Традиційні засоби не забезпечують необхідної оперативності та цифровізації досвіду. Відтак, виникає необхідність розробки інтелектуальної системи, здатної структурувати діагностичний процес, працювати з неповною інформацією та накопичувати локальну базу знань, що стане предметом подальшої розробки.

2 АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

У попередньому розділі було показано, що процес польової діагностики хвороб культур є багатокроковим, залежним від якості спостережень і потребує опори на узгоджені знання. Природно постає питання, які цифрові інструменти вже пропонуються ринку для підтримки цього процесу, яким чином вони працюють із польовими даними та які обмеження залишаються невирішеними.

Для детального дослідження було обрано три типові платформи, що демонструють різні підходи до автоматизації: масовий додаток Plantix, спеціалізований польовий асистент PlantVillage Nuru та інтеграційний сервіс Plant.id. Порівняльний аналіз цих рішень за показниками точності, прозорості алгоритмів та стабільності роботи дозволяє виявити функціональні обмеження сучасного ринку. Саме на усунення цих недоліків і спрямована система, що розробляється в межах цієї роботи.

2.1 Мобільний застосунок Plantix

Plantix (PEAT / HELM AG) належить до найпоширеніших мобільних застосунків для діагностики хвороб культур за зображенням і фактично сформував еталон сценарію «кроп-доктора у смартфоні». У публічних описах його позиціонують як масовий інструмент для фермерів та агрономів, який дає змогу зафіксувати симптоми на рослині за допомогою камери телефону і впродовж кількох секунд отримати попередній діагноз та перелік можливих дій. Декларується підтримка сотень комбінацій «культура–симптом», інтеграція з рекомендаціями щодо захисту та широке охоплення регіонів вирощування, що дозволяє розглядати Plantix як один із наймасштабніших проєктів у своїй ніші [7].

З продуктового погляду Plantix побудований як B2C-рішення, орієнтоване насамперед на кінцевого користувача в полі. Застосунок доступний на платформах Android та iOS; офіційні сторінки наголошують на простоті сценарію використання: користувач фотографує уражену ділянку рослини, надсилає знімок і

отримує набір гіпотез з коротким текстовим поясненням та порадами щодо захисту. Додатковими елементами екосистеми виступають довідкова бібліотека з описами хвороб, сповіщення та допоміжні утиліти (калькулятори живлення, рекомендації з агротехніки), завдяки чому Plantix позиціонується не лише як класифікатор зображень, а як оперативний довідково-консультаційний портал для виробника [7].

Типовий сценарій взаємодії користувача з Plantix можна описати як послідовність кроків «зняв — надіслав — отримав відповідь». Під час польового огляду фермер або агроном запускає застосунок, наводить камеру смартфона на уражений листок чи ділянку рослини, фіксує зображення та підтверджує його відправлення. Далі знімок передається до хмарної інфраструктури, де відбувається оброблення: модель комп'ютерного зору аналізує фото, формує перелік імовірних класів ушкоджень, після чого клієнтський застосунок відображає для користувача одну або кілька гіпотез, супроводжених короткими описами симптомів і пропозиціями щодо подальших дій. За потреби користувач може перейти до розгорнутого довідкового матеріалу, переглянути додаткові ілюстрації, новини або попередні звернення.

Хоча внутрішні технічні деталі Plantix не розкриваються повністю, з відкритих описів і наявних наукових оглядів можна реконструювати загальний підхід. Ядром системи є моделі глибокого навчання для аналізу зображень (мережі класу CNN), навчені на великому корпусі анотованих фотографій хворих і здорових рослин. Ці моделі повертають імовірнісний розподіл по класах ушкоджень, який далі поєднується з інформацією з внутрішньої довідкової бази. На рівні довідника для кожної хвороби або шкідника зберігаються опис симптомів, умови розвитку, рекомендовані заходи, застереження й приклади зображень. Таким чином, користувач фактично отримує не «сирий» вихід моделі, а вже інтерпретовану відповідь, зв'язану з конкретними рекомендаціями [7, 8].

Структурно сервіс можна уявити як багаторівневу систему, де мобільний клієнт відповідає за збір даних та відображення результатів, а основне обчислювальне навантаження переноситься до хмарної частини. На клієнтському рівні реалізовано інтерфейс камери, передобробку зображень, базову валідацію

(наприклад, перевірка якості знімка, орієнтації), авторизацію користувача й доступ до локальних елементів довідника. Серверна частина включає модулі приймання зображень, зберігання їх у сховищі, компонент розпізнавання симптомів, довідковий модуль і механізми персоналізації порад (з урахуванням регіону та культури). Хоча детальну архітектуру розробники не оприлюднюють, наявність масштабного користувацького базису та глобального покриття робить природним припущення про розподілену, мікросервісну організацію, яка дозволяє масштабувати оброблення запитів і оновлення моделей без перерви сервісу.

Важливою віхою розвитку екосистеми Plantix стало залучення інституційного партнера: у 2023 році компанія HELM AG оголосила про придбання контрольної частки в проєкті. У публічних повідомленнях наголошувалося на високій популярності застосунку в окремих регіонах (зокрема, в Індії) та на планах використати глобальну збутову мережу HELM для поширення цифрових рішень, що мають зменшити зловживання засобами захисту за рахунок більш точних рекомендацій. Після цієї угоди Plantix дедалі більше описують не як окремий застосунок, а як елемент цифрової платформи агробізнесу, де функції діагностики, довідкових матеріалів та взаємодії з ринком поєднуються в єдиному інформаційному просторі [9].

Незважаючи на усі переваги, поточне рішення має ряд недоліків:

— інтерфейс працює за принципом «чорної скриньки», не надаючи користувачеві інформації про те, які саме візуальні ознаки вплинули на класифікацію та за яких умов рекомендація є дійсною;

— алгоритми демонструють нестабільність при обробці реальних польових знімків зі складним фоном чи освітленням, які суттєво відрізняються від ідеалізованих даних навчальної вибірки;

— відсутність збереження повної історії діагностичного процесу, включно з проміжними гіпотезами та контекстом, унеможливлює глибокий аналіз помилок та оцінку логіки прийнятих рішень.

Таким чином, одного боку, це реальний, масово застосовуваний інструмент, який демонструє практичну цінність швидкого зменшення невизначеності за

рахунок аналізу зображень і служить природною точкою порівняння для будь-якої нової системи діагностики. З іншого боку, обмеження Plantix окреслюють ніші, в яких є сенс розвивати альтернативні підходи: пояснюваний конвеєр оброблення запиту, робота з мультимодальними даними, явне формування «папок кейсів» із фіксацією усіх кроків діагностики та можливістю внутрішнього аудиту. Саме на закриття цих прогалин орієнтується запропонована в магістерській дисертації система [7, 8].

2.2 Мобільний застосунок PlantVillage Nuru

PlantVillage Nuru (FAO & Penn State) належить до класу мобільних польових асистентів, спеціально орієнтованих на дрібних та середніх виробників у регіонах із нестабільним або дорогим доступом до інтернету. Ключова продуктова обіцянка цього рішення полягає в тому, що діагностику можна виконувати без підключення до мережі на звичайному смартфоні: користувач наводить камеру на листок, застосунок локально оцінює наявність характерних пошкоджень і формує попередній висновок разом із короткими підказками щодо подальших дій. Такий підхід принципово відрізняється від переважно хмарних сервісів, де аналіз зображення виконується на віддалених серверах і потребує стабільного каналу зв’язку [10].

У типовому сценарії використання Nuru виконує роль «інтерактивної інструкції» для польового огляду. Застосунок пропонує користувачеві послідовно обрати культуру та локальну мову, після чого проводить через серію простих кроків: наведення камери на кілька листків, перевірка якості знімка, підтвердження виявлених симптомів. Аналіз зображення відбувається безпосередньо на смартфоні; результати подаються у вигляді попереднього діагнозу (тип хвороби або шкідника) та коротких рекомендацій, доповнених голосовими або текстовими підказками. Така організація процесу особливо цінна для віддалених господарств, де доступ до фахових консультацій обмежений, а зв’язок нестабільний [10].

З архітектурної точки зору Nuru можна розглядати як офлайн-орієнтований клієнтський застосунок, у якому основні моделі комп’ютерного зору розгорнуті безпосередньо на пристрої користувача. Офіційні описи підкреслюють використання моделей детекції об’єктів, які дають змогу не лише класифікувати кадр загалом, а й виявляти окремі ділянки листка з підозрілими симптомами. Це дозволяє будувати більш наочні підказки («зверніть увагу на ці плями/деформації»), що, у свою чергу, підсилює навчальний ефект від роботи з інструментом. Хмарна інфраструктура при цьому використовується переважно для завантаження оновлень моделей, синхронізації агрегованих даних і доступу до розширених довідкових матеріалів, але базові діагностичні обчислення не залежать від постійного підключення [10, 11].

Особливою рисою PlantVillage Nuru є наявність опублікованих академічних оцінок його роботи для конкретних культур. Дослідження, представлені в журналі Frontiers in Plant Science та пов’язаних матеріалах, порівнювали спроможність Nuru розпізнавати симптоми вірусних хвороб касави та пошкодження кліщами з результатами різних груп користувачів (дослідники, консультанти, фермери). Показано, що якість діагностики суттєво зростає за умови дотримання стандартизованого протоколу огляду: коли користувач послідовно перевіряє шість листків на рослині, досягаються діапазони точності близько 74–88% для низки завдань. Крім того, зафіксовано навчальний ефект: уже короткий період практичного використання застосунку підвищує здатність польових працівників самостійно розпізнавати симптоми, навіть без прямої участі експертів [11].

Користувацький досвід у Nuru формується поєднанням трьох компонентів. По-перше, це короткий, але чітко регламентований сценарій огляду, коли застосунок крок за кроком підказує, які листки слід фотографувати і як саме, фактично дисциплінуючи процес збору даних. По-друге, локальне отримання результату з мінімальною затримкою, що дозволяє приймати попередні рішення без витрат часу на передачу даних і очікування відповіді сервера. По-третє, виразна навчальна складова: через багаторазову взаємодію з підказками користувач формує

стабільні уявлення про характерні ознаки хвороб і відмінності між схожими симптомами [11].

У ширшому контексті ініціатив CGIAR/FAO Nuru позиціонують як «розмовного» польового помічника, що не лише класифікує зображення, а й проводить користувача через методику огляду посівів і надає доступ до пов'язаних інформаційних модулів. У описах підкреслюється, що смартфон, «озброєний» Nuru, може поєднувати локальне розпізнавання симптомів із доступом до навчальних матеріалів, рекомендацій та, у відповідних конфігураціях, до даних дистанційного моніторингу (на кшталт WaPOR). Це зміщує акцент із «мобільної нейромережі» на елемент польової методики: інструмент, який допомагає стандартизувати спостереження і приймати типові рішення на місці, коли інтернет або експертна консультація недоступні.

Незважаючи на чисельні переваги даного продукту на ринку, можна виділити наступний перелік ключових характеристик, за якими він поступається своїм аналогам:

— офлайн-архітектура суттєво обмежує кількість підтримуваних культур і симптомів, а розширення цієї бази потребує ресурсномістких кампаній зі збору нових польових даних;

— точність діагностики критично залежить від суворого дотримання протоколу зйомки, оскільки ігнорування користувачем підказок щодо ракурсу чи освітлення різко підвищує ймовірність помилки;

— пояснюваність рішень є недостатньою для формального аудиту через відсутність прозорого логічного ланцюжка, який би пов'язував виявлену візуальну ознаку з правилом прийняття рішення.

Таким чином, PlantVillage Nuru розглядається як еталонний приклад офлайн-орієнтованого польового асистента. З одного боку, він демонструє, як стандартизація огляду й інтеграція навчальних елементів можуть підвищувати відтворюваність діагностики й рівень компетентності користувачів. З іншого боку, його обмеження — вузьке предметне поле, залежність від дотримання протоколу та недостатня прозорість обґрунтувань — підкреслюють важливість побудови

систем, у яких візуальний аналіз є лише однією з ланок керованого конвеєра, узгодженого з базою знань і доменними правилами та забезпеченого повноцінним аудиторським слідом [10, 11].

2.3 Агломерація Plant.id

Під брендами Plant.id та Plant.health компанія Kindwise (раніше FlowerChecker) пропонує низку хмарних сервісів для автоматизованого розпізнавання рослин та діагностики їхнього стану за зображеннями. На відміну від масових мобільних застосунків, орієнтованих безпосередньо на фермера, ці продукти позиціонуються насамперед як комерційний API для інтеграції в сторонні веб- та мобільні системи, а також у рішення класу «смарт-агро» й прецизійного землеробства [13].

Базовим продуктом виступає Plant.id API, який за одним або кількома зображеннями визначає вид рослини, а також може оцінювати її загальний стан. У публічній документації зазначається підтримка десятків тисяч таксонів (понад 35 тис. видів та сортів), що охоплюють як дикорослі, так і культурні рослини. Для доступу до функціональності розпізнавання розробник отримує ключ доступу і викликає REST-ендпоїнти сервісу, передаючи закодовані зображення та додаткові параметри (наприклад, бажану мову описів або запит на оцінку здоров’я рослини) [12, 13].

Комплементарний продукт Plant.health спеціалізується на діагностиці хвороб, шкідників та абіотичних стресів за фото. За заявленими характеристиками, модель охоплює понад 500 класів уражень, включно з грибними, бактеріальними й вірусними захворюваннями, шкідниками, абіотичними розладами та «нешкідливими двійниками» (статуси, що імітують хворобу, але не потребують втручання). Для складних випадків передбачено режими, за яких в одному запиті поєднуються результати Plant.id та Plant.health; у відповідь повертається ранжований список можливих діагнозів із ймовірностями та коротким текстовим описом і базовими рекомендаціями щодо дій [13].

З погляду сценарію використання Plant.id / Plant.health типово не взаємодіє безпосередньо з фермером. Кінцевий користувач працює з власним мобільним застосунком чи веб-порталом агрокомпанії, де реалізовано інтерфейс зйомки та перегляду результатів. Після того як користувач фотографує листок або іншу частину рослини й підтверджує відправлення, бекенд інтеграційної системи формує запит до API, отримує набір гіпотез та їх імовірності й уже на своєму боці вирішує, як представити цю інформацію, які поради додати, чи зберігати кейс у внутрішній базі. Таким чином, Plant.id/Plant.health виступає «вбудованим» класифікаційним модулем, відповідальним за візуальне розпізнавання, тоді як побудова повного конвеєра підтримки прийняття рішень покладається на інтегратора [13, 14].

Для непрофесійних користувачів компанія надає демо-застосунок у вигляді веб-інтерфейсу, реалізованого як прогресивний веб-додаток. Цей інструмент дозволяє завантажити обмежену кількість зображень на тиждень і отримати приклад того самого результату, який повертає API: список можливих видів або хвороб, супровідні описи, зображення для звірки. Мета демо — не стільки створення окремого масового продукту, скільки наочна демонстрація можливостей сервісу для потенційних замовників і розробників, а також джерело зворотного зв’язку для покращення моделей [14].

Важливою складовою екосистеми є також рішення Plant.id Sensor, призначене для вбудовування алгоритмів аналізу зображень у бортові системи сільськогосподарської техніки. У публічному описі йдеться про використання камерних масивів та обчислювального модуля в кабіні машини для реального часу визначення бур’янів, підрахунку рослин і оцінки стану посівів з подальшим формуванням карт для систем точкового внесення. Це розширює сферу застосування технологій Plant.id від «точкових» фото, зроблених користувачем, до потокових даних з польових сенсорів і машинного зору в агро-роботизованих комплексах.

З технічного та наукового погляду Plant.id / Plant.health опирається на глибокі моделі комп’ютерного зору, навчені на великих масивах анотованих зображень. У

корпоративних матеріалах наголошується, що розмітка даних для діагностики хвороб здійснюється за участю фахових фітопатологів, а якість моделей регулярно оцінюється в академічних і прикладних дослідженнях, де Plant.id часто демонструє вищу точність порівняно з альтернативними безкоштовними застосунками для розпізнавання видів [14].

З погляду інтеграційного потенціалу це рішення має низку суттєвих переваг: зрілу хмарну інфраструктуру, масштабованість «за запитами», наявність SDK-прикладів та документації для різних мов програмування, підтримку кількох мов для описів хвороб і рекомендацій, а також чітку модель ліцензування, що дозволяє використовувати сервіс як платну компоненту в комерційних продуктах. Завдяки цьому Plant.id / Plant.health природно вписується в архітектуру B2B та B2B2C-рішень, де необхідно швидко додати функцію розпізнавання рослин і їхніх хвороб без розроблення власних моделей [13].

Серед недоліків даного рішення варто підкреслити наступні:

— робота сервісу в режимі «чорної скриньки» не дозволяє отримати доступ до проміжних правил та структури ознак, що ускладнює використання результатів у формалізованих процедурах аудиту;

— обробка ізольованих зображень без урахування агротехнічного контексту та метеоданих перекладає завдання побудови повноцінного діагностичного кейсу виключно на сторону інтегратора;

— залежність від зовнішньої хмари створює ризики для суверенності даних, вимагає постійного зв’язку та збільшує експлуатаційні витрати, що обмежує застосування системи в умовах суворих безпекових вимог.

У сукупності ці чинники визначають Plant.id / Plant.health як потужний, але доволі вузько сфокусований компонент, який доцільно розглядати не як самостійну систему підтримки прийняття рішень, а як один з можливих модулів у ширшому пояснюваному конвеєрі діагностики хвороб культур.

2.4 Порівняльний аналіз існуючих систем

Цифрові рішення Plantix, PlantVillage Nuru та Plant.id / Plant.health репрезентують три різні підходи до організації діагностики хвороб сільськогосподарських культур. Plantix демонструє модель масового мобільного застосунку для фермерів, орієнтованого на швидке отримання попереднього діагнозу та базових рекомендацій. PlantVillage Nuru є прикладом офлайн-орієнтованого польового асистента з чітко регламентованим протоколом огляду, розробленого насамперед для дрібних виробників у регіонах зі слабкою інфраструктурою зв'язку. Plant.id / Plant.health, своєю чергою, уособлює модель хмарного сервісу та API, який надає функціональність розпізнавання як компонент для вбудовування в інші системи [15].

Для коректного порівняння доцільно розглядати ці системи за низкою узагальнених критеріїв:

— тип рішення та цільовий користувач;

— предметне охоплення;

— режим роботи;

— пояснюваність результатів та робота з невизначеністю;

— інтеграційний потенціал;

— можливості фіксації та подальшого аналізу кейсів.

Узагальнені результати за порівняльним аналізом всіх трьох існуючих рішень наведено в таблиці 2.1.

Таблиця 2.1 — Порівняльна характеристика існуючих систем

Характеристика	Plantix	PlantVillage Nuru	Plant.id / Plant.health
Тип рішення, цільовий користувач	масовий мобільний застосунок, орієнтований на фермерів та агрономів	мобільний польовий офлайн-асистент для фермерів	хмарний сервіс та api, орієнтований на розробників, агрокомпанії та інтеграторів

Характеристика	Plantix	PlantVillage Nuru	Plant.id / Plant.health
Предметне охоплення	широкий перелік культур та комбінацій «культура–симптом»	спершу — окремі культури (насамперед касава, кукурудза), поступове розширення	велика кількість видів рослин, спеціалізовані моделі для оцінки стану та хвороб
Основний вхідний сигнал	окреме фото ураженої ділянки, іноді з мінімальним текстовим контекстом	серія фото за стандартизованим протоколом огляду	одна або кілька фотографій, за потреби — з додатковими параметрами запиту
Режим роботи	аналіз на сервері, потрібне стабільне підключення до інтернету	діагностика виконується локально на смартфоні, інтернет потрібен лише епізодично	хмарна обробка, повна залежність від мережевого з’єднання
Пояснюваність та робота з невизначеністю	кілька гіпотез з короткими текстовими поясненнями, без формалізованих правил	діагноз та поради доповнюються навчальними підказками, але без явної формалізації правил	ранжований список гіпотез з імовірностями та описами, внутрішні моделі недоступні

Характеристика	Plantix	PlantVillage Nuru	Plant.id / Plant.health
Інтеграційний потенціал	переважно самодостатній застосунок; можливі B2B-розширення	самостійний інструмент; для корпоративних сценаріїв потрібна зовнішня надбудова	спроєктований саме як вбудовуваний компонент у ширші інформаційні системи
Робота з кейсами та історичністю	історія звернень на рівні кінцевого користувача, обмежені можливості структурованої аналітики	локальні історії використання, наголос на навчальному ефекті	повна історія запитів формується на боці інтегратора, сервіс повертає лише результати API

Порівняльний аналіз трьох репрезентативних рішень — масового хмарного застосунку Plantix, спеціалізованого офлайн-асистента PlantVillage Nuru та інтеграційного API-сервісу Plant.id — засвідчив, що попри успішне впровадження комп’ютерного зору, існуючі інструменти мають суттєві обмеження для професійного використання. Вони різняться архітектурою та цільовими моделями, проте жоден із них не забезпечує комплексної роботи з глибоким контекстом (історія обробок, метеодані), не пропонує прозорого для аудиту ланцюга прийняття рішень («ознака — правило — висновок») і не підтримує формування структурованих «папок кейсів» на рівні господарства [14]. Відсутність інструменту, який би поєднував візуальну діагностику з формалізованими доменними знаннями та механізмами накопичення досвіду, обґрунтовує необхідність розробки власної системи, орієнтованої на усунення цих прогалин.

Висновки до розділу 2

Аналіз ринку цифрових рішень для діагностики хвороб рослин дозволив виділити три основні класи продуктів: масові мобільні застосунки (Plantix), спеціалізовані офлайн-асистенти (PlantVillage Nuru) та інтеграційні API-сервіси (Plant.id). Усі вони ефективно використовують моделі комп'ютерного зору для розпізнавання симптомів, проте мають спільні системні обмеження. Головними недоліками є робота за принципом «чорної скриньки», висока чутливість до якості зображень та ігнорування глибокого агротехнічного контексту (історії поля, погодних умов, попередніх обробок), що ускладнює формальне пояснення та верифікацію діагнозів у виробничих умовах.

Крім того, існуючі інструменти не забезпечують належної організації діагностичних кейсів: історія звернень часто є поверхневою або її зберігання повністю покладається на сторонніх розробників. Це створює розрив між можливостями технологій та потребами агровиробництва, де необхідний єдиний простір для поєднання візуальних даних, текстових описів та доменних правил. Саме потреба у створенні пояснюваного конвеєра обробки запитів та структурованого сховища для накопичення й аудиту рішень визначає вимоги до системи, що розробляється в наступних розділах.

3 **ФОРМУВАННЯ ВИМОГ ДО СИСТЕМИ**

На основі попереднього аналізу недоліків існуючих рішень, які часто функціонують як «чорні скриньки», у цьому розділі формалізуються вимоги до нової інформаційної системи діагностики хвороб культур. Визначаються ключові функціональні характеристики, необхідні для роботи з мультимодальними та неповними даними, а також проєктується структура бази знань і сховища діагностичних кейсів. Основний акцент зроблено на забезпеченні прозорості конвеєра прийняття рішень, можливості подальшого аудиту результатів та інтеграції із зовнішніми сервісами комп'ютерного зору й мовних моделей для використання у реальних виробничих умовах [16].

3.1 Постановка задачі

З технічної точки зору ставиться задача побудови програмного комплексу, що реалізує декілька послідовних стадій оброблення запиту. На першому етапі виконується перевірка повноти та коректності вхідних даних, а також їх нормалізація до внутрішнього формату. Далі застосовуються базові методи комп'ютерного зору для витягування візуальних ознак зі зображень, після чого здійснюється пошук релевантних записів у внутрішній базі знань про хвороби та шкідників з урахуванням культури, симптоматики та контекстних параметрів. Наступний етап передбачає застосування доменних правил, які уточнюють або відкидають окремі гіпотези на основі агротехнічних обмежень, сезонності, характеру уражень. За необхідності результати можуть додатково уточнюватися за допомогою зовнішньої мовної моделі, після чого формується підсумкова відповідь.

Важливою складовою задачі є забезпечення відтворюваності діагностичних рішень та можливості їх подальшого аудиту. Для кожного звернення система має формувати структуровану «папку кейсу», що включає вихідні вхідні дані, проміжні результати роботи окремих стадій конвеєра, прийняті гіпотези, обраний підсумковий діагноз і сформовані рекомендації. Такі кейси повинні зберігатися у

сховищі, яке дає змогу відновити повний хід діагностичного процесу, аналізувати типові помилки, відстежувати історію звернень по полях і культурах, а також використовувати накопичений масив даних для уточнення бази знань і налаштування доменних правил.

Окремою вимогою постановки задачі є гнучкість архітектури щодо способів розгортання та використання зовнішніх сервісів. Система має орієнтуватися на роботу у вигляді вебзастосунку з користувацьким інтерфейсом та REST-API, які можуть розгортатися як локально в інфраструктурі господарства, так і в хмарному середовищі. Окремі стадії діагностичного конвеєра мають підтримувати можливість переключення між вбудованими реалізаціями (наприклад, спрощені методи аналізу зображень, локальний пошук у базі знань) та інтеграцією з зовнішніми сервісами комп'ютерного зору чи мовними моделями. Таке формулювання задачі дозволяє надалі конкретизувати функціональні та нефункціональні вимоги до системи, а також вимоги до структури бази знань і сховища діагностичних кейсів.

3.2 Функціональні вимоги

Функціональні вимоги визначають, які саме дії має виконувати система діагностики хвороб сільськогосподарських культур та які результати надавати користувачеві й пов'язаним підсистемам. Вони формуються з урахуванням постановки задачі, особливостей предметної області, аналізу існуючих рішень і цільової архітектури, у якій ключову роль відіграє діагностичний конвеєр та структуроване сховище діагностичних кейсів. Деталізація цих вимог дозволяє забезпечити повноту охоплення інформаційних запитів користувачів та встановити чіткі межі операційної діяльності системи в межах обраної предметної області. Викладені параметри формують логічну структуру взаємодії між окремими модулями, що є необхідним для підтримання високої достовірності отримуваних результатів діагностики. Узагальнений перелік основних функціональних вимог наведено в таблиці 3.1.

Таблиця 3.1 — Функціональні вимоги до системи діагностики хвороб культур

Назва функціональної вимоги	Короткий опис
Приймання запиту на діагностику	Система повинна забезпечувати приймання запитів на діагностику від користувача через зручний інтерфейс. Запит має містити щонайменше вказівку культури та текстовий опис симптомів, а також за можливості — зображення уражених рослин та додаткові метадані.
Перевірка та нормалізація вхідних даних	Система повинна виконувати перевірку повноти та коректності вхідних даних, контролювати типи та розміри файлів, обмежувати кількість зображень, а також нормалізувати відомості про культуру, симптоми й метадані до внутрішнього формату, придатного для подальшої обробки.
Оброблення запиту діагностичним конвеєром	Кожен запит повинен послідовно оброблятися у межах багатостадійного конвеєра, що включає виділення візуальних ознак із зображень, аналіз текстового опису симптомів, пошук релевантних записів у базі знань, застосування доменних правил та формування підсумкових гіпотез.
Формування діагностичного результату	Система повинна формувати для користувача структурований результат діагностики у вигляді ранжованого переліку можливих хвороб (чи інших причин ураження), для кожної з яких надається коротке текстове обґрунтування (основні симптоми, які підтримують гіпотезу) та базові рекомендації щодо подальших дій.

Назва функціональної вимоги	Короткий опис
Формування та збереження діагностичного кейсу	За підсумками оброблення кожного запиту система повинна створювати «папку кейсу», що містить вхідні дані, проміжні результати роботи окремих стадій конвеєра, обрані гіпотези та фінальний діагностичний висновок. Ця інформація повинна зберігатися у структурованому сховищі для подальшого перегляду й аналізу.
Отримання кейсу за ідентифікатором	Система повинна забезпечувати можливість отримання повної інформації про раніше виконаний діагностичний випадок за його унікальним ідентифікатором. Користувач або уповноважена особа має мати змогу переглянути як підсумковий результат, так і ключові вхідні дані та проміжні кроки конвеєра.
Перегляд та фільтрація історії звернень	Система повинна надавати засоби отримання переліку діагностичних кейсів із можливістю фільтрації за часом, культурою, типом хвороби чи іншими узагальненими критеріями. Це необхідно для виконання внутрішнього аудиту, аналізу повторюваних проблем та подальшого вдосконалення бази знань.
Управління базою знань	Система повинна підтримувати завантаження, оновлення та валідацію бази знань про хвороби сільськогосподарських культур та пов’язані агротехнічні рекомендації. Для кожної одиниці знань мають бути визначені обов’язкові поля.

Назва функціональної вимоги	Короткий опис
Інтеграція із зовнішніми сервісами аналізу	Система повинна передбачати можливість інтеграції з зовнішніми сервісами аналізу зображень та/або мовними моделями для підсилення окремих стадій конвеєра. Такі інтеграції мають розглядатися як опційні й не повинні порушувати роботу основних функцій у разі їх відсутності.
Засоби контролю працездатності	Система повинна надавати базові засоби контролю власної працездатності та доступності основних компонентів, що є необхідним для експлуатації у виробничому середовищі.
Підтримка користувацького інтерфейсу	Система повинна включати інтерфейс для кінцевого користувача, який забезпечує введення запиту, запуск діагностики, перегляд результатів і, за потреби, перегляд попередніх звернень. Інтерфейс має відображати діагностичну інформацію у зрозумілій формі та коректно відпрацьовувати можливі помилки під час оброблення запитів.

Сформульований перелік охоплює функції, без реалізації яких система діагностики хвороб культур не може забезпечити ні базову роботу з окремими зверненнями, ні накопичення та подальший аналіз діагностичних кейсів. У наступних підрозділах ці функціональні вимоги доповнюються нефункціональними характеристиками та вимогами до структури бази знань і сховища даних.

3.3 Нефункціональні вимоги

Нефункціональні вимоги визначають якісні характеристики системи діагностики хвороб сільськогосподарських культур, що не пов’язані безпосередньо з конкретними функціями, але є критичними для її практичного використання. До таких характеристик належать, зокрема, продуктивність, надійність, масштабованість, безпека даних, зручність використання, підтримуваність, спостережуваність та пояснюваність результатів. Формування цих вимог здійснюється з урахуванням умов експлуатації в агровиробництві, особливостей вхідних даних та ролі системи як елемента більш широкої інформаційної інфраструктури господарства. Узагальнений перелік основних нефункціональних вимог наведено в таблиці 3.2.

Таблиця 3.2 — Нефункціональні вимоги до системи діагностики

Назва нефункціональної вимоги	Короткий опис
Продуктивність та час відгуку	Система повинна формувати діагностичний результат за час, прийнятний для інтерактивної роботи користувача.
Масштабованість	Архітектура має допускати збільшення кількості користувачів, обсягу бази знань і кейсів без суттєвої втрати продуктивності.
Надійність та відмовостійкість	Система повинна залишатися працездатною при часткових збоях компонентів і коректно обробляти помилки без втрати даних.
Цілісність та збереженість даних	Повинна гарантуватися цілісність бази знань і сховища кейсів, узгодженість структурованих даних та пов’язаних зображень.

Назва нефункціональної вимоги	Короткий опис
Безпека та конфіденційність	Необхідно забезпечити захист діагностичних даних від несанкціонованого доступу та дотримання політик конфіденційності.
Пояснюваність та відтворюваність	Результати діагностики мають супроводжуватися стислими поясненнями, а хід міркування — бути відтворюваним для кожного кейсу.
Зручність використання	Інтерфейс повинен бути зрозумілим для фахівців аграрного профілю та підтримувати прості сценарії введення і перегляду даних.
Спостережуваність та журналювання	Система має накопичувати журнали подій і помилок, необхідні для аналізу роботи та виявлення вузьких місць.
Підтримуваність та розширюваність	Внутрішня структура повинна спрощувати оновлення бази знань, конвеєра діагностики та схеми даних без радикальної переробки.
Портативність та варіативність розгортання	Система має підтримувати розгортання в різних середовищах з мінімальними змінами конфігурації та залежностей.

3.4 Вимоги до бази знань та сховища діагностичних

База знань про хвороби сільськогосподарських культур і структуроване сховище діагностичних кейсів є центральними компонентами системи, від яких залежить якість діагностики, можливість аудиту та подальшого вдосконалення конвеєра. Вимоги до цих компонентів пов’язані як із змістом і структурою записів, так і з їх взаємозв’язком, відтворюваністю та підтримкою аналізу в часі.

Узагальнені вимоги до бази знань та сховища діагностичних кейсів наведено в таблиці 3.3.

Таблиця 3.3 — Вимоги до бази знань та діагностичних кейсів

Назва вимоги	Короткий опис
Структурованість записів бази знань	Записи бази знань повинні мати формалізовану структуру з чітко визначеними полями, придатну для автоматизованої обробки.
Повнота та однозначність опису хвороб	Для кожної хвороби мають бути наведені ключові симптоми, типові умови розвитку та обмеження застосування рекомендацій, що зменшує неоднозначність під час діагностики.
Зв’язок із культурами та фазами розвитку	Записи бази знань повинні явно вказувати, до яких культур і фаз розвитку вони застосовні, що дозволяє виключати нерелевантні гіпотези на ранніх етапах конвеєра.
Структурування рекомендацій	Рекомендації мають бути поділені на групи, що полегшує їх подальше використання у плані дій.
Підтримка багатомовного текстового подання	База знань повинна підтримувати можливість використання принаймні української та, за потреби, додаткових мов для назв хвороб, описів симптомів і рекомендацій.
Структуроване подання діагностичного кейсу	Кожен діагностичний кейс має зберігатися як цілісний об’єкт з унікальним ідентифікатором, відомостями про культуру, час створення, вхідними даними, переліком кандидатів і сформованим планом дій.
Зв’язок кейсів із зображеннями та метаданими	Сховище повинно забезпечувати однозначний зв’язок між кейсом і пов’язаними зображеннями, а також зберігання базових метаданих про знімки.

Назва вимоги	Короткий опис
Збереження проміжних результатів конвеєра	Для підтримки відтворюваності діагностики мають зберігатися ключові проміжні результати.
Підтримка пошуку та фільтрації кейсів	Сховище діагностичних кейсів повинно забезпечувати можливість пошуку та фільтрації за основними атрибутами для проведення аналізу й аудиту.

Висновки до розділу 3

У третьому розділі формалізовано вимоги до інформаційної системи, яка має забезпечувати багатостадійну обробку мультимодальних даних та формування обґрунтованих діагностичних гіпотез. Ключовою функціональною особливістю визначено створення для кожного запиту структурованої «папки кейсу», що фіксує вхідні дані, проміжні результати конвеєра та фінальні рекомендації. Окреслено базові можливості системи: від приймання запиту та управління базою знань до збереження історії звернень і опційної інтеграції із зовнішніми сервісами аналізу, що гарантує цілісність процесу діагностики.

Особливий акцент зроблено на нефункціональних вимогах, серед яких пріоритетними є пояснюваність та відтворюваність рішень, що відрізняє систему від «чорних скриньок» і дозволяє проводити внутрішній аудит на основі накопиченого досвіду. Ядром предметної моделі виступають формалізована база знань та сховище кейсів, які забезпечують прив’язку симптомів до контексту конкретних культур. Сформульований комплекс вимог створює фундамент для проєктування архітектури, розробки діагностичного конвеєра та реалізації прототипу в наступних розділах роботи.

4 ВИБІР АРХІТЕКТУРНОГО ПІДХОДУ ТА ТЕХНОЛОГІЙ РЕАЛІЗАЦІЇ

На основі сформульованих у попередньому розділі вимог здійснюється перехід до проєктування архітектури системи, яка має забезпечити реалізацію інтерактивної взаємодії, багатостадійного діагностичного конвеєра та інтеграцію зовнішніх аналітичних сервісів при збереженні можливості накопичення історичних даних. Головним завданням на цьому етапі є вибір таких структурних рішень, що поєднують простоту розгортання прототипу з чітким розділенням відповідальності компонентів, дозволяючи системі масштабуватися та гнучко розвиватися без докорінної перебудови в майбутньому.

4.1 Вибір архітектурного підходу та загальна характеристика архітектури

Для побудови веборієнтованих інформаційних систем у сучасній практиці застосовуються декілька основних архітектурних підходів. Їх доцільно розглядати у трьох взаємопов’язаних вимірах: логічна організація системи, спосіб розгортання та характер мережевої взаємодії.

З погляду логічної організації одним із базових підходів є шарова архітектура. У цьому випадку система поділяється на послідовні шари з чітко визначеними зонами відповідальності: презентаційний шар, що відповідає за взаємодію з користувачем, прикладний шар призначений для реалізації предметної логіки, шар доступу до даних та шар зберігання даних. Такий поділ сприяє дотриманню принципу розділення відповідальностей і полегшує супровід, оскільки зміни в одному шарі, за умови стабільності інтерфейсів, мінімально впливають на інші. Для систем, що опрацьовують складні предметні сценарії та працюють із різними типами даних (текст, зображення, структуровані записи), шарова архітектура є природним вибором [18].

За способом розгортання та ступенем зв’язності компонентів традиційно розрізняють монолітні та розподілені (зокрема, мікросервісні) варіанти побудови.

Монолітна архітектура передбачає, що серверна частина системи реалізується як єдиний застосунок, у межах якого інтегруються функції кількох шарів. Такий підхід спрощує початкову розробку, налагодження та розгортання, оскільки використовується одне програмне оточення, єдина схема версіонування та єдина точка конфігурації. Водночас із зростанням масштабів системи та кількості розробників моноліт може ускладнювати локальне масштабування окремих підсистем і незалежне оновлення окремих функціональних блоків.

Мікросервісний підхід, навпаки, передбачає поділ системи на набір дрібніших сервісів, кожен з яких реалізує обмежений піднабір функцій і розгортається окремо. Така архітектура полегшує незалежний розвиток компонентів, дозволяє масштабувати найнавантаженіші частини системи та підвищує стійкість до відмов окремих сервісів. Однак ці переваги досягаються ціною суттєвого ускладнення інфраструктури, що зумовлює необхідність у централізованому моніторингу, керуванні конфігураціями, а також в організації журналювання та трасування розподілених транзакцій. Для прототипів і систем із порівняно помірним навантаженням це ускладнення не завжди є виправданим.

Окремим виміром є мережевий спосіб взаємодії між компонентами. Для вебзастосунків базовою моделлю є клієнт-серверна архітектура, у межах якої клієнтські компоненти — браузер, окремий користувацький застосунок формують запити до серверної частини, що централізовано опрацьовує ці запити, звертається до сховищ даних і повертає результати. Водночас клієнт-серверна модель не суперечить ні монолітному, ні мікросервісному підходу: вона описує спосіб взаємодії між фронтендом і бекендом, тоді як монолітність або мікросервісність визначає внутрішню організацію саме серверної частини.

З урахуванням попередньо визначених вимог, а саме: необхідності інтерактивної роботи, пояснюваної діагностики, збереження «папок кейсів», підтримки інтеграції з зовнішніми сервісами та відносно обмежених ресурсів для розгортання, доцільним є поєднання таких рішень:

— використання багатошарової архітектури як логічної основи побудови системи;

— реалізація серверної частини у вигляді логічно монолітного застосунку, що спрощує розгортання та супровід прототипу;

— застосування клієнт-серверної моделі взаємодії між користувацьким інтерфейсом і серверною частиною;

— передбачення точок інтеграції з зовнішніми сервісами аналізу зображень, мовними моделями та хмарними сховищами як окремих підключених компонентів.

У межах обраного підходу система діагностики хвороб культур розглядається як клієнт-серверний вебзастосунок, серверна частина якого реалізує багатошаровий підхід із виділенням презентаційного, прикладного, бізнесового та шару даних. Користувач працює з системою через вебінтерфейс, що взаємодіє з серверною частиною за протоколом «запит–відповідь».

У серверній частині логіка діагностики структурована у вигляді конвеєра, який послідовно виконує валідацію вхідних даних, аналіз зображень, пошук у базі знань, застосування доменних правил, формування пояснень і збереження діагностичного кейсу.

Шар даних, своєю чергою, включає базу знань, сховище кейсів та сховище зображень, що можуть реалізовуватися у різних конфігураціях залежно від середовища розгортання.

У наступних підрозділах на основі цієї загальної схеми деталізуються структура та робота діагностичного конвеєра, модель даних і організація сховища кейсів, а також варіанти інтеграції із зовнішніми сервісами та середовищами розгортання.

4.2 Вибір технологічного стеку та середовища реалізації системи

Таким чином, сформульовані у розділі 3 функціональні та нефункціональні вимоги задають низку обмежень до технологій, за допомогою яких реалізується архітектура, описана у підрозділі 4.1. Зокрема, система повинна підтримувати роботу з текстовими описами та зображеннями, надавати вебінтерфейс та програмний інтерфейс для інтеграції, забезпечувати збереження діагностичних

кейсів і бази знань, а також потенційно взаємодіяти з хмарними сервісами. У цьому підрозділі розглядається добір конкретних технологій для реалізації цих вимог: мови програмування та серверного фреймворка, засобів побудови користувацького інтерфейсу, технологій зберігання даних, бібліотек для оброблення зображень і тексту та інфраструктурних рішень для локального й хмарного розгортання.

4.2.1 Вибір мови програмування та серверного фреймворка

Вибір технологічного стеку є критичним етапом, що визначає здатність системи до роботи з мультимодальними даними та інтеграції з алгоритмами штучного інтелекту. У результаті порівняльного аналізу з платформами Java та .NET, які є стандартом для корпоративних рішень, перевагу було надано мові Python. Вона має найбільш розвинуту екосистему бібліотек для обробки зображень і текстів, що дозволяє реалізувати діагностичний конвеєр, серверну логіку та засоби аналізу даних у єдиному середовищі, уникаючи зайвої фрагментації технологій.

При виборі серверного фреймворка розглядалися різні підходи: від повнофункціональних рішень до мінімалістичних мікрофреймворків. Оскільки архітектура системи передбачає відокремлений клієнтський вебзастосунок, використання «важких» фреймворків із вбудованими шаблонізаторами було визнано недоцільним. Водночас класичні мікрофреймворки вимагали б значних ручних зусиль для налаштування валідації даних та документування інтерфейсів, що не відповідає вимогам швидкого прототипування.

Оптимальним вибором для реалізації API став фреймворк FastAPI, що базується на сучасних стандартах анотації типів Python. Він забезпечує декларативний опис структур даних, автоматичну валідацію вхідних запитів та генерацію документації у стандарті OpenAPI. Це дозволяє визначити моделі діагностичних кейсів у єдиному місці й гарантувати їх узгоджене використання на всіх етапах обробки без дублювання коду [19].

Важливою технічною перевагою FastAPI є підтримка асинхронної моделі обробки запитів, що є критичним для системи, яка виконує операції, що

потребують багато ресурсів, з зображеннями та звертається до зовнішніх сервісів. Такий підхід дозволяє ефективно масштабувати навантаження, не блокуючи виконання інших завдань. Обраний стек органічно вписується в шарову архітектуру системи, де API слугує надійним фасадом для складної бізнес-логіки діагностичного конвеєра.

4.2.2 Технології реалізації користувацького інтерфейсу

Користувацький інтерфейс системи проєктувався з огляду на потреби агрономів, для яких пріоритетними є простота введення даних та наочність результатів. Оскільки архітектура передбачає відокремлення презентаційного шару, серед розглянутих варіантів (включно зі складними JS-фреймворками та настільними застосунками) було обрано фреймворк Streamlit. Він дозволяє розробляти повноцінні вебклієнти засобами мови Python, що уніфікує технологічний стек із серверною частиною та суттєво пришвидшує процес прототипування, дозволяючи вносити зміни в інтерфейс без залучення фронтенд-розробників.

Архітектурно застосунок на Streamlit функціонує як незалежний клієнт, що взаємодіє з бекендом FastAPI за протоколом REST. Це забезпечує чітке розмежування відповідальності: інтерфейс формує запит із фото та описом симптомів, а сервер повертає оброблений результат. Такий підхід дозволяє відображати не лише фінальний діагноз, а й метадані кейсу, що сприяє пояснюваності рішень. Крім того, робота через API спрощує розгортання та залишає можливість легкої заміни клієнтського застосунку в майбутньому без втручання в логіку ядра системи.

4.2.3 Підсистема зберігання даних та база даних

Підсистема зберігання даних у системі діагностики хвороб сільськогосподарських культур відіграє центральну роль, оскільки забезпечує

довгострокове накопичення знань та діагностичних кейсів, необхідних як для повсякденної експлуатації, так і для подальшого аналізу якості роботи системи. У межах рівня даних можна виокремити три основні категорії інформації: базу знань про хвороби культур, діагностичні кейси (включно з вхідними параметрами, проміжними результатами та фінальними відповідями) та зображення, що супроводжують запити діагностики. Для кожної з цих категорій обрано окремі підходи до зберігання, які разом утворюють цілісну архітектуру даних.

База знань реалізована у вигляді набору файлів формату YAML, організованих за культурами у відповідній деревоподібній структурі каталогів. Кожен файл містить формалізований опис однієї хвороби: назву, перелік підтримуваних культур, діапазон фаз розвитку, за яких хвороба є релевантною, перелік характерних симптомів, опис типових візуальних патернів та структурований план дій, поділений на діагностичні, агротехнічні, хімічні та біологічні заходи. Для текстових описів симптомів використовується українська мова, орієнтована на кінцевого користувача, тоді як частина візуальних патернів подається англійською для кращої сумісності із зовнішніми сервісами аналізу зображень. Обраний формат YAML є людиночитним, добре інтегрується із системами контролю версій та дозволяє зберігати коментарі й багатомовний контент, що важливо для поступового доповнення та уточнення бази знань.

Для зберігання діагностичних кейсів, сформованих у результаті роботи конвеєра діагностики, застосовано двомодовий підхід, який передбачає можливість роботи як із реляційною базою даних, так і з файловим сховищем. Така організація є компромісом між вимогами до простоти розгортання прототипу та потребами потенційної промислової експлуатації, де важливими стають масштабованість, можливість виконання складних запитів та інтеграція з аналітичними інструментами. На рівні прикладної логіки реалізовано диспетчер збереження, який, керуючись налаштуваннями конфігурації, обирає відповідний режим: у разі увімкнення використання бази даних діагностичні кейси записуються до реляційної СУБД, в іншому випадку — у файлову структуру в межах робочого каталогу. Передбачено також автоматичний перехід до файлового режиму у

випадку збою збереження в базі даних, що підвищує стійкість системи до відмов інфраструктурних компонентів.

У режимі бази даних як основну СУБД обрано PostgreSQL. Це сучасна реляційна система керування базами даних із підтримкою транзакційності, індексування, розширених типів даних та розвинених механізмів оптимізації запитів. Доступ до БД реалізовано за допомогою бібліотеки SQLAlchemy у асинхронній конфігурації, що узгоджується із загальним підходом до асинхронної обробки запитів у серверній частині. На рівні логічної моделі даних у базі даних виокремлено, принаймні, дві основні сутності: таблицю діагностичних кейсів, у якій зберігається метаінформація (ідентифікатор кейсу, культура, дата створення, текстовий опис симптомів, агреговані результати діагностики, часові характеристики виконання конвеєра), та таблицю зображень, яка містить відомості про файли зображень, прив’язані до певного кейсу. Залежно від конфігурації, у базі даних можуть зберігатися як безпосередні бінарні представлення зображень, так і посилання на зовнішнє об’єктне сховище. Така структура дозволяє виконувати подальший аналіз накопичених кейсів за культурами, часом, ключовими симптомами чи результатами діагностики, а також інтегруватися з інструментами бізнес-аналітики [20].

Файлова конфігурація зберігання орієнтована насамперед на етапи розробки, тестування та невеликі розгортання, де розгортання повноцінної реляційної БД могло б бути надмірним. У цьому режимі для кожного діагностичного кейсу створюється окрема папка у структурі каталогу даних, як правило, із групуванням за датою створення та унікальним ідентифікатором випадку. Усередині такої папки зберігаються файли зображень, початковий запит користувача, сформована системою відповідь та додаткові відомості про перебіг виконання конвеєра (наприклад, детальні журнали чи виміряні часові характеристики окремих стадій). Формат збереження структурованих даних у цьому режимі — JSON, що забезпечує сумісність з широким колом інструментів для подальшого аналізу та, водночас, легку читабельність для розробника. Такий підхід дозволяє використовувати

систему без налаштування окремого серверу бази даних, що є важливим для швидкого розгортання прототипу та перевірки гіпотез.

Окремого розгляду потребує зберігання зображень, які можуть становити значні обсяги даних порівняно з текстовими описами та структурованими записами. Архітектура системи передбачає кілька варіантів: зберігання зображень у файловій системі в межах робочого каталогу, зберігання у стовпцях бінарного типу реляційної бази даних та використання зовнішнього об’єктного сховища, зокрема AWS S3. У файловому режимі всі зображення, пов’язані з кейсом, розміщуються у підкаталозі цього кейсу, що забезпечує просту локальну організацію даних. У режимі бази даних може використовуватися або збереження бінарного вмісту безпосередньо в таблиці зображень, або зберігання лише посилань на об’єкти у S3, що є доцільним для хмарних розгортань з великими обсягами зображень. Об’єктне сховище AWS S3, у свою чергу, надає засоби масштабованого, надійного зберігання файлів та формування тимчасових посилань для доступу, що може бути використано як для внутрішньої обробки, так і для надання доступу до оригіналів зображень через інтерфейс системи [21].

Узагальнюючи, обрана архітектура підсистеми зберігання даних поєднує гнучкість конфігурації з можливістю забезпечити вимоги до відтворюваності, аудиту та подальшого аналітичного опрацювання діагностичних кейсів.

4.2.4 Технології оброблення зображень і текстових даних

Оброблення зображень та текстових описів симптомів є ключовим елементом діагностичного конвеєра, оскільки саме на основі цих даних формується первинна гіпотеза щодо можливих хвороб сільськогосподарських культур. Обрані технології мають забезпечувати, з одного боку, достатню виразність для вилучення релевантних ознак, а з іншого — прозорість і керованість процесу, що є важливим з огляду на вимоги до пояснюваності та відтворюваності діагностики.

У частині роботи з текстовими даними основним джерелом знань є формалізовані записи бази знань у форматі YAML, які містять структуровані описи

хвороб, їхніх симптомів і планів дій. Для завантаження та інтерпретації цих записів використовується бібліотека PyYAML, що забезпечує перетворення текстових файлів бази знань у внутрішні структури даних мови Python. Такий підхід дозволяє поєднати людинозрозуміле подання знань (редагування YAML-файлів експертами) з програмною обробкою, не вимагаючи складних проміжних перетворень. Текстові поля записів бази знань та текстовий опис симптомів із запиту користувача опрацьовуються засобами стандартної бібліотеки мови Python (нормалізація, приведення до єдиного регістру, базовий аналіз ключових слів), після чого використовуються у механізмах пошуку відповідності між запитом та профілями хвороб.

Оброблення зображень у прототипі системи реалізовано багаторівнево. На базовому рівні застосовується бібліотека Pillow, яка надає засоби для завантаження, перевірки та первинної трансформації зображень. За її допомогою здійснюється контроль формату та роздільної здатності файлів, нормалізація зображень до уніфікованого розміру, а також вилучення простих візуальних характеристик, які можуть бути використані для побудови узагальненого вектору ознак. Такий підхід забезпечує мінімально необхідний рівень обробки без залежності від зовнішніх сервісів, що є важливим для сценаріїв локального розгортання та роботи в умовах обмеженого мережевого доступу.

Для більш глибокого аналізу зображень архітектурою системи передбачено можливість використання хмарного сервісу AWS Rekognition. Інтеграція реалізується через окремий адаптер, який за наявності відповідних налаштувань застосовує методи розпізнавання користувацьких міток до завантажених зображень. Отримані від сервісу ймовірнісні оцінки наявності певних візуальних патернів зіставляються з профілями хвороб у базі знань і перетворюються на додаткові вагові коефіцієнти для відповідних діагностичних гіпотез. У разі, якщо використання AWS Rekognition вимкнено або сервіс недоступний, конвеєр повертається до локального режиму, в якому враховуються лише ознаки, вилучені за допомогою Pillow. Таким чином реалізується принцип «граційної деградації»,

коли відсутність зовнішнього сервісу не блокує роботу системи, а лише зменшує глибину візуального аналізу.

Щодо текстових пояснень, система поєднує внутрішні шаблонні механізми формування відповідей із можливістю використання зовнішньої мовної моделі. У базовому режимі пояснення будуються на основі структурованих полів бази знань: короткий опис хвороби, типові симптоми, узагальнений план дій. Такий підхід гарантує стабільність формулювань і прямий зв’язок між відповіддю та записами бази знань, що полегшує аудит і валідацію. Для розширеного режиму архітектура передбачає інтеграцію з сервісом AWS Bedrock, який надає доступ до сучасних мовних моделей. Взаємодія з ним організована через асинхронний клієнтський інтерфейс на основі бібліотеки aioboto3. За цієї конфігурації мовна модель використовується для генерації більш деталізованих пояснень, зокрема для розгортання коротких пунктів плану дій у зв’язний текст і адаптації формулювань під контекст конкретного запиту.

4.2.5 Хмарні сервіси та інфраструктура розгортання

Під час проєктування системи діагностики хвороб сільськогосподарських культур важливим завданням є забезпечення гнучких варіантів її розгортання. З одного боку, система має залишатися придатною для локального використання в умовах окремого господарства чи лабораторії, де ресурси обмежені й небажано залежати від зовнішніх сервісів. З іншого боку, архітектура повинна дозволяти розгортання у хмарному середовищі з використанням керованих сервісів, масштабованого зберігання даних та сервісів штучного інтелекту. У цьому контексті як цільову платформу обрано хмарне середовище Amazon Web Services, а також передбачено можливість контейнеризованого та оркестрованого розгортання за допомогою сучасних інструментів інфраструктури.

Концептуально розглядаються два базові сценарії розгортання: локальний та хмарний. Локальний сценарій орієнтований на відносно просту конфігурацію, у якій серверна частина (FastAPI) та користувацький інтерфейс (Streamlit)

запускаються безпосередньо на робочому вузлі, а сховище даних реалізується у вигляді файлової системи або інстансу реляційної бази даних, розгорнутого на тій самій машині. Такий варіант є найбільш доступним на етапі розробки й тестування, а також для невеликих інсталяцій, де кількість користувачів і обсяг даних є обмеженими. При цьому інтеграція з зовнішніми сервісами аналізу зображень і тексту може бути повністю вимкнена, а система працює виключно на локальних засобах оброблення.

Хмарний сценарій у проєктних рішеннях орієнтований на екосистему Amazon Web Services. Для зберігання структурованих даних передбачається використання керованої реляційної бази даних Amazon RDS з рушієм PostgreSQL, що дозволяє забезпечити надійність, резервування та масштабованість без необхідності самостійної підтримки серверів баз даних. Зображення, пов’язані з діагностичними кейсами, доцільно розміщувати в об’єктному сховищі Amazon S3, яке забезпечує високу доступність та можливість еластичного масштабування за обсягом. Таке рознесення даних за типами сховищ добре узгоджується з архітектурою рівня даних: метадані та діагностичні результати зберігаються у реляційній БД, тоді як великі бінарні об’єкти розташовуються в S3 з посиланнями на них у структурі кейсу.

Для розгортання серверних компонентів у хмарному середовищі в якості цільового рішення розглядається використання контейнерних сервісів AWS, зокрема платформи ECS/Fargate або, у більш складних сценаріях, Kubernetes-кластера на базі Amazon EKS. У першому випадку окремі контейнери з серверною частиною та, за потреби, з інтерфейсом користувача можуть бути розгорнуті як служби, масштабування яких виконується автоматично відповідно до навантаження. У випадку використання EKS архітектура дозволяє організувати більш складну оркестрацію декількох сервісів (API, фонові задачі, окремі компоненти аналізу) в межах єдиного Kubernetes-кластера, з можливістю застосування стандартних для цієї технології механізмів відмовостійкості та оновлення. У поточній реалізації прототипу такі схеми розгортання не впроваджено, проте структура застосунку (виділення окремих сервісів,

конфігурація через змінні середовища, явні інтерфейси взаємодії) дозволяє у подальшому перейти до контейнеризованих і оркестрованих варіантів без радикальних змін прикладного коду.

Ключовим елементом підготовки до контейнеризованого розгортання є використання Docker. Проєктом передбачено організацію контейнерів для серверної частини та інтерфейсу користувача, а також допоміжних сервісів (бази даних), які можуть бути об’єднані у єдине середовище за допомогою інструментів на кшталт docker-compose. Така конфігурація є природним «містком» між локальним та хмарним розгортанням: ті самі образи контейнерів можуть запускатися як на робочій станції розробника, так і в середовищі контейнерних сервісів AWS. На момент формування даного опису повноцінна інфраструктура Docker ще перебуває в стадії проєктування, однак архітектурні рішення, зокрема чітке розділення сервісів та використання конфігурації через змінні середовища, вже відповідають вимогам контейнеризованого розгортання.

У контексті керування інфраструктурою у хмарному середовищі доцільним є застосування підходу інфраструктури як коду. Як один з поширених інструментів такого класу розглядається Terraform, який дозволяє описувати ресурси хмарного середовища (віртуальні мережі, кластери контейнерів, інстанси RDS, бакети S3 та інші компоненти) у вигляді декларативних конфігурацій. Для системи діагностики це означає можливість відтворюваного розгортання однакових середовищ для розробки, тестування й промислової експлуатації, а також прозорий облік змін в інфраструктурі через системи контролю версій. На рівні прототипу такі конфігурації можуть залишатися перспективним напрямом розвитку, однак їх доцільно враховувати під час проєктування структури сервісів та їхніх залежностей.

Ще одним важливим аспектом інфраструктури є організація процесів безперервної інтеграції та доставки. Для репозиторіїв, розміщених на платформі GitHub, природним вибором є використання GitHub Actions як механізму автоматизації виконання тестів, статичного аналізу коду, збирання контейнерних образів та розгортання їх у цільових середовищах. У випадку системи діагностики

це дозволяє формалізувати послідовність етапів: перевірка якості коду, запуск автоматизованих тестів, збирання образів для серверної частини й інтерфейсу, оновлення інфраструктури або деплой на контейнерний сервіс. Хоча на момент опису повноцінний конвеєр CI/CD ще не реалізований, його проєктування є логічним продовженням вже прийнятих рішень щодо структури проєкту та використання контейнеризації.

Узагальнюючи, хмарні сервіси та інфраструктура розгортання розглядаються не як додатковий рівень складності, а як невід'ємна складова архітектури системи, що забезпечує їй гнучкість, масштабованість і можливість адаптації до різних сценаріїв використання — від локальної інсталяції в межах одного господарства до розгорнутого хмарного сервісу.

4.3 Вибір бібліотек та допоміжних засобів мови Python

Узагальнені відомості про основні бібліотеки, використані в системі, наведено у таблиці 4.1. Тут зосереджено увагу на їх ролі у загальній архітектурі та ключовому призначенні, без деталізації внутрішньої реалізації окремих компонентів, що буде розглянуто у наступному розділі.

Таблиця 4.1 — Основні використані бібліотеки мови Python

Бібліотека	Основне призначення в системі	Коротка характеристика ролі
fastapi	Побудова прикладного програмного інтерфейсу	Опис REST-ендпоїнтів, валідація даних, інтеграція з Pydantic
uvicorn	Запуск серверної частини	ASGI-сервер для оброблення вебзапитів до FastAPI
pydantic	Моделювання та валідація структур даних	Типізовані моделі запитів, відповідей і внутрішніх сутностей
python-multipart	Оброблення багаточастинних HTTP-запитів	Підтримка завантаження файлів (зображень) через API

Бібліотека	Основне призначення в системі	Коротка характеристика ролі
PyYAML	Робота з базою знань у форматі YAML	Завантаження та розбір формалізованих описів хвороб
Pillow	Базове оброблення зображень	Перевірка, нормалізація та підготовка зображень до аналізу
streamlit	Реалізація користувацького вебінтерфейсу	Інтерактивні форми введення та відображення результатів
prometheus-client	Експорт базових метрик роботи системи	Надання даних для моніторингу у форматі, сумісному з Prometheus
requests	Клієнтські HTTP-запити (допоміжна роль)	Виклик зовнішніх сервісів або локальних API у утилітарних задачах
sqlalchemy	Доступ до реляційної бази даних	Опис моделі збереження кейсів і зображень, побудова запитів
asyncpg	Асинхронний драйвер PostgreSQL	Ефективна взаємодія з БД у асинхронному режимі
psycopg2-binary	Синхронний драйвер PostgreSQL	Альтернативний доступ до БД, зокрема для службових утиліт
alembic	Керування міграціями схеми бази даних	Поетапна еволюція структури БД відповідно до змін моделі
boto3	Синхронна взаємодія з сервісами AWS	Базовий клієнт для S3, Rekognition та інших сервісів
aioboto3	Асинхронна взаємодія з сервісами AWS	Інтеграція з AWS у межах асинхронного діагностичного конвеєра

Бібліотека	Основне призначення в системі	Коротка характеристика ролі
python-jose[cryptography]	Робота з токенами доступу та криптографією	Підтримка JWT і базових механізмів аутентифікації
passlib[bcrypt]	Хешування паролів	Підготовка та перевірка безпечних хешів облікових даних
pytest, pytest-asyncio	Організація модульного та асинхронного тестування	Формування тестового покриття, у тому числі для асинхронних частин
httpx	Тестування HTTP-інтерфейсів	Емуляція клієнтських запитів до API під час тестів

Висновки до розділу 4

У четвертому розділі обґрунтовано вибір технологічного стека, де базовою мовою визначено Python завдяки його розвиненій екосистемі для задач штучного інтелекту. Серверну частину реалізовано на асинхронному фреймворку FastAPI, а інтерактивний вебзастосунок — за допомогою бібліотеки Streamlit, що забезпечує єдність кодової бази та зручність прототипування. Архітектура системи побудована за модульним принципом, розділяючи сервіси комп'ютерного зору, пошуку в базі знань та генерації рекомендацій для ефективної обробки мультимодальних даних.

Технічна реалізація спирається на сучасні хмарні рішення: для аналізу зображень використано гібридний підхід (локальні засоби та AWS Rekognition), а для генерації пояснень — сервіс AWS Bedrock. Зберігання структурованих діагностичних кейсів забезпечує PostgreSQL із підтримкою JSONB, тоді як медіафайли розміщуються у сховищі AWS S3.

5 РОЗРОБКА ТА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

У цьому розділі представлено програмну реалізацію прототипу системи діагностики хвороб рослин, виконану на основі попередньо обґрунтованих архітектурних рішень. Детально описано структуру компонентів, модель даних, алгоритми діагностичного конвеєра та специфіку розгортання у локальному й хмарному середовищах. За результатами тестування та експериментальної оцінки продуктивності підтверджено практичну придатність розробки, ідентифіковано її обмеження та окреслено шляхи подальшого вдосконалення.

5.1 Функціональна структура системи

Функціональна структура прототипу інформаційної системи діагностики хвороб сільськогосподарських культур визначається сукупністю задач, які вона підтримує на рівні кінцевого користувача та сервісних підсистем. У сформованому варіанті реалізації основний акцент зроблено на повному циклі роботи з «діагностичним випадком» — від формування запиту до збереження результатів у вигляді кейсу та можливості його подальшого аналізу. Узагальнене подання функціональної структури наведено у Додатку В.

На рівні користувача система реалізує такі базові функції.

По-перше, формування запиту на діагностику. Користувач за допомогою вебінтерфейсу обирає культуру, за потреби уточнює фазу розвитку та умови вирощування, вводить текстовий опис спостережуваних симптомів українською мовою й завантажує одне чи кілька зображень уражених рослин. Усі ці дані передаються до серверної частини як єдиний діагностичний запит, що ініціює запуск діагностичного конвеєра.

По-друге, виконання діагностичного аналізу. На основі отриманих від користувача даних запускається послідовність етапів оброблення: аналіз зображень підсистемою комп’ютерного зору, пошук релевантних карток хвороб у базі знань, застосування доменних агрономічних правил та формування впорядкованого

списку кандидатних хвороб. На цьому етапі система перетворює вхідні текстові та візуальні дані на набір узгоджених діагностичних гіпотез.

По-третє, формування та подання результатів діагностики. Для найбільш імовірної гіпотези будується структурований план дій, який включає уточнювальні діагностичні кроки, рекомендовані агротехнічні заходи та, за наявності відповідної інформації у базі знань, можливі варіанти застосування хімічних і біологічних засобів захисту. Результати передаються у вигляді структурованої відповіді й відображаються користувацьким інтерфейсом у зручному для сприйняття форматі.

По-четверте, система реалізує створення та зберігання «папок діагностичних кейсів». Для кожного звернення формується запис, що містить вихідні параметри запиту, узагальнену інформацію про перебіг діагностичного конвеєра (зокрема часові характеристики окремих етапів), список кандидатних хвороб та обраний план дій. Зображення, надані користувачем, зберігаються окремо, але з посиланням на відповідний кейс. Така організація забезпечує відтворюваність діагностичних рішень і можливість їх подальшого аудиту.

По-п’яте, на рівні серверної частини реалізовано функції технічного обслуговування та моніторингу. До них належать перевірка доступності сервісу (ендпоїнт контролю стану), отримання базової оперативної інформації про кількість оброблених запитів та часові затримки, а також можливість інтеграції зі стандартними системами збору експлуатаційних метрик. Ці функції орієнтовані насамперед на технічний персонал і використовуються під час розгортання та супроводу прототипу.

По-шосте, передбачено доступ до раніше збережених діагностичних кейсів. Через відповідні програмні інтерфейси можна отримувати інформацію про історичні звернення, аналізувати стабільність роботи системи, виявляти типові сценарії використання та використовувати накопичені кейси як підґрунтя для подальшого розширення бази знань. У поточній версії прототипу такий доступ реалізовано на рівні API, а розширений інтерфейс для кінцевого користувача може бути доданий на наступних етапах розвитку системи.

Отже, функціональна структура системи охоплює повний цикл роботи з діагностичним випадком — від введення вихідних даних, їх оброблення в межах діагностичного конвеєра та формування результатів до збереження інформації у вигляді відтворюваного діагностичного кейсу й технічного моніторингу роботи сервісу.

5.2 Опис бізнес-процесів

Бізнес-процеси, які реалізує прототип інформаційної системи діагностики хвороб сільськогосподарських культур, відображають послідовність дій користувача та програмних компонентів упродовж усього життєвого циклу «діагностичного випадку». На концептуальному рівні ключовими є два взаємопов’язані процеси: діагностика окремого випадку ураження рослин та подальша робота з накопиченими діагностичними кейсами. Перший процес відповідає за перетворення польових спостережень на структуроване діагностичне рішення, другий — за використання сформованих кейсів для аналізу, аудиту та розвитку системи.

5.2.1 Бізнес-процес діагностики окремого випадку

Базовим бізнес-процесом, реалізованим системою, є діагностика конкретного випадку ураження сільськогосподарської культури. Він охоплює послідовний ланцюг дій — від фіксації симптомів у полі до формування плану дій та збереження результатів діагностики у вигляді кейсу.

Процес розпочинається на рівні господарства, де фермер або агроном фіксує прояви хвороби: візуальні симптоми на листі, стеблах, плодах, просторовий характер ураження, динаміку змін у часі. Одночасно, за можливості, уточнюється контекст: культура, фаза розвитку, попередні обробки, погодні умови, технологія вирощування. На цьому етапі інформація є неформалізованою і представлена у вигляді спостережень та фотоматеріалів.

Наступним кроком є введення даних до системи за допомогою вебінтерфейсу. Користувач обирає культуру, за потреби зазначає фазу розвитку, вводить текстовий опис симптомів українською мовою та завантажує одне чи кілька зображень уражених рослин. Після заповнення обов’язкових полів формується запит на діагностику, який передається до серверної частини системи через HTTP-запит.

На стороні серверної частини відбувається первинна обробка запиту. Виконується формальна валідація вхідних даних: перевіряється наявність обов’язкових параметрів, допустимість форматів і розміру файлів зображень, коректність значень полів. У разі успішної валідації для звернення генерується унікальний ідентифікатор, формується внутрішній діагностичний контекст (культура, опис симптомів, метадані, зображення), після чого ініціюється виконання діагностичного конвеєра.

Подальші етапи бізнес-процесу відповідають послідовним стадіям конвеєра. Спочатку підсистема комп’ютерного зору здійснює попередній аналіз зображень, виділяючи узагальнені візуальні ознаки (наявність світлого нальоту, темних «водянистих» плям, дрібних некротичних плям тощо). У розширеній конфігурації, за наявності доступу до хмарної інфраструктури, може додатково залучатися сервіс аналізу зображень AWS Rekognition, який повертає ймовірнісні оцінки для попередньо визначених класів уражень. Результати локальної обробки та зовнішнього аналізу зображень об’єднуються в єдиний вектор візуальних характеристик.

На наступній стадії відбувається пошук у базі знань. Для заданої культури завантажуються картки хвороб, і для кожної з них обчислюється оцінка відповідності спостережуваному випадку на основі текстового опису симптомів користувача та виявлених візуальних ознак. Ця оцінка уточнюється блоком доменних агрономічних правил, що враховують культуру, фазу розвитку та типові патерни перебігу хвороб. У результаті формується впорядкований список кандидатних хвороб із числовими оцінками відповідності.

Для кількох найімовірніших гіпотез відкриваються підсумкові пояснення та формується план дій. У базовому режимі пояснення й рекомендації будуються на основі структурованих полів картки хвороби та заздалегідь заданих шаблонів. У розширеному режимі може використовуватися інтеграція з сервісом мовних моделей (наприклад, AWS Bedrock), який генерує розгорнуті текстові пояснення українською мовою на основі результатів конвеєра. Незалежно від конкретної конфігурації джерелом змісту рекомендацій є база знань та результати оброблення даних у конвеєрі, тоді як мовна модель впливає переважно на форму подання.

Сформована відповідь, що включає перелік кандидатних хвороб із оцінками, пояснення та план дій для найбільш імовірної гіпотези, повертається до клієнтської частини та відображається у вебінтерфейсі. Користувач може зіставити запропоновані гіпотези з власними спостереженнями, за потреби уточнити діагноз додатковими польовими обстеженнями та ухвалити рішення щодо подальших заходів захисту.

Паралельно, у завершальній фазі бізнес-процесу, система створює «папку діагностичного кейсу». У ній фіксуються вхідні дані запиту, результати роботи конвеєра, сформований план дій, а також технічні параметри (часові характеристики етапів, інформація про використані підсистеми, службові повідомлення). Зображення зберігаються у вибраному сховищі (реляційна база даних або файлове/об’єктне сховище) з посиланням на відповідний кейс. Така організація забезпечує відтворюваність діагностичного рішення та створює основу для подальшого аналізу й аудиту.

5.2.2 Бізнес-процес роботи з діагностичними кейсами

Другим важливим бізнес-процесом є робота з уже сформованими діагностичними кейсами, яка охоплює їх накопичення, перегляд, аналіз та використання для удосконалення системи. На відміну від оперативної діагностики, цей процес зазвичай реалізується фахівцями аграрного профілю або технічним персоналом, відповідальним за експлуатацію системи.

На першому етапі відбувається накопичення діагностичних кейсів у процесі повсякденного використання системи. Кожне звернення, оброблене діагностичним конвеєром, породжує окремий запис, який включає вхідні дані, результати аналізу та пов’язані зображення. У такий спосіб формується репозиторій кейсів, що відображає реальну практику застосування системи в різних господарствах, для різних культур та сценаріїв ураження.

Далі, залежно від потреб, може виконуватися перегляд і аналіз накопичених кейсів. У поточній версії прототипу доступ до таких даних реалізовано насамперед через програмні інтерфейси на стороні серверної частини: за допомогою відповідних запитів можна отримати список кейсів за певний період, відібрати кейси для конкретної культури або з заданими характеристиками, а також отримати детальну інформацію щодо окремого випадку. На цій основі агроном або аналітик може порівнювати рішення системи з фактичним перебігом подій у полі, виявляти типові ситуації, для яких діагностика виявилася успішною або, навпаки, потребує коригування.

Ще одним напрямом використання кейсів є аудит та розвиток бази знань. Аналізуючи накопичені записи, фахівці можуть виявляти хвороби, що повторюються, але не мають достатньо деталізованих карток, або навпаки — ситуації, у яких наявні картки не забезпечують достатньої розрізнювальної здатності. Це створює підстави для актуалізації описів симптомів, уточнення візуальних патернів та перегляду рекомендацій. Таким чином, репозиторій кейсів виступає не лише архівом, але й інструментом зворотного зв’язку для подальшого навчання й адаптації системи.

Окремий практичний сценарій пов’язаний з використанням кейсів для навчальних і демонстраційних цілей. На основі реальних або наближених до реальних випадків можна формувати навчальні підбірки для підготовки фермерів, студентів аграрних спеціальностей або молодших фахівців-консультантів. У цьому разі діагностичні кейси виконують роль структурованих прикладів, у яких поєднано польові спостереження, результати аналізу системи та фактичні заходи, що були вжиті.

Перспективним елементом бізнес-процесу є розширення користувацького інтерфейсу функціями перегляду та фільтрації діагностичних кейсів. Такий інтерфейс дозволив би кінцевому користувачеві безпосередньо аналізувати історію власних звернень, порівнювати поточний випадок із попередніми, відстежувати повторюваність хвороб на окремих полях або культурах. У поточній реалізації ці можливості перебувають на рівні закладеної інформаційної моделі та API й можуть бути реалізовані на наступних етапах розвитку системи.

5.3 Архітектура інформаційної системи

Архітектура прототипу інформаційної системи діагностики хвороб сільськогосподарських культур ґрунтується на багатошаровому підході, обґрунтованому у четвертому розділі, та відображає перехід від концептуальної моделі до конкретних програмних компонентів. У межах обраної архітектури чітко розмежовано рівень взаємодії з користувачем, прикладну логіку діагностики, роботу з базою знань, оброблення зображень, інтеграцію із зовнішніми сервісами та зберігання результатів у вигляді діагностичних кейсів.

Загальна структура архітектури може бути подана у вигляді компонентної діаграми, на якій відображено основні підсистеми, інтерфейси між ними та напрямки потоків даних. Відповідну діаграму компонентів архітектури інформаційної системи наведено в Додатку Ж.

Прототип системи реалізовано як сукупність узгоджених програмних компонентів, що безпосередньо відображають виділені логічні шари: презентаційний, прикладний (бізнесовий) та шар даних. Така побудова спрощує подальший розвиток системи, оскільки кожен компонент має чітко окреслену відповідальність і взаємодіє з іншими через стабільні інтерфейси.

На рівні презентаційного шару функціонує компонент користувацького інтерфейсу, реалізований як вебзастосунок. Він забезпечує введення вихідних даних (культури, фази розвитку, текстового опису симптомів українською мовою, одного чи кількох зображень) та відображення результатів діагностики. Усі

взаємодії цього компонента із системою відбуваються через HTTP-запити до серверного програмного інтерфейсу; логіка прийняття рішень повністю зосереджена у серверній частині, що відповідає вимогам до централізованої обробки та відтворюваності діагностичних рішень.

Прикладний шар на стороні сервера представлено компонентом програмного інтерфейсу (API), побудованим на базі вебфреймворка FastAPI. Цей компонент приймає запити від користувацького інтерфейсу, виконує їх формальну валідацію, перетворює вхідні дані у внутрішні моделі, генерує унікальний ідентифікатор діагностичного звернення та ініціює виконання діагностичного конвеєра. Крім того, API надає допоміжні операції для перевірки стану сервісу та отримання раніше збережених діагностичних кейсів, що забезпечує можливість технічного моніторингу й інтеграції з іншими інформаційними системами.

Центральне місце у бізнесовому шарі займає компонент діагностичного конвеєра, реалізований у вигляді оркестратора послідовності етапів оброблення запиту. У межах конвеєра виділено кілька основних стадій: підсистему комп’ютерного зору, механізм пошуку в базі знань, блок доменних правил, модуль ранжування кандидатів із використанням мовної моделі або шаблонного механізму, етап формування підсумкової відповіді та етап збереження діагностичного кейсу. Конвеєр координує виклики відповідних підсистем, агрегує отримані від них оцінки та пояснення, а також фіксує часові характеристики виконання окремих стадій, що надалі використовується у процесі аналізу продуктивності.

Підсистема комп’ютерного зору інкапсулює оброблення візуальної інформації, наданої користувачем. Вона складається з локального блоку попередньої обробки зображень, який на основі простих візуальних ознак (наявність світлого нальоту, темних водянистих ділянок, інтенсивність контурів ушкоджень) формує узагальнений вектор характеристик, та, за потреби, адаптера до хмарного сервісу аналізу зображень. Взаємодія з таким сервісом (зокрема, AWS Rekognition) реалізується як окремий компонент, що отримує зображення, повертає список імовірних класів ушкоджень із числовими оцінками та переводить їх у

внутрішній формат, узгоджений із базою знань. Обидва джерела ознак подаються до діагностичного конвеєра через єдиний інтерфейс, що дозволяє поєднувати локальний та хмарний аналіз без зміни логіки подальших етапів.

Підсистема бази знань відповідає за завантаження формалізованих описів хвороб із файлів, їх перетворення у внутрішні моделі та пошук найімовірніших кандидатів на основі текстового опису симптомів і візуальних ознак. З технічного погляду вона реалізована як окремий сервіс, що надає конвеєру операції з вибірки карток хвороб для конкретної культури та обчислення рейтингових оцінок відповідності. Така організація дозволяє розглядати базу знань як окремий рівень абстракції, який може розвиватися незалежно від інших компонентів, за умови збереження узгодженого формату карток.

Наступним елементом бізнесового шару є підсистема доменних правил та клієнт мовних моделей. Підсистема правил здійснює корекцію оцінок, отриманих від підсистеми бази знань, з урахуванням агрономічних обмежень: культури, фази розвитку, типових візуальних патернів для окремих хвороб. Клієнт мовних моделей забезпечує формування текстових пояснень і планів дій для кінцевого користувача. У базовому режимі використовується детермінований шаблонний механізм, який на основі структурованих полів картки хвороби формує стислий та однорідний за структурою текст відповіді. У розширеній конфігурації реалізовано інтеграційний компонент для взаємодії з сервісом великої мовної моделі (наприклад, AWS Bedrock), який використовує результати конвеєра як вхідні дані для генерації розгорнутих пояснень українською мовою. В обох випадках компонент мовної моделі спроєктовано як надбудову над уже прийнятим діагностичним рішенням, що дає змогу уникнути залежності змісту рекомендацій від зовнішнього сервісу.

Шар даних представлений підсистемою зберігання діагностичних кейсів, яка реалізує єдиний інтерфейс для запису та читання інформації про діагностичні звернення. Усередині цієї підсистеми підтримуються режими роботи з реляційною базою даних (PostgreSQL) та з файловою системою; у хмарній конфігурації додатково використовується об’єктне сховище для зберігання зображень. Компонент доступу до даних інкапсулює логіку роботи з конкретними

механізмами зберігання (SQL-запити, операції з файлами, доступ до об’єктного сховища) та забезпечує для решти системи уніфікований інтерфейс: створення, читання та оновлення записів про кейси та пов’язані з ними зображення. Структура діагностичного кейсу та відповідні моделі даних деталізуються у підрозділі 5.5.

Окрему групу становлять допоміжні компоненти конфігурації, безпеки та моніторингу. Компонент конфігурації відповідає за завантаження параметрів середовища (режим локальної чи хмарної роботи, параметри підключення до бази даних, облікові дані хмарних сервісів) та їх надання іншим частинам системи у стандартизованому вигляді. Засоби безпеки включають базові механізми криптографічного захисту й керування токенами доступу, необхідні для інтеграції з зовнішніми сервісами. Компонент моніторингу відповідає за збирання ключових експлуатаційних метрик (кількість оброблених запитів, час відповіді, частота помилок) та їх експорт у форматі, сумісному зі стандартними системами спостереження.

У сукупності зазначені компоненти утворюють цілісну архітектурну модель, у якій кожен елемент має чітко визначену роль, а взаємодія між компонентами здійснюється через добре визначені інтерфейси. Така організація полегшує подальшу еволюцію прототипу: кожну з підсистем (комп’ютерний зір, базу знань, правила, інтеграцію з хмарними сервісами, зберігання даних) можна розвивати незалежно, зберігаючи при цьому узгодженість загальної архітектури інформаційної системи.

5.4 Алгоритмічні засади діагностичного конвеєра

Діагностичний конвеєр є центральною підсистемою розробленої системи, яка реалізує послідовне перетворення вхідних даних користувача (текстових описів симптомів та зображень) на впорядкований набір діагностичних гіпотез із поясненнями та планом дій. Конвеєр складається з кількох логічно відокремлених етапів, що працюють над єдиним діагностичним контекстом і поступово збагачують його новою інформацією.

5.4.1 Загальна послідовність оброблення запиту

Після надходження запиту від користувача серверна частина системи формує внутрішній об’єкт діагностичного контексту. До нього включаються: ідентифікатор випадку, час звернення, культура, текстовий опис симптомів, метадані (за наявності — фаза розвитку, умови вирощування) та масив зображень уражених рослин. На цьому ж етапі виконуються формальні перевірки: перевіряється наявність обов’язкових полів, коректність форматів файлів, допустимість розміру зображень. У разі успішної валідації запускається власне конвеєр.

На першій стадії з вхідних даних виділяються ознаки: з тексту — ключові слова та фрагменти, що описують симптоми; із зображень — агреговані візуальні характеристики. На другій стадії ці ознаки порівнюються з інформацією, наявною в базі знань, і для кожної хвороби формується попередня оцінка відповідності. На третій стадії до цих оцінок застосовуються доменні правила, які враховують агрономічний контекст (культура, стадія розвитку, типові патерни перебігу хвороб).

На четвертій стадії формується впорядкований список гіпотез, для кількох найімовірніших хвороб будуються пояснення, а для хвороби з найвищою оцінкою — план дій. Нарешті, на заключній стадії результат повертається користувачеві через вебінтерфейс, а весь діагностичний контекст фіксується у сховищі кейсів. Деталізовану послідовність взаємодій між компонентами під час оброблення одного запиту відображено у Додатку Д.

5.4.2 Оброблення візуальних даних

Підсистема опрацювання зображень виконує роль джерела високорівневих візуальних ознак, які надалі зіставляються з патернами, описаними в картках хвороб. Її робота організована так, щоб система могла функціонувати як у повністю

локальній конфігурації, так і з використанням зовнішніх хмарних сервісів аналізу зображень.

У локальному режимі для кожного зображення виконуються базові операції комп’ютерного зору: нормалізація розміру та яскравості, виділення контрастних ділянок, оцінювання частки дуже світлих і темних пікселів, виявлення підвищеної «зернистості» тощо. За результатами цих операцій встановлюється наявність або відсутність таких загальних патернів, як світлий порошистий наліт, темні «водянисті» плями, дрібні некротичні плямки. Для кожного патерна формується узагальнений індикатор, і в результаті утворюється вектор візуальних ознак, який у подальшому може безпосередньо співвідноситися з полями карток хвороб.

У розширеній конфігурації до локальних ознак додаються результати хмарної класифікації. Зображення передається до зовнішнього сервісу аналізу, який повертає перелік найбільш ймовірних класів ушкоджень із числовими оцінками впевненості. Для кожного такого класу у базі знань задається відповідність певним хворобам. На основі цієї відповідності до вектора ознак додаються додаткові компоненти, що відображають підтримку окремих гіпотез високорівневим візуальним аналізом.

Обидва джерела візуальної інформації — локальні ознаки та результати хмарного сервісу — зводяться до єдиного внутрішнього формату, який не залежить від конкретної реалізації підсистеми комп’ютерного зору. Це полегшує заміну й розвиток алгоритмів на цьому етапі без модифікації решти конвеєра.

5.4.3 Пошук у базі знань та застосування доменних правил

Під час пошуку у базі знань система для кожної хвороби оцінює ступінь узгодженості між даними картки та описом конкретного випадку. Цей процес базується на поєднанні трьох джерел інформації: текстового опису симптомів, візуальних патернів і, за потреби, результатів зовнішнього аналізу зображень.

На текстовому рівні виділяються ключові слова та фрази з опису симптомів користувача, які співвідносяться з описами симптомів у картках хвороб.

Враховується як наявність або відсутність таких збігів, так і їхня кількість та інформативність. У найпростішому випадку кожен релевантний збіг надає хворобі додатковий внесок до загальної оцінки; важливіші терміни можуть мати більшу вагу.

На візуальному рівні аналізується збіг між вектором візуальних ознак випадку та переліком патернів, указаних у картці хвороби. Якщо характерні для конкретної хвороби патерни (наприклад, поєднання білого нальоту та певного типу плям) виявляються на зображеннях, оцінка на користь цієї хвороби істотно підвищується.

У разі використання хмарного сервісу аналізу зображень до уваги беруться й його результати. Якщо сервіс із високою впевненістю «припускає» наявність ушкодження, пов’язаного з певною хворобою з бази знань, відповідна гіпотеза отримує додаткову підтримку. Такий внесок, як правило, більший за внесок окремих текстових або локальних візуальних ознак, але все одно залишається збалансованим відносно інших джерел інформації.

Після акумулювання всіх внесків для кожної хвороби формується узагальнена оцінка відповідності. Щоб зробити ці оцінки порівнюваними та придатними для інтерпретації, застосовується нелінійне згладжування, яке переводить їх у обмежений інтервал значень та зменшує вплив поодиноких надмірно великих внесків. У результаті формується набір значень, які можна розглядати як відносні міри довіри системи до окремих гіпотез.

Після цього до отриманих оцінок послідовно застосовується система доменних агрономічних правил. На цьому етапі:

— відсіюються хвороби, що не можуть уражати задану культуру;

— коригуються оцінки з урахуванням зазначеної фази розвитку (наприклад, хвороби, характерні лише для пізніх фаз, отримують знижені оцінки, якщо запит стосується ранніх фаз);

— можуть застосовуватися додаткові коригування для випадків, коли поєднання візуальних ознак особливо характерне для окремої хвороби.

Результатом роботи цього блоку є впорядкований список кандидатних хвороб, у якому одночасно враховано і інформаційні ознаки (текстові та візуальні), і агрономічні обмеження, зафіксовані в базі знань і правилах.

5.4.4 Формування пояснень, відповіді та збереження кейсу

Після ранжування гіпотез система переходить до формування відповіді, орієнтованої на кінцевого користувача. На вхід цього етапу надходить впорядкований список хвороб із підсумковими оцінками відповідності, посиланнями на відповідні картки бази знань та діагностичний контекст.

Для кількох найімовірніших хвороб будуються пояснення. У базовому випадку вони формуються шляхом комбінування найбільш релевантних елементів картки (ключових симптомів, характерних візуальних ознак, типових умов розвитку) з інформацією про вхідні дані запиту (культура, описані симптоми, виявлені патерни). За такою схемою пояснення вказує, чому саме ця хвороба вважається ймовірною, на які ознаки слід звернути увагу при додатковому огляді посівів і які альтернативні гіпотези варто зважати.

Далі на основі картки хвороби з найвищою оцінкою формується план дій. Він структуровано подається у вигляді кількох блоків: уточнювальні діагностичні кроки, агротехнічні заходи, можливі хімічні та біологічні засоби захисту. До плану додаються рекомендації щодо необхідності перевірки локальних регуляторних вимог та дотримання вимог безпеки при роботі з засобами захисту рослин.

Паралельно з формуванням відповіді для користувача система створює запис у сховищі діагностичних кейсів. У ньому фіксуються вхідні дані запиту, перелік кандидатних хвороб із оцінками та поясненнями, згенерований план дій, а також узагальнені технічні відомості про перебіг оброблення (часові характеристики, інформація про використані режими й зовнішні сервіси). Зображення, надіслані користувачем, зберігаються в обраному сховищі й пов’язуються з відповідним кейсом. У разі недоступності основної бази даних задіюється резервний файловий

режим, у межах якого створюється окрема «папка кейсу» з копіями запиту, відповіді та зображень.

5.5 Опис класів та інформаційної моделі системи

Інформаційна модель прототипу системи діагностики хвороб сільськогосподарських культур охоплює дві основні групи сутностей: картки хвороб у базі знань і діагностичні кейси, що відображають окремі звернення користувачів. Перша група забезпечує накопичення відносно стабільних предметних знань, друга — фіксує результати роботи діагностичного конвеєра та формує «цифровий слід» реальних діагностичних ситуацій. Узагальнену схему сутностей та зв’язків подано у Додатку Е.

5.5.1 Сутності бази знань хвороб

База знань організована як набір карток хвороб, структурованих за культурами. Логічно це відповідає сутності «картка хвороби», яка описує окреме захворювання певної культури і містить такі групи атрибутів.

По-перше, ідентифікаційні атрибути. Кожній хворобі надається унікальний стабільний ідентифікатор, що використовується всередині системи під час зіставлення результатів комп’ютерного зору, пошуку у базі знань та формування діагностичних відповідей. Додатково зберігаються назва хвороби українською мовою та, за потреби, скорочені або альтернативні позначення.

По-друге, атрибути, пов’язані з симптоматикою. У картці фіксується перелік характерних ознак ураження, сформульованих природною мовою: опис типу плям, нальоту, деформацій, некрозів, їхнього розташування на рослині, швидкості розвитку тощо. Ці описи використовуються під час текстового пошуку, коли система порівнює введений користувачем опис симптомів із наявними картками.

По-третє, візуальні патерни, що можуть бути виявлені підсистемою комп’ютерного зору. Для цього в картці зазначається набір узагальнених

візуальних характеристик (наприклад, світлий порошистий наліт, темні водянисті плями, дрібні некротичні точки), які відповідають бінарним або числовим ознакам, що формуються на етапі аналізу зображень. Наявність таких патернів у картці дозволяє конвеєру прямо зіставляти результати оброблення фотографій із відповідними хворобами.

По-четверте, агрономічний контекст. Для кожної хвороби може бути задано перелік культур, на яких вона зустрічається, а також інтервали фаз розвитку, у яких захворювання є найбільш імовірним. Ця інформація використовується блоком доменних правил для коригування оцінок відповідності з урахуванням введених користувачем даних про культуру і фазу розвитку.

По-п’яте, рекомендації щодо реагування. Картка містить структурований набір рекомендацій, поділений на логічні групи: уточнювальні діагностичні кроки, агротехнічні заходи, можливі хімічні засоби захисту, біологічні препарати тощо. Завдяки такому поділу конвеєр може автоматично формувати для користувача впорядкований план дій для хвороби, що отримала найвищу оцінку.

Фізично картки хвороб зберігаються у вигляді текстових файлів формату YAML, згрупованих за культурами. Під час запуску системи вони завантажуються та перетворюються у внутрішні структури даних, що відповідають описаним вище атрибутам.

5.5.2 Сутності сховища діагностичних кейсів

Друга ключова група сутностей утворює сховище діагностичних кейсів. Логічною центральною сутністю тут є «Діагностичний кейс», який відображає повний цикл оброблення одного звернення користувача до системи.

До атрибутів діагностичного кейсу належать:

— ідентифікатор кейсу (стабільний унікальний код, у форматі UUID) та часові мітки створення і, за потреби, останнього оновлення;

— описові поля запиту: культура, текстовий опис симптомів, інші параметри середовища (фаза розвитку, тип посіву тощо, якщо вони були задані);

— результати діагностики: упорядкований список кандидатних хвороб із числовими оцінками відповідності та короткими поясненнями;

— сформований план дій для хвороби з найвищою оцінкою, структурований за блоками (діагностичні кроки, агротехніка, хімічні та біологічні засоби захисту);

— технічна інформація про перебіг оброблення запиту: часові характеристики окремих етапів конвеєра, відомості про використані зовнішні сервіси, службові позначки про режим роботи (локальний чи хмарний).

Окремою сутністю виступають «Зображення кейсу». Кожен запис цього типу містить посилання на відповідний діагностичний кейс, вихідне ім’я файлу, а також спосіб фізичного зберігання зображення. У локальному режимі це може бути безпосередньо бінарне зображення у базі даних або шлях до файлу на диску; у хмарній конфігурації — посилання на об’єкт у зовнішньому сховищі (наприклад, комірку в об’єктному сховищі).

У реляційній базі даних ці сутності реалізуються як окремі таблиці зі зв’язком «один до багатьох» між діагностичним кейсом та його зображеннями. У файловому режимі аналогічна структура відтворюється на рівні файлової системи: для кожного кейсу створюється окремий каталог, у якому розміщуються файли з вхідним запитом, отриманою відповіддю, службовою інформацією та пов’язаними зображеннями. Така організація зберігання дозволяє забезпечити відтворюваність результатів діагностики й спростити аудит роботи системи.

5.5.3 Зв’язки між базою знань та діагностичними кейсами

Окрему роль в інформаційній моделі відіграють зв’язки між статичною базою знань та динамічним сховищем діагностичних кейсів. На концептуальному рівні кожен запис про кандидатну хворобу в межах діагностичного кейсу містить посилання на відповідну картку бази знань через її унікальний ідентифікатор. Завдяки цьому забезпечується узгодженість між описами хвороб, що використовуються для пошуку й пояснення, та результатами конкретних діагностичних рішень.

Такий підхід дає змогу, по-перше, відтворити повний контекст діагностики, навіть якщо з часом картки хвороб доповнюються або уточнюються. По-друге, він створює підґрунтя для подальшого аналізу накопичених кейсів: можна визначати, які хвороби найчастіше зустрічаються для певних культур, які комбінації симптомів виявляються проблемними для діагностики, у яких випадках користувачі звертаються до системи повторно. По-третє, така структура полегшує поступове розширення бази знань новими картками на основі накопичених кейсів та експертних правок.

Вона демонструє, що розроблена система оперує узгодженим набором сутностей, у межах якого статична експертна інформація та динамічні дані про реальні звернення користувачів не ізольовані, а органічно пов’язані між собою. Це створює передумови для подальшого розвитку прототипу в напрямі більш повноцінної аналітичної системи підтримки прийняття рішень в аграрній галузі.

5.6 Життєвий цикл роботи інформаційної системи

Життєвий цикл роботи інформаційної системи діагностики хвороб сільськогосподарських культур охоплює послідовність станів, у яких перебувають її основні інформаційні об’єкти та програмні компоненти в процесі експлуатації. До ключових елементів цього циклу належать: діагностичні кейси як одиниці обліку звернень користувачів, база знань про хвороби, а також сам сервіс діагностики як довготривало працюючий програмний компонент. Узгоджене функціонування цих елементів забезпечує відтворюваність результатів, можливість накопичення досвіду та подальший розвиток системи.

Діаграма діагностичного кейсу зображено у Додатку Б.

5.6.1 Життєвий цикл діагностичного кейсу

Базовою одиницею роботи розробленої системи є діагностичний кейс. Це відповідає одному зверненню користувача із запитом на діагностику. Життєвий

цикл такого кейсу можна описати низкою послідовних стадій, а саме ініціацією, оброблення запиту, успішного завершення і подальших аналітичних станів.

На етапі ініціації формується діагностичний запит. Користувач через вебінтерфейс задає культуру, за потреби — фазу розвитку та контекстні параметри, вводить текстовий опис симптомів і завантажує зображення уражених рослин. Після надходження даних до серверної частини запиту присвоюється унікальний ідентифікатор, що надалі використовується для логічного об’єднання всіх пов’язаних із ним артефактів (внутрішніх структур конвеєра, записів у сховищі, файлів зображень).

На етапі оброблення запиту діагностичний кейс перебуває у проміжному стані: дані вже прийняті системою, але остаточне рішення ще не сформовано. У цей період послідовно виконуються етапи діагностичного конвеєра: виділення візуальних ознак із зображень, пошук відповідних записів у базі знань, застосування доменних правил, ранжування гіпотез та формування пояснень і плану дій. У процесі оброблення додатково накопичуються службові дані — часові характеристики виконання окремих стадій, інформація про використані режими аналізу (локальний або хмарний) тощо, які надалі включаються до структури діагностичного кейсу.

Після успішного завершення оброблення відбувається перехід кейсу до завершеного стану. На цьому етапі формується структурований запис у сховищі: фіксуються вхідні параметри запиту, впорядкований список кандидатних хвороб з оцінками відповідності та поясненнями, сформований план дій, а також узагальнені технічні метадані. Паралельно з цим зображення зберігаються у вибраному фізичному сховищі (реляційна база даних, локальна файлова система або об’єктне сховище у хмарній інфраструктурі) з посиланнями на відповідний кейс. Якщо основне сховище тимчасово недоступне, задіюється резервний файловий режим, у межах якого для кожного кейсу створюється окрема «папка» зі структурованим набором файлів, синхронізованих зі схемою основної бази даних.

На подальших етапах життєвого циклу завершений кейс може переходити до стану перегляду та аналізу. Через відповідні інтерфейси доступу (API або

адміністративні засоби) технічний персонал або аналітики можуть отримувати доступ до історії звернень, аналізувати відповідність діагностичних рішень експертним оцінкам, відстежувати стабільність роботи системи та виявляти типові помилки. У перспективі передбачається можливість формального введення стану архівації, за якого старі кейси переводитимуться до окремого розділу сховища з метою розвантаження оперативної бази даних без втрати історичної інформації.

Таким чином, життєвий цикл кейсу охоплює щонайменше такі логічні стани: «ініційований» (створення запиту), «у обробленні» (виконання діагностичного конвеєра), «завершений» (збереження результатів) та «використовуваний» (подальший перегляд і аналіз). Визначення цих станів і відповідних переходів створює основу для реалізації формалізованих механізмів аудиту, архівації та керування життєвим циклом даних.

5.6.2 Життєвий цикл бази знань та конфігурації системи

База знань про хвороби та конфігураційні параметри сервісу належать до класу відносно стабільних даних, які, однак, потребують періодичного оновлення. Для них також доцільно розглядати окремий життєвий цикл, що включає етапи підготовки, розгортання, експлуатації та модифікації.

На стадії підготовки бази знань формуються та редагуються картки хвороб у текстовому форматі з фіксованою структурою полів (ідентифікація, симптоматика, візуальні патерни, обмеження за культурами й фазами розвитку, рекомендації щодо дій тощо). Ця робота, як правило, виконується фахівцями предметної області з використанням допоміжних інструментів контролю версій.

Стадія розгортання полягає в завантаженні оновлених карток до робочого середовища сервісу та їх підключенні до діагностичного конвеєра. У поточній реалізації це здійснюється шляхом оновлення набору текстових файлів у структурі проєкту та перезапуску сервісу, після чого база знань повторно завантажується у пам’ять і стає доступною для запитів. На цьому ж етапі застосовуються зміни у конфігураційних параметрах, що визначають режим роботи підсистем

комп’ютерного зору, способи зберігання кейсів, рівень деталізації відлагоджувальної інформації тощо.

У фазі експлуатації база знань використовується як довідковий ресурс для всіх діагностичних запитів, а конфігурація — як джерело параметрів для роботи окремих компонентів. При цьому результати опрацювання реальних кейсів утворюють зворотний зв’язок: аналіз виявлених помилок, типових неправильних діагнозів або систематичних зміщень у рекомендаціях може виявити необхідність корекції описів хвороб, доповнення переліку візуальних патернів або уточнення агрономічних обмежень.

Стадія модифікації передбачає внесення змін до змісту бази знань і конфігураційних файлів з подальшим повторенням циклу «підготовка — розгортання — експлуатація». Таким чином забезпечується еволюційний розвиток інформаційної складової системи без зміни її програмного каркаса: додавання нових хвороб, розширення підтримуваних культур, уточнення рекомендацій тощо відбувається переважно на рівні даних.

5.6.3 Експлуатаційний цикл сервісу діагностики

Життєвий цикл сервісу діагностики як програмної системи визначається послідовністю етапів розгортання, штатної роботи, моніторингу, оновлення та реакції на відмови. Цей цикл тісно пов’язаний із моделлю розгортання (локальною або хмарною) та обраними засобами інфраструктурної підтримки.

На етапі розгортання виконується підготовка середовища виконання: встановлення залежностей, налаштування доступу до бази даних та зовнішніх сервісів, параметризація режимів роботи конвеєра. У хмарних сценаріях додатково конфігуруються керовані сервіси зберігання даних і об’єктні сховища для зображень, а також інтеграція з сервісами комп’ютерного зору та мовних моделей.

У фазі штатної експлуатації сервіс працює як довготривалий процес, що приймає запити діагностики, ініціює виконання конвеєра, зберігає результати у сховищі кейсів та повертає відповіді користувачам. Паралельно збираються

відлагоджувальні й експлуатаційні метрики: час оброблення окремих етапів, кількість оброблених запитів, статистика використання зовнішніх сервісів тощо. Наявність таких даних є передумовою для подальшого вдосконалення продуктивності та надійності системи.

Моніторинг і супровід включають періодичний контроль доступності сервісу, аналіз журналів подій і відлагоджувальних даних, а також перевірку коректності інтеграції з хмарними сервісами та базою даних. У разі виявлення збоїв або деградації продуктивності можуть застосовуватися оперативні заходи: переключення на резервний режим збереження кейсів, тимчасове відключення окремих зовнішніх сервісів, рестарт або масштабування компонентів у хмарному середовищі.

Окремою стадією життєвого циклу є оновлення версій програмного забезпечення. Воно охоплює внесення змін до коду, конфігурацій і, за потреби, до структури сховища даних з подальшим повторним розгортанням сервісу. При цьому важливо забезпечити узгодженість між новими версіями компонентів та вже накопиченою історією кейсів, що може вимагати виконання міграційних процедур для бази даних або перетворень файлових структур.

Узагальнена схема відображення системи в хмарному середовищі AWS показано у Додатку К.

5.7 Обробка та передача даних

Обробка та передача даних у прототипі системи діагностики хвороб сільськогосподарських культур організовані таким чином, щоб забезпечити узгодженість форматів на всіх етапах проходження діагностичного запиту — від введення даних користувачем до збереження результатів у сховищах. Основними типами даних, з якими працює система, є структуровані JSON-повідомлення, зображення у стандартних графічних форматах, текстові картки хвороб у форматі YAML, а також записи в реляційній базі даних із використанням структурованих полів. Узагальнений потік даних зображено у Додатку И.

5.7.1 Формати даних на основних інтерфейсах

На межі «користувач — інформаційна система» як основний формат використовується HTTP-запит із структурованими параметрами. Користувацький вебінтерфейс формує звернення до серверної частини у вигляді запиту, що містить поля для зазначення культури, текстового опису симптомів, додаткових метаданих (фаза розвитку, за потреби — інші агротехнічні параметри), а також один або кілька файлів-зображень. Для передавання зображень застосовується стандартне кодування вебформ із підтримкою бінарних вкладень, тоді як текстові параметри можуть бути представлені як частина тіла запиту у форматі JSON або як окремі поля форми.

Відповідь сервісу діагностики повертається у форматі JSON. Структура цього повідомлення включає унікальний ідентифікатор діагностичного кейсу, упорядкований список кандидатних хвороб із числовими оцінками їхньої відповідності, короткими текстовими поясненнями та посиланнями на відповідні картки бази знань. Окремим блоком подається структурований план дій для найбільш імовірної хвороби, поділений на діагностичні кроки, агротехнічні заходи, а також, за наявності, рекомендації щодо хімічних та біологічних засобів захисту. Додатково у відповідь можуть включатися службові поля з попередженнями та узагальненою відлагоджувальною інформацією про перебіг діагностичного конвеєра.

База знань про хвороби зберігається у вигляді набору текстових файлів формату YAML, згрупованих за культурами. Кожна картка хвороби містить структуровані поля, що відображають ідентифікаційні атрибути, опис симптомів, візуальні патерни, агрономічні обмеження та рекомендації щодо реагування. Під час ініціалізації сервісу ці файли завантажуються та перетворюються у внутрішні об’єкти, з якими надалі працюють модулі пошуку та формування рекомендацій.

У реляційній базі даних для зберігання діагностичних кейсів використовуються як стандартні скалярні поля (ідентифікатор, культура, текст

опису симптомів, часові мітки), так і структуровані поля, що дозволяють зберігати складні об’єкти у вигляді вкладених структур. У таких полях фіксуються списки кандидатних хвороб з оцінками, сформований план дій та узагальнені відлагоджувальні дані. Зображення можуть зберігатися або безпосередньо як бінарні поля, або опосередковано — у вигляді посилань на об’єкти в зовнішньому сховищі.

У резервному файловому режимі для кожного діагностичного кейсу створюється окремий каталог, у якому розміщуються файли request.json та response.json з копіями вхідного запиту та відповіді сервісу, а також підкаталог із зображеннями, де зберігаються файли у вихідних графічних форматах. Така організація дозволяє уніфікувати роботу з даними між реляційним сховищем і файловою системою та полегшує подальший аудит і відтворення діагностичних рішень.

5.7.2 Потоки даних у межах системи

Узагальнений потік даних між основними компонентами системи відповідає типовій моделі клієнт-серверного вебзастосунку з виділеним діагностичним конвеєром і підсистемами зберігання.

На першому етапі дані передаються від користувацького інтерфейсу до серверного API. Вебінтерфейс збирає введені користувачем параметри й формує HTTP-запит до спеціалізованого ендпоїнта діагностики. На цьому етапі у тілі запиту одночасно присутні текстові поля, що описують культуру та симптоми, та один або кілька файлів-зображень. Серверний компонент приймає запит, виконує формальну валідацію структури та типів даних, перетворює їх у внутрішнє представлення у вигляді об’єкта діагностичного контексту та ініціює роботу конвеєра.

На наступному етапі відбувається внутрішній обмін даними між етапами діагностичного конвеєра. Підсистема комп’ютерного зору одержує зображення у вигляді масиву байтів або стандартного об’єкта зображення та повертає

узагальнений вектор візуальних ознак — набір індикаторів, що відображають наявність певних патернів (світлий наліт, темні водянисті плями, дрібні некротичні ушкодження тощо). Паралельно з цим текстовий опис симптомів залишається доступним у складі того самого контексту.

Модуль пошуку у базі знань одержує на вхід текст опису симптомів, вектор візуальних ознак та підмножину карток хвороб для відповідної культури, завантажених з YAML-файлів. У процесі оброблення для кожної картки обчислюється числова оцінка відповідності, після чого формується набір проміжних структур, що містять ідентифікатори хвороб, значення оцінок та службову інформацію. Ці структури передаються до блоку доменних правил, який коригує оцінки з урахуванням агрономічного контексту та формує впорядкований список кандидатів.

На завершальних етапах конвеєра модуль формування пояснень і планів дій одержує впорядкований список кандидатних хвороб разом з посиланнями на їх картки та формує структуровані текстові пояснення і план дій для найбільш імовірної хвороби. Ці дані разом із узагальненою службовою інформацією включаються до об’єкта діагностичного контексту, який надалі використовується і для формування відповіді користувачеві, і для збереження кейсу у сховищі.

Фінальним кроком є передавання даних до підсистеми зберігання. На цьому етапі частина полів контексту відображається на реляційні структури бази даних (ідентифікатор кейсу, культура, текст опису симптомів, структуровані поля із списком кандидатів, планом дій та службовою інформацією), а зображення зберігаються або у відповідних таблицях, або в зовнішньому сховищі з реєстрацією посилань у базі даних. У разі використання файлового сховища ті самі дані записуються до JSON-файлів та графічних файлів у окремому каталозі, що відповідає конкретному кейсу. Формат відповіді користувачеві формується на основі того самого контексту шляхом відбору полів, релевантних для відображення у вебінтерфейсі.

5.7.3 Взаємодія з зовнішніми сервісами аналізу даних

Окремий напрямок обробки та передавання даних пов’язаний із взаємодією системи з зовнішніми хмарними сервісами — насамперед сервісом аналізу зображень і сервісом мовних моделей. Ця взаємодія організована таким чином, щоб мінімізувати залежність внутрішніх структур даних від конкретних API сторонніх провайдерів.

Для використання сервісу комп’ютерного зору вихідні зображення, отримані від користувача, перетворюються у масиви байтів стандартного графічного формату та надсилаються до зовнішнього API згідно з його вимогами. Відповідь сервісу повертається у форматі JSON, що містить перелік виявлених об’єктів або класів ушкоджень із числовими оцінками впевненості. На рівні внутрішніх структур системи ця відповідь перетворюється на набір додаткових ознак — значень, що пов’язують ідентифікатори зовнішніх класів із відповідними картками хвороб з бази знань. Надалі ці ознаки розглядаються як частина вектора візуальних характеристик і обробляються конвеєром нарівні з локально обчисленими патернами.

У випадку використання сервісу мовних моделей до нього передається узагальнений контекст діагностики: культура, текстовий опис симптомів, впорядкований список гіпотез із оцінками відповідності та, за потреби, витяги з карток хвороб. Цей контекст кодується у вигляді текстового запиту та/або JSON-структури відповідно до інтерфейсу зовнішнього сервісу. Відповідь мовної моделі зазвичай повертається у форматі JSON або структурованого тексту, який містить пояснення до гіпотез і уточнений план дій. На внутрішньому рівні ці дані інтегруються в уже наявну структуру діагностичної відповіді, причому зовнішня модель розглядається лише як джерело формулювань, а не як окремий елемент, що змінює логіку прийняття рішень.

Важливою особливістю взаємодії з зовнішніми сервісами є те, що в основних сховищах системи не фіксуються детальні технічні деталі протоколів обміну даними, а лише узагальнені результати їх роботи: візуальні ознаки, що

підтримують окремі гіпотези, текстові пояснення та структуровані рекомендації. Це дозволяє, з одного боку, зберігати необхідну інформацію для відтворюваності та аудиту діагностичних рішень, а з іншого — не дублювати внутрішні формати зовнішніх API у власній інформаційній моделі.

Узагальнюючи, можна відзначити, що обробка та передача даних у розробленій системі організовані на основі чітко визначених форматів і потоків, що охоплюють усі етапи — від введення інформації користувачем, її перетворення в межах діагностичного конвеєра та взаємодії з зовнішніми сервісами до збереження результатів у сховищах і відображення їх у користувацькому інтерфейсі. Це забезпечує узгодженість і відтворюваність роботи системи та створює основу для подальшого розширення її функціональності.

5.8 Поведінкова модель системи

Поведінкова модель системи діагностики хвороб сільськогосподарських культур описує динаміку взаємодії її компонентів у часі під час опрацювання типових запитів. На відміну від статичних структурних моделей, що фіксують склад і зв’язки між елементами, поведінкова модель фокусується на послідовності викликів, переходах між станами та варіантах гілкування залежно від конфігурації та результатів виконання окремих операцій.

5.8.1 Типовий сценарій діагностики одного випадку

Базовим сценарієм, що визначає поведінку системи, є опрацювання одного запиту на діагностику, сформованого користувачем. У часовому вимірі цей сценарій можна подати як послідовність взаємодій між такими логічними учасниками: користувач, користувацький інтерфейс, серверний API, діагностичний конвеєр, підсистеми комп’ютерного зору та пошуку в базі знань, блок доменних правил, модуль формування пояснень, підсистема зберігання кейсів і, за потреби, зовнішні хмарні сервіси.

Процес починається з того, що користувач у вебінтерфейсі формує запит: обирає культуру, за потреби зазначає фазу розвитку, вводить текстовий опис симптомів, завантажує одне або кілька зображень уражених рослин і ініціює надсилання запиту. Користувацький інтерфейс перетворює введені дані на HTTP-звернення до серверного ендпоїнта діагностики.

Серверний API приймає запит, виконує первинну валідацію структури та типів даних, генерує унікальний ідентифікатор діагностичного випадку та створює внутрішній діагностичний контекст. Далі запускається діагностичний конвеєр, який працює над цим контекстом.

На першому етапі конвеєра підсистема комп'ютерного зору отримує зображення з контексту та виконує над ними серію операцій: локальний аналіз для виявлення узагальнених візуальних патернів, а в розширеному режимі — додаткові виклики зовнішнього сервісу комп'ютерного зору. Результатом цього етапу є вектор візуальних ознак, який додається до контексту.

На другому етапі модуль пошуку в базі знань завантажує релевантні картки хвороб для заданої культури, порівнює текстовий опис симптомів і візуальні ознаки з інформацією в картках і формує для кожної хвороби числову оцінку відповідності. Сформований набір кандидатних хвороб із проміжними оцінками передається до блоку доменних правил.

На третьому етапі блок доменних правил коригує оцінки з урахуванням агрономічних обмежень (культура, фаза розвитку, характерний перебіг). Частина гіпотез може бути відсіяна, а інші — підвищити або знизити свої оцінки. У результаті формується впорядкований список кандидатних хвороб, упорядкований за скоригованими оцінками відповідності.

На четвертому етапі модуль формування пояснень і планів дій отримує впорядкований список кандидатів і відповідні картки з бази знань. У базовому режимі пояснення та план дій формуються на основі структурованих полів карток із використанням шаблонів; у розширеній конфігурації можливе додаткове звернення до сервісу мовних моделей, який повертає уточнені формулювання. Отримані пояснення та план дій додаються до діагностичного контексту.

На п’ятому етапі підсистема зберігання кейсів створює запис у сховищі: з діагностичного контексту відбираються поля, що мають бути довготривало збережені (вхідні параметри, перелік кандидатів, план дій, узагальнені технічні метадані), а зображення зберігаються в обраному фізичному сховищі з реєстрацією посилань у структурі кейсу.

Після успішного завершення збереження формується відповідь для користувача у форматі JSON, що містить ідентифікатор кейсу, список кандидатних хвороб із оцінками та поясненнями, структурований план дій і, за потреби, короткі службові повідомлення. Вебінтерфейс отримує цю відповідь і відображає результати у зручному для сприйняття форматі.

Таким чином, у типовому сценарії поведінкова модель системи визначається послідовним проходженням діагностичного контексту через низку підсистем із чітко визначеним порядком викликів і передаванням проміжних результатів від одного етапу до іншого.

5.8.2 Варіанти поведінки залежно від конфігурації

Поведінка системи у часі залежить від обраної конфігурації, насамперед від того, чи використовуються зовнішні хмарні сервіси та який режим зберігання даних увімкнено.

У локальному режимі підсистема комп’ютерного зору обмежується аналізом зображень засобами локальних алгоритмів, без виклику зовнішнього API. Це означає, що етап формування візуальних ознак складається лише з локальних операцій, а гілка, пов’язана з викликом хмарного сервісу, у поведінковій моделі відсутня. Аналогічно, модуль формування пояснень працює в шаблонному режимі, не надсилаючи запити до сервісу мовних моделей. Уся обробка відбувається всередині застосунку, а зовнішні виклики обмежуються лише зверненням до бази даних або файлової системи.

У розширеному режимі до поведінкової моделі додаються додаткові гілки. Після локальної обробки зображень здійснюється виклик хмарного сервісу аналізу

зображень; на часовій осі діаграми послідовності це відображається як асинхронний або синхронний виклик із очікуванням результату та подальшою інтеграцією отриманих оцінок у вектор ознак. На етапі формування пояснень можливий додатковий виклик сервісу мовних моделей: спочатку формуються структуровані дані (список гіпотез, ключові ознаки, рекомендації), які передаються до зовнішнього сервісу, а потім його відповідь інтегрується в остаточне формулювання для користувача.

Ще одним джерелом варіативності є вибір режиму зберігання кейсів. Якщо використовується реляційна база даних, то після формування діагностичного контексту поведінкова модель передбачає послідовність викликів до підсистеми доступу до БД: створення запису діагностичного кейсу, створення записів для пов’язаних зображень та фіксація транзакції. Якщо ж конфігурація передбачає використання лише файлового сховища, діаграма послідовності включає іншу гілку: створення каталогу кейсу, запис JSON-файлів із запитом і відповіддю та збереження зображень у файловій системі.

Таким чином, поведінкова модель має дерево варіантів, у якому базовий стовбур сценарію (від формування запиту до повернення відповіді) є спільним, а окремі гілки додаються залежно від того, активовано локальний чи хмарний режим аналізу та який спосіб зберігання даних обрано.

5.8.3 Обробка відмов і виняткових ситуацій

Важливою складовою поведінкової моделі є реакція системи на відмови й виняткові ситуації. У прототипі передбачено кілька типових сценаріїв, що змінюють стандартний хід виконання.

По-перше, відмови на етапі приймання та валідації запиту. Якщо виявлено відсутність обов’язкових полів, некоректний формат зображень або інші критичні помилки вводу, серверний API не запускає діагностичний конвеєр, а формує відповідь із повідомленням про помилку. На діаграмі послідовності це

відображається як раннє завершення сценарію з поверненням помилки без переходу до внутрішніх підсистем.

По-друге, відмови зовнішніх сервісів. У разі недоступності сервісу аналізу зображень конвеєр може продовжити роботу, спираючись лише на локальні ознаки, а у випадку відсутності відповіді від сервісу мовних моделей — використати шаблонний механізм формування пояснень. У поведінковій моделі це відповідає альтернативним гілкам: замість переходу до зовнішнього учасника взаємодія повертається до внутрішнього механізму оброблення.

По-третє, відмови на етапі збереження кейсів. Якщо під час спроби запису в реляційну базу даних виникає помилка (недоступність сервера, порушення цілісності тощо), підсистема зберігання ініціює перехід до резервного файлового режиму. На діаграмі послідовності це відображається як розгалуження: невдала спроба запису в базу даних із подальшим викликом файлового сховища, де створюється каталог кейсу та зберігаються відповідні файли. При цьому зовнішня поведінка для користувача залишається незмінною: відповідь із результатами діагностики повертається навіть у разі відмови основного сховища.

Нарешті, у поведінковій моделі передбачається можливість накопичення відлагоджувальної інформації про виняткові ситуації. У разі виникнення помилок система доповнює діагностичний контекст відповідними службовими повідомленнями, які за наявності успішного збереження можуть бути зафіксовані у сховищі й використані під час подальшого аналізу надійності прототипу.

Узагальнено поведінкова модель системи демонструє, що навіть у мінімальній реалізації прототип не обмежується прямолінійною послідовністю операцій, а включає розгалуження залежно від конфігурації та механізми обробки відмов. Це є важливою передумовою для подальшого розвитку системи в напрямі підвищення її надійності, адаптивності та придатності до експлуатації в реальних умовах аграрного виробництва.

5.9 Реалізація системи

Реалізація прототипу системи діагностики хвороб сільськогосподарських культур виконується з урахуванням багатошарової архітектури, обґрунтованої у четвертому розділі. Програмна система складається з серверної частини, що реалізує діагностичний конвеєр і програмний інтерфейс взаємодії, користувацького вебінтерфейсу, підсистеми доступу до бази даних, бази знань у вигляді текстових карток та адаптерів до зовнішніх хмарних сервісів. Значна частина логіки реалізована мовою Python з використанням сучасних бібліотек для веброзробки, роботи з даними та інтеграції з інфраструктурою Amazon Web Services.

5.9.1 Особливості програмної реалізації

Серверна частина системи реалізована як вебсервіс на основі фреймворку FastAPI, що надає REST-інтерфейс для основних операцій. Ключовим ендпоїнтом є сервіс діагностики, який приймає дані про культуру, текстовий опис симптомів та зображення, ініціює виконання діагностичного конвеєра й повертає впорядкований список гіпотез із поясненнями та планом дій. Допоміжні ендпоїнти забезпечують перевірку стану сервісу та отримання інформації про раніше оброблені діагностичні кейси.

Формальна валідація запитів і побудова структурованих відповідей виконуються за допомогою моделей, описаних засобами бібліотеки Pydantic. Це дозволяє жорстко фіксувати формат даних на межі «клієнт — сервер», автоматично перевіряти наявність обов’язкових полів, типи та діапазони значень, а також генерувати зрозумілі повідомлення про помилки у випадку некоректних запитів.

Бізнесова логіка діагностики зосереджена у спеціалізованому модулі оркестратора, який реалізує діагностичний конвеєр як послідовність етапів оброблення єдиного діагностичного контексту. На вхід оркестратор отримує нормалізовані дані запиту, далі послідовно викликає підсистему комп’ютерного зору, механізм пошуку в базі знань, рушій доменних правил та модуль формування

пояснень. Завдяки такій структурі конвеєр залишається розширюваним: окремі етапи можуть модифікуватися або замінюватися без зміни загального протоколу взаємодії між компонентами.

Підсистема комп’ютерного зору реалізована у двох варіантах. У локальному режимі застосовуються прості операції попередньої обробки зображень і виділення високорівневих патернів (наявність світлого нальоту, темних водянистих ділянок, дрібних некротичних плям) на основі бібліотеки Pillow. У розширеному режимі передбачена інтеграція з сервісом AWS Rekognition: зображення передаються до хмарного класифікатора, який повертає ймовірнісні оцінки для заздалегідь визначених класів уражень. Обидва джерела інформації узгоджуються у вигляді єдиного набору візуальних ознак, що надалі використовуються під час пошуку в базі знань.

База знань про хвороби організована у вигляді текстових карток, структурованих мовою YAML. Для кожної хвороби зберігаються ідентифікаційні дані, опис симптомів природною мовою, узагальнені візуальні патерни, обмеження щодо культури та фази розвитку, а також рекомендації щодо реагування. Під час ініціалізації сервісу ці картки завантажуються й перетворюються у внутрішні моделі, з якими працює модуль пошуку. Сам пошук реалізує RAG-подібний підхід: поєднується ключовева відповідність між текстовим описом симптомів і полями картки з додатковими ваговими внесками, отриманими від підсистеми комп’ютерного зору [17].

Рушій доменних правил реалізує набір агрономічних правил, що уточнюють попередні оцінки відповідності хвороб: відсіюються хвороби, непритаманні заданій культурі, коригуються оцінки залежно від фази розвитку, можуть застосовуватися додаткові бонуси чи штрафи за наявність характерних патернів. Результатом роботи цього блоку є впорядкований список кандидатних хвороб, у якому одночасно враховано інформацію з бази знань, візуальних ознак та агрономічного контексту.

Модуль формування пояснень і планів дій реалізує два режими роботи. У базовому режимі пояснення і рекомендації конструюються шаблонним чином на

основі структурованих полів картки хвороби: ключові симптоми, типові умови розвитку, рекомендовані агротехнічні, хімічні та біологічні заходи. У розширеній конфігурації передбачено виклик сервісу мовних моделей AWS Bedrock, якому передається структурований контекст (культура, опис симптомів, перелік гіпотез із оцінками, витяги з карток). Отримана від мовної моделі відповідь інтегрується у загальну структуру відповіді, при цьому логіка вибору гіпотез і плану дій залишається на стороні діагностичного конвеєра.

Шар даних реалізовано із застосуванням ORM-підходу на базі SQLAlchemy. Моделі бази даних описують сутності діагностичного кейсу та пов'язаних із ним зображень. Застосовується асинхронний доступ до реляційної бази даних PostgreSQL із використанням драйвера asyncpg; міграції схеми виконуються засобами Alembic. Це дозволяє забезпечити узгодженість інформаційної моделі, описаної в підрозділі 5.3, з фізичною структурою сховища. Зображення зберігаються або безпосередньо в базі даних, або в об'єктному сховищі AWS S3 із фіксацією посилань у таблицях, залежно від конфігурації розгортання.

Користувацький інтерфейс реалізовано як вебзастосунок на базі Streamlit. Він надає засоби введення параметрів діагностичного запиту (культура, опис симптомів, зображення), надсилає їх до серверного REST-інтерфейсу та відображає у відповідь сформований системою список можливих хвороб і рекомендований план дій. У результаті фронтенд залишається тонким клієнтом, який не містить предметної логіки діагностики, а лише організовує взаємодію користувача із серверною частиною.

Конфігураційні параметри системи (режим локальної чи хмарної роботи, параметри підключення до бази даних, ключі доступу до сервісів AWS, увімкнення чи вимкнення інтеграцій) задаються через змінні середовища та файли налаштувань і завантажуються у єдину модель конфігурації. Це дозволяє реалізувати гнучкі сценарії розгортання без модифікації програмного коду.

Таким чином, програмна реалізація системи відображає архітектурні рішення, описані в попередніх розділах: серверна частина побудована як багатошаровий вебсервіс із чітким розмежуванням API-рівня, бізнесової логіки

діагностики, сховища даних і інтеграції з зовнішніми сервісами, тоді як користувацький інтерфейс реалізує єдиний вхідний канал для кінцевих користувачів.

5.9.2 Тестування та оцінювання якості системи

Тестування та оцінювання якості прототипу системи діагностики хвороб сільськогосподарських культур проводилося з метою перевірки коректності реалізації основних сценаріїв, а також отримання орієнтовних показників продуктивності, точності та масштабованості. З огляду на статус прототипу, акцент зроблено на критично важливих шляхах проходження діагностичного конвеєра та типових конфігураціях розгортання.

Автоматизоване тестування реалізовано на основі фреймворку pytest із використанням засобів підтримки асинхронних викликів. Тестові сценарії згруповані за основними підсистемами: API-рівень (перевірка структури запитів і відповідей, обробка помилкових даних), діагностичний конвеєр (послідовність етапів і коректність формування діагностичного контексту), підсистема комп’ютерного зору (формування візуальних ознак для контрольних зображень), пошук у базі знань і рушій доменних правил (стабільність результатів для фіксованих сценаріїв), підсистема зберігання кейсів (запис і читання даних у різних режимах).

Автоматизовані тести доповнено ручним сценарним тестуванням, під час якого перевірялися робота вебінтерфейсу, взаємодія з хмарними сервісами та коректність формування «папок кейсів» у базі даних і файловому сховищі. Формальні метрики покриття тестами у поточній версії не розраховувалися, однак тестовий набір охоплює основні бізнес-процеси, пов’язані з діагностикою одного випадку.

Оцінювання продуктивності здійснювалося за часом отримання повної відповіді на діагностичний запит за різних конфігурацій підключення зовнішніх сервісів. У «легкому» режимі, коли використовується лише локальний пошук у базі

знань без хмарного аналізу зображень та мовної моделі, час відповіді становить порядку десятків мілісекунд, що забезпечує майже миттєву реакцію системи. Додавання сервісу аналізу зображень збільшує час відповіді до сотень мілісекунд, а використання мовної моделі для формування розгорнутих пояснень — до одиниць секунд.

Отримані результати свідчать, що домінуючим чинником затримок є виклики мовної моделі, тоді як внесок локального пошуку та операцій роботи зі сховищем є порівняно незначним.

Формальне кількісне оцінювання точності діагностики на репрезентативній вибірці польових даних у межах роботи не проводилося. Натомість виконано низку орієнтовних експериментів на основі тестових зображень і синтетичних сценаріїв, які показали, що за умови чіткого формулювання симптомів і наявності підтримки з боку моделей комп'ютерного зору правильна хвороба, як правило, потрапляє до верхніх позицій у сформованому списку кандидатів. Для розпливчастих описів симптомів або культур, не охоплених навчальними даними хмарних сервісів, якість діагностики знижується, що окреслює природні обмеження поточного підходу.

Масштабованість системи на цьому етапі оцінювалася аналітично з урахуванням обраної інформаційної моделі та засобів зберігання. Використання реляційної бази даних PostgreSQL для структурованих даних і об'єктного сховища для зображень дозволяє нарощувати обсяг накопичених діагностичних кейсів без зміни логічної структури сховища; обмежувальними факторами є насамперед дискові ресурси та параметри індексації. Навантажувальне тестування з моделюванням великої кількості паралельних користувачів не проводилося, однак архітектурні рішення (асинхронний вебсервіс, можливість горизонтального масштабування та контейнеризації) задають резерв для подальшого підвищення пропускної спроможності у разі промислового впровадження.

5.10 Інструкція користувача

Розроблений прототип системи діагностики хвороб сільськогосподарських культур орієнтований на роботу у режимі вебзастосунку з тонким клієнтом. У цьому підрозділі подано короткі рекомендації щодо запуску системи у локальному середовищі, формування діагностичного запиту, інтерпретації отриманих результатів та типових сценаріїв її використання. Загальна інформація про варіанти використання системи користувачами у різних ролях наведена в Додатку Г.

Запуск системи у локальному середовищі без інтеграції із сервісами AWS передбачає попереднє встановлення інтерпретатора Python відповідної версії та інсталяцію необхідних бібліотек. Рекомендовано створити окреме віртуальне середовище, після чого встановити залежності з файлу вимог засобами стандартного менеджера пакетів. Конфігураційні параметри задаються через змінні середовища або файл налаштувань, при цьому у локальному режимі вимикаються інтеграції з AWS Rekognition та AWS Bedrock, а підсистема зберігання налаштовується на використання локальної бази даних або файлового сховища. Серверна частина запускається як вебсервіс на основі FastAPI (наприклад, через вебсервер uvicorn), користувацький вебінтерфейс — як окремий процес на основі Streamlit. Після запуску компонентів користувач отримує доступ до системи через веббраузер за локальною адресою.

Формування діагностичного запиту здійснюється через головну форму вебінтерфейсу. Користувач обирає культуру із запропонованого переліку, за потреби вказує додаткові параметри (наприклад, фазу розвитку рослин) та вводить текстовий опис спостережуваних симптомів українською мовою. Опис має бути максимально конкретним (вказувати характер плям, наявність нальоту, темп поширення ураження, залучені органи рослин тощо), оскільки саме на цих даних ґрунтується текстова частина діагностики.

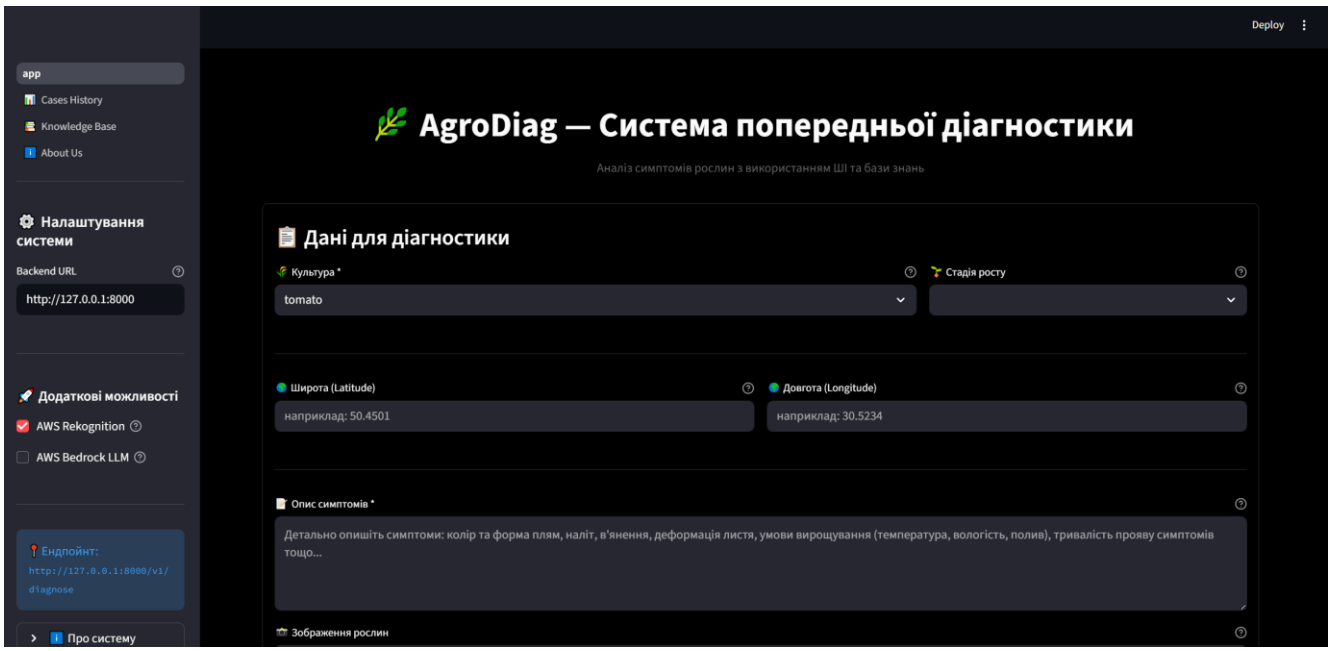

Рисунок 5.1 — Головне вікно вебінтерфейсу системи діагностики

Далі користувач завантажує одне або кілька зображень уражених рослин; рекомендовано використовувати поширені формати (JPEG, PNG), дотримуватися помірної роздільної здатності та уникати надто великих файлів, що можуть ускладнювати оброблення. Обов’язковими для коректної роботи системи є вказання культури, текстового опису симптомів та наявність хоча б одного зображення (у конфігураціях, де працює підсистема комп’ютерного зору). Після заповнення форми користувач ініціює діагностику натисканням відповідної кнопки.

Отримана від системи відповідь відображається у вебінтерфейсі у структурованому вигляді. Зазвичай вона містить:

— впорядкований список кандидатних хвороб із числовими оцінками відповідності (умовні бали або відсоткові значення, що відображають відносну впевненість системи);

— короткі пояснення для кількох найімовірніших гіпотез, побудовані на основі бази знань і результатів роботи діагностичного конвеєра;

— рекомендований план дій для хвороби з найвищою оцінкою, структурований за блоками (уточнювальні діагностичні кроки, агротехнічні заходи, за наявності даних — можливі хімічні та біологічні засоби захисту);

— застереження щодо необхідності врахування місцевих регуляторних вимог, інструкцій із безпечного застосування засобів захисту рослин та консультативний характер рекомендацій прототипу.

Під час інтерпретації результатів користувач має оцінювати список гіпотез комплексно. Висока числова оцінка певної хвороби означає, що опис симптомів і виділені візуальні ознаки добре узгоджуються з відповідною карткою бази знань, однак не замінює професійного агрономічного висновку. За необхідності користувач може використати пояснення та рекомендовані діагностичні кроки для додаткового польового обстеження, а у разі суттєвих розбіжностей між спостереженнями та пропозиціями системи — критично оцінити коректність вхідних даних (якість зображень, повнота опису симптомів).

Як типовий сценарій використання можна розглядати ситуацію польової діагностики. Фахівець господарства або фермер у полі фіксує симптоми ураження, робить кілька фотографій проблемних рослин, після повернення до офісу або за наявності мобільного доступу до локально розгорнутої системи заповнює форму діагностичного запиту, отримує перелік гіпотез і попередній план дій. Далі, з урахуванням власного досвіду та можливих додаткових оглядів, користувач приймає управлінське рішення щодо доцільності й терміновості застосування захисних заходів.

Другим сценарієм є аналіз накопичених діагностичних кейсів. У цьому разі система використовується не лише для безпосередньої діагностики, а й як інструмент фіксації історії звернень: за допомогою відповідних функцій серверної частини можна отримати інформацію про раніше оброблені випадки, порівняти їх із новими ситуаціями, оцінити час реакції системи та сталість діагностичних рішень. Такий режим є корисним для внутрішнього моніторингу, навчання персоналу та подальшого вдосконалення бази знань.

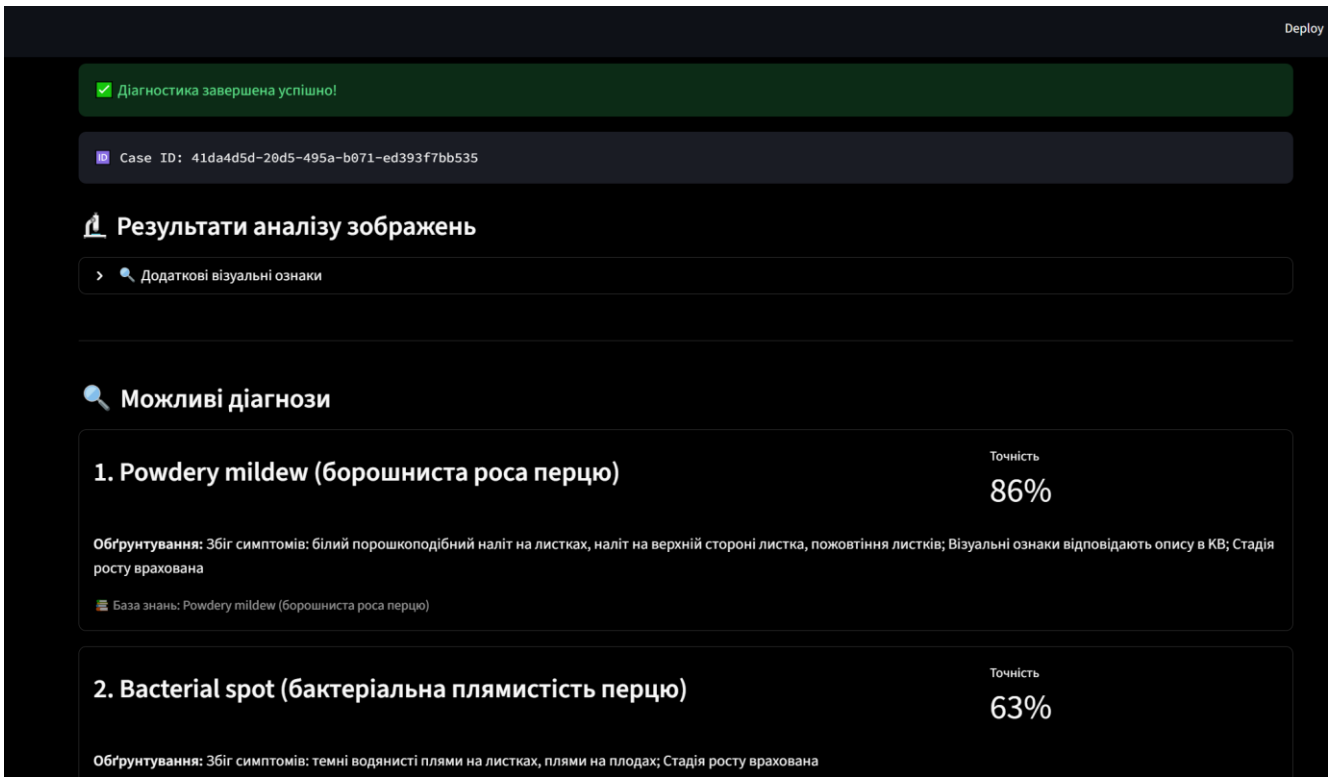

Рисунок 5.2 — Відображення результатів у веб інтерфейсі системи

Узагальнено, настанову користувача можна сформулювати як послідовність кроків: коректно ввести вихідні дані (культура, максимально конкретний опис симптомів, якісні зображення), запустити діагностику, уважно проаналізувати запропоновані гіпотези й план дій та використовувати результати системи як інструмент підтримки прийняття рішень, а не як єдине джерело остаточного агрономічного висновку. Це відповідає поточному статусу прототипу й забезпечує безпечне та обґрунтоване його застосування в умовах практичного рослинництва.

Висновки до розділу 5

У п’ятому розділі було представлено програмну реалізацію прототипу системи, що охоплює повний цикл діагностики: від обробки запиту до збереження результатів у структурованому сховищі кейсів. Архітектура рішення підтримує як автономне, так і хмарне розгортання (із залученням сервісів AWS), а алгоритмічне ядро поєднує пошук у базі знань, доменні правила та зовнішні моделі штучного інтелекту для аналізу мультимодальних даних. Така організація забезпечує не лише

формування обґрунтованих гіпотез, а й створення «цифрового сліду» для подальшого аудиту та вдосконалення системи.

Результати тестування підтвердили коректність виконання бізнес-процесів, прийнятну швидкодію та потенціал до масштабування. Водночас ідентифіковано низку обмежень, зокрема залежність якості рекомендацій від повноти вхідних даних та необхідність верифікації точності на репрезентативних польових вибірках. Загалом створений прототип довів практичну життєздатність запропонованих архітектурних рішень і слугує технічною основою для подальшого розвитку та впровадження системи в аграрному секторі.

6 РОЗРОБЛЕННЯ СТАРТАП-ПРОЄКТУ

6.1 Опис ідеї проєкту

Ідея стартап-проєкту полягає у створенні інформаційної системи діагностики хвороб сільськогосподарських культур, яка працює як «цифровий агроном-консультант». Користувач формує запит, задаючи культуру, текстовий опис симптомів і завантажуючи зображення уражених рослин, після чого система повертає впорядкований список можливих хвороб, пояснення причин такого вибору та структурований план дій. Ключовою особливістю є не лише автоматизована діагностика, а й пояснюваність результатів, збереження «папок діагностичних кейсів» та орієнтація на реальні умови роботи агровиробників, включно з обмеженим або нестабільним доступом до мережі.

Концепція продукту спирається на реалізований діагностичний конвеєр, що поєднує аналіз текстових описів симптомів, оброблення зображень, пошук у базі знань та доменні агрономічні правила. Передбачено можливість локальної роботи (без підключення до хмарних сервісів) та розширеної конфігурації з використанням сервісів AWS Rekognition і AWS Bedrock для покращення аналізу зображень та формування пояснень. Завдяки цьому продукт може позиціонуватися як гнучкий інструмент, придатний як для дослідної експлуатації в окремих господарствах, так і для масштабування до рівня сервісу для агрохолдингів або консультативних компаній.

Цільовими користувачами стартап-продукту є:

— фермерські та сімейні господарства, які не мають постійного доступу до висококваліфікованих агрономів, але потребують оперативних діагностичних підказок;

— середні та великі агропідприємства, зацікавлені у стандартизації процесів діагностики та формуванні єдиного «цифрового архіву» випадків ураження;

— агроконсалтингові компанії та дистриб’ютори засобів захисту рослин, які можуть використовувати систему як елемент сервісної підтримки клієнтів;

— освітні та дослідницькі установи, що потребують інструментів візуалізації й накопичення діагностичних кейсів для навчання й аналізу.

На відміну від низки існуючих мобільних застосунків, що виконують переважно «чорнову» класифікацію зображень, запропонована система орієнтована на поєднання візуального та текстового опису, використання формалізованої бази знань і збереження повного діагностичного контексту. Це дозволяє підвищити прозорість рішень, забезпечити можливість їх подальшого аудиту та інтегрувати систему в більш широкі процеси управління посівами й аналізу ризиків.

Структурований опис основних елементів ідеї стартап-проєкту наведено в таблиці 6.1.

Таблиця 6.1 — Структурий опис елементів ідеї стартапу

Елемент ідеї	Напрями застосування	Вигоди для користувача	Відмінність від існуючих рішень
Автоматизована діагностика за фото і текстом	Оперативна оцінка стану посівів у полі, у виробничих лабораторіях	Швидке отримання попереднього діагнозу без очікування виїзду спеціаліста	Поєднання текстового опису, зображень і бази знань, а не лише «фото-сканер»
Формування «папок діагностичних кейсів»	Документування історії уражень, аналіз ефективності заходів	Відтворюваність рішень, можливість повернутися до конкретного випадку, використання кейсів для навчання	Структуроване сховище кейсів з повним контекстом, а не тільки разові «знімки»

Елемент ідеї	Напрями застосування	Вигоди для користувача	Відмінність від існуючих рішень
Пояснювані рекомендації та план дій	Підтримка прийняття рішень щодо захисних заходів	Не лише назва хвороби, а й аргументація, на що звернути увагу та які кроки доцільні	Фокус на поясненості діагностики й структурованому плані дій
Гібридна локально-хмарна модель	Робота в умовах обмеженого Інтернету, а також у повністю хмарному середовищі	Гнучкість розгортання, можливість почати з локальної конфігурації та надалі підключити хмарні сервіси	Передбачений сценарій автономної роботи без обов’язкового постійного доступу до зовнішньої інфраструктури
Орієнтація на український агросектор	Підтримка української мови, адаптація до локальних культур та умов ведення господарства	Вища релевантність для користувачів, які працюють із локальними культурами та нормативними обмеженнями	Локалізація бази знань та інтерфейсу з урахуванням специфіки національного агросектору

Для подальшого аналізу доцільно виокремити техніко-економічні характеристики ідеї проєкту та оцінити їх як з погляду потенційних конкурентів, так і з точки зору сильних та слабких сторін. Узагальнення відповідних характеристик наведено в таблиці 6.2.

Таблиця 6.2 — Визначення сильних і слабких характеристик ідеї проєкту

Техніко-економічна характеристика	Потенційні конкуренти	W — слабка сторона	S — сильна сторона
Використання гібридного діагностичного конвеєра	Мобільні застосунки типу Plantix, Plant.id	Потреба в коректному заповненні текстового опису	Більш пояснювані та контрольовані результати
Наявність структурованого сховища діагностичних кейсів	Системи управління полем, окремі CRM-рішення	Необхідність організації захищеного зберігання даних	Можливість аналізу історії уражень, навчання персоналу, підтримка аудиту та покращення бази знань
Підтримка локального та хмарного режимів роботи	Переважно суто хмарні рішення	Складніша конфігурація розгортання	Адаптивність до умов обмеженого доступу до Інтернету, зниження вхідного бар’єру для невеликих господарств
Інтеграція з AWS Rekognition і мовними моделями (опційно)	Спеціалізовані хмарні сервіси аналізу зображень	Вартість використання хмарних сервісів	Підвищення точності діагностики без самостійного навчання глибоких моделей
Орієнтація на український аграрний ринок та локальні культури	Глобальні рішення з обмеженою локалізацією	Відносно вужчий початковий ринок	Вища релевантність для місцевих користувачів

Таким чином, ідея стартап-проєкту базується на поєднанні технологічно здійсненного прототипу, реальної потреби аграрного сектору в інструментах підтримки прийняття рішень та можливості поступового масштабування від локальних інсталяцій до хмарного сервісу.

6.2 Технологічний аудит ідеї проєкту

Технологічний аудит дає змогу оцінити, наскільки ідея стартап-проєкту технічно здійсненна на основі сучасних засобів розроблення програмного забезпечення, хмарних сервісів та наявної реалізації прототипу. У межах цього підрозділу аналізуються ключові компоненти системи діагностики хвороб сільськогосподарських культур з погляду зрілості використаних технологій, їх доступності, масштабованості та придатності для подальшого промислового впровадження.

Реалізований прототип побудовано на сучасному стеку відкритих технологій: мова програмування Python використовується для реалізації серверної логіки, вебфреймворк FastAPI — для побудови REST-інтерфейсу, бібліотеки SQLAlchemy та драйвери PostgreSQL — для роботи з реляційною базою даних, Streamlit — для створення вебінтерфейсу. Підсистема бази знань організована на основі текстових файлів формату YAML, що спрощує підтримку й розширення знань фахівцями предметної області. Для інтеграції з хмарними сервісами аналізу зображень та мовними моделями застосовуються клієнтські бібліотеки AWS (зокрема, для сервісів Rekognition і Bedrock). Такий вибір технологій відповідає усталеній практиці розроблення веборієнтованих систем підтримки прийняття рішень і не вимагає використання рідкісних або експериментальних засобів.

Окремо слід відзначити доступність засобів контейнеризації та інфраструктурної автоматизації, які можуть бути залучені на наступних етапах розвитку стартапу. Поточна реалізація прототипу вже орієнтована на можливість розгортання у вигляді окремих сервісів (API, вебінтерфейс, база даних), що є природною передумовою для використання Docker-контейнерів та оркестраторів

на кшталт Kubernetes або керованих рішень (наприклад, Amazon EKS). Хоча такі засоби можуть не бути повністю задіяними на стадії мінімально життєздатного продукту, їх застосування не потребує фундаментальної перебудови архітектури.

Результати узагальненого технологічного аудиту основних компонентів ідеї проєкту наведено в таблиці 6.3.

Таблиця 6.3 — Результати технологічного аудиту стартап-проєкту

Компонент / функція системи	Технології реалізації (приклади)	Наявність технологій	Доступність технологій
Серверний API та діагностичний конвеєр	Python, FastAPI, внутрішні сервісні модулі	Висока	Відкритий код, велика спільнота, докладна документація
Користувацький вебінтерфейс	Streamlit, HTTP(S)-доступ через браузер	Висока	Відкриті бібліотеки, низький поріг входу для розробників
Реляційне зберігання структурованих даних	PostgreSQL, SQLAlchemy, async-драйвери	Висока	Відкритий код, підтримка керованих сервісів (зокрема, Amazon RDS)
Сховище діагностичних кейсів у файловій системі	Стандартні засоби файлової системи, серіалізація у форматі JSON	Висока	Не потребує спеціалізованих засобів, доступно у більшості цільових середовищ
База знань про хвороби	YAML-файли, власні структури оброблення	Висока	Використовуються відкриті формати, підтримка редагування без зміни програмного коду

Компонент / функція системи	Технології реалізації (приклади)	Наявність технологій	Доступність технологій
Оброблення зображень у локальному режимі	Pillow та базові алгоритми комп’ютерного зору	Висока	Відкриті бібліотеки, можливість запуску на типових серверних конфігураціях
Хмарний аналіз зображень (опційно)	AWS Rekognition, клієнтські бібліотеки AWS	Висока (для користувачів сервісу AWS)	Комерційний сервіс, доступний у більшості регіонів, тарифікація за обсягом використання
Формування пояснень на основі мовних моделей (опційно)	AWS Bedrock, інтеграція з LLM-клієнтом	Середня–висока (залежно від регіону й моделей)	Комерційний сервіс, керований доступ, потребує облікового запису та належної конфігурації
Контейнеризація та оркестрація	Docker, потенційно Kubernetes / Amazon EKS	Висока	Де-факто стандарт для промислового розгортання, доступні керовані платформи
Засоби моніторингу й журналювання	Стандартні бібліотеки логування	Висока	Відкриті інструменти, інтеграція з поширеними системами моніторингу

З наведеного огляду видно, що всі ключові елементи ідеї стартап-проєкту можуть бути реалізовані на основі зрілих і доступних технологій, значна частина яких уже використана в прототипі. Потенційні технологічні ризики пов’язані переважно з вартістю та регіональною доступністю окремих хмарних сервісів, а також із необхідністю забезпечення надійної інфраструктури розгортання й моніторингу при масштабуванні до значної кількості користувачів. Водночас відсутність критичної залежності від власних глибоких моделей машинного навчання, можливість локальної роботи та використання відкритих форматів даних знижують бар’єри входу та підвищують технологічну стійкість стартап-проєкту.

6.3 Аналіз ринкових можливостей запуску стартап-проєкту

Аналіз ринкових можливостей запуску стартап-проєкту, що ґрунтується на системі діагностики хвороб сільськогосподарських культур, покликаний оцінити потенціал попиту, структуру цільових сегментів, а також основні зовнішні фактори, які можуть сприяти або перешкоджати комерціалізації продукту. Особливістю такого аналізу є поєднання загальних тенденцій розвитку цифрових технологій в аграрному секторі з урахуванням специфіки діагностики хвороб рослин, сезонності робіт і неоднорідності структури агровиробників.

Сегмент цифрових рішень для агросектору характеризується зростанням інтересу до систем підтримки прийняття рішень, моніторингу посівів, аналізу супутникових та дронових знімків, а також до інструментів, що знижують залежність від окремих експертів. Водночас рівень цифровізації різних категорій виробників суттєво відрізняється: великі агрокомпанії активніше впроваджують складні IT-рішення, тоді як малі та середні господарства часто обмежені в ресурсах та експертизі, але мають гостру потребу в практичних інструментах діагностики й консультування.

Запропонований продукт орієнтований на нішу прикладної діагностики хвороб культур за фото та текстовим описом із пояснюваними рекомендаціями. Ця ніша частково перетинається з функціональністю окремих мобільних застосунків і

сервісів, проте суттєво відрізняється завдяки фокусу на агрономічному контексті, збереженні «папок кейсів» та підтримці гібридного режиму роботи (локального й хмарного).

Узагальнену характеристику ринкового середовища стартап-проєкту наведено в таблиці 6.4.

Таблиця 6.4 — Попередня характеристика потенційного ринку стартап-проєкту

Показник стану ринку	Характеристика
Рівень цифровізації агросектору	Нерівномірний: високий у великих підприємств, середній або низький у малих і середніх господарств
Наявність існуючих рішень для діагностики хвороб	Присутні мобільні застосунки та окремі онлайн-сервіси, переважно орієнтовані на глобальний ринок
Доступність інтернет-з’єднання в полі	Обмежена або нестабільна в частини господарств, особливо у віддалених регіонах
Кадрова забезпеченість кваліфікованими агрономами	Дефіцит профільних фахівців, нерівномірний розподіл експертизи між регіонами та категоріями господарств
Чутливість до втрат урожаю через хвороби	Висока: хвороби можуть спричиняти значні економічні збитки, особливо у високорентабельних культурах
Готовність до впровадження інновацій	Зростаюча, але залежна від розміру господарства, доступу до консультацій та довіри до цифрових рішень
Конкурентне середовище	Наявні міжнародні гравці та універсальні платформи, локалізовані рішення розвинені слабше

Структура потенційних клієнтів стартап-проєкту є неоднорідною й охоплює як кінцевих виробників, так і сервісні та посередницькі організації. Характеристику основних груп наведено в таблиці 6.5.

Таблиця 6.5 — Характеристика потенційних клієнтів стартап-проєкту

Потреба, що формує ринок	Цільова аудиторія	Відмінності у поведінці цільових груп	Вимоги споживачів до продукту
Оперативна діагностика хвороб на полі	Малі фермерські та сімейні господарства	Обмежені ресурси, нерегулярний доступ до агрономів, висока чутливість до ціни	Простота використання, можливість роботи при слабкому інтернеті
Стандартизація діагностики та накопичення історії випадків	Середні та великі агропідприємства	Більш формалізовані процеси, наявність IT-підрозділів, більші вимоги до інтеграцій	Масштабованість, інтеграція з внутрішніми системами, надійність зберігання даних
Підтримка клієнтів і післяпродажний сервіс	Агроконсалтингові компанії, дистриб’ютори ЗЗР	Орієнтація на сервісну диференціацію, потреба у швидкій обробці великої кількості кейсів	Брендування, можливість спільного використання кейсів з клієнтами
Навчання та аналіз типових випадків	Освітні та наукові установи	Фокус на накопиченні знань та візуалізації, менша чутливість до комерційних умов	Аналітичні можливості, прозорість алгоритмів

На ринкові перспективи істотно впливають зовнішні загрози, пов’язані з технологічними, економічними та поведінковими факторами. Їх узагальнено у таблиці 6.6.

Таблиця 6.6 — Основні фактори загроз для стартап-проєкту

Фактор	Зміст загрози	Можлива реакція компанії
Конкуренція з боку глобальних мобільних застосунків	Наявність рішень, що вже впізнавані на ринку й мають базу користувачів	Диференціація через пояснюваність, локалізацію, роботу в гібридному режимі та фокус на «папках діагностичних кейсів»
Низька цифрова готовність частини фермерів	Обмежена готовність змінювати звичні практики, недовіра до автоматизованої діагностики	Спрощення інтерфейсу, навчальні матеріали, демонстраційні проєкти з партнерами, залучення аграрних консультантів
Залежність від інтернет-доступу й хмарних сервісів	Неможливість стабільного використання хмарних компонентів у віддалених регіонах	Підтримка повноцінного локального режиму роботи, можливість асинхронної синхронізації даних
Вартість використання хмарних сервісів	Зростання операційних витрат при масштабуванні кількості запитів із використанням Rekognition/LLM	Оптимізація режимів використання, кешування результатів, диференційовані тарифи, можливість відмови від хмарних модулів
Регуляторні обмеження щодо рекомендацій із ЗЗР	Необхідність врахування локальних норм і заборон щодо засобів захисту рослин	Введення загальних застережень, адаптація бази знань до регіональних норм, співпраця з локальними експертами

Водночас ринкове середовище містить низку можливостей, на яких може базуватися стратегія розвитку стартапу. Основні з них наведено в таблиці 6.7.

Таблиця 6.7 — Основні фактори ринкових можливостей

Фактор	Зміст можливості	Можлива реакція компанії
Тренд на цифровізацію агросектору	Зростання попиту на цифрові інструменти аналізу стану посівів та підтримки прийняття рішень	Позиціонування продукту як елемента цифрової трансформації господарств, інтеграція з іншими агротех-рішеннями
Дефіцит профільних агрономів	Потреба компенсувати нестачу експертів за допомогою інструментів підтримки рішень	Акцент на ролі системи як «цифрового асистента», співпраця з консультантами і навчальними центрами
Поширення хмарних сервісів та доступність інфраструктури	Спрощення розгортання й масштабування інформаційних систем для середніх і великих підприємств	Використання керованих хмарних сервісів, пропозиція рішення як сервісу (SaaS) для організаційного сегмента
Інтерес до даних та аналітики в агросекторі	Зростання цінності історичних даних про ураження та ефективність заходів	Розвиток функцій аналітики на основі «папок кейсів», можливість побудови додаткових сервісів на зібраних даних
Можливості партнерств із дистриб’юторами ЗЗР й агросервісами	Зацікавленість у додаткових сервісах для клієнтів та диференціації на ринку	Моделі спільного брендування, інтеграція у сервісні пакети партнерів, спільні пілотні проєкти

Узагальнюючи, аналіз ринкових можливостей свідчить, що стартап-проєкт на основі системи діагностики хвороб сільськогосподарських культур відповідає

актуальним тенденціям розвитку агротехнологій та цифрових сервісів. Водночас для успішного виходу на ринок необхідно врахувати наявні загрози, пов’язані з конкуренцією, цифровою готовністю цільової аудиторії, вартістю хмарних сервісів та регуляторними обмеженнями, і закласти механізми їх пом’якшення у подальшу ринкову стратегію, що розглядається у наступному підрозділі.

6.4 Розроблення ринкової стратегії продукту

Ринкова стратегія стартап-проєкту, що базується на системі діагностики хвороб сільськогосподарських культур, має враховувати неоднорідність цільової аудиторії, наявність глобальних конкурентів, технологічні обмеження та специфіку аграрного ринку. З огляду на результати попереднього аналізу доцільно орієнтуватися на поетапний вихід на ринок: початкове впровадження в сегменті невеликих і середніх господарств, пілотні впровадження з сервісними організаціями та подальший розвиток у напрямі хмарної сервісної моделі для більших агропідприємств і партнерських мереж.

Важливим аспектом успішної реалізації цієї стратегії є побудова довірчих відносин зі споживачами, оскільки аграрний сектор характеризується певним консерватизмом щодо впровадження нових технологій. Тому на ранніх етапах акцент робиться не стільки на масових продажах, скільки на демонстрації надійності та прозорості діагностичних рішень через пілотні проєкти та пряму взаємодію з клієнтами. Такий підхід дозволить не лише накопичити базу реальних успішних кейсів для маркетингового просування, а й удосконалити алгоритми системи на основі зворотного зв’язку, сформувавши репутацію експертного партнера як фундамент для подальшого масштабування бізнесу.

Ключовими елементами ринкової стратегії є вибір пріоритетних цільових груп, формування базової стратегії розвитку продукту, визначення характеру конкурентної поведінки та позиціонування системи в сприйнятті цільових користувачів. Відповідні рішення узагальнюються в таблицях 6.8–6.11.

Таблиця 6.8 — Вибір цільових груп потенційних споживачів

Опис профілю цільової групи	Готовність споживачів сприйняти продукт	Орієнтовний рівень попиту	Інтенсивність конкуренції	Простота входу у сегмент
Малі фермерські та сімейні господарства	Середня: потреба висока, але цифрова готовність нерівномірна	Стабільний попит у сезон, чутливість до ціни	Помірна: конкуренція з мобільними застосунками	Порівняно проста за умови простого інтерфейсу та доступної ціни
Середні агропідприємства	Вища: наявні базові цифрові інструменти	Зростаючий попит на систематизовану діагностику	Помірна–висока	Потребує демонстрацій, інтеграцій та сервісної підтримки
Великі агрохолдинги	Висока, але з високими вимогами до інтеграції та масштабованості	Потенційно значний при доведеній ефективності	Висока: конкуренція з комплексними agtech-рішеннями	Складніший вхід, доцільні пілотні проєкти та індивідуальні переговори

Опис профілю цільової групи	Готовність споживачів сприйняти продукт	Орієнтовний рівень попиту	Інтенсивність конкуренції	Простота входу у сегмент
Агроконсалтингові компанії, дистриб'ютори ЗЗР	Висока: зацікавленість у сервісній диференціації	Попит залежить від структури клієнтської бази	Помірна	Вхід через партнерства, спільні пілоти, white-label моделі
Освітні та наукові установи	Досить висока для навчальних і дослідницьких цілей	Обмежений, але стабільний	Низька	Порівняно проста інтеграція, акцент на аналітичних можливостях

На основі такого структурування доцільно виділити пріоритетні сегменти для початкового етапу: малі та середні господарства, а також сервісні організації, що працюють з великою кількістю клієнтів. Це дозволяє поєднати швидші цикли впровадження з можливістю отримання репрезентативного набору діагностичних кейсів для подальшого вдосконалення бази знань.

З урахуванням характеристик ринку та обраних цільових груп формулюється базова стратегія розвитку стартап-проєкту (таблиця 6.9).

Таблиця 6.9 — Визначення базової стратегії розвитку стартап-проєкту

Обрана альтернатива розвитку	Стратегія охоплення ринку	Ключові конкурентні позиції	Базова стратегія
Етап запуску MVP для малих і середніх господарств	Селективне охоплення окремих регіонів та партнерських господарств	Пояснюванісь діагностики, локалізація, можливість локального розгортання	Вихід на ринок через демонстраційні проєкти, збір кейсів, уточнення бази знань
Розширення через партнерства з агроконсалтинговими компаніями	Охоплення клієнтських баз партнерів	Інтеграція в сервісні пакети, «цифровий асистент» для консультантів	Спільні програми впровадження, white-label рішення для B2B-клієнтів
Масштабування для середніх і великих підприємств як хмарного сервісу	Сегментований підхід за категоріями клієнтів	Масштабованість, підтримка хмарних сервісів AWS, сховище кейсів для аналітики	Перехід до SaaS-моделі з розширеними функціями аналітики та управління доступом

Важливою складовою ринкової стратегії є визначення характеру конкурентної поведінки. На ринку вже присутні рішення, які виконують автоматизований аналіз зображень, однак рівень локалізації, прозорість алгоритмів і можливості роботи в гібридному режимі залишають простір для диференціації. Узагальнення підходу до конкурентної поведінки наведено в таблиці 6.10.

Таблиця 6.10 — Визначення стратегії конкурентної поведінки

Ознака	Характеристика для стартап-проєкту
Статус «першопрохідця» на цільовому ринку	Не є абсолютним «першопрохідцем» глобально, але може виступати новатором у локальному сегменті зі специфікою українського ринку
Орієнтація на нових чи існуючих споживачів	Орієнтація на частково «нових» споживачів для цифрових продуктів (малі господарства) та на «існуючих» для agtech-сегмента (середні/великі підприємства)
Ступінь копіювання характеристик конкурентів	Вибіркове: запозичуються базові очікувані функції (діагностика за фото), але робиться акцент на додаткових можливостях (пояснення, кейси, гібридний режим)
Обрана стратегія конкурентної поведінки	Диференціація: акцент на поєднанні діагностики з веденням «цифрового архіву» випадків і можливістю локальної роботи

Завершальним елементом ринкової стратегії є позиціонування продукту в сприйнятті цільової аудиторії. Воно має відображати як ключові очікування користувачів, так і ті аспекти, які відрізняють продукт від існуючих аналогів. Узагальнення підходу до позиціонування наведено в таблиці 6.11.

Центральна ідея позиціонування полягає у зміщенні акценту з простої автоматизації розпізнавання на забезпечення обґрунтованості та прозорості агрономічного рішення. На відміну від масових мобільних застосунків, які часто функціонують як «чорна скринька» та орієнтовані на миттєвий результат без пояснень, розроблювана система представляється як професійний інструмент «другої думки», що посилює, а не замінює експертизу агронома.

Таблиця 6.11 — Стратегія позиціонування продукту на ринку

Вимоги цільової аудиторії	Базова стратегія розвитку	Ключові конкурентні переваги	Цільові асоціації
Простота використання, надійність результатів, зрозумілі рекомендації	Поступове розгортання від MVP до масштабованого сервісу	Пояснюваність діагностики, підтримка української мови, робота в локальному та хмарному режимах	«Надійний», «зрозумілий», «польовий асистент»
Можливість інтеграції в існуючі процеси та системи	Розвиток у напрямі сервісу для організаційних клієнтів	Структуроване сховище кейсів, готовність до інтеграцій, гнучкі сценарії розгортання	«Професійний», «інтегрований», «аналітичний»
Прийнятна вартість володіння та прозора модель використання	Диференціація тарифів за функціональністю та режимами	Можливість базового використання без дорогих хмарних компонентів, поетапне підключення додаткових сервісів	«Доступний», «масштабований за потребою», «керований витратами»

Таким чином, ринкова стратегія продукту ґрунтується на поєднанні селективного виходу на пріоритетні сегменти, диференціації через пояснюваність

та гібридну архітектуру, а також чіткому позиціонуванні як інструмента практичної підтримки прийняття рішень у польових умовах.

6.5 Розроблення маркетингової програми стартап-проєкту

Маркетингова програма стартап-проєкту, що базується на системі діагностики хвороб сільськогосподарських культур, формалізує комплекс заходів у вимірах продукту, ціни, збуту та комунікацій. На відміну від попередніх підрозділів, де розглядалися ринкові можливості та стратегічні орієнтири, у даному підрозділі акцент зроблено на практичних аспектах виходу продукту на ринок та підтримки його життєвого циклу.

Продуктовий компонент маркетинг-міксу ґрунтується на пропозиції веборієнтованого сервісу діагностики з можливістю розгортання у локальному та хмарному середовищах. У початковій конфігурації передбачається вебінтерфейс для формування запиту на діагностику, перегляду результатів та роботи з «папками діагностичних кейсів». Надалі як потенційний напрям розвитку розглядається створення мобільного клієнта, що використовуватиме той самий серверний функціонал. Модель ліцензування може включати підписку для окремих господарств, оплати за обсяг використання (наприклад, кількість діагностичних кейсів з повним хмарним аналізом) та спеціальні умови для партнерських організацій (у тому числі формати white-label для агроконсалтингових компаній і дистриб’юторів засобів захисту рослин).

Для цільових споживачів основними перевагами продукту є можливість оперативної діагностики уражень, зниження залежності від наявності експерта «тут і зараз», накопичення історичних даних про випадки, а також підвищення якості прийняття рішень щодо захисту культур. Для агрокомпаній додаткову цінність створює структуроване сховище кейсів, на основі якого можуть формуватися аналітичні звіти.

Цінова політика має бути адаптована до різних сегментів ринку. Для малих господарств доцільним є простий і прозорий тариф із невисоким порогом входу,

тоді як для середніх і великих підприємств більш релевантними є гнучкі пакетні пропозиції або корпоративні ліцензії, що враховують масштаби використання. Межі цін визначаються співвідношенням із наявними аналогами та альтернативами (традиційний консалтинг, виїзд агронома), а також платоспроможністю відповідних сегментів.

Система збуту передбачає поєднання прямих цифрових каналів (онлайн-реєстрація та підключення сервісу), продажів через партнерів (агродистриб'ютори, консалтингові компанії), а також присутність на галузевих заходах (виставки, демонстраційні поля, навчальні заходи). Комунікаційна політика орієнтується на демонстрацію практичної корисності продукту, прозорості алгоритмів та пояснюваності рекомендацій, що є ключовими факторами довіри фермерів та агрономів. Основні елементи маркетингової програми узагальнено у таблицях 6.12–6.16.

Таблиця 6.12 — Визначення ключових переваг концепції потенційного продукту

Потреба	Вигода, яку забезпечує продукт	Ключові переваги перед конкурентами
Оперативна діагностика хвороб у польових умовах	Швидке отримання попереднього діагностичного висновку та плану дій	Поєднання аналізу тексту й зображень, можливість роботи в локальному та хмарному режимах
Зменшення залежності від доступності агронома в полі	Можливість отримати структуровану рекомендацію без очного виїзду фахівця	Пояснювані результати, що можуть бути верифіковані експертом у відкладеному режимі

Потреба	Вигода, яку забезпечує продукт	Ключові переваги перед конкурентами
Накопичення історії випадків для подальшого аналізу	Формування «папок діагностичних кейсів» для аудиту, навчання та аналітики	Структуроване сховище кейсів, орієнтоване на багаторазове використання
Зменшення втрат урожаю через несвоєчасну або некоректну діагностику	Підвищення якості та швидкості прийняття рішень щодо заходів захисту	Інтеграція агрономічного контексту (культура, фаза розвитку, типові патерни) у процес діагностики
Потреба сервісних компаній у диференціації своїх послуг	Можливість пропонувати клієнтам додатковий цифровий сервіс підтримки рішень	Підтримка партнерських моделей, включно з брендуванням і гнучкими схемами ліцензування

Таблиця 6.13 — Опис трьох рівнів моделі продукту

Рівень	Сутність та складові
Товар за задумом	Цифровий інструмент підтримки прийняття рішень у діагностиці хвороб культур, який дозволяє на основі фото та опису симптомів отримати пояснюваний перелік гіпотез і базовий план дій.
Товар у реальному виконанні	Веборієнтований сервіс із користувацьким інтерфейсом для формування запиту, серверною частиною з діагностичним конвеєром, базою знань і сховищем кейсів; підтримка локального й хмарного розгортання, інтеграція із зовнішніми сервісами комп’ютерного зору та мовних моделей у розширеному режимі.

Рівень	Сутність та складові
Товар із підкріпленням	Навчальні матеріали й інструкції, технічна підтримка, можливість оновлення бази знань, консультаційний супровід у межах партнерських програм, перспективний мобільний доступ та аналітичні звіти на основі накопичених кейсів.

Таблиця 6.14 — Визначення меж встановлення ціни

Рівень цін на аналоги	Рівень цін на замінники	Рівень доходів цільових клієнтів	Межі ціни продукту
Міжнародні мобільні застосунки з діагностики за фото	Традиційні консультаційні послуги агрономів, виїзди на поля	Малі фермери з обмеженим бюджетом на цифрові сервіси	Ціна має бути співставною або нижчою за вартість базових аналогів, орієнтована на доступну підписку
Спеціалізовані хмарні рішення для агросектору	Внутрішні ІТ-системи агропідприємств та консалтинг «під ключ»	Середні й великі господарства з більшими, але регламентованими бюджетами	Ціновий рівень може бути вищим, але виправданим завдяки масштабованості
Нішеві сервіси моніторингу посівів	Власні розробки або адаптації відкритих інструментів	Сервісні організації, що працюють із широкою клієнтською базою	Межі ціни визначаються співвідношенням «вартість володіння — цінність для клієнтів сервісної організації»

Таблиця 6.15 — Формування системи збуту

Специфіка закупівельної поведінки клієнтів	Функції постачальника	Глибина каналу збуту	Оптимальна система збуту
Малі фермери приймають рішення швидко, орієнтуючись на практичну користь	Надання простого механізму підключення, демо-доступу, базової підтримки	Короткий канал	Прямі онлайн-продажі через вебсайт/портал із можливістю самостійної реєстрації
Середні господарства орієнтуються на досвід упровадження	Проведення демонстрацій, надання тестових доступів, допомога в інтеграції	Середній канал	Збут через партнерів (агроконсалтинг, дистриб'ютори ЗЗР) з підтримкою спільних проєктів
Великі компанії приймають рішення колегіально	Підготовка пілотних проєктів, адаптація конфігурацій, узгодження умов ліцензування	Поглиблений канал з індивідуальними переговорними стадіями	Індивідуальні контракти, інтеграційні проєкти, можлива кастомізація рішень
Сервісні компанії інтегрують продукт у власні пакети послуг	Забезпечення API-доступу, брендування, консультаційної підтримки	Канал через B2B-партнерів	Моделі white-label, включення системи до складу сервісних пакетів партнерів

Таблиця 6.16 — Концепція маркетингових комунікацій

Специфіка поведінки клієнтів	Канали комунікацій	Позиціонування	Завдання рекламного повідомлення
Малі фермери орієнтуються на практичний результат та рекомендації колег	Галузеві виставки, демонстраційні поля, онлайн-спільноти фермерів, таргетовані кампанії	«Простий і надійний помічник у полі»	Показати, що продукт дає швидку й зрозумілу відповідь, яку можна використати у реальних польових умовах
Середні та великі господарства очікують структурованих даних і звітності	Професійні конференції, галузеві медіа, вебінари, прямі презентації для менеджменту	«Інструмент для системного управління ризиками хвороб»	Підкреслити можливість накопичення кейсів, аналізу, інтеграції в існуючі управлінські процеси
Сервісні організації шукають засоби диференціації своїх послуг	Персоналізовані презентації, партнерські програми, професійні мережі консультантів	«Цифровий сервіс, що підсилює експертні консультації»	Пояснити, як продукт допомагає підвищити цінність консалтингових послуг і вибудувати довгострокові відносини з клієнтами

Узагальнюючи, маркетингова програма стартап-проєкту формує цілісний підхід до виведення на ринок цифрового продукту, що ґрунтується на поєднанні пояснюваної діагностики, гнучких моделей використання та партнерських каналів

збуту. Це створює передумови для поступового нарощування клієнтської бази й переходу від прототипу до стабільного комерційного сервісу.

Висновки до розділу 6

У шостому розділі розроблено концепцію стартап-проєкту на базі створеної системи, ключова цінність якої полягає у поєднанні мультимодальної діагностики (текст + фото), пояснюваності рішень та веденні структурованих «папок кейсів» для аудиту. Аналіз ринкового середовища виявив конкурентні переваги гібридної архітектури в умовах цифровізації агросектору, хоча й підкреслив ризики, пов’язані з конкуренцією та залежністю від якості вхідних даних. Пріоритетними групами споживачів визначено малі та середні господарства й агроконсалтингові компанії, для яких запропоновано стратегію поетапного виходу на ринок — від пілотних впроваджень MVP до масштабування через партнерські канали.

Розроблений маркетинговий комплекс передбачає гнучку цінову політику та просування продукту з акцентом на його практичну надійність і здатність до інтеграції у виробничі процеси. Підтверджено наявність об’єктивних передумов для комерціалізації прототипу, що дозволяє розглядати його як основу для повноцінного бізнесу. Успішна реалізація стартапу залежатиме від подальшої конкретизації фінансової моделі, розширення бази знань та проведення польових випробувань, що логічно поєднує технічні досягнення роботи з бізнес-потребами галузі.

ВИСНОВКИ

У магістерській дисертації вирішено актуальну науково-прикладну задачу підвищення ефективності діагностики хвороб сільськогосподарських культур шляхом створення інтелектуальної інформаційної системи підтримки прийняття рішень. Результати роботи базуються на комплексному підході, що поєднує сучасні методи комп'ютерного зору, пошук у базі знань та можливості генеративного штучного інтелекту.

На початковому етапі дослідження було проведено глибокий аналіз предметної області та існуючих технологічних рішень для агросектору. Встановлено, що наявні на ринку програмні продукти часто функціонують за принципом «чорної скриньки», надаючи користувачу лише кінцевий результат без пояснення логіки діагностики, або ж ігнорують важливі контекстні дані, такі як текстовий опис симптомів чи фаза розвитку рослини. Це обумовило необхідність розробки нової системи, яка б забезпечувала прозорість процесу прийняття рішень та мультимодальність вхідних даних.

Для реалізації поставленої мети було сформовано перелік функціональних та нефункціональних вимог, ключовими з яких стали висока точність розпізнавання, швидкодія та здатність системи адаптуватися до умов роботи з нестабільним інтернет-з'єднанням. На основі цих вимог спроєктовано мікросервісну архітектуру системи, яка передбачає чіткий поділ на клієнтську частину, реалізовану за допомогою бібліотеки Streamlit, та серверну частину на базі фреймворку FastAPI. Обґрунтовано вибір технологічного стека, що включає мову програмування Python, реляційну базу даних PostgreSQL для зберігання структурованих діагностичних кейсів та хмарні сервіси Amazon Web Services для забезпечення масштабованості.

Центральним науково-технічним результатом роботи стала розробка та програмна реалізація діагностичного конвеєра. Цей механізм послідовно поєднує валідацію вхідних даних, візуальний аналіз зображень за допомогою сервісу AWS Rekognition та локальних алгоритмів, пошук релевантної інформації в базі знань за методом RAG та застосування агрономічних правил. Такий підхід дозволив

мінімізувати ймовірність помилок, властивих окремим методам, та забезпечити формування рекомендацій.

Практичну цінність роботи підтверджено створенням діючого прототипу системи, який підтримує діагностику 50 видів хвороб для 10 ключових сільськогосподарських культур. Проведене тестування на 3 основних культурах і 7 відповідних хворобах засвідчило, що система забезпечує високу релевантність результатів завдяки інтеграції великих мовних моделей AWS Bedrock, які генерують зрозумілі для користувача пояснення та плани дій.

Важливою складовою дисертації стала розробка стартап-проєкту, в рамках якого проаналізовано ринковий потенціал системи. Визначено цільову аудиторію продукту, до якої увійшли фермерські господарства та агрономи-консультанти, а також розроблено стратегію комерціалізації. Доведено, що впровадження розробленої системи є економічно доцільним, оскільки вона дозволяє суттєво знизити ризики втрати врожаю та оптимізувати витрати на засоби захисту рослин.

Таким чином, у роботі досягнуто поставленої мети: створено та всебічно досліджено інформаційну систему, яка є завершеним програмним продуктом, готовим до пілотного впровадження, та має значний потенціал для подальшого масштабування й функціонального розширення.

ПЕРЕЛІК ДЖЕРЕЛ ТА ПОСИЛАНЬ

1. M.Bogonos, A.Chmil, R.Nazarkina. Agricultural Outlook Ukraine 2024-2033 Report-summary. Kyiv School of Economics. URL: https://kse.ua/wp-content/uploads/2024/04/UA-Outlook-2024-2033-Report-1.pdf.
2. Janna Beckerman, Tom Creswell. Symptoms and Signs for Plant Problem Diagnosis — An Illustrated Glossary / Department of Botany and Plant Pathology, Purdue University. URL: https://www.extension.purdue.edu/extmedia/BP/BP-164-W.pdf.
3. Nigel Russell Curry, James Kirwan. The Role of Tacit Knowledge in Developing Networks for Sustainable Agriculture. Sociologia Ruralis. URL: https://doi.org/10.1111/soru.12048.
4. Gabriel Laliberte, Ben Goldberg. AI, data and innovation in agriculture: the future of innovative farming. Valtech. URL: https://www.valtech.com/blog/ai-data-and-the-future-of-farming/.
5. S. Hernández, Juan L. López B. Uncertainty quantification for plant disease detection using Bayesian deep learning. Sciencedirect. URL: https://doi.org/10.1016/j.asoc.2020.106597.
6. Jayme Garcia Arnal Barbedo. A review on the main challenges in automatic plant disease identification based on visible range images. Biosystems Engineering. URL: https://doi.org/10.1016/j.biosystemseng.2016.01.017.
7. Plantix. URL: https://plantix.net/en/.
8. Jingyi Yan, Huarui Wu, Zhihua Diao. Recent Developments and Applications of Crop Disease Detection, Prediction, and Early Warning: A review. *Engineering*. URL: https://doi.org/10.1016/j.eng.2025.10.032.
9. HELM takes majority of shares in agritech startup Plantix. *Helmag.com*. URL: https://www.helmag.com/news-media/news/detail/helm-takes-majority-of-shares-in-agritech-startup-plantix.
10. Mrisho L., Mbilinyi N, Ndalahwa M. Accuracy of a smartphone-based object detection model, PlantVillage Nuru, in identifying the foliar symptoms of the viral

diseases of cassava-CMD and CBSD. *Frontiers in Plant Science*. URL: https://doi.org/10.3389/fpls.2020.590889.

11. Crowd2map Tanzania. Introduction to PlantVillage Nuru. Crowd2Map Tanzania. URL: https://crowd2map.org/wp-content/uploads/2021/02/INTRODUCTION-TO-PLANTVILLAGE-NURU-POWER-POINT.-2.pdf.
12. Wäldchen J., Mäder P. Plant Species Identification Using Computer Vision Techniques: A Systematic Literature Review. Arch Computat Methods Eng. URL: https://doi.org/10.1007/s11831-016-9206-z.
13. Plant Id Official. URL: https://plant.id/.
14. Plant Health Official. URL: http://kindwise.com/plant-health.
15. Yosuke Toda, Fumio Okura. How Convolutional Neural Networks Diagnose Plant Disease. *Plant Phenomics*. URL: https://doi.org/10.1155/2019/9237136.
16. Samek, Wojciech & Montavon, Grégoire & Vedaldi, Andrea & Hansen, Lars & Müller, Klaus-Robert. Explainable AI: Interpreting, Explaining and Visualizing Deep Learning. 10.1007/978-3-030-28954-6
17. Patrick Lewis, Ethan Perez, Aleksandra Piktus. Retrieval-augmented generation for knowledge-intensive NLP tasks. NIPS'20: Proceedings of the 34th International Conference on Neural Information Processing Systems. URL: https://proceedings.neurips.cc/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf.
18. Martin R. C., Clean Architecture: A Craftsman's Guide to Software Structure and Design. Prentice Hall.
19. FastAPI Documentation. URL: https://fastapi.tiangolo.com/
20. PostgreSQL Global Development Group. PostgreSQL 16.1 Documentation. URL: https://www.postgresql.org/docs/
21. Amazon Web Services. AWS Well-Architected Framework. URL: https://aws.amazon.com/architecture/well-architected/.