Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

РОЗРОБЛЕННЯ ВЕБЗАСТОСУНКУ ВИЗНАЧЕННЯ СТУПЕНЯ СТИГЛОСТІ ФРУКТІВ ЗА ДОПОМОГОЮ ЗАСОБІВ КОМП'ЮТЕРНОГО ЗОРУ
(тема)

Виконав:
здобувач 4 року навчання,
групи ІТІНФ-22-3
Петрова М. Ф.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Творошенко І. С.
(посада, прізвище, ініціали)

Допускається до захисту

Завідувач кафедри інформатики _____ Кобилін О. А.
(підпис) (прізвище, ініціали)

2026 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту

Кафедра Інформатики

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Петровій Марії Федірівні
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення вебзастосунку визначення ступеня стиглості фруктів за допомогою засобів комп'ютерного зору

затверджена наказом університету від 26 травня 2026 року № 435Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 22 червня 2026 р.
3. Вихідні дані до роботи науково-методична та науково-технічна література з комп'ютерного зору та машинного навчання, відкриті набори цифрових зображень фруктів, матеріали інтернет-мережі, документація TensorFlow для побудови моделі класифікації зображень.

4. Перелік питань, що потрібно опрацювати в роботі
1. Аналіз сучасних вебзастосунків і наукових джерел щодо визначення ступеня стиглості фруктів засобами комп'ютерного зору.
2. Проєктування структури вебзастосунку, бази даних і алгоритму оброблення зображень.
3. Реалізація вебзастосунку для класифікації зображень фруктів і збереження результатів аналізу.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність проблеми, постановка задачі, схеми бази даних і структури вебзастосунку, блок-схеми алгоритмів, UML-діаграми, архітектура моделі, графіки навчання, матриці помилок.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	13.04.2026	
2	Аналіз завдання, підбір літератури	14.04.26-16.04.26	
3	Аналіз літератури з проблеми, що вирішується	17.04.26-22.04.26	
4	Аналіз існуючих методів розпізнавання	23.04.26-27.04.26	
5	Формування кроків вирішення проблеми	28.04.26-02.05.26	
6	Програмна реалізація	03.05.26-23.05.26	
7	Аналіз отриманих результатів	24.05.26-31.05.26	
8	Оформлення пояснювальної записки	01.06.26-09.06.26	
9	Перевірка на нормоконтроль	10.06.26-19.06.26	
10	Перевірка на плагіат	11.06.26-30.06.26	
11	Рецензування	20.06.26-30.06.26	
12	Підготовка презентації та доповіді	20.06.26-30.06.26	
13	Занесення роботи в електронний архів	30.06.26-09.07.26	
14	Попередній захист кваліфікаційної роботи	30.06.26-09.07.26	

Дата видачі завдання 13 квітня 2026 р.

Здобувач _____
(підпис)

Керівник роботи _____ _____доц. Творошенко І. С.___
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 85 с., 8 табл., 21 рис., 1 дод., 62 джерела.

БАЗА ДАНИХ, ВЕБЗАСТОСУНОК, КОМП'ЮТЕРНИЙ ЗІР, СТУПІНЬ СТИГЛОСТІ, ФРУКТ, ЦИФРОВЕ ЗОБРАЖЕННЯ.

Об'єктом роботи є процес визначення ступеня стиглості фруктів за цифровими зображеннями засобами комп'ютерного зору.

Метою роботи є розроблення вебзастосунку для визначення ступеня стиглості фруктів за цифровими зображеннями засобами комп'ютерного зору.

Під час роботи використано: методи комп'ютерного зору, класифікації цифрових зображень, клієнт-серверну архітектуру, засоби React, FastAPI, SQLite та UML-моделювання.

Практична цінність вебзастосунку полягає в автоматизації визначення ступеня стиглості фруктів за цифровими зображеннями.

Розроблено вебзастосунок, що забезпечує завантаження зображення фрукта, визначення ступеня його стиглості, збереження результатів і перегляд історії перевірок.

Область застосування: побутове використання, навчальні цілі, контроль якості плодово-овочевої продукції.

COMPUTER VISION, DATABASE, DEGREE OF RIPENESS, DIGITAL IMAGE, FRUIT, WEB APPLICATION.

The object of the work is the process of determining the degree of fruit ripeness from digital images using computer vision tools.

The goal of the work is to develop a web application for determining the degree of fruit ripeness from digital images using computer vision tools.

The following were used during the work: computer vision methods, digital image classification, client-server architecture, React, FastAPI, SQLite, and UML modeling.

The practical value lies in the automation of determining the degree of fruit ripeness from digital images.

A web application has been developed for uploading a fruit image, determining its degree of ripeness, saving results, and viewing the history of checks.

Applications: household use, educational purposes, quality control of fruit and vegetable products.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 7
Вступ 8
1 Аналіз існуючих вебзастосунків визначення певного ступеня конкретного фактору на об’єктах за допомогою засобів комп’ютерного зору 9
1.1 Аналіз сучасних вебзастосунків визначення певного ступеня конкретного фактору на об’єктах за допомогою засобів комп’ютерного зору 9
1.2 Аналіз літературних джерел щодо апробації результатів розроблення вебзастосунків визначення певного ступеня конкретного фактору на об’єктах за допомогою засобів комп’ютерного зору, зокрема ступеня стиглості фруктів 15
1.3 Постановка задачі 21
2 Розроблення структури вебзастосунку визначення ступеня стиглості фруктів за допомогою засобів комп’ютерного зору 23
2.1 Особливості структури бази даних вебзастосунку визначення ступеня стиглості фруктів за допомогою засобів комп’ютерного зору 23
2.2 Моделювання структури вебзастосунку визначення ступеня стиглості фруктів 30
3 Розроблення вебзастосунку визначення ступеня стиглості фруктів за допомогою засобів комп’ютерного зору 41
3.1 Вибір інструментальних засобів для реалізації поставленої задачі 41
3.2 Етапи розроблення вебзастосунку визначення ступеня стиглості фруктів за допомогою засобів комп’ютерного зору 47
3.3 Тестування розробленого вебзастосунку визначення ступеня стиглості фруктів та аналіз результатів 59

3.4 Перспективи подальшої роботи 70
Висновки 74
Перелік джерел посилання 77
Додаток А Порівняльна характеристика сучасних вебзастосунків для аналізу плодово-овочевої продукції 84

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

% – відсоток

ШІ – штучний інтелект

API – Application Programming Interface (інтерфейс програмного застосунку, що використовується для взаємодії між програмними компонентами)

Backend – серверна частина вебзастосунку, що відповідає за оброблення запитів, роботу з моделлю комп’ютерного зору, базою даних і файловою системою

CNN – Convolutional Neural Network (згорткова нейронна мережа)

Frontend – клієнтська частина вебзастосунку, з якою безпосередньо взаємодіє користувач

SQLite – легка реляційна система керування базами даних, що зберігає базу у вигляді окремого файлу

UML – Unified Modeling Language (уніфікована мова моделювання, що використовується для опису структури та поведінки програмних систем)

YOLO – You Only Look Once (модель комп’ютерного зору для виявлення та локалізації об’єктів на зображеннях)

ВСТУП

Комп’ютерний зір є одним із напрямів сучасних інформаційних технологій, що використовується для автоматизованого аналізу цифрових зображень. Його застосування дає змогу розпізнавати об’єкти, визначати їхні візуальні ознаки та формувати результат на основі оброблення зображення. Такі підходи активно використовуються у задачах класифікації, контролю якості та аналізу стану об’єктів.

Визначення ступеня стиглості фруктів є важливим завданням, оскільки від цього залежить якість продукту, можливість його споживання, зберігання або подальшого використання. Традиційне оцінювання за зовнішнім виглядом часто є суб’єктивним і може залежати від умов освітлення, якості зображення та досвіду людини. Наявні програмні рішення переважно орієнтовані на загальне розпізнавання фруктів і овочів або визначення їхнього стану за спрощеними категоріями, тому задача визначення саме ступеня стиглості фруктів потребує окремого розгляду.

Тема роботи пов’язана з розробленням вебзастосунку для визначення ступеня стиглості фруктів засобами комп’ютерного зору. Основними проєктними рішеннями є використання клієнт-серверної архітектури, моделі комп’ютерного зору, бази даних для збереження результатів аналізу та вебінтерфейсу для взаємодії з користувачем.

Актуальність роботи полягає в необхідності автоматизації процесу визначення ступеня стиглості фруктів за цифровими зображеннями. Розроблення такого вебзастосунку дає змогу зробити оцінювання стиглості більш зручним для користувача, забезпечити збереження результатів аналізу та створити основу для подальшого використання системи в побутових, навчальних і прикладних задачах контролю якості плодово-овочевої продукції.

1 АНАЛІЗ ІСНУЮЧИХ ВЕБЗАСТОСУНКІВ ВИЗНАЧЕННЯ ПЕВНОГО СТУПЕНЯ КОНКРЕТНОГО ФАКТОРУ НА ОБ'ЄКТАХ ЗА ДОПОМОГОЮ ЗАСОБІВ КОМП'ЮТЕРНОГО ЗОРУ

1.1 Аналіз сучасних вебзастосунків визначення певного ступеня конкретного фактору на об'єктах за допомогою засобів комп'ютерного зору

Сучасні засоби комп'ютерного зору дедалі активніше застосовуються у вебзастосунках для аналізу зображень харчових об'єктів. Найпоширенішими напрямами є класифікація фруктів і овочів, оцінювання їхньої якості, визначення свіжості або псування, а також виявлення окремих ознак стиглості. Вебформат подібних рішень є особливо зручним, оскільки дає змогу працювати із завантаженими зображеннями без встановлення спеціалізованого локального програмного забезпечення. Оскільки вузькоспеціалізованих відкритих вебзастосунків саме для визначення ступеня стиглості фруктів небагато, доцільно розглянути також суміжні рішення для оцінювання стану фруктів і овочів, що функціонально близькі до тематики кваліфікаційної роботи.

До аналізу в межах цього підрозділу включено не лише завершені вебзастосунки, а й демонстраційні вебсервіси, дослідницькі прототипи та проєкти з відкритим доступом до опису функціональних можливостей. Такий підхід є доцільним, оскільки у сфері комп'ютерного зору значна частина прикладних рішень реалізується саме у вигляді експериментальних або демонстраційних вебінтерфейсів. Їх аналіз дає змогу оцінити наявні технічні підходи, типи вхідних даних, формат подання результатів і рівень придатності таких рішень до практичного використання.

Одним із прикладів є вебзастосунок Banana Ripeness Detection [1] (рис. 1.1). У відкритому описі проєкту зазначено, що він призначений для прогнозування стадії стиглості банана, класифікує банани на три категорії та має вебінтерфейс для тестування зображень у реальному часі.

Рисунок 1.1 – Інтерфейс вебзастосунку Banana Ripeness Detection

Також автори [1] вказують, що модель може бути використана і для прогнозування стану інших фруктів. Перевагою цього рішення є вузька орієнтація саме на задачу оцінювання стиглості, що безпосередньо відповідає тематиці роботи. Разом із тим у відкритому описі не наведено кількісних показників точності класифікації, тому це рішення складно об’єктивно порівнювати з іншими аналогами за метриками якості розпізнавання. Крім того, не розкрито, наскільки модель є стійкою до зміни умов освітлення, фону та якості вхідних зображень, що важливо для реального вебвикористання.

Ще одним прикладом є вебзастосунок sBuah [2], який класифікує зображення фруктів за двома станами: свіжий або зіпсований (рис. 1.2).

У репозиторії вказано, що за допомогою використання алгоритму машинного навчання модель демонструє точність понад 97%, а сам застосунок було розгорнуто як вебрішення на основі Flask. Такий підхід є практично цінним, оскільки дає змогу автоматизувати первинне оцінювання якості фруктів за зображенням. Його сильною стороною є наявність відкрито поданої кількісної оцінки якості розпізнавання. Водночас недоліком є те, що застосунок не визначає кілька послідовних стадій стиглості, а розв’язує більш спрощену задачу бінарної класифікації стану об’єкта.

Рисунок 1.2 – Результат аналізу стиглості та дефектів у реальному часі

Також у відкритому описі не уточнено, наскільки модель зберігає точність за використання зображень зі складним тлом або в неконтрольованих умовах знімання.

Серед подібних рішень можна виокремити Fruit Quality Classification – вебзастосунок ідентифікації фруктів [3] і класифікації їх за якістю (рис. 1.3).

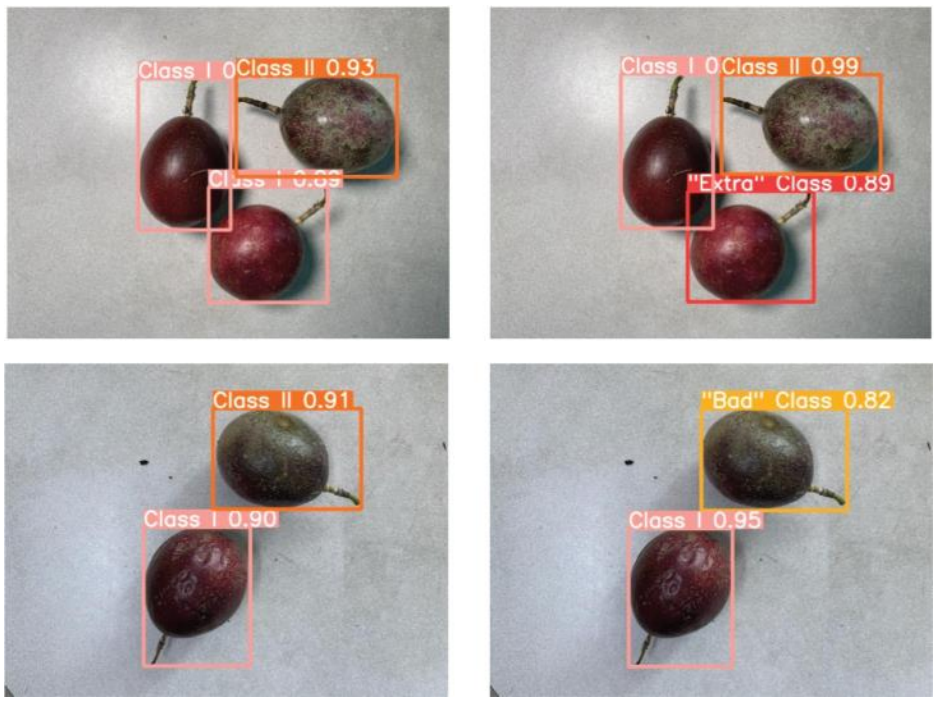

Рисунок 1.3 – Результат оброблення даних нейромережею з класифікації фруктів за класами якості

У відкритому описі зазначено, що для проєкту використано набір даних із понад 29000 зображень 14 різних фруктів і овочів, класифікованих як якісні або зіпсовані. Для навчання моделі було застосовано ResNet50 із SGD-оптимізатором, наведено точність 91,3% на тренувальному наборі та 90,99% на тестовому наборі. Також автори [3] вказують, що застосунок розгорнуто у середовищі Streamlit. Сильною стороною цього рішення є поєднання багатокласового розпізнавання об’єкта з оцінюванням його стану. Проте, як і в попередньому випадку, основний акцент зроблено на виявленні зіпсованих об’єктів, а не на детальному поділі фруктів за кількома рівнями стиглості.

Окремої уваги заслуговує інтерактивний вебзастосунок для класифікації фруктів і овочів за зображеннями Fruit & Vegetable Classifier App (рис. 1.4) [4].

Рисунок 1.4 – Результат роботи Fruit & Vegetable Classifier App

У його описі зазначено, що застосунок побудовано на Streamlit, використовує модель VGG-16 із застосуванням перенесеного навчання та дає змогу завантажувати зображення або тестувати зразки з подальшою класифікацією на 36 окремих категорій.

Крім цього, користувачеві може бути показано прогноз із рівнем упевненості моделі. У характеристиках продуктивності вказано точність на навчальній вибірці близько 97% та точність на валідаційній вибірці близько 92%. Перевагою такого рішення є широкий спектр підтримуваних класів, а також наявність зручного вебінтерфейсу. Однак воно орієнтоване насамперед на загальну класифікацію плодово-овочевих об’єктів, а не на визначення ступеня стиглості конкретного фрукта. Крім того, з відкритого опису не випливає, наскільки модель адаптована до роботи з реальними користувацькими зображеннями, що можуть суттєво відрізнятися від навчальної вибірки.

До суміжних рішень також належить Vegetable Classification & Detection, який позиціонується як вебінструмент для класифікації та детекції овочів (рис. 1.5) [5].

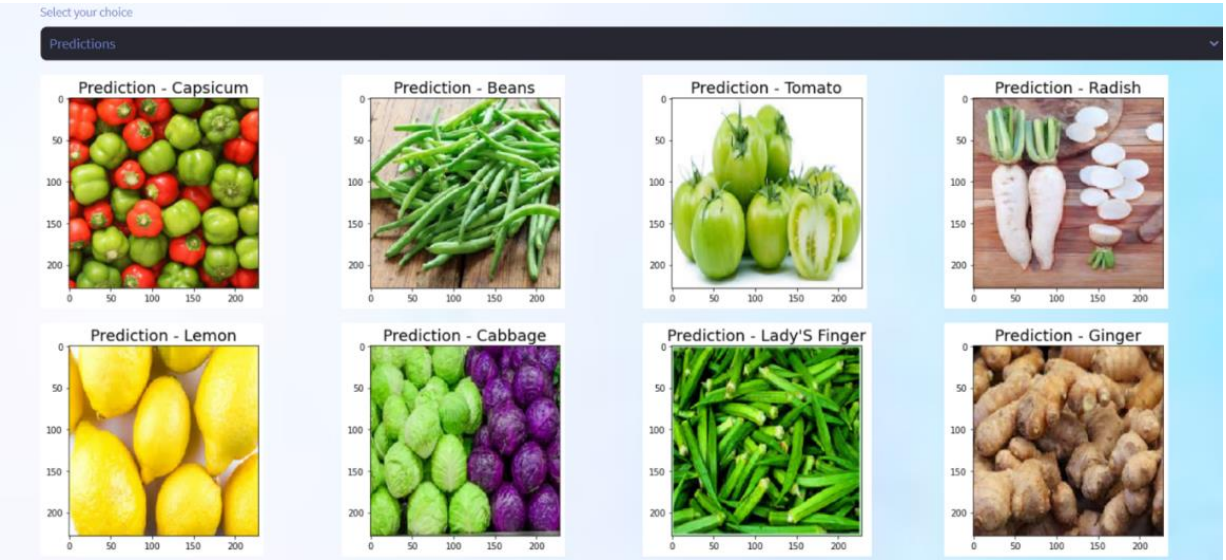

Рисунок 1.5 – Інтерфейс вебінструменту Vegetable Classification & Detection

У відкритому описі проєкту зазначено, що він побудований із використанням Streamlit, TensorFlow та OpenCV, а для оброблення зображень і відеопотоків застосовуються моделі CNN і YOLO. Автори вказують, що рішення здатне класифікувати й детектувати овочі на зображеннях і у відеопотоці в реальному часі, тобто поєднує дві важливі задачі комп’ютерного зору: класифікацію та локалізацію об’єктів.

Перевагою цього застосунку є використання сучасного інструментарію та орієнтація на роботу не лише зі статичними зображеннями, а й з відеопотоком. Стосовно тематики даної роботи його слід розглядати як суміжний приклад, оскільки він не орієнтований на фрукти та не виконує визначення рівня стиглості.

Порівняння розглянутих вебзастосунків дає змогу виявити кілька характерних тенденцій. Насамперед слід зазначити, що найбільш близькими до тематики даної кваліфікаційної роботи є рішення, спрямовані на аналіз стану плодів за зображенням, зокрема Banana Ripeness Detection, sBuah та Fruit Quality Classification. Водночас навіть ці застосунки переважно реалізують або спрощену бінарну класифікацію стану об’єкта, або визначення обмеженої кількості категорій, що не завжди відповідає потребі більш детального поділу за ступенями стиглості.

Крім того, значна частина доступних вебрішень орієнтована на суміжні задачі комп’ютерного зору, зокрема загальну класифікацію фруктів і овочів, виявлення дефектів, поділ на свіжі та зіпсовані об’єкти або локалізацію продуктів на зображенні. Такі підходи є важливими з практичної точки зору, однак лише частково відповідають постановці задачі визначення саме ступеня стиглості фруктів. Окремо слід відзначити, що не всі проаналізовані рішення містять відкрито оприлюднені показники ефективності. Наявність показників якості, зокрема точності на навчальній, валідаційній та тестовій вибірках, є важливою для об’єктивного порівняння застосунків і оцінювання практичної придатності моделей. Відсутність таких даних у частини рішень ускладнює їх змістовне зіставлення за рівнем якості розпізнавання. Основні характеристики проаналізованих вебзастосунків наведено в таблиці А.1.

Включення до аналізу рішень для овочів і загальної плодово-овочевої продукції є обґрунтованим, оскільки такі застосунки реалізують подібну загальну схему роботи: завантаження зображення користувачем, його оброблення моделлю комп’ютерного зору та повернення результату у вигляді класу, стану об’єкта або рівня упевненості прогнозу.

Це свідчить про наявність сформованих технічних підходів до створення подібних вебзастосунків, але водночас підтверджує потребу у вужчій спеціалізації рішення саме для визначення ступеня стиглості фруктів.

Аналіз сучасних вебзастосунків показав, що засоби комп’ютерного зору вже активно використовуються для класифікації фруктів і овочів, оцінювання їхнього стану, виявлення дефектів і часткового визначення стиглості. Разом із тим більшість доступних рішень орієнтовані або на загальне розпізнавання об’єктів, або на спрощене визначення стану за двома категоріями, зокрема «свіжий / зіпсований» або «якісний / зіпсований».

Спеціалізовані вебзастосунки, призначені саме для багаторівневого визначення ступеня стиглості фруктів, у відкритому доступі є обмежено.

Крім того, не всі рішення містять відкрито оприлюднені метрики якості, що ускладнює їх об’єктивне порівняння. Усе це підтверджує доцільність розроблення власного вебзастосунку, орієнтованого саме на визначення ступеня стиглості фруктів засобами комп’ютерного зору у зручному для користувача вебформаті.

1.2 Аналіз літературних джерел щодо апробації результатів розроблення вебзастосунків визначення певного ступеня конкретного фактору на об’єктах за допомогою засобів комп’ютерного зору, зокрема ступеня стиглості фруктів

Аналіз літературних джерел [6 – 17] засвідчує, що визначення ступеня стиглості фруктів засобами комп’ютерного зору є актуальним напрямом досліджень у галузі штучного інтелекту, машинного навчання та автоматизованого контролю якості сільськогосподарської продукції.

У працях [6 – 14] цю задачу розглядають як різновид автоматизованого аналізу візуальних ознак об’єкта. Насамперед ідеться про колір, текстуру поверхні, форму та зовнішні зміни, що виникають у процесі дозрівання.

Значна частина досліджень [10 – 15] зосереджена переважно на моделі та експериментальній оцінці її якості. При цьому питання створення зручного вебзастосунку для кінцевого користувача, реалізації інтерфейсу, вебдоступності та інтеграції в прикладний сервіс зазвичай залишаються поза основною увагою авторів.

Першу групу джерел становлять оглядові праці [6, 7], що формують загальне уявлення про сучасний стан проблеми. У систематичному огляді узагальнено результати досліджень із використанням традиційних і глибоких моделей навчання для неруйнівного визначення стиглості бананів.

У праці [6] показано, що серед візуальних ознак найчастіше використовують саме колірні характеристики, а камери залишаються основним засобом отримання зображень. Огляд, присвячений виявленню та розпізнаванню фруктів на основі глибокого навчання, розширює це бачення. У ньому акцентовано увагу на проблемах якості наборів даних, складному тлі, перекритті плодів, різних масштабах об’єктів та необхідності використання легких моделей. Праці [6, 7] дають змогу виявити не лише переваги сучасних підходів, а й типові обмеження, які слід ураховувати під час проєктування прикладної системи. Водночас оглядові роботи [6, 7] не дають готового рішення для побудови вебзастосунку, а лише окреслюють спектр методів і проблем. Тому вони потребують подальшої конкретизації в межах прикладної задачі.

Другу групу становлять праці [8, 9], у яких ключову роль відіграють колірні ознаки та класичні методи комп’ютерного зору. Базовим у цьому контексті є дослідження, присвячене автоматичному визначенню стадій стиглості манго [8]. Воно демонструє, що ще до широкого поширення сучасних глибоких моделей задача оцінювання стиглості могла розв’язуватися за допомогою цифрового оброблення зображень, виділення характерних ознак та подальшої класифікації. Такі підходи є зрозумілими, відносно простими в реалізації та менш вимогливими до обчислювальних ресурсів.

Проте їхня слабкість полягає у високій залежності від контрольованих умов знімання, однорідного тла, стабільного освітлення та якості попередньої сегментації зображення. Саме тому високий результат, отриманий у межах одного експерименту, не завжди можна прямо перенести на реальні умови використання, де освітлення, фон і положення об’єкта можуть суттєво змінюватися.

Окремо слід виділити дослідження, у яких колірні характеристики використовуються вже в поєднанні з сучасними моделями глибокого навчання [9, 10]. У праці [9] розглянуто класифікацію стиглості бананів на основі RGB-зображень. Використано набір даних із 1565 зображень, зібраних протягом 20 днів. Це дослідження демонструє, що кольорові зображення справді є інформативною основою для визначення ступеня стиглості. Разом із тим наведені результати отримано на спеціально сформованому наборі даних, а отже, не до кінця зрозуміло, наскільки модель буде стійкою за зміни освітлення, складного тла або інших умов, характерних для реального використання. Крім того, з такого опису не випливає, чи передбачалася інтеграція моделі у вебзастосунок або інший сервіс для кінцевого користувача.

Третю групу джерел [10 – 13] становлять дослідження, у яких для визначення стиглості плодів безпосередньо застосовуються моделі глибокого навчання. У роботі [10] запропоновано підхід до автоматичного визначення стиглості бананів на основі розпізнавання зображень, а заявлена точність становить 98,8%. Такий результат свідчить про високу ефективність нейромережевих підходів для цієї задачі. Водночас сам по собі високий відсоток точності ще не гарантує універсальності моделі, якщо не розкрито повною мірою склад набору даних, кількість класів, різноманітність умов знімання та поведінку моделі на зображеннях поза межами навчального набору. Отже, робота [10] переконливо демонструє потенціал методу, але залишає відкритим питання його узагальнювальної здатності.

Подібна ситуація простежується і в дослідженнях, присвячених іншим культурам [11, 12]. У праці [11] запропоновано визначення стиглості полуниці в реальному часі на основі поєднання глибокого навчання та доповненої реальності; у доступному описі наведено точність 0,95.

Робота [12], що присвячена полуниці, подає кілька метрик якості: прецизійність 0,97, повноту 0,94 та середню точність виявлення при порозі IoU 0,5 на рівні 0,98. Ці результати підтверджують, що сучасні моделі можуть забезпечувати високу якість розпізнавання не лише за однією метрикою, а й за системою показників.

Однак обмеження таких праць полягає в тому, що вони здебільшого орієнтовані на конкретний вид плодів. Через це їхні результати не можна автоматично переносити на інші фрукти.

Крім того, без детального опису умов знімання, кількості класів і структури набору даних складно оцінити, наскільки такі моделі будуть стабільними за використання в неконтрольованому середовищі.

Аналогічні підходи застосовуються й до близької предметної області – овочевої продукції [13]. У дослідженні [13] запропоновано метод визначення стиглості томатів на основі вдосконаленої моделі YOLOv8, для якої наведено 95,8% прецизійності та 91,7% точності на тестовому наборі. Одержані результати свідчать про ефективність сучасних моделей виявлення об’єктів для задач, пов’язаних із визначенням ступеня стиглості. Водночас це дослідження стосується саме томатів, а не фруктів у вузькому розумінні теми даної роботи, тому його доцільно розглядати як суміжний приклад. Додатково слід урахувати, що висока якість на тестовому наборі не завжди означає таку саму ефективність за роботи в реальних умовах із неоднорідним тлом, різною якістю зображення та варіативністю зовнішнього вигляду об’єктів.

Четверту групу становлять праці [14, 15], у яких оцінювання стиглості поєднується з класифікацією плодово-овочевих об’єктів або з аналізом якості продукції.

У дослідженні [14] запропоновано підхід на основі глибокого навчання, що поєднує класифікацію фруктів і овочів та оцінювання їх стиглості. Саме така постановка є особливо близькою до логіки побудови прикладного сервісу, оскільки користувачеві зазвичай потрібне не лише визначення типу об’єкта, а й оцінювання його стану. Водночас із такого дослідження не випливає, якою мірою модель адаптована до реального вебвикористання. Також не уточнено, чи передбачено інтерфейс для завантаження зображень і наскільки система є зручною для кінцевого користувача. Наукова частина в роботі є достатньою, але прикладна вебреалізація розкрита не достатньо.

Практичний аспект доступності технології для реального впровадження розглянуто у праці, присвяченій спрощеним підходам глибокого навчання для оцінювання якості фруктів у малих господарствах [15]. Це дослідження акцентує увагу на важливому напрямі – побудові не лише точної, а й доступної системи. Разом із тим спрощення моделі зазвичай пов’язане з компромісом між обчислювальною складністю та якістю розпізнавання, а без детального порівняння на складних наборах даних важко оцінити, наскільки такий підхід буде стійким у ширшому спектрі умов. Крім того, як і в більшості розглянутих джерел [10, 14, 15], основний акцент зроблено на моделі, а не на побудові повноцінного вебзастосунку.

П’яту групу утворюють суміжні дослідження, у яких аналізується не стільки стиглість, скільки свіжість, псування або строк придатності плодів [16, 17]. У праці [16] запропоновано комбіновану систему оцінювання стиглості фруктів на основі штучного ольфакторного сенсора та глибокого навчання; для неї наведено 97,39% точності на валідаційному наборі та 82,20% – на тестовому. Це дослідження розширює уявлення про можливі ознаки стиглості та демонструє перспективність поєднання візуальних і невізуальних даних. Водночас такий підхід не може бути безпосередньо перенесений у межі цієї кваліфікаційної роботи, оскільки тут розглядається саме вебзастосунок на основі комп’ютерного зору, без використання додаткових сенсорних пристроїв.

У роботі [17] глибоке навчання застосовується для класифікації зіпсованих фруктів і визначення строку їх придатності. Це дослідження розширює контекст аналізу, оскільки демонструє зв’язок між стиглістю, свіжістю та часовими змінами стану плоду. Однак така постановка задачі є суміжною, а не тотожною визначенню ступеня стиглості, тому її результати доцільно використовувати насамперед для обґрунтування ширшої прикладної значущості візуального аналізу плодів, а не як прямий зразок для побудови системи багаторівневої оцінки стиглості .

Аналіз літературних джерел [6 – 17] свідчить, що найбільш інформативними для визначення ступеня стиглості фруктів залишаються колірні характеристики, текстурні ознаки поверхні та візуальні зміни зовнішнього стану плоду.

У сучасних дослідженнях [10 – 14] найвищі результати переважно демонструють моделі глибокого навчання, зокрема згорткові нейронні мережі та моделі сімейства YOLO. При цьому високі значення точності в окремих роботах не слід тлумачити як універсальний показник придатності моделі. Результати істотно залежать від кількості класів, умов знімання, однорідності тла, обсягу й структури набору даних та способу його формування.

Окремо слід підкреслити, що більшість розглянутих праць зосереджені передусім на моделі та експерименті [10, 14, 15], тоді як питання реалізації зручного вебзастосунку для кінцевого користувача висвітлюється значно рідше.

Саме це створює підстави для подальшого розроблення спеціалізованого вебзастосунку для визначення ступеня стиглості фруктів засобами комп’ютерного зору.

1.3 Постановка задачі

Автоматизоване визначення ступеня стиглості фруктів є актуальним практичним завданням у сфері комп’ютерного зору та інтелектуальних програмних систем. Традиційне візуальне оцінювання стиглості плодів людиною значною мірою залежить від суб’єктивного сприйняття, умов освітлення, досвіду користувача та особливостей конкретного об’єкта.

Проведений аналіз сучасних вебзастосунків (підрозділ 1.1) показав, що більшість наявних рішень орієнтовані або на загальну класифікацію фруктів і овочів, або на визначення свіжості та псування продукції, тоді як спеціалізованих вебзастосунків саме для визначення ступеня стиглості фруктів у відкритому доступі небагато.

Аналіз літературних джерел [6 – 17] засвідчив, що сучасні моделі комп’ютерного зору здатні забезпечувати високі показники якості розпізнавання, однак у більшості досліджень основний акцент зроблено на моделі та експерименті, а питання прикладної вебреалізації та зручності використання для кінцевого користувача висвітлено обмежено. У зв’язку з цим доцільним є розроблення вебзастосунку, який дасть змогу визначати ступінь стиглості фруктів за цифровими зображеннями засобами комп’ютерного зору.

Об’єктом роботи є процес визначення ступеня стиглості фруктів за цифровими зображеннями засобами комп’ютерного зору.

Метою роботи є розроблення вебзастосунку для визначення ступеня стиглості фруктів за цифровими зображеннями засобами комп’ютерного зору.

Для досягнення поставленої мети необхідно вирішити такі завдання:

− проаналізувати сучасні вебзастосунки для визначення ступеня стиглості фруктів і суміжних характеристик плодово-овочевої продукції засобами комп’ютерного зору;

– проаналізувати літературні джерела щодо використання засобів комп’ютерного зору для визначення ступеня стиглості фруктів і суміжних задач оцінювання стану плодово-овочевої продукції;

– визначити візуальні ознаки, релевантні для оцінювання ступеня стиглості фруктів на цифрових зображеннях;

– обрати та обґрунтувати модель комп’ютерного зору для автоматизованого розпізнавання ступеня стиглості фруктів;

– сформувати вимоги до вхідних даних, структури програмного рішення та функціональних можливостей вебзастосунку;

– реалізувати вебзастосунок для визначення ступеня стиглості фруктів;

– провести тестування розробленого вебзастосунку та оцінити якість визначення ступеня стиглості фруктів;

– виконати аналіз отриманих результатів і визначити напрями подальшого вдосконалення розробленого рішення.

2 РОЗРОБЛЕННЯ СТРУКТУРИ ВЕБЗАСТОСУНКУ ВИЗНАЧЕННЯ СТУПЕНЯ СТИГЛОСТІ ФРУКТІВ ЗА ДОПОМОГОЮ ЗАСОБІВ КОМП'ЮТЕРНОГО ЗОРУ

2.1 Особливості структури бази даних вебзастосунку визначення ступеня стиглості фруктів за допомогою засобів комп'ютерного зору

Для забезпечення збереження результатів аналізу в проєктованому вебзастосунку визначення ступеня стиглості фруктів передбачено використання бази даних. Її наявність дає змогу не лише фіксувати результати класифікації окремих зображень, а й організувати історію перевірок, отримувати детальну інформацію про кожний виконаний аналіз, формувати статистичні відомості щодо роботи системи та створити основу для подальшого розширення функціональності вебзастосунку. Таким чином, база даних у структурі системи виконує функцію централізованого збереження результатів розпізнавання, довідкової інформації та супровідних службових даних.

Для розв'язання цієї задачі обрано базу даних SQLite [18]. Такий вибір зумовлений простотою інтеграції із серверною частиною вебзастосунку, відсутністю потреби в окремому сервері бази даних та достатньою функціональністю для системи навчального призначення. Крім того, SQLite зручно використовувати на етапі проєктування та реалізації вебзастосунку, оскільки база даних зберігається у вигляді окремого файла [19]. Це спрощує розгортання, резервне копіювання та перенесення системи між різними середовищами.

Під час проєктування структури даних слід враховувати, що вебзастосунок працює не лише з результатом одного прогнозу, а з кількома взаємопов'язаними інформаційними сутностями. До них належать завантажені зображення, результати розпізнавання, перелік фруктів, рівні стиглості, рекомендації користувачеві та відомості про використану версію

моделі. Саме тому організація бази даних має бути достатньо гнучкою, щоб підтримувати як поточні функції системи, так і її подальший розвиток.

Логічна структура бази даних вебзастосунку може бути подана у вигляді взаємозв’язку кількох основних сутностей: «Зображення», «Результати аналізу», «Фрукти», «Рівні стиглості» та «Версії моделі». Центральною сутністю є результат аналізу, оскільки саме він поєднує завантажене користувачем зображення, визначений фрукт, встановлений рівень стиглості, рівень упевненості моделі та службову інформацію про використану модель. Такий підхід дає змогу зберігати не лише підсумковий результат для користувача, а й технічні дані, необхідні для подальшого аналізу роботи системи. Логічну схему бази даних подано на рисунку 2.1. Вона відображає основні сутності та зв’язки між ними.

Рисунок 2.1 – Логічна схема бази даних

У зазначеній схемі сутність «Зображення» містить інформацію про завантажений файл, зокрема його назву, шлях у файловій системі, формат і дату завантаження.

Сутність «Результати аналізу» зберігає підсумковий результат роботи моделі: прогнозований клас, визначений фрукт, рівень стиглості, рівень упевненості та рекомендацію для користувача.

Сутності «Фрукти» та «Рівні стиглості» мають довідковий характер і використовуються для впорядкування можливих значень.

Сутність «Версії моделі» дає змогу фіксувати, яка саме модель або її версія була використана під час конкретного аналізу. Перелік основних інформаційних сутностей бази даних вебзастосунку та їх призначення наведено в таблиці 2.1.

Таблиця 2.1 – Основні інформаційні сутності бази даних вебзастосунку

Сутність	**Призначення**	**Основні дані, що зберігаються**
1	**2**	**3**
Результати аналізу	Збереження підсумків роботи моделі комп’ютерного зору	Ідентифікатор аналізу, прогнозований клас, фрукт, рівень стиглості, рівень упевненості, дата аналізу, рекомендація
Фрукти	Збереження переліку фруктів, які підтримуються системою	Назва фрукта, опис, службовий код або ідентифікатор
Рівні стиглості	Збереження можливих градацій стиглості	Назва рівня стиглості, опис, рекомендація для користувача
Версії моделі	Фіксація інформації про модель, яка виконувала аналіз	Назва моделі, версія, дата оновлення, опис

Продовження таблиці 2.1

1	2	3
Зображення	Збереження відомостей про завантажені користувачем файли	Назва файлу, шлях до файлу, формат, дата завантаження

Як видно з таблиці 2.1, центральною сутністю бази даних є результат аналізу, оскільки він поєднує інформацію про завантажене зображення, визначений фрукт, рівень стиглості та використану модель.

Особливістю запропонованої структури є те, що зображення не варто зберігати безпосередньо в базі даних. Це пояснюється тим, що графічні файли можуть істотно збільшувати її обсяг і ускладнювати операції доступу до інформації. Раціональнішим є збереження завантажених зображень у файловій системі, тоді як у базі даних фіксується лише шлях до відповідного файлу та пов’язані з ним результати аналізу. Такий підхід забезпечує економніше використання ресурсів, зменшує навантаження на сховище даних та спрощує роботу із завантаженими файлами.

Інформаційний потік у межах роботи вебзастосунку з базою даних подано на рисунку 2.2.

Рисунок 2.2 – Схема інформаційного потоку під час аналізу зображення

На першому етапі користувач завантажує цифрове зображення фрукта через вебінтерфейс. Після цього серверна частина зберігає файл у відповідному каталозі та передає його на попереднє оброблення. Далі модель комп’ютерного зору виконує класифікацію зображення і визначає прогнозований клас. На основі цього класу система формує зрозумілий для користувача результат: назву фрукта, ступінь стиглості, рівень упевненості моделі та рекомендацію. Після завершення аналізу відповідні дані записуються до бази даних, що дає змогу надалі переглядати історію перевірок і формувати статистику. База даних у цій схемі виконує роль сховища результатів аналізу та джерела даних для подальшого перегляду історії й формування статистики.

З погляду логіки організації даних варто передбачити виокремлення основної сутності результату аналізу та кількох допоміжних сутностей довідкового характеру. Основна сутність акумулює інформацію про конкретне завантажене зображення, результат класифікації, рівень упевненості моделі, технічний клас прогнозу, рекомендацію для користувача, дату виконання аналізу та інші параметри, необхідні для подальшого перегляду й оцінювання результатів. Допоміжні сутності використовуються для збереження переліку підтримуваних фруктів, можливих рівнів стиглості, а також службової інформації про версії моделі. Така організація даних дає змогу уникнути дублювання, підвищити узгодженість записів та спростити оновлення системи в разі розширення переліку фруктів або зміни класифікаційних категорій.

Запропонована база даних може містити кілька взаємопов’язаних таблиць, що відповідають основним інформаційним сутностям системи: зображенням, результатам аналізу, фруктам, рівням стиглості та версіям моделі. Водночас центральною таблицею є таблиця результатів аналізу, оскільки саме вона поєднує дані про завантажене зображення, визначений фрукт, рівень стиглості, упевненість моделі та службову інформацію про виконаний прогноз.

У межах цього підрозділу детально подано структуру саме цієї основної таблиці. Вона повинна містити ключові поля, необхідні для збереження інформації про кожний виконаний прогноз. Її структуру подано в таблиці 2.2.

Таблиця 2.2 – Структура основної таблиці результатів аналізу

Поле	**Тип даних**	**Призначення**
id	INTEGER	Унікальний ідентифікатор запису
image_path	TEXT	Шлях до завантаженого зображення у файловій системі
fruit_name	TEXT	Назва фрукта, визначеного системою
ripeness_level	TEXT	Визначений рівень стиглості
predicted_class	TEXT	Технічна назва прогнозованого класу моделі
confidence	REAL	Рівень упевненості моделі
recommendation	TEXT	Текстова рекомендація для користувача
model_version	TEXT	Версія моделі, використаної для аналізу
created_at	DATETIME	Дата і час виконання аналізу

Наведена структура дає змогу зберігати як користувацькі дані, необхідні для відображення результату, так і службову інформацію, потрібну для аналізу роботи моделі.

Важливою особливістю структури бази даних є поєднання користувацьких і службових даних в одному інформаційному середовищі. З одного боку, система повинна зберігати зрозумілий для користувача результат: назву фрукта, встановлений ступінь стиглості та текстову рекомендацію. З іншого боку, для аналізу роботи моделі потрібно фіксувати технічне позначення прогнозованого класу, рівень упевненості та, за потреби, імовірності для всіх можливих класів.

Завдяки цьому база даних може використовуватися не лише як засіб збереження результатів, а й як основа для контролю якості роботи моделі, порівняння прогнозів та подальшого вдосконалення системи.

Окремої уваги потребує питання масштабованості проєктованої структури. Хоча на початковому етапі вебзастосунок може підтримувати обмежену кількість фруктів і рівнів стиглості, база даних має бути придатною до розширення. У подальшому може виникнути потреба додати нові фрукти, збільшити кількість градацій стиглості, реалізувати збереження історії змін моделей, врахувати дані про користувачів або впровадити механізми оцінювання коректності результатів. Тому під час проєктування потрібно передбачити таку організацію даних, яка дозволить масштабувати систему без суттєвого перегляду її базової структури.

Запропонована база даних має забезпечувати реалізацію кількох основних функцій вебзастосунку. По-перше, вона підтримує збереження результатів аналізу кожного завантаженого зображення. По-друге, така організація даних забезпечує перегляд історії перевірок і отримання детальної інформації про окремий аналіз. По-третє, на основі накопичених записів можна формувати статистику за видами фруктів, ступенями стиглості, частотою розпізнавання певних класів і середнім рівнем упевненості моделі. По-четверте, збережені дані можуть стати основою для подальшої аналітики, пов’язаної з оцінюванням стабільності роботи моделі в різних умовах використання.

Під час проєктування серверної частини слід передбачити набір запитів, які забезпечуватимуть взаємодію з базою даних. Запит для аналізу зображення має не лише ініціювати роботу моделі, а й формувати новий запис про виконаний аналіз. Окремі запити забезпечують отримання історії аналізів, перегляд деталей конкретного результату та формування узагальненої статистики. Наявність таких сценаріїв використання підтверджує, що база даних має бути зорієнтована не лише на просте збереження записів, а й на накопичення результатів у межах вебзастосунку.

Розроблена структура бази даних дає змогу впорядковано зберігати результати аналізу та пов’язані з ними службові дані. Поділ інформації на окремі сутності, зокрема зображення, результати аналізу, фрукти, рівні стиглості та версії моделі, забезпечує логічну організацію даних і спрощує подальше розширення системи. Зберігання графічних файлів у файловій системі з фіксацією шляхів до них у базі даних зменшує навантаження на сховище та робить роботу із завантаженими зображеннями більш зручною. Такий підхід дозволяє реалізувати історію перевірок, перегляд окремих результатів, формування статистичних відомостей і подальше вдосконалення функціональних можливостей вебзастосунку.

2.2 Моделювання структури вебзастосунку визначення ступеня стиглості фруктів

Моделювання структури вебзастосунку визначення ступеня стиглості фруктів здійснюється з урахуванням того, що така система повинна поєднувати засоби веброзроблення, машинного навчання, збереження даних та оброблення графічної інформації в межах єдиного функціонального середовища. На відміну від автономної моделі класифікації, вебзастосунок має забезпечувати повний цикл взаємодії з користувачем: від завантаження зображення до повернення результату розпізнавання, його пояснення, збереження та подальшого перегляду. Саме тому під час проєктування системи передбачено не лише модуль класифікації, а й клієнтську частину, серверну частину, файлову систему та базу даних.

Основою проєктованого вебзастосунку є клієнт-серверна архітектура. Її використання забезпечує логічне розмежування функцій між інтерфейсом користувача та обчислювальною частиною системи. Клієнтська частина відповідає за взаємодію з користувачем, приймання вхідних даних, попередній перегляд зображення та відображення результатів аналізу.

Серверна частина забезпечує приймання файлів, їх перевірку, попереднє оброблення, виклик моделі комп’ютерного зору, формування відповіді та взаємодію з базою даних.

Як основу клієнтської частини передбачено використання React [20], оскільки ця технологія дає змогу створювати зручний інтерактивний інтерфейс користувача. Для серверної частини обрано FastAPI [21], що є придатним для швидкого оброблення запитів, реалізації API та взаємодії з модулем машинного навчання. У структурі системи також передбачено модель комп’ютерного зору для класифікації зображень, базу даних SQLite для збереження результатів аналізу та файлову систему для роботи із завантаженими зображеннями. Загальну структурну схему вебзастосунку подано на рисунку 2.3. Вона відображає основні компоненти системи та зв’язки між ними.

На рисунку 2.3 користувач взаємодіє лише з клієнтською частиною вебзастосунку. Після завантаження зображення frontend передає файл на backend, де виконуються перевірка, попереднє оброблення, передавання даних до моделі комп’ютерного зору та формування результату [22]. Серверна частина також відповідає за запис результатів у базу даних і збереження файлів у файловій системі. Такий підхід є важливим з погляду безпеки та підтримання системи, оскільки користувач не має прямого доступу ні до моделі, ні до бази даних, ні до внутрішньої структури збереження файлів.

Функціональна повнота вебзастосунку забезпечується сукупністю взаємопов’язаних модулів. Модуль введення даних забезпечує завантаження зображення через вебінтерфейс. Модуль попереднього оброблення відповідає за перевірку типу файлу, збереження зображення у файловій системі та приведення його до формату, придатного для подальшого аналізу [23].

Модуль класифікації використовує навчену модель комп’ютерного зору для визначення прогнозованого класу.

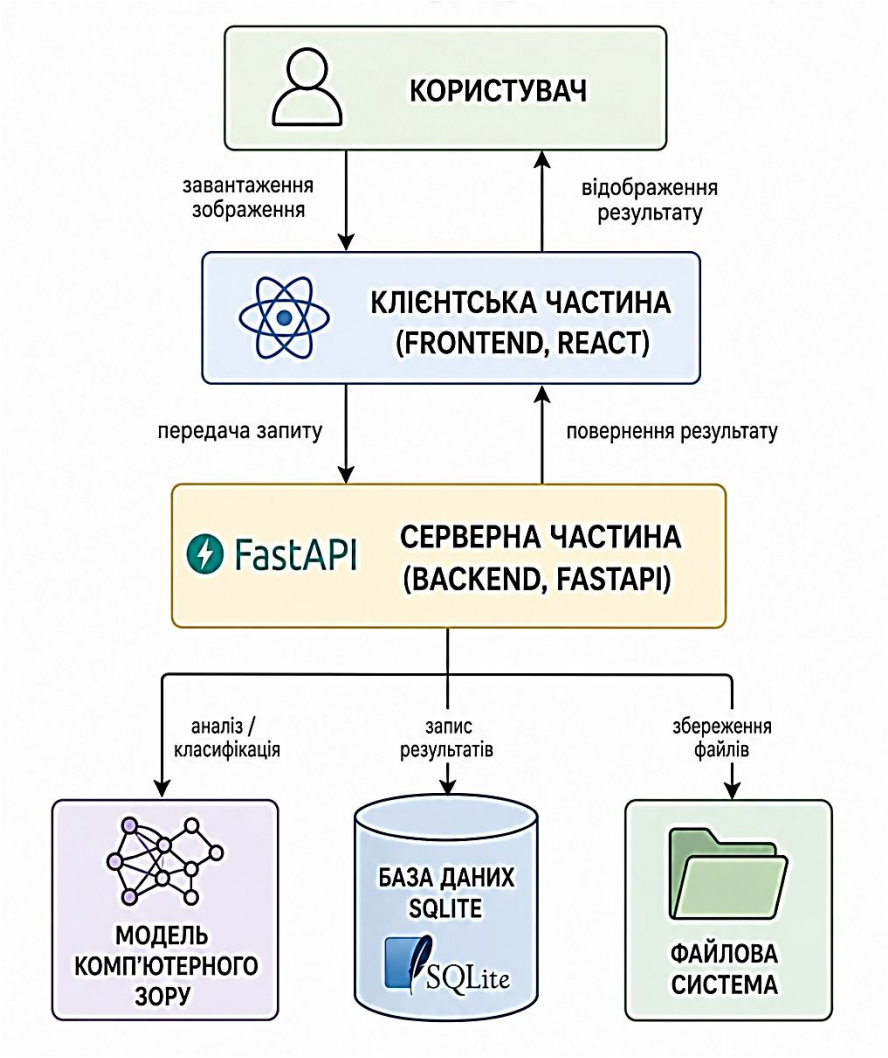

Рисунок 2.3 – Загальна структурна схема вебзастосунку

Модуль інтерпретації результату перетворює технічний клас моделі на зрозуміле повідомлення для користувача. Окремо передбачено модуль збереження результатів, модуль історії аналізів і модуль статистики. Основні компоненти вебзастосунку та їх призначення узагальнено в таблиці 2.3.

Як видно з таблиці 2.3, кожний компонент виконує окрему функцію, однак повноцінна робота вебзастосунку досягається саме завдяки їх узгодженій взаємодії. Клієнтська частина забезпечує зручність використання, серверна частина керує основною логікою, модель виконує розпізнавання, а база даних і файлова система відповідають за збереження результатів та вхідних даних. Однією з ключових частин моделювання структури вебзастосунку є формалізація алгоритму його роботи.

Таблиця 2.3 – Основні компоненти вебзастосунку та їх призначення

Компонент	**Призначення**	**Основні функції**
1	**2**	**3**
Клієнтська частина	Забезпечення взаємодії користувача із системою	Завантаження зображення, попередній перегляд, відображення результату, перегляд історії
Серверна частина	Оброблення запитів і керування логікою роботи системи	Приймання файлів, перевірка даних, виклик моделі, формування відповіді, робота з базою даних
Модель комп’ютерного зору	Автоматизоване визначення класу зображення	Класифікація фрукта та його ступеня стиглості
База даних	Збереження результатів аналізу та службової інформації	Запис результатів, збереження історії, формування статистики
Файлова система	Збереження завантажених зображень	Збереження файлів і передавання шляхів до бази даних
Модуль історії	Надання доступу до попередніх перевірок	Перегляд списку виконаних аналізів і деталей окремого запису
Модуль статистики	Узагальнення накопичених результатів	Підрахунок кількості аналізів, частоти класів, середнього рівня упевненості

У загальному вигляді алгоритм функціонування системи можна описати як послідовність етапів оброблення даних. Спочатку користувач завантажує фото фрукта через вебінтерфейс. Далі клієнтська частина передає зображення на сервер. Серверна частина перевіряє коректність файлу, зберігає його у файловій системі та виконує попереднє оброблення. Після цього підготовлене зображення передається до навченої моделі комп'ютерного зору.

Модель формує прогнозований клас, на основі якого серверна частина визначає назву фрукта, ступінь стиглості та рівень упевненості прогнозу. Після цього система формує текстову рекомендацію, записує результат аналізу до бази даних і повертає відповідь користувачеві. Така логіка забезпечує повний цикл роботи вебзастосунку: від отримання вхідного зображення до збереження результату та його відображення у вебінтерфейсі. Алгоритм роботи вебзастосунку подано у вигляді блок-схеми на рисунку 2.4.

Блок-схема на рисунку 2.4 ілюструє послідовність проходження запиту від користувача до серверної частини, модуля класифікації, бази даних і назад до користувача. Вона також показує, що процес аналізу зображення не обмежується лише викликом моделі, а охоплює перевірку файлу, попереднє оброблення, інтерпретацію результату та його збереження.

Для формалізації процесу розпізнавання використано спрощену математичну модель роботи системи. Нехай вхідне зображення позначено як x. Після його отримання серверна частина виконує попереднє оброблення, яке можна описати перетворенням $T(x)$, де T – функція приведення зображення до потрібного формату. Далі навчена модель комп'ютерного зору f формує вектор імовірностей.

$$P = f(T(x)) = (p_1, p_2, \dots, p_n), \tag{2.1}$$

де p_i – імовірність належності зображення до i-го класу;

n – кількість підтримуваних класів.

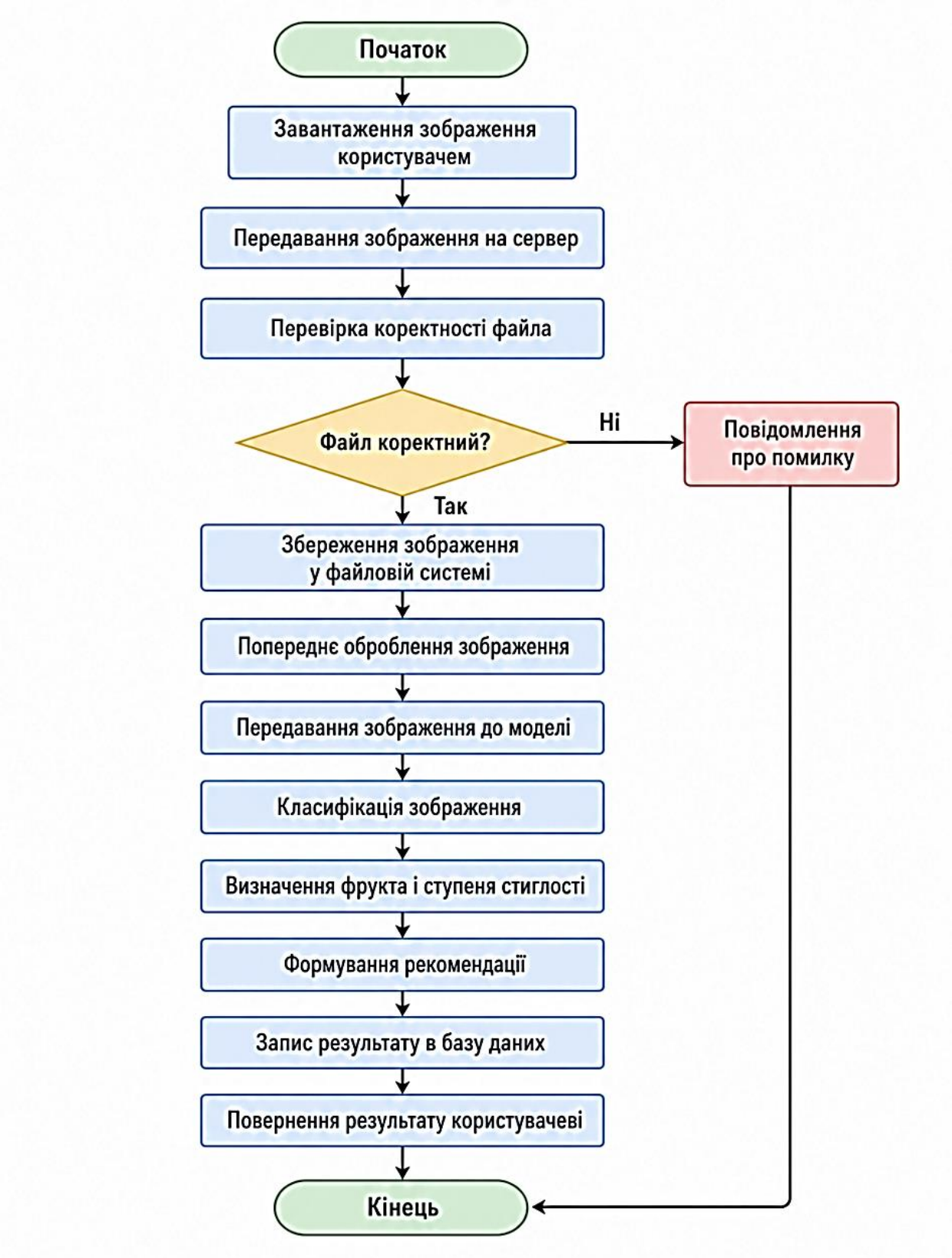

Рисунок 2.4 – Блок-схема алгоритму роботи вебзастосунку

Остаточний результат розпізнавання визначається як клас із найбільшою імовірністю.

$$c^* = \arg\max_i p_i. \quad (2.2)$$

Після цього технічний клас c^* перетворюється на пару значень «тип фрукта – ступінь стиглості» та використовується для формування текстової рекомендації користувачеві. Така формалізація є достатньою для опису логіки прийняття рішення, оскільки вона показує основні етапи перетворення вхідного зображення на підсумковий результат.

Особливістю проєктованого вебзастосунку є використання об’єднаних класів, які одночасно кодують назву фрукта і ступінь його стиглості. Наприклад, технічний клас типу mango_ripe містить інформацію про фрукт «манго» і стан «стиглий». Такий підхід дає змогу одній моделі одночасно розв’язувати дві взаємопов’язані задачі: визначення типу об’єкта та оцінювання його стиглості.

Використання об’єднаних класів спрощує загальну структуру системи, оскільки немає потреби окремо запускати модель ідентифікації фрукта та модель визначення ступеня стиглості. Після отримання прогнозу серверна частина перетворює технічний клас на користувацький результат, що містить зрозумілу назву фрукта, ступінь стиглості, рівень упевненості моделі та коротку рекомендацію.

Під час моделювання вебзастосунку необхідно врахувати можливість подальшого розширення системи. На початковому етапі система може підтримувати обмежену кількість фруктів і рівнів стиглості, наприклад банан, манго, апельсин і полуницю, а для кожного з них – три стани: недостиглий, стиглий і перестиглий. Така структура забезпечує баланс між функціональною повнотою та складністю реалізації.

Водночас архітектура має залишатися відкритою до подальшого розвитку. У майбутньому до системи можуть бути додані нові фрукти, нові рівні стиглості, інші версії моделі або додаткові сервісні функції. Це підтверджує сенс використання клієнт-серверної архітектури, модуля класифікації, бази даних і файлової системи для збереження зображень.

Окремим аспектом другого розділу є об’єктно-орієнтоване моделювання етапів вирішення задачі. Для цього використано стандартні засоби UML [24], які дають змогу подати структуру системи, її компоненти та сценарії взаємодії в узагальненому вигляді. UML-моделювання допомагає чітко визначити межі системи, ролі її елементів і послідовність виконання основних операцій.

Одним із важливих елементів такого моделювання є діаграма варіантів використання. Вона відображає основні сценарії взаємодії користувача з вебзастосунком і дає змогу показати функціональні можливості системи з позиції зовнішнього користувача. У межах проєктованого вебзастосунку користувач може завантажити зображення фрукта, запустити аналіз, переглянути результат визначення ступеня стиглості, переглянути історію попередніх аналізів, отримати детальну інформацію про окремий результат і переглянути узагальнену статистику. UML-діаграму варіантів використання вебзастосунку подано на рисунку 2.5.

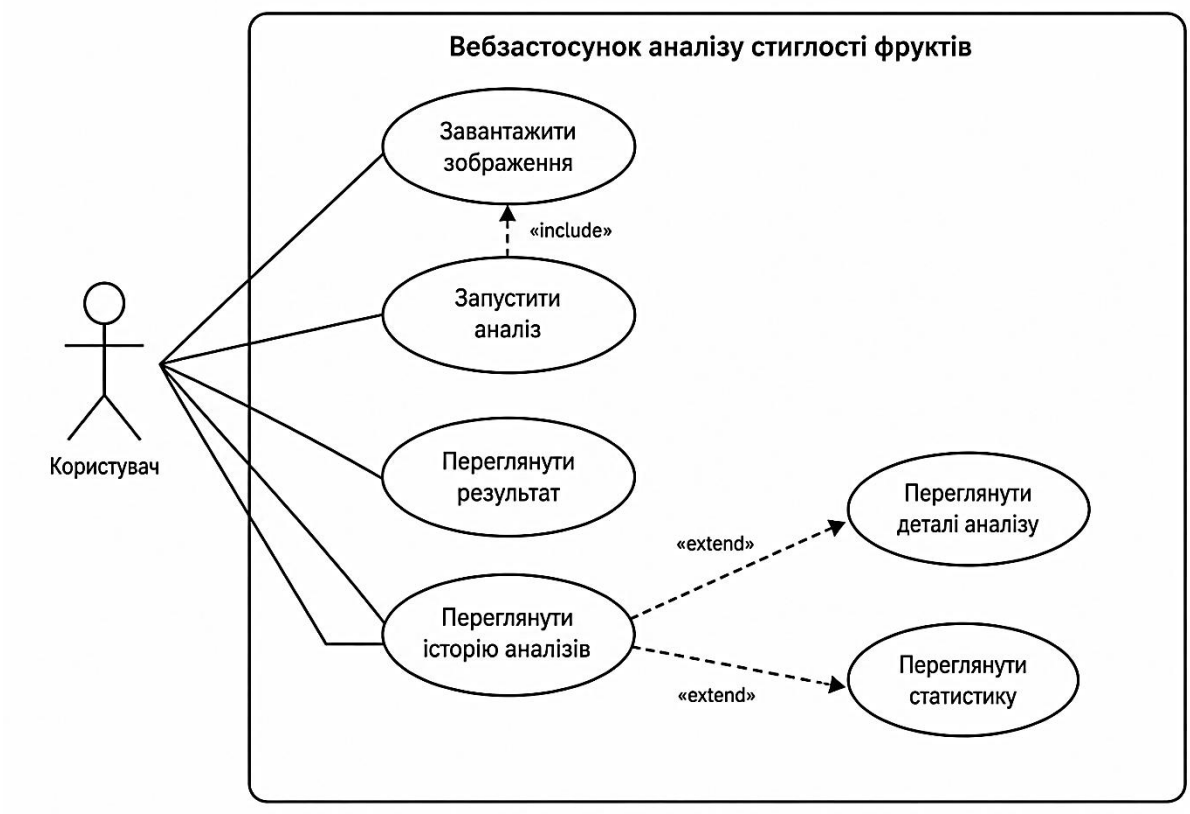

Рисунок 2.5 – UML-діаграма варіантів використання вебзастосунку

На рисунку 2.5 актором виступає користувач, який взаємодіє з основними функціями системи. До ключових варіантів використання належать завантаження зображення, запуск аналізу, перегляд результату, перегляд історії аналізів, перегляд деталей окремого запису та отримання статистики. Така діаграма допомагає окреслити функціональні межі вебзастосунку й показує, які дії має підтримувати система на рівні користувацької взаємодії.

Для відображення внутрішньої будови системи використано діаграму компонентів. Вона показує, з яких основних програмних частин складається вебзастосунок і як ці частини взаємодіють між собою. Діаграму компонентів вебзастосунку подано на рисунку 2.6.

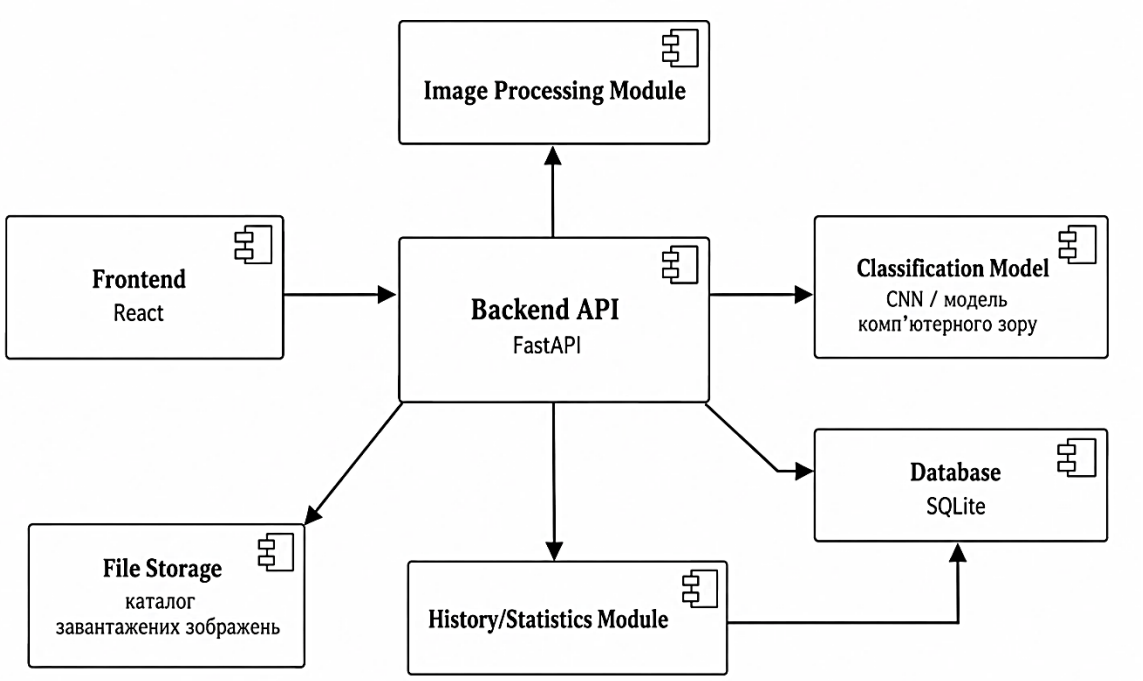

Рисунок 2.6 – UML-діаграма компонентів вебзастосунку

На рисунку 2.6 відображено клієнтську частину, серверну частину, модель комп'ютерного зору, базу даних і файлову систему. Діаграма показує, що frontend надсилає запити на backend, backend взаємодіє з моделлю класифікації та базою даних, а також звертається до файлової системи для збереження завантажених зображень.

Це дає змогу представити вебзастосунок як систему взаємопов’язаних компонентів, а не як окрему модель розпізнавання. Для деталізації проходження одного запиту використано діаграму послідовності, яку подано на рисунку 2.7.

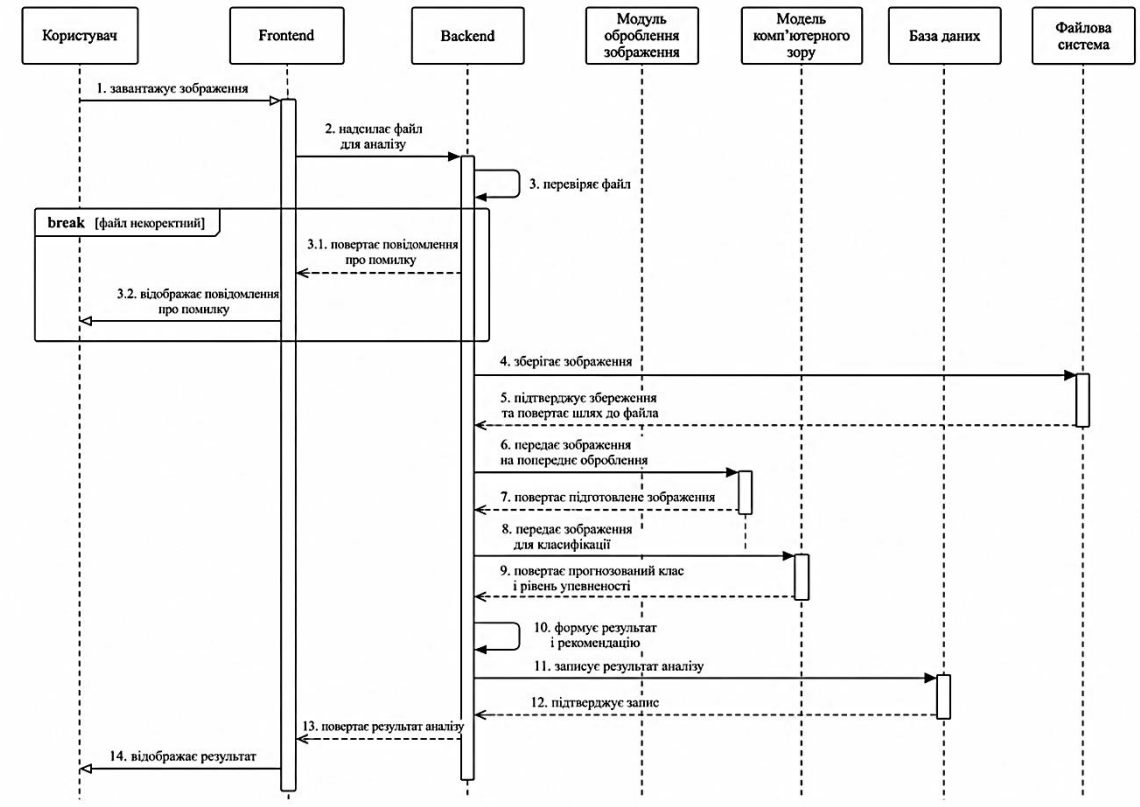

Рисунок 2.7 – UML-діаграма послідовності виконання аналізу зображення

На рисунку 2.7 показано, як користувач ініціює аналіз, як клієнтська частина передає файл на сервер, як сервер виконує перевірку, звертається до модуля оброблення зображення, передає дані моделі, отримує результат, записує його в базу даних і повертає відповідь користувачеві. Така діаграма є важливою для опису вебзастосунку, оскільки вона демонструє часову послідовність взаємодії між компонентами в межах одного запиту.

З погляду практичного використання проєктована структура повинна підтримувати не лише разове визначення ступеня стиглості, а й додаткові сценарії роботи. До них належать перегляд історії аналізів, отримання деталей конкретного запису та формування узагальненої статистики.

Наявність таких функцій робить вебзастосунок більш повноцінним, оскільки користувач отримує не лише відповідь на окремий запит, а й доступ до накопичених результатів.

Збережені результати можуть використовуватися для аналізу повторюваності певних прогнозів, порівняння результатів для різних фруктів і відстеження стабільності роботи моделі в часі. Це також створює основу для подальшого вдосконалення системи, зокрема оновлення моделі, розширення набору класів і поліпшення рекомендацій для користувача.

Запропонована структура вебзастосунку забезпечує повний цикл оброблення зображення: від його завантаження користувачем до отримання результату класифікації, формування рекомендації та збереження даних для подальшого перегляду. Розподіл системи на клієнтську частину, серверну частину, модуль комп’ютерного зору, базу даних і файлову систему дає змогу чітко розмежувати функції окремих компонентів і спростити подальшу підтримку програмного рішення. Використання об’єднаних класів, які одночасно містять інформацію про фрукт і рівень його стиглості, дозволяє реалізувати розпізнавання в межах однієї моделі без ускладнення архітектури. Подана структура також підтримує розширення функціональності системи, зокрема додавання нових фруктів, рівнів стиглості, версій моделі, історії аналізів і статистичних можливостей. Таким чином, побудована модель вебзастосунку створює основу для практичної реалізації системи визначення ступеня стиглості фруктів засобами комп’ютерного зору.

3 РОЗРОБЛЕННЯ ВЕБЗАСТОСУНКУ ВИЗНАЧЕННЯ СТУПЕНЯ СТИГЛОСТІ ФРУКТІВ ЗА ДОПОМОГОЮ ЗАСОБІВ КОМП'ЮТЕРНОГО ЗОРУ

3.1 Вибір інструментальних засобів для реалізації поставленої задачі

Реалізація вебзастосунку для визначення ступеня стиглості фруктів потребувала поєднання засобів машинного навчання, оброблення зображень, веброзроблення та збереження даних. Добір технологій здійснювався з урахуванням їх сумісності, придатності до інтеграції навченої моделі у вебсередовище, складності розгортання та можливості подальшого розвитку системи. Використаний набір інструментів охоплює весь цикл роботи програмного рішення: підготовку даних і навчання моделі, оброблення користувацьких зображень, виконання прогнозування, збереження отриманих результатів та їх відображення у вебінтерфейсі.

Основною мовою програмування для реалізації обчислювальної та серверної частин обрано Python [25]. Вона використовується для завантаження навченої моделі, попереднього оброблення зображень, формування вхідних даних, виконання прогнозування та роботи з базою даних. Застосування однієї мови для основних серверних операцій спрощує інтеграцію окремих модулів, налагодження програмного коду й подальший супровід вебзастосунку. Важливою перевагою Python у межах поставленої задачі є також наявність бібліотек для машинного навчання, комп'ютерного зору, числових обчислень і створення прикладних програмних інтерфейсів.

Для створення моделі комп'ютерного зору використано TensorFlow і Keras [26, 27]. TensorFlow виконує основні операції побудови, навчання та використання нейронної мережі, тоді як Keras надає високорівневі засоби опису її архітектури. Поєднання цих технологій дало змогу застосувати навчання з перенесенням і адаптувати попередньо навчену модель до предметної області кваліфікаційної роботи.

Основою класифікатора стала архітектура EfficientNetB0 [28, 29]. Її базова частина використовується для виділення характерних ознак із вхідного зображення, а додані класифікаційні шари формують прогноз відповідно до набору класів вебзастосунку. EfficientNetB0 обрано з огляду на співвідношення якості класифікації та обчислювальної складності. Порівняно з більш ресурсоємними архітектурами вона має менший обсяг обчислень, що полегшує інтеграцію навченої моделі із серверною частиною.

Фінальна версія моделі розпізнає три види фруктів: банан, манго та полуницю. Для кожного виду передбачено три ступені стиглості: недостиглий, стиглий і перестиглий.

У результаті сформовано дев’ять вихідних класів:

- banana_unripe;
- banana_ripe;
- banana_overripe;
- mango_unripe;
- mango_ripe;
- mango_overripe;
- strawberry_unripe;
- strawberry_ripe;
- strawberry_overripe.

Назва кожного технічного класу одночасно кодує вид фрукта та його стан. Завдяки цьому одна модель одночасно визначає вид фрукта та ступінь його стиглості. Після отримання прогнозу серверна частина розділяє технічне позначення на два зрозумілі для користувача значення, а також формує відповідну текстову рекомендацію.

Перед подаванням до моделі кожне зображення перетворюється до колірної моделі RGB і масштабується до розміру 224×224 пікселів. Уніфікація вхідних даних необхідна для відповідності структурі входу EfficientNetB0 та однакового оброблення всіх користувацьких файлів.

Під час навчання також застосовано аугментацію даних: горизонтальне віддзеркалення, невеликі повороти, масштабування та зміну контрастності. Такі перетворення збільшують різноманітність навчальних прикладів без додаткового збирання зображень і зменшують залежність моделі від конкретного ракурсу, відстані до об’єкта або умов освітлення.

Після базової частини EfficientNetB0 використано шар глобального усереднення ознак, шар регуляризації Dropout і повнозв’язний вихідний шар. Останній містить дев’ять нейронів відповідно до кількості класів і використовує функцію активації softmax. Для вхідного зображення модель формує вектор із дев’яти значень, кожне з яких характеризує оцінену імовірність належності зображення до певного класу. Клас із найбільшим значенням приймається як результат прогнозування, а відповідна імовірність використовується для відображення рівня упевненості моделі. Архітектуру навченої моделі класифікації зображень подано на рисунку 3.1.

Навчання моделі виконувалося в середовищі Google Colab [30]. Воно дало змогу запускати програмний код у браузері, підключати Google Drive для збереження набору даних і результатів експериментів, а також використовувати графічний прискорювач без окремого налаштування локального середовища. Такий формат був зручним для багаторазового запуску навчання, зміни параметрів і порівняння отриманих результатів.

Після завершення навчання модель збережено у форматі .keras. Порядок вихідних класів зафіксовано в окремому файлі формату JSON. Роздільне збереження моделі та переліку класів дає змогу коректно інтерпретувати індекс прогнозованого класу під час виконання аналізу на сервері та полегшує заміну версії моделі без суттєвого переписування програмної логіки. Серверну частину вебзастосунку реалізовано за допомогою FastAPI. Вона приймає зображення від клієнтської частини, перевіряє тип і коректність файлу, зберігає його у файловій системі, виконує попереднє оброблення та передає підготовлені дані до моделі.

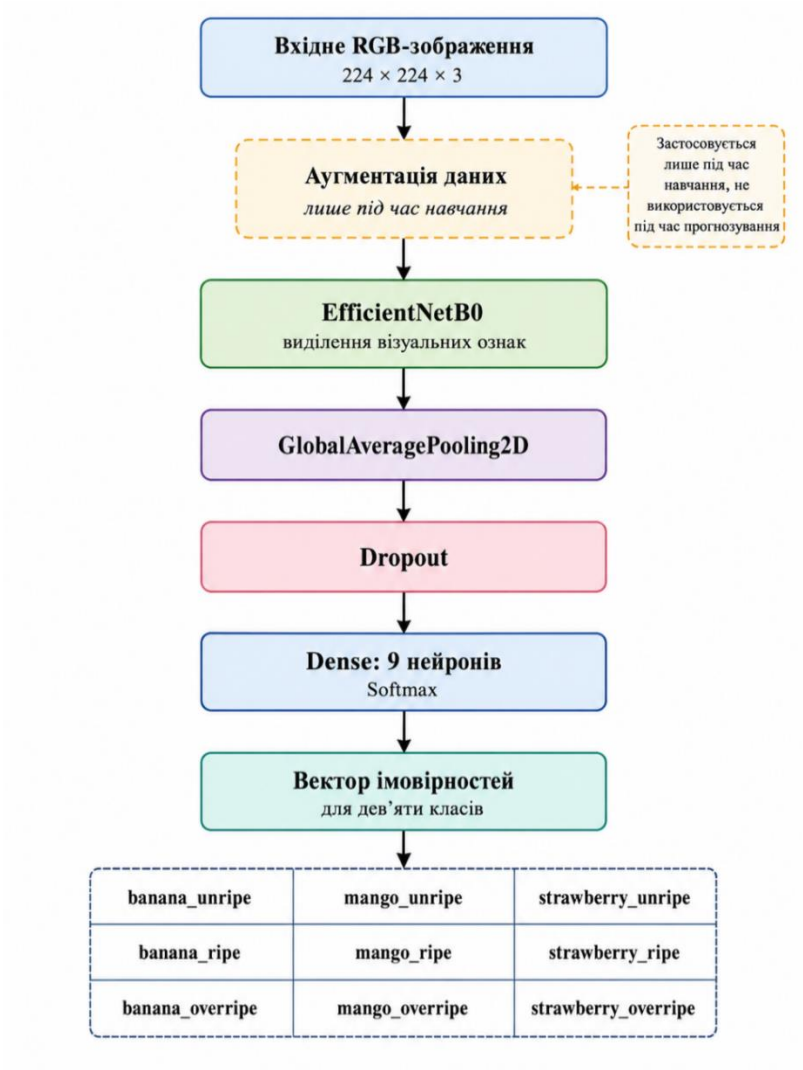

Рисунок 3.1 – Архітектура навченої моделі класифікації зображень

Після прогнозування сервер інтерпретує технічну назву класу, визначає фрукт і ступінь стиглості, формує рекомендацію, записує результат до бази даних і повертає відповідь клієнтові у форматі JSON.

Для запуску FastAPI-застосунку використано Uvicorn [31]. Він приймає мережеві запити та передає їх програмі серверної частини. Розмежування логіки API і механізму запуску сервера спрощує організацію проєкту та дає змогу незалежно змінювати внутрішню реалізацію оброблення запитів.

Попереднє оброблення користувацьких зображень на сервері виконується за допомогою Pillow і NumPy [32, 33]. Бібліотека Pillow відкриває файл, перетворює його у формат RGB і змінює розмір відповідно до вимог моделі.

NumPy формує числовий масив пікселів і додає вимір пакета, необхідний для передавання даних нейронній мережі. Таким чином, ці бібліотеки забезпечують перехід від графічного файла до числового подання, з яким працює модель.

Клієнтську частину вебзастосунку створено за допомогою React. Компонентний підхід дає змогу розділити інтерфейс на незалежні функціональні блоки: завантаження зображення, його попередній перегляд, відображення результату, історію аналізів і статистичні відомості. Обмін із сервером відбувається без повного перезавантаження сторінки, тому користувач отримує результат у межах поточного інтерфейсу.

Для створення та запуску React-проєкту використано Vite [34]. Він спрощує структуру клієнтського проєкту, керування залежностями та запуск локального середовища розроблення. Візуальне оформлення сторінок реалізовано засобами CSS [35], за допомогою яких визначено розташування елементів, вигляд кнопок і форм, подання результатів аналізу, історії та статистики.

Результати аналізу зберігаються в базі даних SQLite. У записі фіксуються назва завантаженого файлу, шлях до зображення, визначений фрукт, ступінь стиглості, технічний клас, рівень упевненості, імовірності за класами, сформована рекомендація, версія моделі та час виконання аналізу. Самі графічні файли розміщуються в каталозі uploads, а база даних містить шлях до відповідного зображення. Таке розділення запобігає надмірному збільшенню файлу бази даних і спрощує відображення фотографій в історії аналізів.

SQLite відповідає призначенню поточної версії вебзастосунку, оскільки не потребує окремого сервера бази даних і легко підключається до Python-коду. Її можливостей достатньо для збереження результатів, отримання історії перевірок, перегляду окремих записів і формування узагальненої статистики.

У разі подальшого збільшення кількості користувачів або обсягу інформації структура системи допускає перехід до серверної системи керування базами даних.

Розроблення й налагодження програмного коду виконувалося у Visual Studio Code. У цьому середовищі організовано серверну та клієнтську частини проєкту, налаштовано запуск програмних модулів і редагування конфігураційних файлів. Перевірка маршрутів FastAPI здійснювалася через Swagger UI [36], що автоматично формує інтерактивну документацію API та дає змогу надсилати тестові запити без використання клієнтського інтерфейсу. Поведінку цілісного вебзастосунку перевірено у браузері після одночасного запуску серверної та клієнтської частин. Взаємозв'язок використаних інструментальних засобів у структурі програмного рішення наведено на рисунку 3.2.

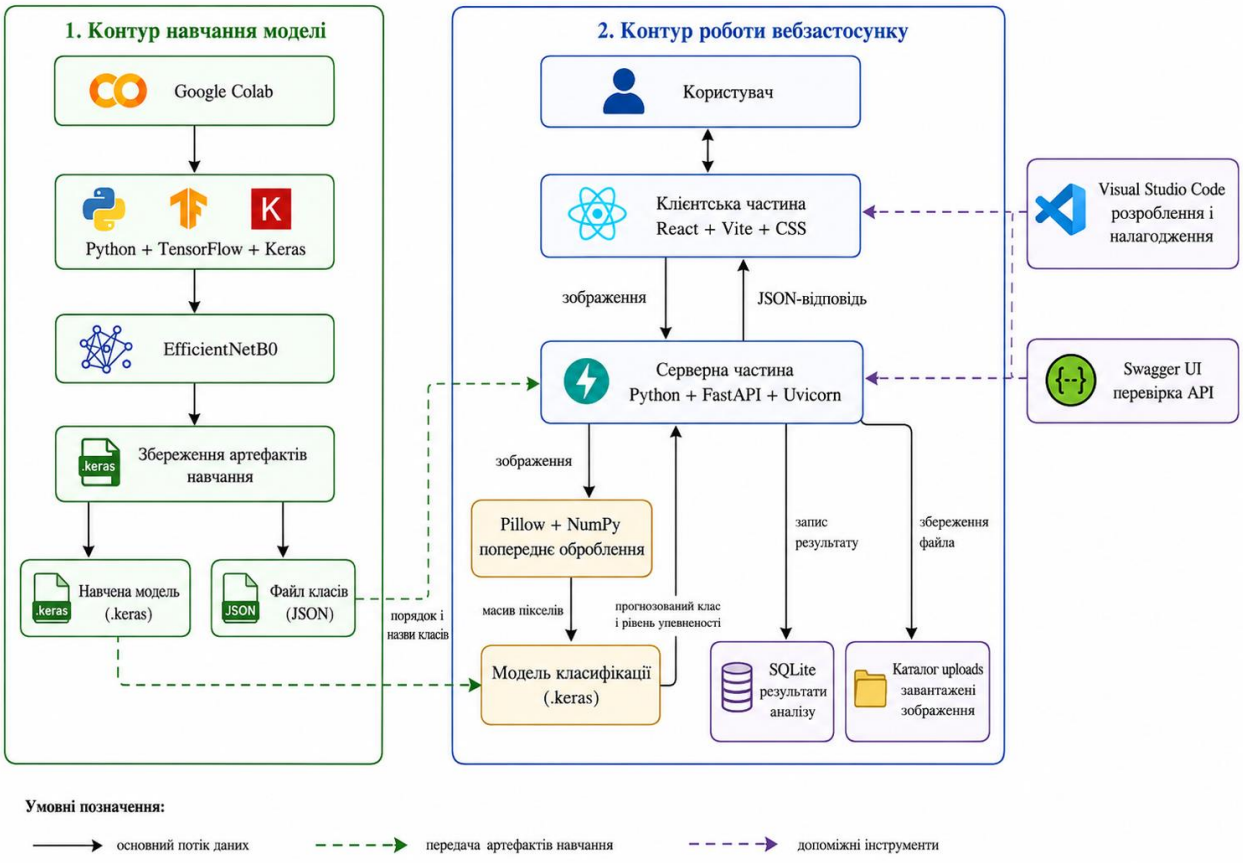

Рисунок 3.2 – Взаємозв'язок інструментальних засобів у структурі вебзастосунку

Як видно з рисунка 3.2, технологічний процес складається з двох взаємопов’язаних контурів. У першому виконується навчання моделі та формування артефактів у форматах .keras і JSON. У другому ці артефакти використовуються серверною частиною для оброблення користувацьких зображень, прогнозування, інтерпретації класів і збереження результатів. Клієнтська частина забезпечує взаємодію з користувачем, а Visual Studio Code і Swagger UI підтримують розроблення, налагодження та перевірку системи.

Обрані інструментальні засоби відповідають функціональним і технічним потребам вебзастосунку. Їх поєднання забезпечує інтеграцію навченої моделі із серверним API, підготовку вхідних зображень, збереження результатів аналізу та роботу інтерактивного користувацького інтерфейсу. Сформована технологічна основа використовується на наступних етапах реалізації програмного рішення.

3.2 Етапи розроблення вебзастосунку визначення ступеня стиглості фруктів за допомогою засобів комп’ютерного зору

Розроблення вебзастосунку здійснювалося послідовно – від перевірки найпростішого способу оцінювання стиглості за кольором до створення програмної системи, що поєднує модель комп’ютерного зору, серверний API, клієнтський інтерфейс, базу даних і файлове сховище.

Такий підхід дав змогу на кожному кроці перевіряти працездатність окремих рішень, виявляти їх обмеження та використовувати одержані результати під час формування наступної версії системи. Загальну послідовність етапів розроблення наведено на рисунку 3.3.

Перший прототип був орієнтований на визначення стиглості банана за його колірними характеристиками.

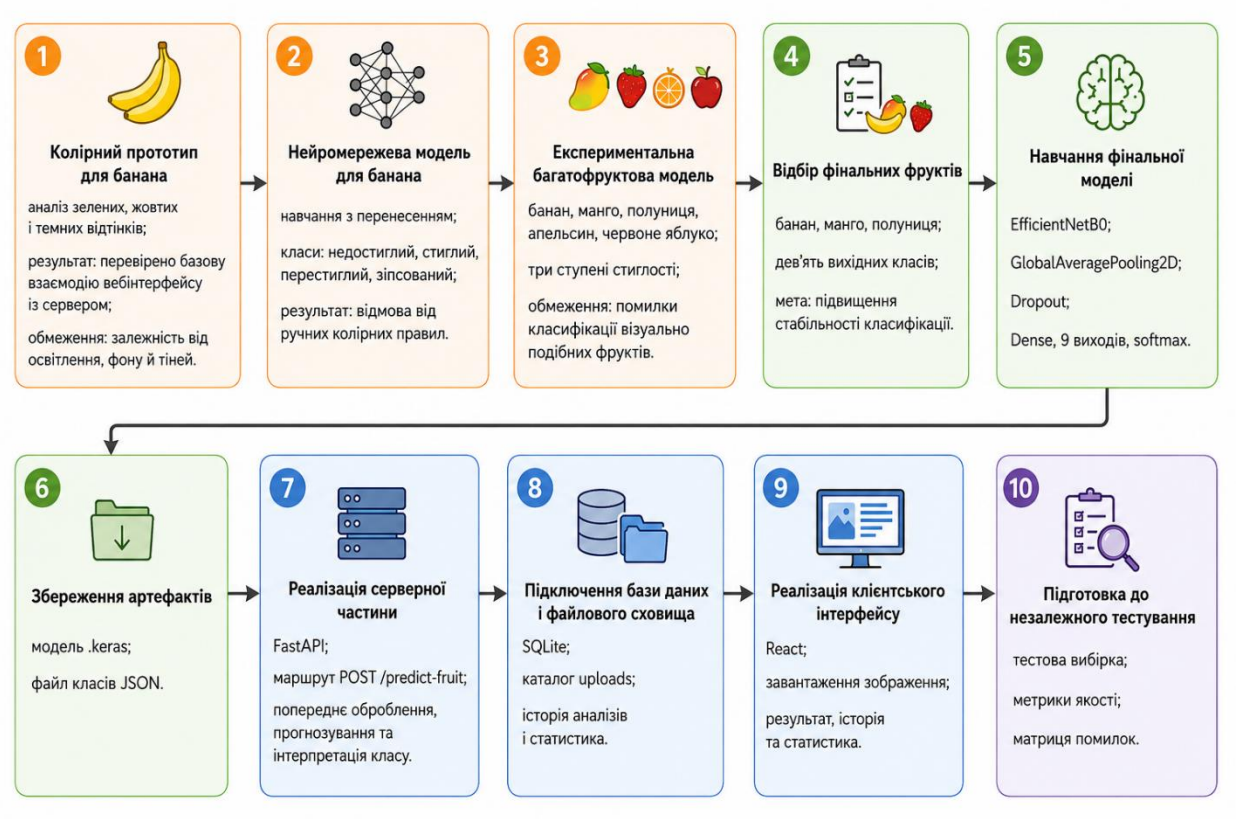

Рисунок 3.3 – Послідовність етапів розроблення вебзастосунку

Після завантаження зображення воно перетворювалося на числовий масив, а пікселі розподілялися за наперед заданими діапазонами зелених, жовтих і темних відтінків. Переважання зеленого кольору трактувалося як ознака недостиглого банана, жовтого – стиглого, а коричневих або темних ділянок – перестиглого.

Ця версія виконувала роль демонстраційного прототипу й дала змогу перевірити базовий сценарій взаємодії користувача із системою: завантаження файлу, передавання його на сервер, оброблення та повернення результату у вебінтерфейс. Водночас експерименти показали, що класифікація за жорстко заданими колірними межами істотно залежить від освітлення, балансу білого, тіней, фону та ракурсу. Однаковий плід міг отримувати різні результати залежно від умов знімання, а сторонні об’єкти відповідного кольору впливали на підсумкову оцінку. Через це підхід на основі ручних правил не забезпечував потрібної стійкості.

Виявлені обмеження стали підставою для переходу до моделі машинного навчання. Друга версія системи також працювала лише з бананом, проте стан плоду вже визначався не за вручну встановленими порогами, а за ознаками, які нейронна мережа виділяла з навчальних зображень. Для експерименту було сформовано категорії недостиглого, стиглого, перестиглого та зіпсованого банана. Навчання з перенесенням дало змогу використати ознаки, сформовані попередньо навченою згортковою мережею, й адаптувати їх до поставленої задачі.

Цей етап підтвердив, що нейромережева модель краще придатна до аналізу візуально складних зображень, ніж сукупність колірних правил. Водночас робота лише з одним видом фрукта не відповідала бажаній функціональності майбутнього вебзастосунку. Тому подальші експерименти були спрямовані на створення багатокласової моделі, здатної одночасно розпізнавати вид плоду та його стан.

До розширеного експериментального набору було включено банан, манго, полуницю, апельсин і червоне яблуко. Для кожного виду формувалися класи недостиглого, стиглого та перестиглого стану. Така організація передбачала, що вихідний клас містить відразу дві характеристики об’єкта. Наприклад, клас mango_ripe означає, що на зображенні розпізнано стигле манго.

Використання однієї моделі для двох взаємопов’язаних характеристик спростило загальну логіку системи, однак збільшення кількості класів виявило нову проблему – міжкласову подібність. Під час перевірки експериментальної моделі окремі зображення манго класифікувалися як апельсин або червоне яблуко. Помилки були пов’язані з подібною формою плодів, близькими відтінками шкірки, неоднорідними фонами та різними умовами формування наборів даних. Отриманий результат показав, що механічне збільшення кількості підтримуваних фруктів не завжди поліпшує прикладну цінність системи. За недостатньо збалансованих даних воно, навпаки, може знижувати стабільність прогнозування.

Для фінальної версії було залишено три види фруктів: банан, манго та полуницю. Вони мають достатньо виразні візуальні відмінності, а їхні зміни під час дозрівання помітні на звичайних цифрових зображеннях. Для кожного фрукта передбачено три ступені стиглості: недостиглий, стиглий і перестиглий. У результаті вихідний простір моделі складається з дев'яти класів. Це рішення стало компромісом між функціональною різноманітністю та необхідністю зберегти прийнятну відокремлюваність категорій. Склад класів фінальної моделі наведено в таблиці 3.1.

Таблиця 3.1 – Класи моделі для визначення ступеня стиглості фруктів

Технічний клас	**Фрукт**	**Ступінь стиглості**
banana_unripe	Банан	Недостиглий
banana_ripe	Банан	Стиглий
banana_overripe	Банан	Перестиглий
mango_unripe	Манго	Недостиглий
mango_ripe	Манго	Стиглий
mango_overripe	Манго	Перестиглий
strawberry_unripe	Полуниця	Недостиглий
strawberry_ripe	Полуниця	Стиглий
strawberry_overripe	Полуниця	Перестиглий

Назви каталогів набору даних, сформованого на основі відкритих наборів зображень бананів, манго та полуниці [37–39], відповідали технічним класам моделі. Зображення стиглих бананів, наприклад, розміщувалися в каталозі banana_ripe, а зображення недостиглої полуниці – у strawberry_unripe. Така структура дала змогу автоматично формувати мітки класів під час завантаження даних засобами TensorFlow та зменшила кількість ручних операцій.

Набір зображень було поділено на навчальну та валідаційну вибірки. Навчальні дані використовувалися для коригування параметрів нейронної мережі, тоді як валідаційні давали змогу контролювати її поведінку після кожної епохи та своєчасно виявляти ознаки перенавчання.

Для підвищення різноманітності прикладів застосовувалися горизонтальне віддзеркалення, невеликі повороти, масштабування та зміна контрастності. Аугментація використовувалася лише під час навчання й не входила до процесу прогнозування користувацьких зображень.

Усі вхідні файли перетворювалися до колірної моделі RGB і масштабувалися до розміру 224×224 пікселів. Окрема нормалізація перед EfficientNetB0 не виконувалася, оскільки масштабування значень пікселів реалізовано всередині цієї архітектури шаром Rescaling. Така організація попереднього оброблення відповідає структурі використаної моделі та запобігає повторному масштабуванню даних.

Базою фінального класифікатора стала EfficientNetB0 без початкового вихідного шару. На початку навчання ваги базової частини було зафіксовано, тому вона використовувалася як засіб виділення візуальних ознак. До неї додано шар глобального усереднення, Dropout для зниження ризику перенавчання та повнозв'язний шар із дев'ятьма виходами й функцією активації softmax. Така конфігурація адаптувала попередньо навчену архітектуру до сформованого набору класів. Фрагмент програмного коду побудови фінальної архітектури моделі наведено в лістингу 3.1.

Лістинг 3.1 Побудова фінальної архітектури моделі класифікації:

```
import tensorflow as tf
from tensorflow.keras import layers, models
base_model = tf.keras.applications.EfficientNetB0(
    input_shape=(224, 224, 3),
    include_top=False,
    weights="imagenet"
)
# На початковому етапі ваги базової моделі не змінюються
base_model.trainable = False
model = models.Sequential([
```

```
        layers.Input(shape=(224, 224, 3)),
        base_model,
        layers.GlobalAveragePooling2D(),
        layers.Dropout(0.3),
        layers.Dense(9, activation="softmax")
    ])
    model.compile(
        optimizer=tf.keras.optimizers.Adam(
            learning_rate=0.001
        ),
        loss="sparse_categorical_crossentropy",
        metrics=["accuracy"]
    )
```

У наведеному фрагменті EfficientNetB0 використовується без стандартного класифікаційного блока. Після базової частини мережі додано глобальне усереднення ознак, шар Dropout для зменшення ризику перенавчання та вихідний шар із дев’ятьма нейронами, що відповідають класам фінальної моделі. На початковому етапі навчання ваги EfficientNetB0 зафіксовано, тому оновлюються лише параметри доданих класифікаційних шарів.

Після завершення навчання нейронну мережу збережено у форматі .keras, а порядок класів – у файлі JSON. Ці два артефакти виконують різні функції. Файл .keras містить архітектуру та навчені параметри моделі, тоді як JSON-файл дає серверній частині змогу зіставити індекс вихідного нейрона з технічною назвою класу. Розділення цих даних полегшує заміну моделі й запобігає жорсткому закріпленню переліку класів у програмному коді.

Наступна частина роботи була пов’язана з інтеграцією моделі в серверний застосунок. Для приймання користувацьких зображень реалізовано маршрут POST /predict-fruit.

Після надходження запиту сервер перевіряє файл, формує для нього унікальне ім’я та зберігає в каталозі uploads. Далі зображення відкривається засобами Pillow, перетворюється до RGB, масштабується до 224×224 пікселів і перетворюється на масив NumPy з додаванням виміру пакета.

Підготовлений масив передається моделі. На основі отриманого вектору імовірностей сервер визначає індекс найбільш імовірного класу та відповідний рівень упевненості. Назва класу визначається за переліком, завантаженим із JSON-файлу, після чого з неї виділяються назва фрукта й ступінь стиглості. Поряд з основним прогнозом визначаються три класи з найвищими імовірностями, що дає змогу показати користувачеві розподіл альтернативних результатів. Ключові операції попереднього оброблення зображення, виконання прогнозування та інтерпретації класу наведено в лістингу 3.2.

Лістинг 3.2 Фрагмент серверного оброблення зображення та формування прогнозу:

```
image = Image.open(io.BytesIO(contents))
image = image.convert("RGB")
image = image.resize((224, 224))

image_array = np.asarray(image, dtype=np.float32)
image_array = np.expand_dims(image_array, axis=0)

probabilities = model.predict(
    image_array,
    verbose=0
)[0]

predicted_index = int(np.argmax(probabilities))
predicted_class = class_names[predicted_index]
```

```
confidence = float(probabilities[predicted_index])

fruit, ripeness = predicted_class.split("_", 1)
```

У наведеному фрагменті завантажене зображення перетворюється до колірної моделі RGB, масштабується до розміру 224×224 пікселів і подається моделі у вигляді числового масиву. За отриманим вектором імовірностей визначаються прогнозований клас і рівень упевненості. Технічна назва класу розділяється на назву фрукта та ступінь його стиглості.

Технічний клас моделі не подається користувачеві безпосередньо. Серверна частина перетворює його на зрозумілий результат. Наприклад, mango_ripe інтерпретується як фрукт «манго» зі ступенем стиглості «стиглий». До цих даних додаються рівень упевненості та рекомендація щодо споживання або подальшого зберігання плоду. Якщо модель формує прогноз з недостатнім рівнем упевненості, у відповіді додається попередження про необхідність використати інше зображення. Фрагмент функції формування рекомендацій подано в лістингу 3.3.

Лістинг 3.3 Формування рекомендації відповідно до визначеного ступеня стиглості:

```
FRUIT_LABELS = {
    "banana": "Банан",
    "mango": "Манго",
    "strawberry": "Полуниця"
}
def get_recommendation(fruit: str, ripeness: str) -> str:
    fruit_name = FRUIT_LABELS.get(fruit, fruit)
    recommendations = {
        "unripe": (
            f"{fruit_name} ще недостиглий. "
```

```
            "Його краще залишити достигати."
        ),
        "ripe": (
            f"{fruit_name} стиглий. "
            "Його можна вживати зараз."
        ),
        "overripe": (
            f"{fruit_name} перестиглий. "
            "Його бажано використати найближчим часом."
        )
    }
    return recommendations.get(
        ripeness,
        "Не вдалося сформувати рекомендацію."
    )
```

Функція зіставляє технічне позначення ступеня стиглості з текстовою рекомендацією для користувача. Назва фрукта попередньо перетворюється з технічного позначення на україномовний варіант. Окремо в серверній частині перевіряється рівень упевненості прогнозу. Якщо він нижчий за встановлений поріг, до відповіді додається повідомлення з рекомендацією завантажити чіткіше зображення.

Алгоритм оброблення одного користувацького зображення у серверній частині наведено на рисунку 3.4.

Блок-схема відображає не лише виклик моделі, а весь цикл роботи із запитом: отримання та перевірку файлу, його збереження, підготовку зображення, прогнозування, інтерпретацію класу, формування рекомендації, запис результату й повернення відповіді. Окрема гілка передбачає завершення оброблення з повідомленням про помилку, якщо формат файлу не підтримується або зображення неможливо відкрити.

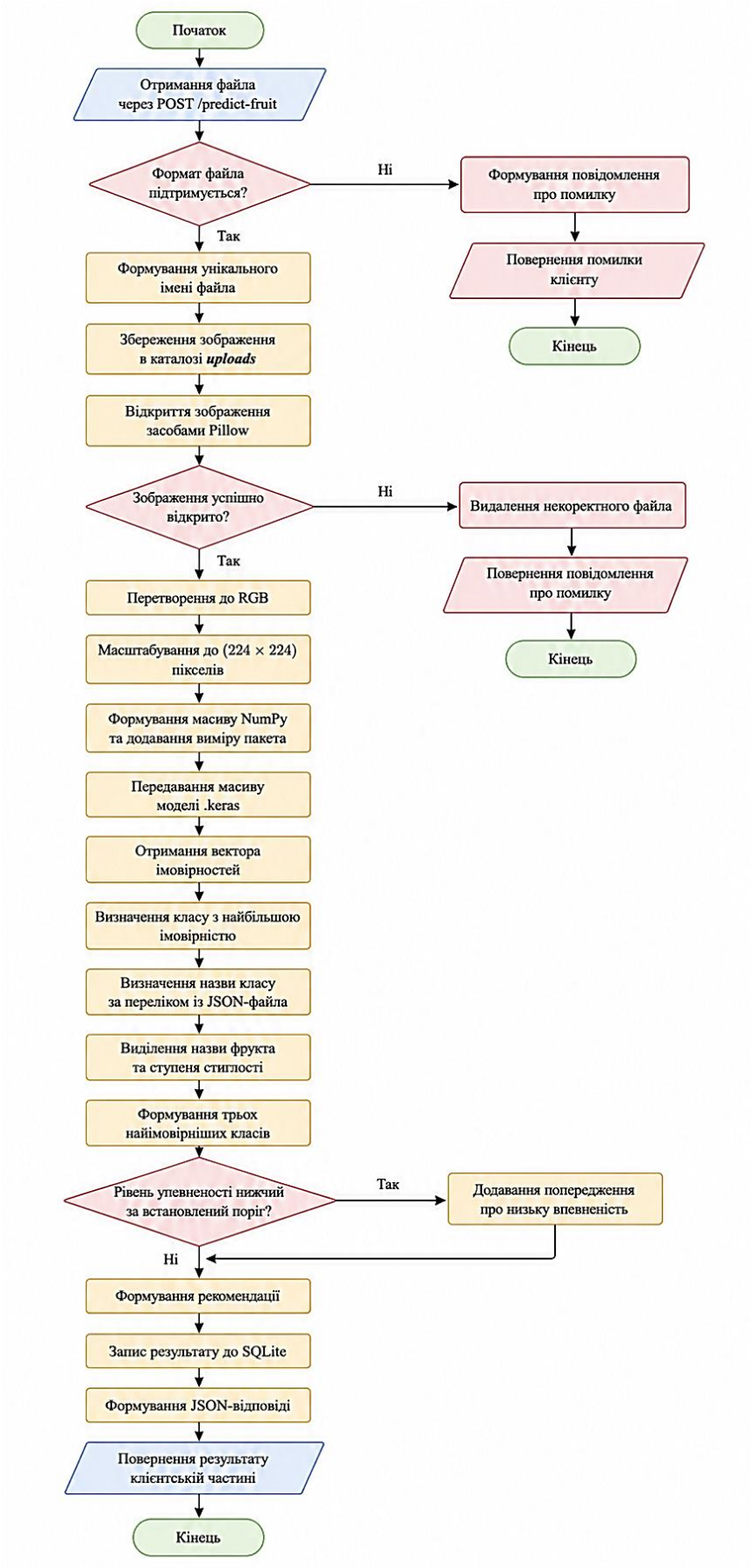

Рисунок 3.4 – Блок-схема алгоритму оброблення зображення в серверній частині вебзастосунку

Для накопичення результатів використано SQLite. Кожний успішний аналіз створює новий запис у таблиці predictions, де зберігаються шлях до файлу, визначений фрукт, ступінь стиглості, технічний клас, рівень упевненості, імовірності класів, рекомендація, версія моделі та дата виконання запиту. Така структура пов’язує прогноз із конкретним зображенням і підтримує подальший перегляд результатів.

Крім основного маршруту прогнозування, у серверній частині реалізовано запити для роботи з накопиченими даними:

- POST /predict-fruit – аналіз зображення та збереження результату;
- GET /history – отримання переліку попередніх аналізів;
- GET /history/{id} – перегляд повної інформації про вибраний запис;
- GET /stats – отримання узагальнених статистичних даних;
- DELETE /history – очищення історії разом із пов’язаними файлами.

Ці маршрути розширюють функціональність системи порівняно з простим сервісом класифікації. Вебзастосунок не лише виконує окремий прогноз, а й зберігає результати, підтримує їх повторний перегляд і формує статистичні показники.

Клієнтську частину розроблено на основі React. Головна сторінка містить область завантаження файлу, попередній перегляд вибраного зображення, кнопку запуску аналізу та блок відображення відповіді. Після завершення оброблення користувач отримує назву фрукта, визначений ступінь стиглості, рівень упевненості, рекомендацію та три класи з найвищими імовірностями. За низького рівня упевненості додатково відображається попередження.

Окремі компоненти клієнтської частини відповідають за історію та статистику. В історії відображаються мініатюра завантаженого зображення, фрукт, ступінь стиглості, рівень упевненості, номер запису й дата аналізу. Статистичний блок містить загальну кількість виконаних перевірок, середній рівень упевненості, кількість результатів для кожного фрукта та їх розподіл за ступенями стиглості.

Після успішного прогнозування клієнтська частина повторно надсилає запити до маршрутів історії та статистики. Завдяки цьому дані на сторінці оновлюються без її повного перезавантаження. Користувач одразу бачить новий запис і зміни в узагальнених показниках. Функціональну структуру реалізованого вебзастосунку подано на рисунку 3.5.

На рисунку 3.5 показано взаємодію клієнтських компонентів із маршрутами серверної частини, модулем класифікації, SQLite та файловим сховищем. На відміну від рисунка 3.2, де основну увагу приділено використаним технологіям, ця схема відображає функціональні модулі готової системи та дані, якими вони обмінюються.

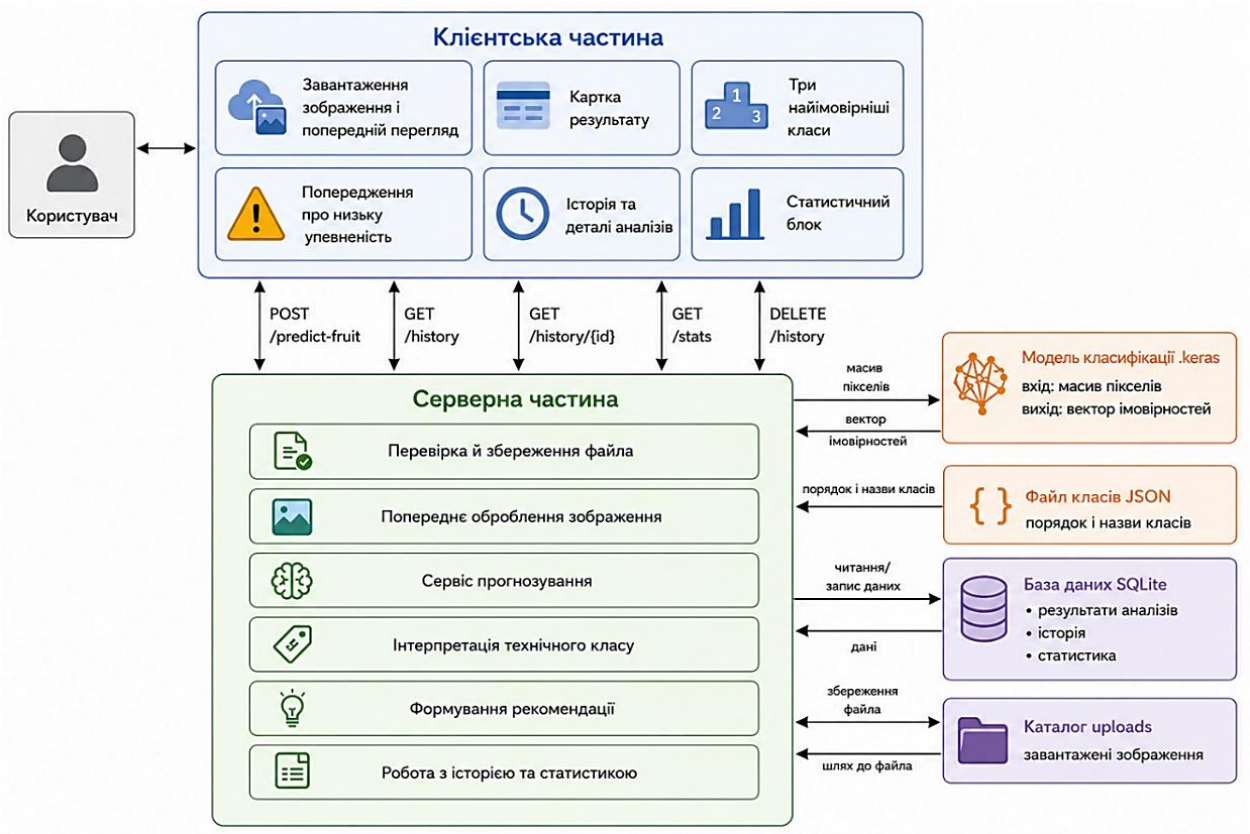

Рисунок 3.5 – Функціональна структура реалізованого вебзастосунку

У результаті послідовного вдосконалення було створено вебзастосунок, що підтримує дев’ять класів, виконує аналіз користувацьких зображень, зберігає результати й завантажені файли, відображає історію та формує статистику.

Перехід від колірного прототипу до нейромережевої моделі, а згодом – до цілісної клієнт-серверної системи визначався не лише розширенням функціональності, а й результатами перевірки попередніх рішень.

Для використання вебзастосунку користувач відкриває головну сторінку системи, обирає зображення фрукта та перевіряє його в області попереднього перегляду. Після натискання кнопки запуску аналізу файл передається до серверної частини, де виконується попереднє оброблення, прогнозування за допомогою навченої моделі та формування результату. У вебінтерфейсі відображаються визначений фрукт, ступінь його стиглості, рівень упевненості моделі, текстова рекомендація та три класи з найвищими імовірностями. Після завершення аналізу результат автоматично зберігається в історії, а статистичні показники оновлюються без перезавантаження сторінки.

3.3 Тестування розробленого вебзастосунку визначення ступеня стиглості фруктів та аналіз результатів

Тестування розробленого вебзастосунку виконувалося з метою перевірки якості роботи моделі комп’ютерного зору та коректності функціонування програмної системи загалом. У межах цього підрозділу оцінюється не лише здатність моделі правильно класифікувати зображення фруктів, а й працездатність вебзастосунку як цілісної системи, що включає клієнтську частину, серверний API, базу даних, файлове сховище, історію аналізів і статистичні показники.

Перевірка проводилася у трьох напрямах. Спочатку було проаналізовано результати навчання та валідації моделі, отримані під час підготовки класифікатора. Далі виконано незалежне тестування на нових зображеннях, які не використовувалися під час навчання та валідації.

Окремо проведено функціональне тестування вебзастосунку, спрямоване на перевірку основних сценаріїв взаємодії користувача із системою.

Першим етапом став аналіз динаміки навчання моделі. Навчання виконувалося протягом десяти епох. У процесі навчання спостерігалося поступове підвищення точності та зменшення значення функції втрат.

На першій епосі точність на навчальній вибірці становила 0,77, а на валідаційній – 0,93. На десятій епосі ці значення досягли відповідно 0,98 та 0,97. Одночасно функція втрат зменшилася з 0,75 до 0,08 на навчальній вибірці та з 0,33 до 0,09 на валідаційній вибірці. Динаміку навчання моделі доцільно розглядати за двома показниками: точністю та значенням функції втрат. Зміну точності на навчальній і валідаційній вибірках відображено на рисунку 3.6, а зміну функції втрат – на рисунку 3.7.

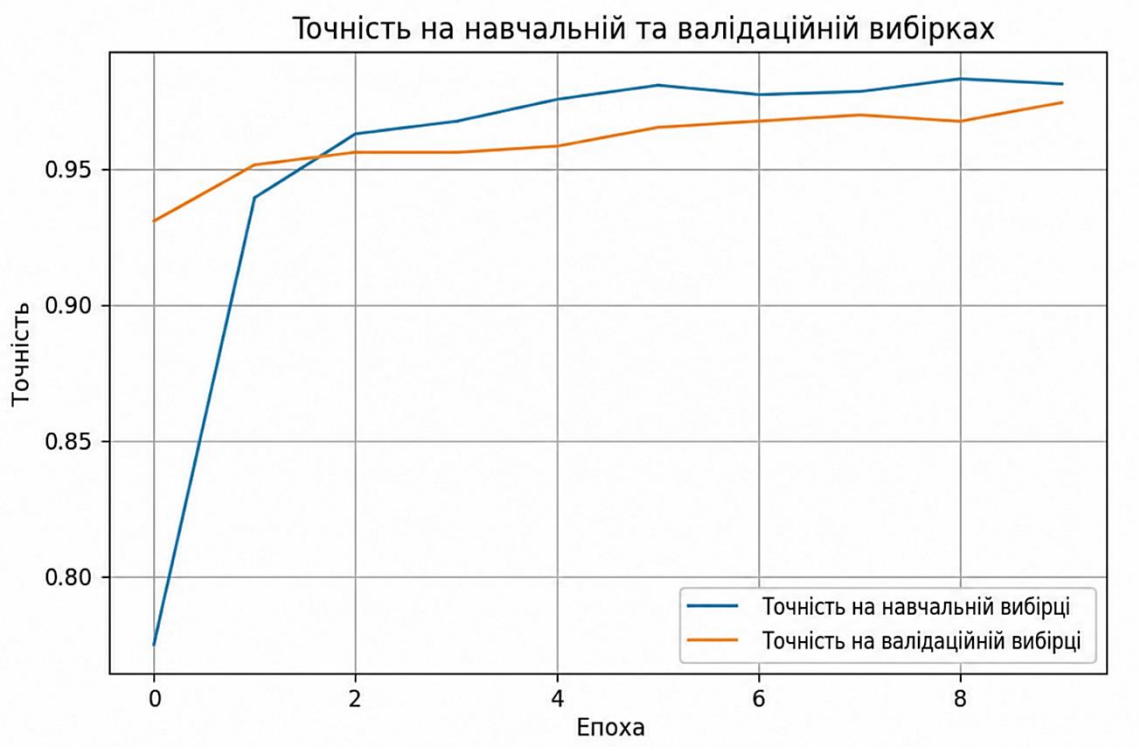

Рисунок 3.6 – Зміна точності моделі на навчальній та валідаційній вибірках

Аналіз графіків свідчить про стабільне збіження моделі в межах виконаного навчального циклу.

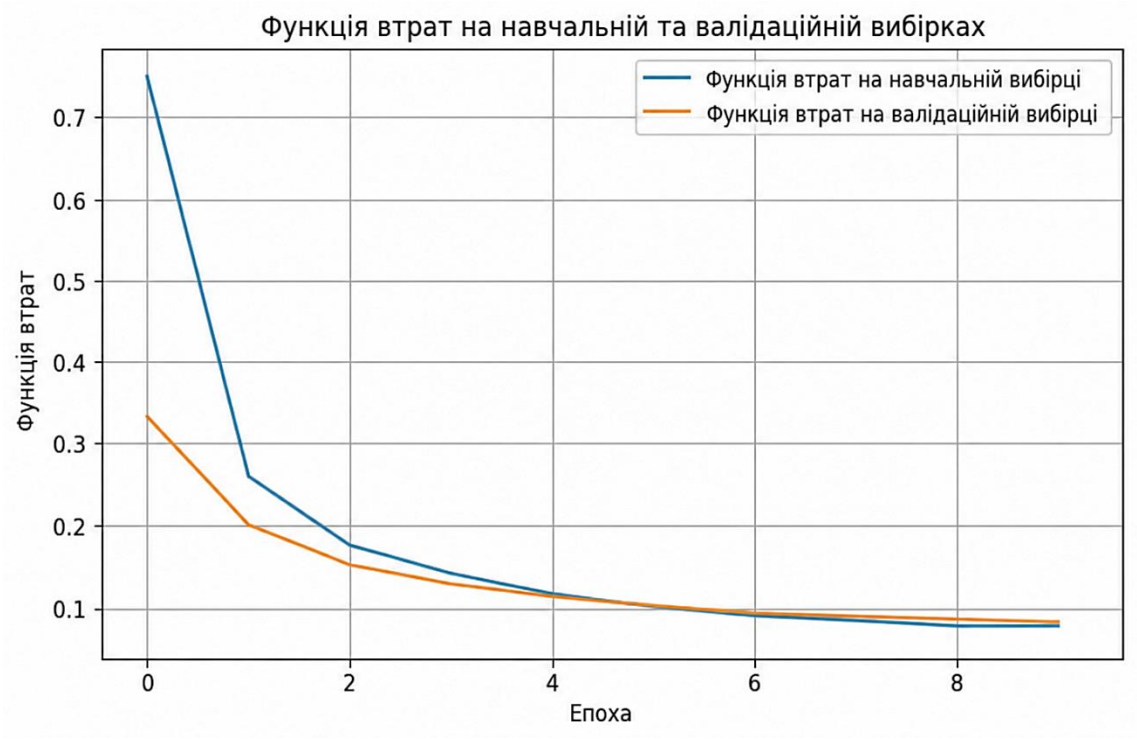

Рисунок 3.7 – Зміна функції втрат моделі на навчальній та валідаційній вибірках

Криві точності на навчальній і валідаційній вибірках залишалися близькими одна до одної, а значення функції втрат на обох вибірках поступово зменшувалися. Це дає підстави вважати, що модель не лише запам’ятовувала навчальні приклади, а й формувала узагальнені ознаки, придатні для класифікації зображень із валідаційної вибірки.

За результатами внутрішньої перевірки валідаційна точність моделі становила 97,47 %.

Водночас результати валідації не можна розглядати як остаточну оцінку якості моделі. Валідаційна вибірка була сформована з того самого загального набору даних, що й навчальна, тому вона характеризує поведінку моделі переважно в межах підготовленого середовища даних. Для об’єктивнішої перевірки здатності моделі працювати з новими зображеннями було проведено незалежне тестування на окремій тестовій вибірці. Додаткову інформацію про внутрішню роботу моделі надали матриця помилок і класифікаційний звіт.

Матриця помилок на рисунку 3.8 дає змогу оцінити, між якими класами модель найчастіше припускалася помилок під час внутрішньої перевірки.

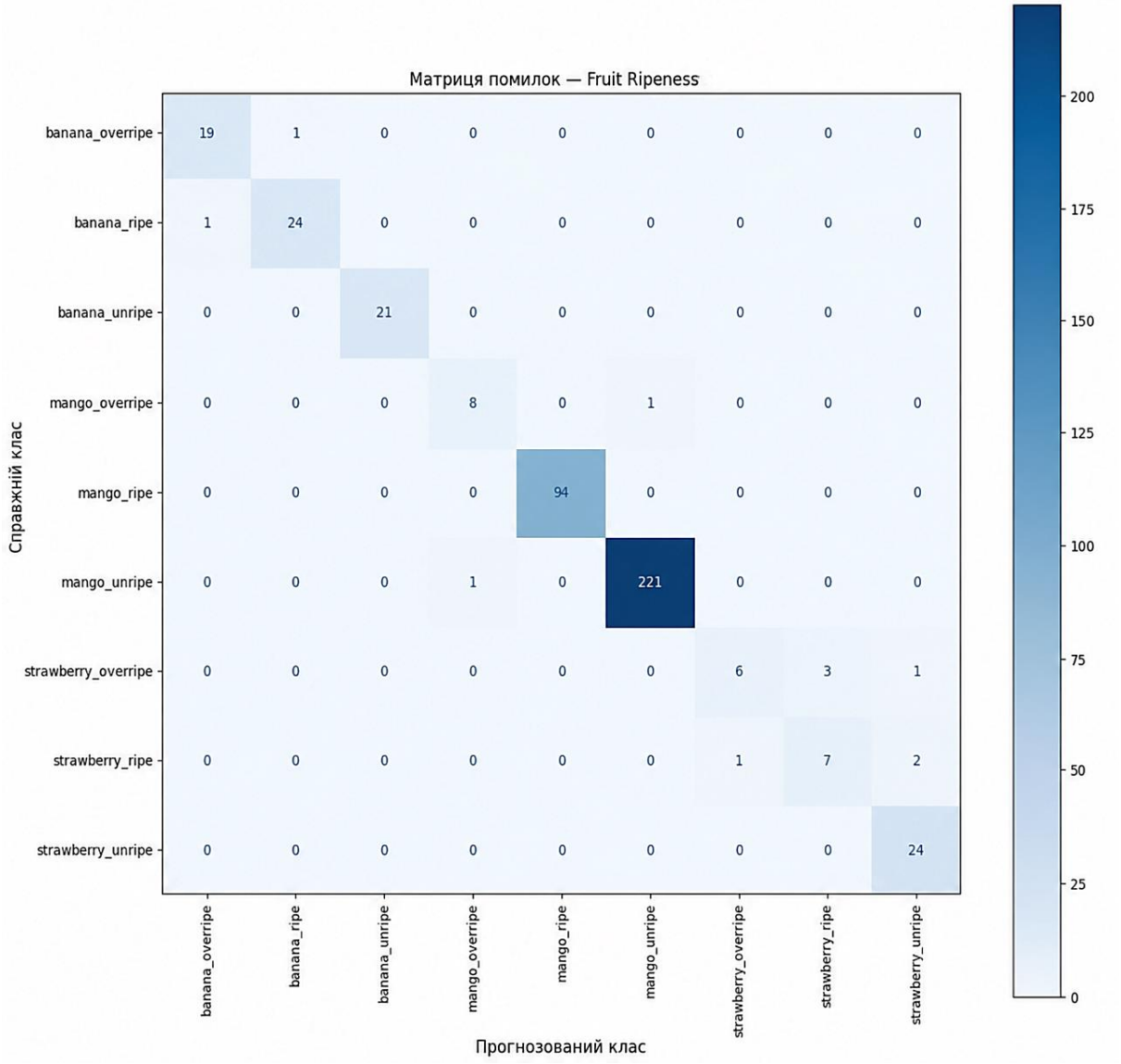

Рисунок 3.8 – Матриця помилок моделі за результатами внутрішньої перевірки

Для деталізації результатів внутрішньої перевірки за окремими класами використано таблицю 3.2.

За результатами внутрішньої перевірки модель показала найвищі результати для класів banana_unripe, mango_ripe та mango_unripe, для яких прецизійність, повнота та *F*1-міра досягли значення 1,00.

Таблиця 3.2 – Показники якості моделі за окремими класами

Клас	**Прецизійність**	**Повнота**	***F*1-міра**	**Кількість зображень**
banana_overripe	0,95	0,95	0,95	20
banana_ripe	0,96	0,96	0,96	25
banana_unripe	1,00	1,00	1,00	21
mango_overripe	0,89	0,89	0,89	9
mango_ripe	1,00	1,00	1,00	94
mango_unripe	1,00	1,00	1,00	222
strawberry_overripe	0,86	0,60	0,71	10
strawberry_ripe	0,70	0,70	0,70	10
strawberry_unripe	0,89	1,00	0,94	24
Загальна точність			0,97	435
Макроусереднене значення	0,92	0,90	0,90	435
Зважене середнє значення	0,97	0,97	0,97	435

Високі показники також отримано для більшості класів банана та манго. Водночас слабші результати спостерігалися для окремих класів полуниці, зокрема strawberry_overripe і strawberry_ripe. Для strawberry_overripe *F*1-міра становила 0,71, а для strawberry_ripe – 0,70. Це свідчить про складність відмежування суміжних ступенів стиглості полуниці, де візуальні відмінності між класами можуть бути менш виразними.

Загальна точність за результатами внутрішньої перевірки становила 0,97. Макроусереднена *F*1-міра дорівнювала 0,90, а зважене середнє значення *F*1-міри – 0,97. Така різниця між макроусередненим і зваженим показниками пояснюється нерівномірною кількістю зображень у класах. Зокрема, класи манго мали значно більшу кількість прикладів, тому вони сильніше впливали на зважене середнє значення.

Після аналізу внутрішніх результатів було проведено незалежне тестування фінальної моделі. Фінальна модель класифікує зображення за дев’ятьма класами: banana_unripe, banana_ripe, banana_overripe, mango_unripe, mango_ripe, mango_overripe, strawberry_unripe, strawberry_ripe та strawberry_overripe. Незалежна тестова вибірка містила 232 зображення, які не використовувалися під час навчання та валідації.

За результатами незалежного тестування значення функції втрат становило 0,47, а загальна точність класифікації – 0,84, тобто приблизно 84,05 %. Макроусереднена *F*1-міра дорівнювала 0,82, а зважене середнє значення *F*1-міри – 0,83. Отримані результати підтверджують працездатність моделі, однак демонструють нижчу якість порівняно з валідаційною вибіркою. Така різниця є очікуваною, оскільки незалежне тестування проводилося на нових зображеннях, які могли відрізнятися за освітленням, фоном, ракурсом, якістю знімання та візуальними особливостями плодів. Результати незалежного тестування за класами узагальнено в таблиці 3.3. Вони дають змогу порівняти прецизійність, повноту та *F*1-міру для кожного з дев’яти класів фінальної моделі.

Найкращі результати на незалежній тестовій вибірці модель показала для класів strawberry_unripe, banana_unripe та mango_unripe. Для strawberry_unripe прецизійність становила 0,93, повнота – 1,00, а *F*1-міра – 0,96. Для banana_unripe відповідні значення дорівнювали 0,97, 0,93 і 0,95. Для mango_unripe усі три показники становили 0,92. Це свідчить, що модель найнадійніше розпізнає недостиглі плоди, які мають виразніші візуальні ознаки та краще відмежовуються від інших ступенів стиглості.

Достатньо високі результати також отримано для класів banana_overripe та strawberry_overripe. Для banana_overripe прецизійність становила 1,00, повнота – 0,76, а *F*1-міра – 0,86. Для strawberry_overripe відповідні значення дорівнювали 0,75, 0,95 і 0,84. Це означає, що модель здатна розпізнавати перестиглі плоди, однак для окремих фруктів частина таких зображень може класифікуватися як стиглий стан.

Таблиця 3.3 – Показники якості моделі за результатами незалежного тестування

Клас	**Прецизійність**	**Повнота**	***F*1-міра**	**Кількість зображень**
banana_overripe	1,00	0,76	0,86	29
banana_ripe	0,71	1,00	0,83	25
banana_unripe	0,97	0,93	0,95	30
mango_overripe	0,86	0,33	0,48	18
mango_ripe	0,67	0,87	0,75	30
mango_unripe	0,92	0,92	0,92	24
strawberry_overripe	0,75	0,95	0,84	22
strawberry_ripe	0,95	0,67	0,78	27
strawberry_unripe	0,93	1,00	0,96	27
Загальна точність			0,84	232
Макроусереднене значення	0,86	0,83	0,82	232
Зважене середнє значення	0,86	0,84	0,83	232

Найскладнішим класом виявився mango_overripe. Для нього прецизійність становила 0,86, повнота – лише 0,33, а *F*1-міра – 0,48 при 18 тестових зображеннях. Низьке значення повноти означає, що значна частина фактично перестиглих манго була віднесена моделлю до інших класів, переважно до mango_ripe. Імовірною причиною є візуальна подібність стиглого та перестиглого манго, недостатня репрезентативність прикладів цього класу, а також нечіткість самої межі між станами ripe і overripe.

Певні труднощі спостерігалися і для класів strawberry_ripe, banana_ripe та mango_ripe. Для strawberry_ripe *F*1-міра становила 0,78, для banana_ripe – 0,83, а для mango_ripe – 0,75.

Ці результати показують, що найбільша кількість помилок виникає саме під час відмежування стиглого стану від суміжних категорій, особливо коли зовнішні ознаки переходу між рівнями стиглості виражені нечітко. Розподіл правильних і помилкових прогнозів на незалежній тестовій вибірці відображено на рисунку 3.9. За цією матрицею можна простежити, які класи модель розпізнає стабільно, а між якими ступенями стиглості виникає найбільше плутанини.

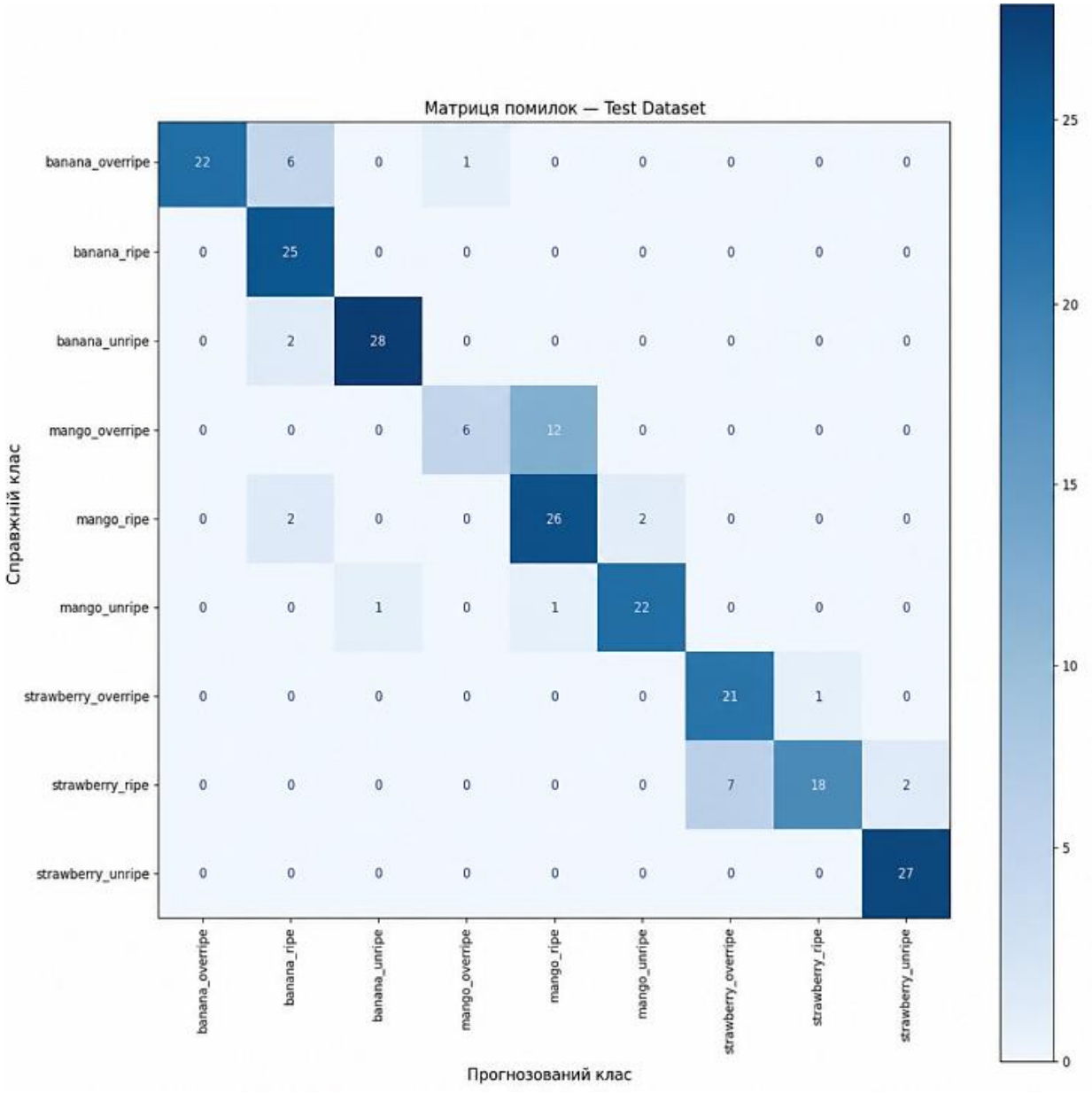

Рисунок 3.9 – Матриця помилок моделі за результатами незалежного тестування

Аналіз матриці помилок підтверджує, що більшість неправильних прогнозів виникає між суміжними ступенями стиглості одного й того самого фрукта. Наприклад, перестигле манго в окремих випадках класифікувалося як стигле манго, стигла полуниця – як перестигла, а перестиглий банан – як стиглий. Це свідчить, що модель переважно правильно визначає тип фрукта, але іноді помиляється в оцінюванні точного ступеня його стиглості. Отже, головна складність задачі полягає не стільки у розпізнаванні банана, манго чи полуниці, скільки у відмежуванні близьких стадій дозрівання.

Окремим напрямом перевірки було функціональне тестування вебзастосунку як програмної системи. Якщо тестування моделі оцінює правильність класифікації зображень, то функціональне тестування перевіряє коректність основних сценаріїв взаємодії користувача із системою.

У межах такої перевірки аналізувалися завантаження зображення через вебінтерфейс, попередній перегляд файлу, передавання його на сервер, виконання прогнозування, відображення результату, формування рекомендації, збереження запису до бази даних, оновлення історії та статистики, а також реакція системи на некоректні дії користувача. Перелік перевірених сценаріїв і фактичну реакцію системи узагальнено в таблиці 3.4.

У процесі функціонального тестування підтверджено, що при завантаженні коректного зображення серверна частина приймає файл, перевіряє його, зберігає в каталозі uploads, виконує попереднє оброблення та передає підготовлені дані до моделі. Після прогнозування користувачеві відображаються назва фрукта, ступінь стиглості, рівень упевненості моделі, текстова рекомендація та три класи з найвищими імовірностями.

Після кожного успішного аналізу вебзастосунок створює новий запис у базі даних. Перевірка показала, що записи коректно зберігаються, відображаються в історії аналізів і можуть бути переглянуті детальніше. У блоці історії користувач бачить мініатюру зображення, визначений фрукт, ступінь стиглості, рівень упевненості, номер запису та дату виконання аналізу.

Таблиця 3.4 – Результати функціонального тестування вебзастосунку

Сценарій тестування	**Очікуваний результат**	**Отриманий результат**
1	**2**	**3**
Завантаження коректного зображення фрукта	Файл успішно обирається та відображається в області попереднього перегляду	Зображення коректно завантажується та відображається у вебінтерфейсі
Надсилання зображення на сервер для аналізу	Файл передається на сервер без помилок	Зображення успішно передається до серверної частини
Виконання прогнозу для коректного зображення	Система повертає назву фрукта, ступінь стиглості та рівень упевненості	Результат класифікації відображається коректно
Формування рекомендації для користувача	Після прогнозу відображається текстова рекомендація	Рекомендація формується та виводиться у вебінтерфейсі
Збереження результату в базі даних	Після аналізу створюється новий запис у таблиці predictions	Запис успішно додається до бази даних
Відображення історії аналізів	Після виконання прогнозу новий запис з’являється в історії	Історія оновлюється без перезавантаження сторінки
Відображення статистики	Після нового аналізу статистичні показники оновлюються	Статистика коректно перераховується та відображається
Завантаження файлу, що не є зображенням	Система повинна відхилити файл і повідомити про помилку	Некоректний файл не приймається, користувач отримує повідомлення
Відсутність вибраного файлу перед запуском аналізу	Аналіз не виконується, користувач отримує попередження	Система не запускає оброблення без файлу

Продовження таблиці 3.4

1	2	3
Низький рівень упевненості моделі	Система повинна вивести додаткове попередження	Попередження про низький рівень упевненості відображається
Отримання деталей окремого аналізу	Відображаються повні дані вибраного запису	Детальна інформація за ідентифікатором запису повертається коректно
Очищення історії аналізів	Історія та пов’язані файли видаляються	Історія очищується, файли видаляються з uploads

Статистичний блок також оновлюється коректно: у ньому відображаються загальна кількість аналізів, середній рівень упевненості, кількість результатів за фруктами та розподіл за ступенями стиглості.

Під час перевірки некоректних сценаріїв було підтверджено, що система не запускає аналіз без вибраного файлу, відхиляє файли, які не є зображеннями, і відображає відповідні повідомлення користувачеві. Також перевірено очищення історії аналізів із видаленням пов’язаних зображень із каталогу uploads. Наявність таких перевірок є важливою для стабільної роботи вебзастосунку, оскільки користувач може виконувати не лише стандартні, а й помилкові або неповні дії.

Узагальнюючи результати тестування, можна зазначити, що розроблений вебзастосунок демонструє достатню функціональну цілісність і прийнятну якість класифікації для навчального програмного рішення. Внутрішні результати навчання та валідації показали високу точність моделі, однак незалежне тестування дало більш реалістичну оцінку її здатності працювати з новими зображеннями. Найкраще модель розпізнає недостиглі фрукти, тоді як основні труднощі виникають під час відмежування стиглого та перестиглого станів, особливо для манго.

Функціональне тестування підтвердило коректну роботу ключових сценаріїв: завантаження зображення, виконання прогнозування, збереження результатів, відображення історії, формування статистики та оброблення некоректних запитів.

Розроблене рішення можна вважати працездатним і придатним для подальшого вдосконалення. Основними напрямами поліпшення є розширення й збалансування набору даних, збільшення кількості прикладів для проблемних класів, зокрема mango_overripe, а також подальше уточнення меж між близькими ступенями стиглості.

3.4 Перспективи подальшої роботи

Розроблений вебзастосунок для визначення ступеня стиглості фруктів підтвердив працездатність обраного підходу: модель комп’ютерного зору була інтегрована із серверною частиною, клієнтським інтерфейсом, базою даних і файловим сховищем. Проведене тестування показало, що система здатна виконувати класифікацію зображень, зберігати результати аналізу, формувати історію перевірок і надавати користувачеві зрозумілу рекомендацію. Водночас отримані результати свідчать, що створене рішення варто розглядати як основу для подальшого розвитку, а не як остаточно завершений програмний продукт.

Одним із головних напрямів удосконалення є розширення та покращення набору даних. Незалежне тестування показало, що модель загалом забезпечує прийнятну якість класифікації, однак для окремих класів, насамперед близьких за зовнішніми ознаками ступенів стиглості, результати є нижчими. Тому подальша робота має бути спрямована на збільшення кількості зображень у проблемних класах, кращу збалансованість вибірки та розширення умов знімання.

До набору даних варто включати фотографії, зроблені за різного освітлення, на різному тлі, з різної відстані та під різними кутами. Це дозволить зменшити залежність моделі від умов отримання зображення й підвищити її здатність працювати з реальними користувацькими фото.

Подальше вдосконалення моделі класифікації також є важливим завданням. Найбільші труднощі під час тестування виникали на межі між стиглим і перестиглим станами, особливо для класу mango_overripe. Це свідчить про потребу в точнішому опрацюванні ознак, за якими модель розмежовує суміжні ступені стиглості. У майбутньому можна дослідити вплив тонкого налаштування базової частини EfficientNetB0, зміни гіперпараметрів навчання, інших стратегій аугментації та добору альтернативних архітектур нейронних мереж. При цьому важливо орієнтуватися не лише на підвищення загальної точності, а й на зменшення кількості помилок у класах, які виявилися найскладнішими.

Окремою перспективою є розширення предметної області вебзастосунку. У фінальній версії система працює з трьома фруктами: бананом, манго та полуницею. Такий вибір був обґрунтований потребою забезпечити стабільнішу класифікацію та уникнути надмірної плутанини між візуально подібними об’єктами. Надалі перелік підтримуваних фруктів можна поступово розширювати, додаючи яблука, апельсини, груші, персики, виноград або інші плоди. Однак таке розширення має супроводжуватися якісною підготовкою нових даних, оскільки збільшення кількості класів без належного балансування може погіршити роботу моделі.

Перспективним є також уточнення градації ступенів стиглості. Поточна система використовує три стани: недостиглий, стиглий і перестиглий. Такий поділ є зрозумілим для користувача й достатнім для навчального вебзастосунку. Проте для окремих фруктів у практичному застосуванні може бути корисним більш детальний поділ, наприклад виділення проміжних станів дозрівання.

Водночас збільшення кількості градацій ускладнює класифікацію та потребує точнішої розмітки зображень. Тому це питання потребує окремого експериментального оцінювання.

Подальшого розвитку потребує і користувацький інтерфейс. На поточному етапі вебзастосунок уже підтримує завантаження зображення, відображення результату, рекомендації, історії та статистики. У майбутньому інтерфейс можна зробити більш інформативним: додати розгорнуте відображення імовірностей за всіма класами, фільтрацію історії, порівняння результатів за різними фруктами, а також короткі підказки щодо правильного фотографування. Корисним було б і пояснення попереджень про низький рівень упевненості, щоб користувач краще розумів, у яких випадках результат потребує обережної інтерпретації.

Ще одним напрямом розвитку є розширення серверної та інформаційної частини системи. У подальшому можна додати облікові записи користувачів, що дозволить зберігати персональну історію аналізів і розмежовувати доступ до даних. Також доцільно передбачити можливість позначення користувачем некоректного результату. Такі приклади могли б накопичуватися окремо й використовуватися для подальшого поліпшення набору даних або донавчання моделі. У разі збільшення кількості користувачів і запитів варто також розглянути перехід від SQLite до серверної системи керування базами даних, придатної для багатокористувацького режиму.

Для наближення вебзастосунку до умов практичного використання важливо також оцінити його швидкодію та зручність роботи на різних пристроях. Оскільки користувачі найчастіше фотографують фрукти за допомогою смартфонів, перспективним є створення адаптивної мобільної версії інтерфейсу. Крім того, подальше розгортання системи у хмарному середовищі дало б змогу використовувати її без прив’язки до локального комп’ютера, а оптимізація серверної частини дозволила б скоротити час відповіді моделі.

У ширшій перспективі вебзастосунок може бути розвинений від системи визначення стиглості до інструмента комплексного аналізу плодово-овочевої продукції. До можливих напрямів належать виявлення дефектів поверхні, ознак псування, оцінювання придатності до споживання та формування рекомендацій щодо зберігання. Реалізація таких функцій потребуватиме нових наборів даних, розширення системи класів і, можливо, використання складніших багатозадачних моделей.

Отже, розроблений вебзастосунок демонструє можливість ефективного використання методів машинного навчання для автоматизованого визначення стиглості фруктів. Отримані результати підтверджують працездатність запропонованого рішення та доцільність його використання як основи для створення більш функціональних систем у цій предметній області.

ВИСНОВКИ

У рамках кваліфікаційної роботи розроблено та реалізовано вебзастосунок для визначення ступеня стиглості фруктів за цифровими зображеннями засобами комп’ютерного зору.

Під час виконання кваліфікаційної роботи:

− проведено аналіз сучасних вебзастосунків і демонстраційних рішень, пов’язаних із класифікацією фруктів, овочів та оцінюванням стану плодово-овочевої продукції. Установлено, що більшість наявних рішень орієнтована на загальне розпізнавання об’єктів або спрощене визначення стану за категоріями «свіжий / зіпсований», тоді як спеціалізовані вебзастосунки для багаторівневого визначення ступеня стиглості фруктів представлені обмежено;

− проаналізовано літературні джерела щодо використання комп’ютерного зору [40 – 47] та моделей машинного навчання [48 – 54] для визначення стиглості фруктів і суміжних задач. Аналіз показав, що найефективнішими для таких задач є підходи на основі глибокого навчання [55 – 61], однак їхня якість залежить від набору даних, умов знімання, кількості класів і візуальної подібності між суміжними станами стиглості;

− розроблено структуру бази даних вебзастосунку. Визначено основні інформаційні сутності: результати аналізу, фрукти, рівні стиглості, версії моделі та зображення. Для основної таблиці результатів аналізу сформовано структуру з 9 полів, що забезпечує збереження результатів перевірок, перегляд історії та подальше формування статистичних даних;

− виконано моделювання структури вебзастосунку. Для опису системи побудовано структурні схеми, блок-схеми алгоритмів і UML-діаграми, що відображають взаємодію користувача, клієнтської частини, серверної частини, моделі комп’ютерного зору, бази даних і файлової системи. Це дало змогу формалізувати логіку роботи системи перед її програмною реалізацією;

– для реалізації вебзастосунку використано TensorFlow, Keras, EfficientNetB0, FastAPI, React, SQLite, Pillow і NumPy. Фінальна модель класифікує зображення за 9 класами: для трьох фруктів – банана, манго та полуниці – передбачено три ступені стиглості: недостиглий, стиглий і перестиглий. Використання об'єднаних технічних класів дало змогу однією моделлю визначати і вид фрукта, і ступінь його стиглості.

Реалізований вебзастосунок забезпечує завантаження зображення фрукта, попередній перегляд файлу, передавання зображення на сервер, виконання прогнозування, відображення результату, формування рекомендації, збереження запису до бази даних, перегляд історії аналізів, отримання деталей окремого запису та відображення статистики.

Проведено тестування моделі та вебзастосунку. За результатами навчання протягом 10 епох точність на навчальній вибірці зросла з 0,77 до 0,98, а на валідаційній – з 0,93 до 0,97. Валідаційна точність моделі становила 97,47%. Незалежне тестування виконано на 232 зображеннях, які не використовувалися під час навчання та валідації. Загальна точність на незалежній тестовій вибірці становила 84,05%, макроусереднена $F1$-міра – 0,82, а зважене середнє значення $F1$-міри – 0,83.

Аналіз результатів показав, що модель найкраще розпізнає недостиглі фрукти. Найскладнішим класом виявився mango_overripe, для якого $F1$-міра становила 0,48. Основні помилки виникали між суміжними ступенями стиглості одного й того самого фрукта, що свідчить про складність візуального відмежування стиглого та перестиглого станів.

Функціональне тестування охопило 12 сценаріїв роботи вебзастосунку. Перевірено завантаження зображення, передавання файлу на сервер, виконання прогнозування, формування рекомендації, збереження результату, оновлення історії та статистики, перегляд деталей аналізу, очищення історії та реакцію системи на некоректні дії користувача. За результатами перевірки основні функції вебзастосунку працюють коректно.

Таким чином, мету роботи досягнуто шляхом створення програмного рішення, що поєднує модель класифікації зображень, серверну частину, клієнтський інтерфейс, базу даних і файлове сховище.

Практична цінність отриманих результатів полягає в автоматизації попереднього визначення ступеня стиглості фруктів за цифровими зображеннями. Розроблений вебзастосунок може бути використаний у навчальних цілях, для побутового оцінювання стану фруктів, а також як основа для подальшого створення систем контролю якості плодово-овочевої продукції.

Подальше вдосконалення роботи доцільно спрямувати на розширення та збалансування набору даних, збільшення кількості прикладів для проблемних класів, додавання нових фруктів, покращення інтерфейсу, адаптацію системи до мобільних пристроїв і підвищення стійкості моделі до різних умов освітлення, фону та якості зображень.

Результати роботи апробовано у вигляді тез доповіді під час XIX Міжнародної науково-практичної конференції «New paradigms of interdisciplinary research: theories, ideas and developments» [62].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Banana Ripeness Detection Using CNN. URL: https://github.com/Abhayparashar31/brd (дата звернення 18.05.2026).

2. capstone-dicoding-sbuah. URL: https://github.com/fadillarizalul/capstone-dicoding-sbuah (дата звернення 18.05.2026).

3. Fruit Quality Classification Project. URL: https://github.com/awojidetola/Fruit-Quality-Classification (дата звернення 18.05.2026).

4. Fruit & Vegetable Classification. URL: https://github.com/hill0106/Fruit-Vegetable-Classification (дата звернення 18.05.2026).

5. Vegetable Classification & Detection. URL: https://github.com/C-Logesh-Perumal-29/Vegetable_Classification_And_Detection (дата звернення 18.05.2026).

6. Baglat P., Hayat A., Mendonça F., Gupta A., Mostafa S.S., and Morgado-Dias F. (2023) Non-Destructive Banana Ripeness Detection Using Shallow and Deep Learning: A Systematic Review, *Sensors*, 23(2), 738.

7. Xiao F., Wang H., Xu Y., and Zhang R. (2023) Fruit Detection and Recognition Based on Deep Learning for Automatic Harvesting: An Overview and Review, *Agronomy*, 13(6), 1625.

8. Mim F.S., Galib S.M., Hasan M.F., and Jerin S.A. (2018) Automatic detection of mango ripening stages – An application of information technology to botany, *Scientia Horticulturae*, 237, pp. 156-163.

9. Martínez-Mora O., Capuñay-Uceda O., Caucha-Morales L., Sánchez-Ancajima R., Ramírez-Morales I., Córdova-Márquez S., and Cuenca-Mayorga F. (2025) Artificial Vision-Based Dual CNN Classification of Banana Ripeness and Quality Attributes Using RGB Images, *Processes*, 13(7), 1982.

10. Yang L., Cui B., Wu J., Xiao X., Luo Y., Peng Q., and Zhang Y. (2024) Automatic Detection of Banana Maturity—Application of Image Recognition in Agricultural Production, *Processes*, 12(4), 799.

11. Chai J.J.K., Xu J.-L., and O'Sullivan C. (2023) Real-Time Detection of Strawberry Ripeness Using Augmented Reality and Deep Learning, *Sensors*, 23(17), 7639.

12. Yue Y., Xu S., and Wu H. (2025) A Strawberry Ripeness Detection Method Based on Improved YOLOv8, *Applied Sciences*, 15(11), 6324.

13. Yang Z., Li Y., Han Q., Wang H., Li C., and Wu Z. (2025) A Method for Tomato Ripeness Recognition and Detection Based on an Improved YOLOv8 Model, *Horticulturae*, 11(1), 15.

14. Tapia-Mendez E., Cruz-Albarran I.A., Tovar-Arriaga S., and Morales-Hernandez L.A. (2023) Deep Learning-Based Method for Classification and Ripeness Assessment of Fruits and Vegetables, *Applied Sciences*, 13(22), 12504.

15. Zárate V. and Cáceres Hernández D. (2024) Simplified Deep Learning for Accessible Fruit Quality Assessment in Small Agricultural Operations, *Applied Sciences*, 14(18), 8243.

16. Zhao M., You Z., Chen H., Wang X., Ying Y., and Wang Y. (2024) Integrated Fruit Ripeness Assessment System Based on an Artificial Olfactory Sensor and Deep Learning, *Foods*, 13(5), 793.

17. Sofana Reka S., Bagelikar A., Venugopal P., Ravi V., and Devarajan H. (2024) Deep Learning-Based Classification of Rotten Fruits and Identification of Shelf Life, *Computers, Materials and Continua*, 78(1), pp. 781-794.

18. SQLite. Appropriate Uses For SQLite. URL: https://www.sqlite.org/whentouse.html (дата звернення 19.05.2026).

19. sqlite3 — DB-API 2.0 interface for SQLite databases. URL: https://docs.python.org/3/library/sqlite3.html (дата звернення 19.05.2026).

20. React Quick Start. URL: https://react.dev/learn (дата звернення 19.05.2026).

21. FastAPI. URL: https://fastapi.tiangolo.com/ (дата звернення 19.05.2026).

22. Using FormData Objects. URL: https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest_API/Using_FormData_Objects (дата звернення 19.05.2026).

23. Image Processing in OpenCV. URL: https://docs.opencv.org/4.x/d2/d96/tutorial_py_table_of_contents_imgproc.html (дата звернення 19.05.2026).

24. Unified Modeling Language, Version 2.5.1. URL: https://www.omg.org/spec/UML/2.5.1/About-UML (дата звернення 19.05.2026).

25. Python 3 Documentation. URL: https://docs.python.org/3/ (дата звернення 14.06.2026).

26. Transfer learning and fine-tuning. URL: https://www.tensorflow.org/guide/keras/transfer_learning (дата звернення 14.06.2026).

27. TensorFlow API Documentation. URL: https://www.tensorflow.org/api_docs (дата звернення 14.06.2026).

28. EfficientNet B0 to B7. URL: https://keras.io/api/applications/efficientnet/ (дата звернення 14.06.2026).

29. Image classification via fine-tuning with EfficientNet. URL: https://keras.io/examples/vision/image_classification_efficientnet_fine_tuning/ (дата звернення 14.06.2026).

30. Google Colab FAQ. URL: https://research.google.com/colaboratory/faq.html (дата звернення 14.06.2026).

31. Uvicorn Documentation. URL: https://www.uvicorn.org/ (дата звернення 14.06.2026).

32. Pillow Documentation. URL: https://pillow.readthedocs.io/ (дата звернення 14.06.2026).

33. numpy.ndarray. URL: https://numpy.org/doc/stable/reference/generated/numpy.ndarray.html (дата звернення 14.06.2026).

34. Getting Started. URL: https://vite.dev/guide/ (дата звернення 14.06.2026).

35. CSS: Cascading Style Sheets. URL: https://developer.mozilla.org/en-US/docs/Web/CSS (дата звернення 14.06.2026).

36. Swagger Documentation. URL: https://swagger.io/docs/ (дата звернення 14.06.2026).

37. Shahriar S. M. Banana Ripeness Classification Dataset. Kaggle. URL: https://www.kaggle.com/datasets/shahriar26s/banana-ripeness-classification-dataset (дата звернення 14.06.2026).

38. srabon00. Mango Ripening Stage Classification. Kaggle. URL: https://www.kaggle.com/datasets/srabon00/mango-ripening-stage-classification (дата звернення 14.06.2026).

39. El-Bendary N., Elhariri E. Strawberry-DS. Mendeley Data. URL: https://data.mendeley.com/datasets/z6dtfdpzz8/1 (дата звернення 14.06.2026).

40. Daradkeh Y.I., Gorokhovatskyi V., Tvoroshenko I., and Zeghid M. (2022) Tools for fast metric data search in structural methods for image classification, *IEEE Access*, 10, pp. 124738-124746.

41. Gorokhovatskyi V., Tvoroshenko I., Kobylin O., and Vlasenko N. (2023) Search for visual objects by request in the form of a cluster representation for the structural image description, *Advances in Electrical and Electronic Engineering*, 21(1), pp. 19-27.

42. Гороховатський В.О., Творошенко І.С., Чмутов Ю.В. (2022) Застосування систем ортогональних функцій для формування простору ознак у методах класифікації зображень, *Сучасні інформаційні системи*, 6(3), С. 5-12.

43. Гороховатський В., Передрій О., Творошенко І., Марков Т. (2023) Матриця відстаней для множини компонентів структурного опису як інструмент для створення класифікатора зображень, *Сучасні інформаційні системи*, 7(1), С. 5-13.

44. Pomazan V., Tvoroshenko I., and Gorokhovatskyi V. (2023) Development of an application for recognizing emotions using convolutional neural networks, *International Journal of Academic Information Systems Research*, 7(7), pp. 25-36.

45. 6. Pomazan V., Tvoroshenko I., and Gorokhovatskyi V. (2023) Handwritten character recognition models based on convolutional neural networks, *International Journal of Academic Engineering Research*, 7(9), pp. 64-72.

46. Tvoroshenko I., Gorokhovatskyi V., Kobylin O., and Tvoroshenko A. (2023) Application of deep learning methods for recognizing and classifying culinary dishes in images, *International Journal of Academic and Applied Research*, 7(9), pp. 57-70.

47. Daradkeh Y.I., Gorokhovatskyi V., Tvoroshenko I., Gadetska S., and Al-Dhaifallah M. (2023) Statistical data analysis models for determining the relevance of structural image descriptions, *IEEE Access*, vol. 11, pp. 126938-126949.

48. Gorokhovatskyi V., Tvoroshenko I., and Yakovleva O. (2024) Transforming image descriptions as a set of descriptors to construct classification features, *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 33, no. 1, pp. 113-125.

49. Gorokhovatskyi V., Tvoroshenko I., Yakovleva O., Hudáková M., and Gorokhovatskyi O. (2024) Application a committee of Kohonen neural networks to training of image classifier based on description of descriptors set, *IEEE Access*, vol. 12, pp. 73376-73385.

50. Tvoroshenko I., Pomazan V., Gorokhovatskyi V., and Kobylin O. (2023) Application of video data classification models using convolutional neural networks, *International Journal of Academic and Applied Research*, 7(11), pp. 134-145.

51. Gorokhovatskyi V., Tvoroshenko I. (2023) Identification of visual objects by the search request. *International scientific symposium «INTELLIGENT SOLUTIONS-S». Computational intelligence (results, problems and perspectives). Decision making theory: proceedings of the international symposium, September 28, 2023, Kyiv-Uzhorod, Ukraine*, pp. 25-27.

52. Daradkeh Y.I., Gorokhovatskyi V., Tvoroshenko I., and Zeghid M. (2024) Improving the effectiveness of image classification structural methods by compressing the description according to the information content criterion, *Computers, Materials & Continua*, vol. 80, no. 2, pp. 3085-3106.

53. Gorokhovatskyi V., Chmutov Y., Tvoroshenko I., and Kobylin O. (2025) Reducing computational costs by compressing the structural description in image classification methods, *Advanced Information Systems*, vol. 9, no. 1, pp. 5–12.

54. Gorokhovatskyi V., Tvoroshenko I., Yakovleva O., and Hudáková M. (2025) Image description compression in classification structural methods, *IEEE Access*, vol. 13, pp. 43631-43641.

55. Yakovleva O., Matúšová S., Tvoroshenko I., and Isaiev Y. (2024) Visitor counting based on video stream analysis from surveillance cameras to solve various business problems, *Verejná správa a regionálny rozvoj ekonómia, manažment a marketing*, XX(1), pp. 67-87.

56. Bohdan N., Tvoroshenko I., Gorokhovatskyi V., and Kobylin O. (2025) Development of a hybrid method to enhance context memory for a chatbot application based on large language models, *International Journal of Academic Information Systems Research*, 9(10), pp. 7-18.

57. Suprun A., Tvoroshenko I., Gorokhovatskyi V., and Yakovleva O. (2025) Development and research of a method for the combined use of large language models for text generation, *International Journal of Academic and Applied Research*, 9(10), pp. 249-263.

58. Gorokhovatskyi V., Tvoroshenko I., Yakovleva O., and Hudáková M. (2026) Feature space quantization and application of metrics in structural methods of image recognition, *IEEE Access*, vol. 14, pp. 17812-17824.

59. Гороховатський В.О., Творошенко І.С. (2025) Оцінювання значущості ознак для підвищення продуктивності структурних методів класифікації зображень. *Проблеми інформатики та моделювання (ПІМ-2025). Тези двадцять п'ятої міжнародної науково-технічної конференції (25 – 28 вересня 2025 року)*. *Харків: НТУ «ХПІ»*, С. 38-43.

60. Larin I., Tvoroshenko I., Gorokhovatskyi V., and Liubchenko V. (2025) Research and comparison of facial color normalization methods relative to an etalon image, *International Journal of Academic and Applied Research*, 9(11), pp. 36-47.

61. Гороховатський В., Творошенко І. (2026) Продуктивні моделі аналізу даних у методах розпізнавання зображень: монографія. Харків: ХНУРЕ. 153 с.

62. Petrova M. (2026) Development of a web application for determining the degree of fruit ripeness using computer vision. *New paradigms of interdisciplinary research: theories, ideas and developments: proceedings of the XIX International Scientific and Practical Conference*, *May 12-15, 2026, Prague, Czech Republic,* pp. 42-45.