Міністерство освіти і науки України
Чернівецький національний університет
імені Юрія Федьковича
Інститут фізико-технічних та комп'ютерних наук
(повна назва інституту/факультету)
Кафедра інформаційних технологій та комп'ютерної фізики
(повна назва кафедри)

Інформаційна система з розпізнавання стану фруктів

Дипломна робота
Рівень вищої освіти - перший (бакалаврський)

Виконав:
студент 4 курсу, групи 417
спеціальності
126 – інформаційні системи та технології
(шифр і назва спеціальності)
Палагута М. І. _____
(прізвище та ініціали)
Керівник: доктор технічних наук, доцент
Баловсяк Сергій Васильович _____
(науковий ступінь, вчене звання, прізвище, ім'я, по-батькові)
Рецензент_____
_____
(науковий ступінь, вчене звання, прізвище, ім'я, по-батькові)

До захисту допущено:
Протокол засідання кафедри № 19
від „ 17 ” червня 2021 р.
зав. кафедри _____ доц. Борча М.Д.

Чернівці – 2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРНІВЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ЮРІЯ ФЕДЬКОВИЧА

Інститут фізико-технічних та комп'ютерних наук
Кафедра інформаційних технологій та комп'ютерної фізики

ЗАТВЕРДЖУЮ
Завідувач кафедри
докт. фіз.-мат. наук, доц.
_____М. Д. Борча
”___” _____ 2021 р.

ІНФОРМАЦІЙНА СИСТЕМА З РОЗПІЗНАВАННЯ СТАНУ ФРУКТІВ

ЛИСТ ЗАТВЕРДЖЕННЯ

УЗГОДЖЕНО
Керівник роботи
докт. техн. наук, доцент
_____ С.В. Баловсяк
“___” _____ 2021 р.

Виконавець
студент 4-го курсу
_____ М. І. Палагута
“___” _____ 2021 р.

2021

ЗАВДАННЯ
НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Палагуті Михайлу Іллічу
(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Інформаційна система з розпізнавання стану фруктів

керівник роботи *Баловсяк Сергій Васильович, докт. техн. наук, доцент*,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджена наказом вищого навчального закладу від "__"_____202_ року №_____

2. Строк подання студентом проекту (роботи) *27 травня 2021 р.*

3. Вихідні дані до проекту (роботи) *Мета роботи – створення інформаційної системи для розпізнавання стану фруктів за їх цифровими зображеннями. Розпізнавання зображень виконати згортковими штучними нейронними мережами. Навчання нейронних мереж виконати методом зворотного розповсюдження помилки. Початкові зображення зчитувати з графічних файлів або з відеокамер. Дослідити вплив параметрів навчання нейронної мережі та розміру навчальної вибірки на точність розпізнавання об'єктів. Мова програмування – Python.*

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1) описати принцип роботи штучної нейронної мережі
2) описати методи навчання нейронної межі
3) розробити алгоритм розпізнавання зображень фруктів
4) створити програму для розпізнавання зображень фруктів
5) дослідити ефективність розробленої програми

5. Перелік графічного матеріалу
1) Структура підсистеми розпізнавання
2) Класи підсистеми надання послуг
3) Діаграма послідовностей класифікації зображення

Студент _____ *Палагута М.І.*
(підпис) (прізвище та ініціали)

Керівник проекту (роботи) _____ *Баловсяк С.В.*
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Кваліфікаційна робота виконана студентом групи 417 Палагутою Михайлом Іллічем. Тема «Інформаційна система з розпізнавання стану фруктів». Робота направлена на здобуття ступеня бакалавр за спеціальністю 126 «Інформаційні системи та технології».

Метою кваліфікаційної роботи є створення та розробка інформаційної системи, що виконує надання інформаційних послуг розпізнавання станів фруктів на зображеннях за платіжною підпискою. Об’єктом дослідження є розпізнавання станів фруктів на зображеннях з застосуванням нейронних мереж. В результаті виконання кваліфікаційної роботи було розроблено інформаційну систему, яка надає послуги аналізу зображень та облік платежів за ці послуги.

Бакалаврська робота містить: кількість сторінок – 73, таблиць – 4, рисунків – 58, додатків – 0, використаних джерел – 8.

Ключові слова: інформаційна система, он-лайн сервіс, машинне навчання, класифікація фруктів, аналіз зображень, система платіжних підписок, веб-платформа.

ABSTRACT

The qualification work has been performed by a student of the 417 group Palahuta Mykhailo. The topic is "Information system for fruit’s state recognition". The work is aimed at obtaining a bachelor's degree in specialty 126 "Information Systems and Technologies".

The purpose of the qualification work is to create and develop an information system that provides subscription based service of fruit’s state classification on images. As a result of the qualification work, an information system has been developed providing image analysis and subscription accounting functionalities.

Number of pages - 73, tables - 4, figures - 58, appendices - 0, sources - 8.

Keywords: information system, on-line service, machine learning, fruits classification, image analysis, paid subscriptions system, web platform.

ЗМІСТ

ВСТУП

Кваліфікаційна робота «Інформаційна система з розпізнавання стану фруктів» представляє собою інноваційну ідею створення виділеного, надійного та безпечного веб сервісу з аналізу фруктів на зображеннях. Клієнтам пропонується можливість швидко й дешево додати функціонал класифікації станів фруктів на зображеннях у свою систему, при цьому відповідаючи найвищим стандартам безпеки. Потенційних клієнтів у такого застосунку є багато, а саме: системи автоматизації супермаркетів, ферм, садів, розумні холодильники, мобільні застосунки для людей з вадами зору та інші.

Основною метою даної роботи, є створення розширюваної та надійної платформи, яку можна швидко і просто інтегрувати в системи замовників. Така платформа повинна містити дві підсистеми. Підсистема надання послуг, яка відповідає за облік клієнтів та стан їх підписок. Підсистема розпізнавання, яка власне опрацьовує зображення фруктів.

Задачі роботи можна поділити на дві групи. Перш за все, це задачі створення підсистеми розпізнавання:

- Збір, попередня обробка та оцінка дата сету станів фруктів.
- Побудова засобів попередньої обробки зображень.
- Інтеграція моделі виявлення фруктів Mask R-CNN.
- Навчання моделі.
- Оцінка точності моделі.

Також важливими є задачі щодо побудови веб-застосунку, який веде облік клієнтів та підписок. А саме:

- Проектування бази даних.
- Проектування архітектури системи.
- Розробка додатку.
- Інтеграція додатку з платіжним сервісом.
- Тестування застосунку.

Предметом дослідження є багатошарові згорткові та повнозв’язні нейронні мережі, технології проектування розподілених програмних систем в мережі Інтернет, методи захисту веб застосунків.

РОЗДІЛ 1
ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Системний аналіз об’єкту дослідження та предметної області

1.1.1 Аналіз мети функціонування системи

Предметною областю системи розпізнавання стану фруктів, є ряд підприємств, кожен з яких має свою унікальну проблему.

А. Супермаркет. Загальновідомою є складність відслідковування станів фруктів на полицях або складах супермаркетів, магазинів, тощо. Ця проблема включає в себе своєчасне переміщення фруктів, які вже достигли, з складів закладу, до полиць та як і викидання зіпсованих або перестиглих фруктів з полиць.

Б. Сад. Автоматизація збору урожаю, звітність та статистика спілості плодів – ці задачі вимагають точної та надійної системи класифікації, яка може працювати з великими зображеннями різної складності.

В. Розумний холодильник. Схожий з проблемою супермаркетів, розумний холодильник повинен вміти визначати зіпсовані продукти і повідомити власника про їх наявність.

Г. Мобільний застосунок для людей з вадами зору. Не менш важливою, є комп’ютеризована підтримка людей з різними фізичними вадами. Доцільною проблемою, будуть випадки вад зору, таких як дальтонізм, сліпота та інші. Можливість самостійно зорієнтуватись в станах фруктів, пропонованих такій людині на ринку, або ж магазині, просто сканувавши фрукт телефоном, може значно полегшити життя таким користувачам.

Запропонована інформаційна система може безпечно і надійно сприяти вирішенню всіх вищесказаних проблем одразу.

Варто зауважити, що діяльність даного застосунку регулюється регламентом Європейського Парламенту і Ради про захист персональних даних, GDPR, адже підсистема надання послуг обробляє та зберігає персональні дані працівників та конфіденційні дані компаній-клієнтів.

Моделювання області застосування в область даних відображена ER-діаграмою на рисунку 1.1.

Система аналізу зображень фруктів буде оперувати як SAAS сервер, який надає послуги іншим застосункам-клієнтам через мережу Інтернет. Функціонал надаватиметься через RESTful API інтерфейс, що дозволяє вільно вза’ємодіяти з клієнтами застосовуючи захищений мережевий протокол HTTPS. Оформлення та адміністрування підписки на сервіс, для клієнтів, виконується через підсистему надання послуг. Клієнти отримують перманентні вхідні дані, за допомогою яких, вони можуть отримати ключ API для їх застосунку. API ключі видаються окремо під кожен обліковий запис в межах компанії та можуть бути відключеними за запитом користувача або адміністратора.

Користувачі, які мають активну підписку на послуги, надсилають серверу картинку, фрукти на якій потрібно розпізнати. Транспортування картинки мережею інтернет здійснюється у форматі BLOB об’єктів, тобто великих масивів байтів. Також при запиті клієнт вказує категорію фрукта, який потрібно розпізнати. У відповідь користувач отримує значення з наперед визначеного переліку станів фруктів у форматі текстової стрічки та число, яке відповідає ймовірності того, що стан розпізнано правильно.

Модуль обліку підписок забезпечує автоматичне поновлення підписки для клієнтів до моменту ручного зупинення підписки клієнтом. Відмінити підписку на сервіс можна в будь-який момент.

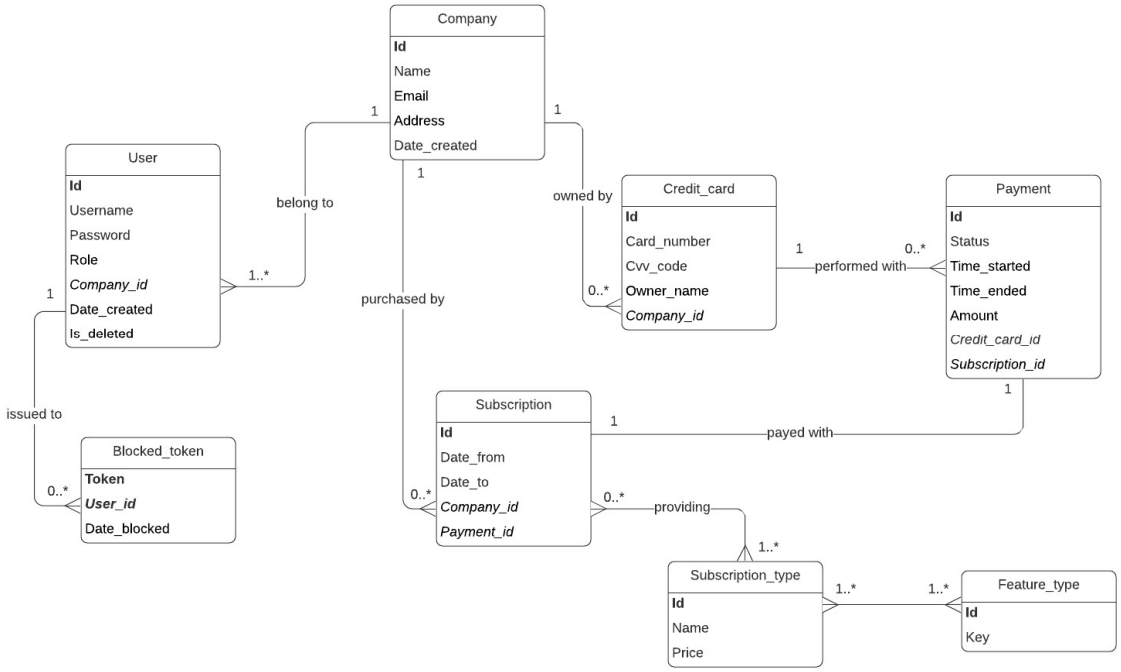


Рисунок 1.1 – Логічна структура бази даних

Декомпозиція системи відображена на рисунках 1.2-1.6, як набір IDEF0 діаграм, які демонструють підсистеми розпізнавання та надання послуг і їх складові.

А. Підсистема розпізнавання. Підсистема розпізнавання базується на наперед натренованих нейронних мережах, які визначають стан фруктів. Аналіз зображень складається з послідовності кроків, де одним з найважливіших кроків є сегментація. Сегментація, слугує попередньою обробкою зображення до його опрацювання нейронними мережами і знаходить на зображення всі підтримувані системою фрукти, та визначає їх точне положення на картинці. Результати такої оброки передаються до нейронних мереж. Завдяки комплексній попередній обробці, модуль підтримує одночасний аналіз як одного, так і багатьох фруктів. Основною функцією даної підсистеми є класифікація станів фруктів зображених на картинці. Тренування моделей використовуваних для розпізнавання виконується окремою програмою в середовищі Python. Декомпозиція підсистеми відображена на рисунку 1.5.

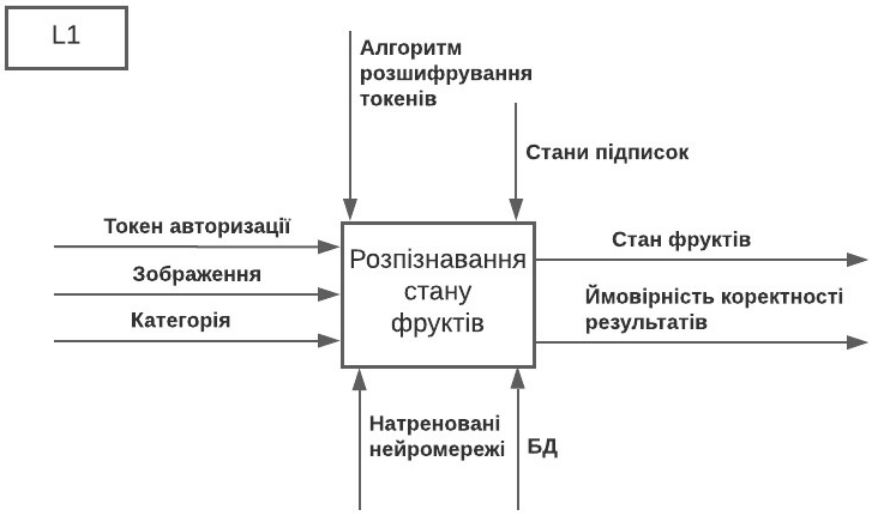


Рисунок 1.2 – IDEF0 контекстна діаграма застосунку

Рисунок 1.3 – IDEF0 діаграма підсистем застосунку

Б. Підсистема надання послуг. Надання послуг виконується через RESTful API сервер, який слугує центральною точкою прийняття рішень в системі. Підсистема розпізнавання звертається до даного сервера і сервер перевіряє чи має користувач, який надіслав запит, доступ до послуг і системи загалом. Використання такого архітектурного підходу підвищує безпечність та стійкість системи в публічному просторі мережі Інтернет.

Також підсистема надання послуг забезпечує облік підписок на послуги запропонованого сервісу, проведення планових платежів та адміністрацію облікових записів. Підсистема використовує базу даних для збереження облікових та платіжних даних користувачів та пропонує клієнтам веб інтерфейс для виконання всіх перелічених вище функцій. Декомпозиція даної підсистеми відображена на рисунку 1.6.

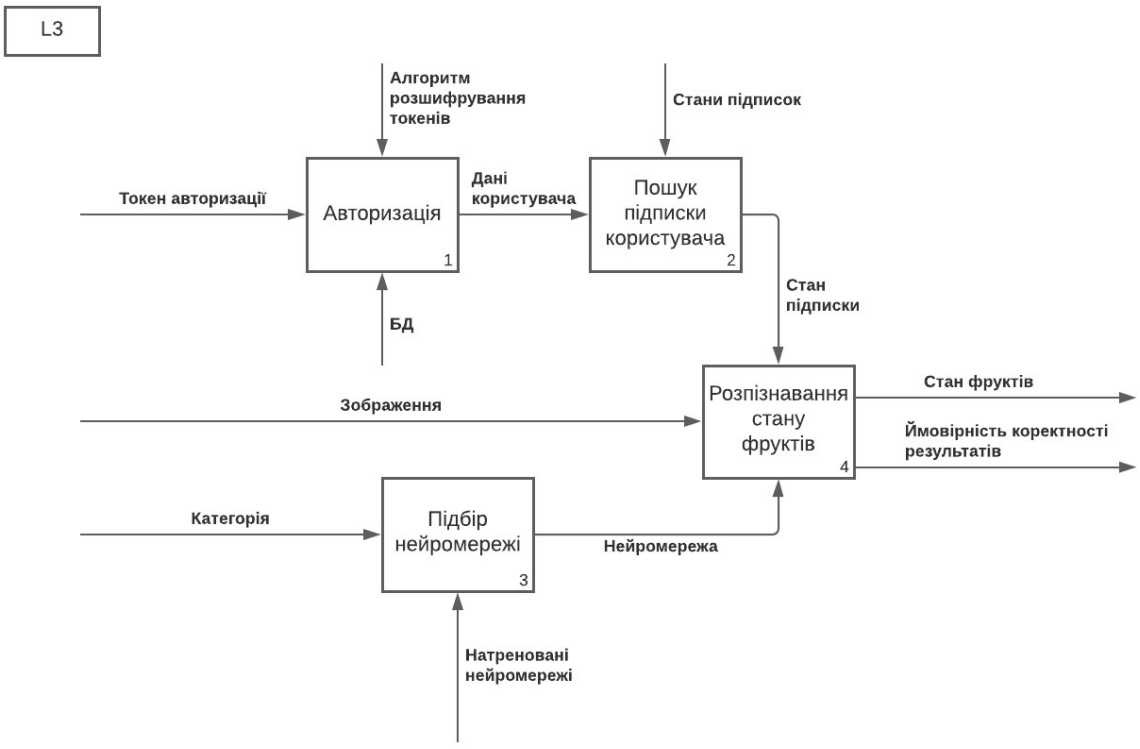


Рисунок 1.4 – IDEF0 діаграма первинної декомпозиції підсистем

Рисунок 1.5 – Декомпозиція підсистеми розпізнавання

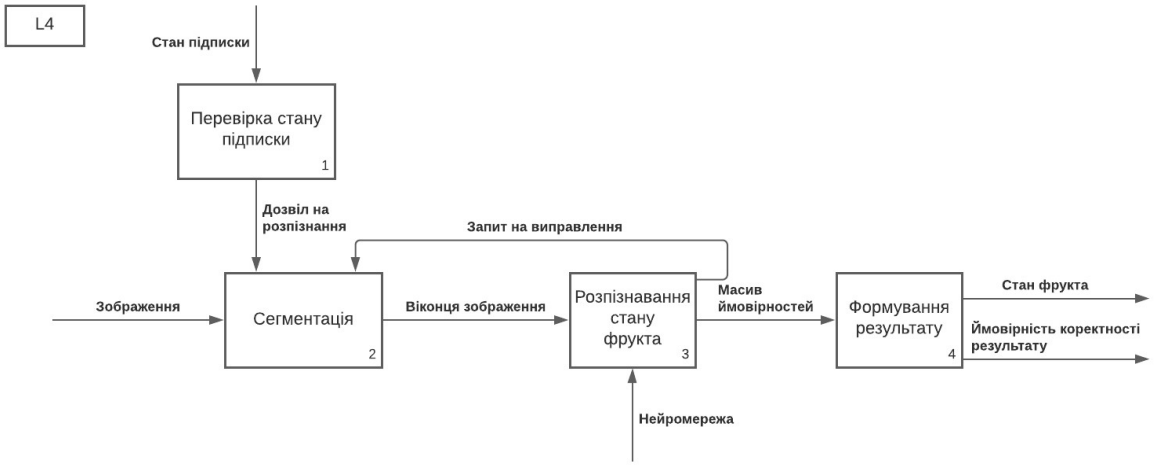


Рисунок 1.6 – Декомпозиція підсистеми надання послуг

1.1.2 Забезпечення якості

Перед кваліфікаційною роботою висуваються три основні критерії якості:

- Безпечність.
- Надійність.
- Розширюваність.

Задля забезпечення досягнення всіх критеріїв якості використовуватимуться такі методи тестування і аналізу, як:

- Автоматизоване модульне тестування коду.
- Ручне тестування в режимі чорного ящику.
- Ручне тестування в режимі сірого ящику.

- Статичний аналіз коду засобами інструментів SonarQube/SonarLint.

Не менш важливим способом досягнення бажаної якості є скорочення циклів відгуків зацікавлених осіб, що дає змогу вчасно і точно визначити неполадки системи чи її відхилення від основних вимог якості. Такий спосіб буде максимально залученим в роботу.

Якість підсистеми розпізнавання забезпечується шляхом збору додаткового тестового дата сету, результати розпізнавання якого формують точність роботи моделі у вигляді числа в межах від 0 до 1. Також, при навчанні, до вхідних даних додається гаусовий шум для збільшення стійкості моделі [1-3].

Варіанти побудови системи залежать від складу й досвіду команди розробників, наявності фреймворків та бібліотек, вимог безпеки і так далі.

З погляду структури команди, яка у випадку даної кваліфікаційної розробки складає одну людину, доцільним було б використання монолітної архітектури рішення, яка б складалась з одного багатофункціонального застосунку.

В той же час, засоби для побудови надійних додатків з машинного навчання та веб додатків є абсолютно різними. Машинне навчання найкраще розвинуте в середовищах Python, Matlab, R. Веб та серверна розробка найбільш надійно й ефективно здійснюється з застосуванням таких технологій як Java, Node.js, Golang.

В даній роботі, я надаю перевагу акцентуванню уваги на безпеці та надійності застосунку, що відповідає обранню розподіленої архітектури, адже вона сприяє значному покращенню супроводження та розширення окремих частин системи, фізично відділяє систему, яка займається персональними та платіжними даними користувачів від системи загалом, забезпечує надійність кожної з систем, особливо системи, яка працює з платіжним сервісом.

Недоліком розподіленого підходу побудови даного застосунку, як набору окремих програм, є збільшення загального обсягу етапу розробки через

потребу побудови засобів спілкування програм між собою через внутрішні мережі середовища розгортання.

1.2 Постановка задачі

1.2.1 Вимоги до системи

Мета розроблення інформаційної системи полягає у створенні розширюваної платформи з розпізнавання стану фруктів на зображеннях, а саме, створення програмної системи обліку клієнтів та їх платежів й розробка механізму створення та використання моделей розпізнавання фруктів та їх станів.

Призначення системи – надання інформаційних послуг аналізу зображень фруктів замовникам через мережу Інтернет на основі платних підписок. Така система є початковим прототипом повномасштабної реалізації ідеї SaaS служби з розпізнавання станів фруктів.

Місце застосування даного програмного продукту не є обмеженим й передбачає інтеграцію в системи замовників, які автоматизовують такі завдання як слідкування за станами фруктів на полицях магазинів, збір стиглих плодів в садах та багато інших.

Реалізація передбачає веб додаток для підсистеми надання послуг, який повинен обслуговувати запити через веб інтерфейс та RESTful точки доступу. Підсистема розпізнавання передбачає базові моделі розпізнавання станів, засоби та алгоритми додавання нових моделей.

Вимоги до процесу розробки передбачають постійну комунікацію з зацікавленими особами використовуючи методики та засоби методологій сімейства Agile, а саме Kanban. Дана розробка потребує повноцінного набору етапів проектування, розробки та тестування, які відображені на рисунках 1.7 - 1.8. Через природу даної кваліфікаційної роботи, етапи розділяються на два паралельні потоки, кожен з яких, передбачає етапи розробки відповідного типу систем, а саме розробка моделей машинного навчання та розробка розподілених інформаційних систем.

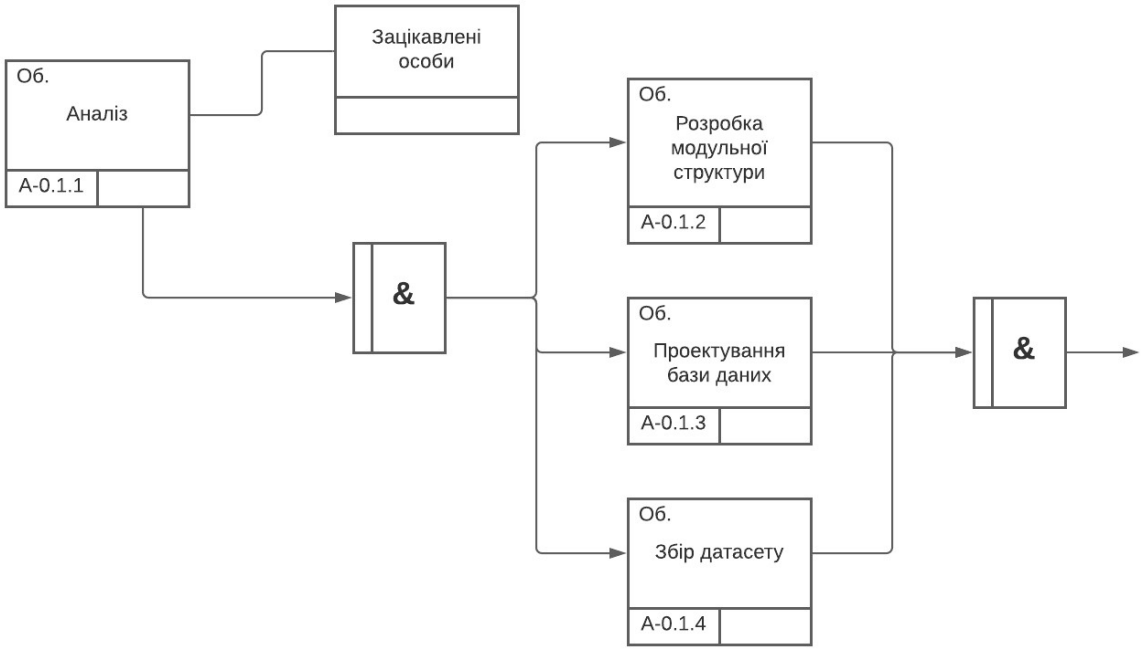


Рисунок 1.7 – IDEF3 контекстна діаграма процесу розробки

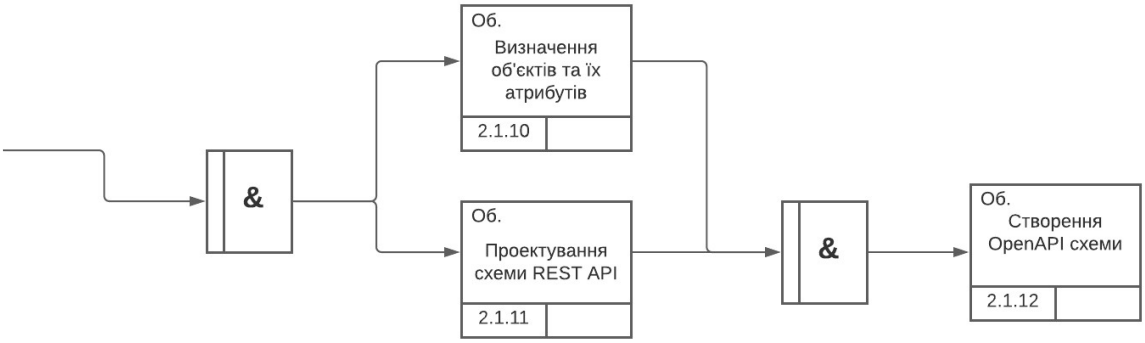


Рисунок 1.8 – IDEF3 діаграма декомпозиції етапу аналізу

Рисунок 1.9 – IDEF3 декомпозиція етапу розробки модульної структури

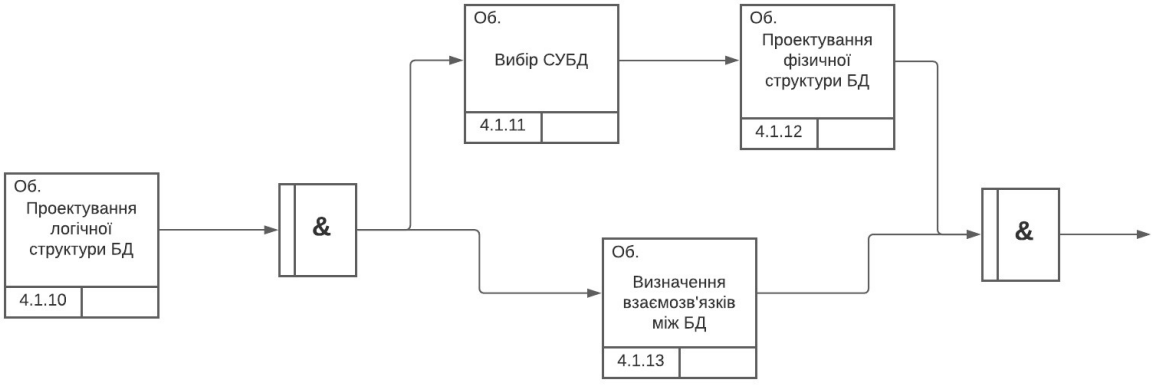


Рисунок 1.10 – IDEF3 діаграма проектування логічної структури БД

Рисунок 1.11 – IDEF3 діаграма декомпозиції етапу тренування моделей

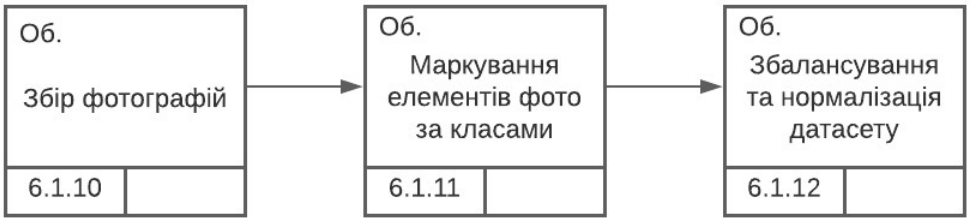


Рисунок 1.12 – IDEF3 діаграма - опис етапу збору дата сету

А. Функціональні вимоги до системи.

1. Первинне введення інформації в БД.
2. Зміна вмісту БД:
   - введення нових даних;
   - зміна існуючих даних;
   - архівація даних.
3. Забезпечення захисту й безпеки даних, зокрема:
   - розмежування прав доступу до ресурсів сервера;
   - контроль використання API ключів;
   - перевірка введеної інформації;
   - забезпечення цілісності БД.
4. Виведення знайденої інформації.
5. Проведення платежів.
6. Інтеграція веб-хук функцій з платіжною системою для оброки подій, що відбуваються з платежем або підпискою.
7. Розпізнавання стану фруктів на зображеннях.
8. Реєстрація, відновлення даних автентифікації.
9. Керування користувацькою структурою організації.
10. Відміна підписки.
11. Перегляд стану підписки.

База даних міститиме такі таблиці, як:

1. Користувач.
2. Підписка.

3. Платіжна карта.

4. API ключ.

5. Компанія.

Система буде забезпечувати розпізнавання стану фруктів шляхом опрацювання зображень нейронними мережами й облік підписок на дані послуги, а саме:

- забезпечувати розпізнання стану фруктів за зображенням;
- надавати й одержувати накопичену інформацію про стан підписки клієнта;
- забезпечувати обмеження доступу до ресурсів системи;
- забезпечувати авторизацію і дотримання оплати підписок;
- забезпечувати платежі та автоматичне подовження підписок;
- забезпечувати регулярну та захищену роботу сервісу.

Б. Нефункціональні вимоги

1. Платформа незалежність серверного застосунку. Веб додаток повинен підтримувати всі відомі ОС, такі як: MacOS X, Windows, Ubuntu (та інші основні дистрибутиви Linux).

2. Веб додаток повинен підтримувати середовище запуску Java програм – JVM, версії 8 та вище.

3. Надійність. Програмний продукт повинен бути надійним.

4. Відмовостійкість. Програма повинна містити механізми щодо уникнення всіх передбачуваних збоїв системи, партнерів та середовищ.

5. Безпечність. Система повинна містити механізми авторизації, автентифікації, та відповідати базовим стандартам безпеки, включаючи захист від XSS, SQL Injection, CSRF та інших поширених видів атак.

6. Узгодженість даних. Система повинна завжди забезпечувати повну узгодженість даних.

7. Розширюваність. Програмний продукт повинен дозволяти швидко й ефективно додавати новий функціонал та підтримку розпізнавання нових видів фруктів.

8. Браузери. Веб додаток повинен підтримувати всі сучасні браузери для пристроїв з мінімальною шириною монітора 800 пікселів.

9. Моделі розпізнавання повинні легко переноситися з однієї платформи на іншу, без втрат інформації про структуру мережі та її ваги.

10. Модель розпізнавання повинна знаходити та класифікувати фрукти в межах однієї секунди.

11. Енергетична та ресурсна ефективність. Енерговитрати та вимоги до технічних ресурсів повинні бути в межах норми відповідно до виконуваних задач.

В. Вимоги до системи обліку користувачів

Інформаційна система запропонована в цій кваліфікаційній роботі містить специфічні вимоги щодо структури користувацької системи через те, що платформа націлена на роботу з компаніями, як користувачами.

Програмний продукт виділяє два типи користувачів, серед яких:

- Root користувач. Користувач такого типу може бути тільки один для однієї компанії. Цей обліковий запис створюється автоматично при реєстрації компанії в систему та має захищений доступ. Можливості цього користувача – керування оплатами та підписками компанії та адміністрування будь-якої кількісті account користувачів для компанії;
- Account користувач. Цей тип користувача надається працівнику або застосунку клієнта. Він створюється і адмініструється root користувачем компанії, та надає можливості генерації API ключів доступу до системи розпізнавання. Кожен ключ доступу має жорстку прив’язку до account користувача, який його згенерував.

На рисунку 1.13 відображено користувацьку структуру для однієї компанії-клієнта програмного продукту.

Рисунок 1.13 – Користувацька структура одного клієнта

1.2.2 Очікувані ефекти від впровадження

Впровадження даного SaaS он-лайн сервісу повинно спростити й пришвидшити інтеграцію функцій розпізнавання станів фруктів на зображеннях в системи замовників, при цьому надавши гарантії ефективності і надійності платформи. Це сприятиме розширенню національного ринку надання спеціальних інформаційних послуг існуючим програмним продуктам, створить можливості розвитку автоматизованих систем керування фруктами у різних ситуаціях.

При цьому, власники та інвестори даної розробки отримають стабільний та передбачуваний дохід з підписок й договірні та кооперацій зв’язки з компаніями, які потребують даних послуг.

Висновки до розділу 1

Проаналізувавши потреби й можливості ринку автоматизації роботи з фруктами, очевидною стає актуальність й інноваційність ідеї створення централізованої, ефективної системи розпізнавання станів фруктів на зображеннях у формі SaaS сервіса-платформи.

Розглянувши архітектурні варіанти побудови такої системи, з можливими способами її реалізації, я прийняв рішення слідувати шаблонам розподіленої архітектури інформаційних систем для даної розробки аби забезпечити максимально зручне технічне супроводження проекту, розширюваність, надійність та безпеку. При цьому передбачається два застосунки, веб додаток з застосуванням Java технологій та підсистема розпізнавання в середовищі Python.

Для побудови нейронних мереж, прийнято рішення використовувати послідовність з декількох кроків, включаючи сегментацію та власне класифікацію зображення, використовуючи набір тренований мереж, одну для кожного типу фрукта. Таки чином спроектована підсистема розпізнавання за визначенням має кращі властивості розширюваності.

Розробка даного програмного продукту стане прототипом та основою для подальшого розширення можливостей (зокрема кількості фруктів які підтримуються) й удосконалення методів розпізнавання, класифікації станів фруктів на зображеннях. Не менш важливою, буде роль в тестуванні даної бізнес-моделі сервісу, з подальшим розвиток шляхів вза’ємодії, включаючи безкоштовні плани підписки, річні закупівлі і так далі.

Як результат, дана платформа має шанс покращити фермерство, садівництво, магазинний менеджмент, та життя людей з вадами зору на глобальному рівні.

РОЗДІЛ 2
ПОБУДОВА СИСТЕМИ

2.1 Моделювання системи

2.1.1 Вхідні дані

Вхідні дані системи поділяються на вхідні дані застосунку клієнта та вхідні дані власне клієнта. Вхідні дані власне клієнта – це вхідні дані додатку підсистеми надання послуг. Такими даними є:

- Інформація про компанію (адреса, назва і т.д.);
- Інформація окремого працівника компанії (найменння користувача, адреса електронний скриньки для передачі автентифікаційних даних і т.д.);
- Платіжні дані компанії (номер картки, CVV2 код, рік і місяць закінчення дії картки).

Інший тип вхідних даних – вхідні дані підсистеми розпізнавання. Для виконання розпізнавання користувач надає зображення, яке потрібно проаналізувати, у формі масиву байтів та API ключ, який передається HTTP заголовком “Authorization” для підтвердження наявності доступу до послуг.

Вхідні дані підсистеми надання послуг візуалізовано набором діаграм потоків даних різного рівня на рисунках 2.1-2.4.

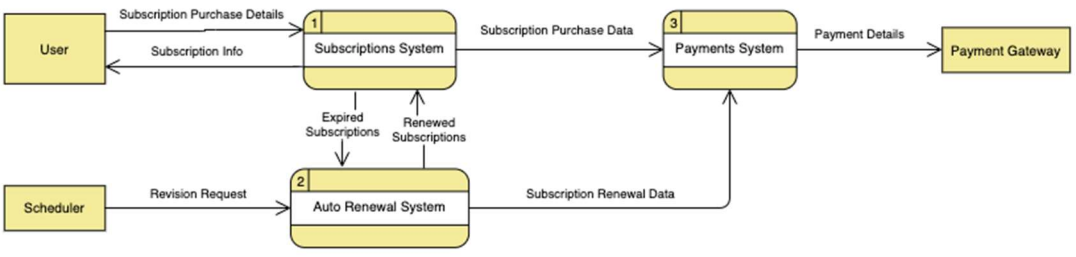


Рисунок 2.1 - Контекстна діаграма потоків даних підсистеми надання послуг

Ще однією зовнішньою сутністю, яка вносить дані в систему є платіжний сервіс Stripe. Ця сутність повідомляє систему надання послуг про події які відбуваються з платежами або підписками користувачів. Інтеграція передбачає обробку таких подія, як:

1) “Checkout completed”. Ця подія доводить до відома додаток про завершення проходження етапу заповнення і перевірки платіжних даних користувачем, передаючи його ідентифікатор та номер сформованої для нього підписки;
2) “Payment Success” – це подія, яка сповіщує підсистему про успішність первинної оплати за підписку, або оплати щодо її щомісячного поновлення. При отриманні такої події, додаток надасть доступ до послуг клієнтові, або поновить його;
3) “Payment Failure”. Дана подія надходить у випадку провалу первинної або поновлювальної оплати за підписку. Ця подія передбачає призупинення надання послуг;
4) “Subscription Canceled”. Відповідає відміні підписки клієнта через неможливість платіжної системи провести поновлювальний щомісячний платіж. Stripe виконає 5 спроб поновити підписку перед тим, як визнати поновлення неможливим;

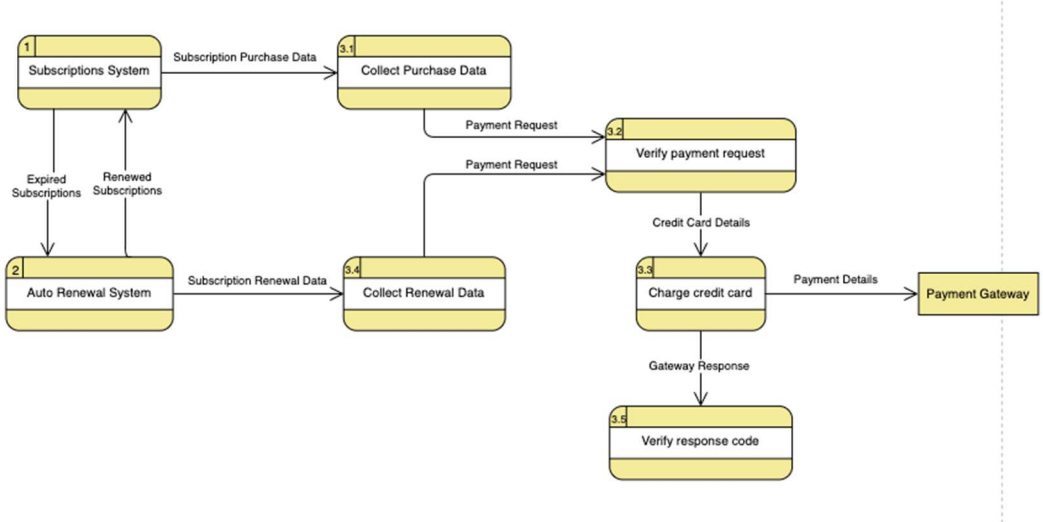


Рисунок 2.2 - Декомпозиція потоків даних системи платежів

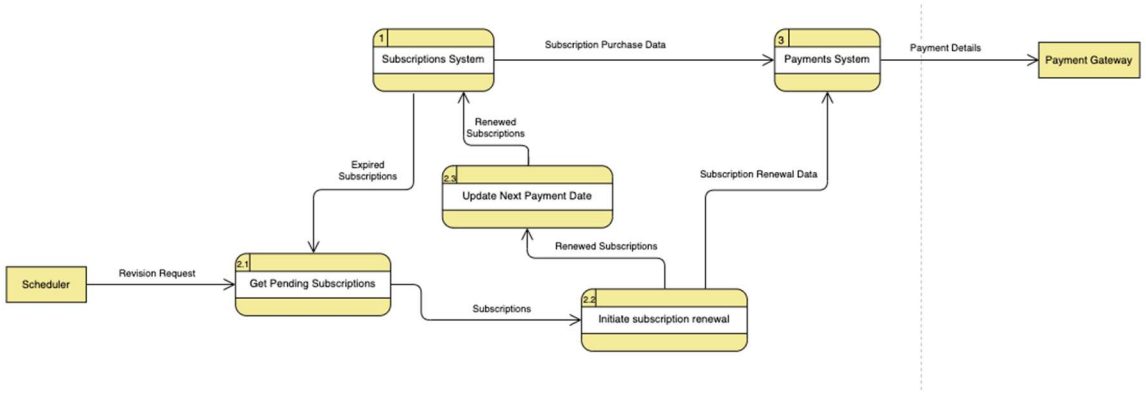


Рисунок 2.3 - Декомпозиція потоків даних системи автоматичного поновлення підписки

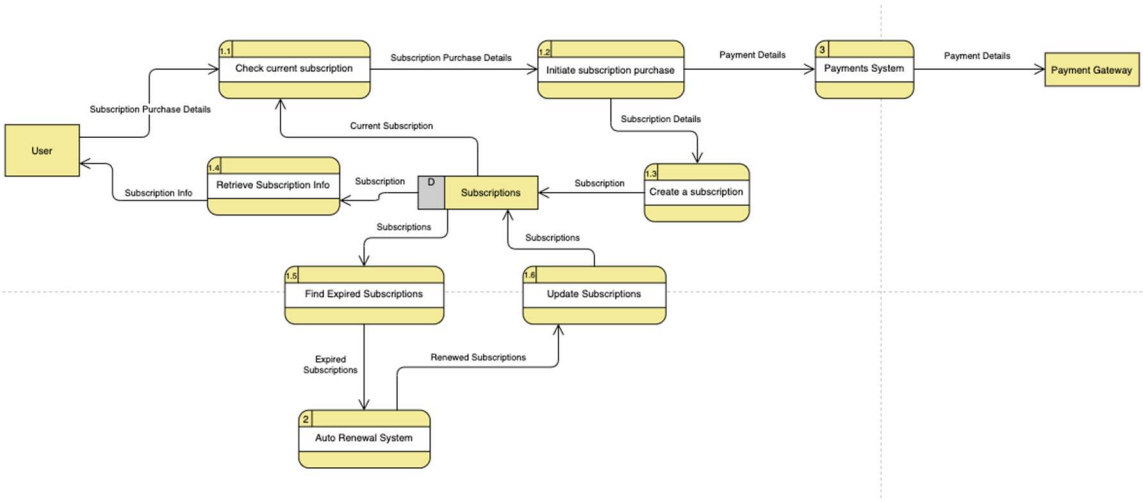


Рисунок 2.4 - Декомпозиція потоків даних системи обліку підписок

2.1.2 Вихідні дані

А. Вихідні дані для root користувача.

1. Стан підписки на послуги.
2. Список доданих account користувачів.
3. Дані окремого користувача компанії.

Б. Вихідні дані для account користувача.

1. Документація щодо інтеграції послуг в клієнтські застосунки.
2. API ключ доступу.

3. Інформація надана підсистемою розпізнавання за запитом.

В. Вихідні дані для незареєстрованого користувача.

1. Опис платіжного плану.

2. Опис послуг.

3. Дані щодо відновлення доступу до облікового запису.

Г. Вихідні дані підсистеми розпізнавання.

1. Перелік підтримуваних фруктів знайдених на наданому клієнтом зображенні.

2. Стан фрукта як стрічка, що містить значення із переліку можливих.

3. Точність (або впевненість) результату класифікації стану.

Вихідні дані підсистеми надання послуг відображені на діаграмах потоків даних (рисунки 2.1-2.4).

Також, система надсилає вихідні дані до платіжного сервісу Stripe, що включає в себе платіжну інформацію компанії-клієнта (номер картки, CVV2 код і т.д.), опис наданих інформаційною системою послуг (за які здійснюється оплата) та деякі необхідні дані клієнтів.

2.1.3 Функції та структура системи

Структура системи передбачає два бізнес-домени, які фізично розділені на різні простори імен та мережі. Вони відображені на загальній діаграмі архітектури системи (рисунок 2.5).

Прикладні застосунки, які працюють з запитами клієнтів та їх додатків, розміщені в домені «SP Domain», що є абревіатурою від Service Provider Domain (з англ. - домен надавача послуг). Цей домен передбачає розміщення в окремому AWS записі, з власним простором імен та обмеженою внутрішньою мережею. Підсистема має дві точки доступу, забезпечені сервісом API Gateway від Amazon Web Services, які є захищеними та надійними реверсивними проксі серверами.

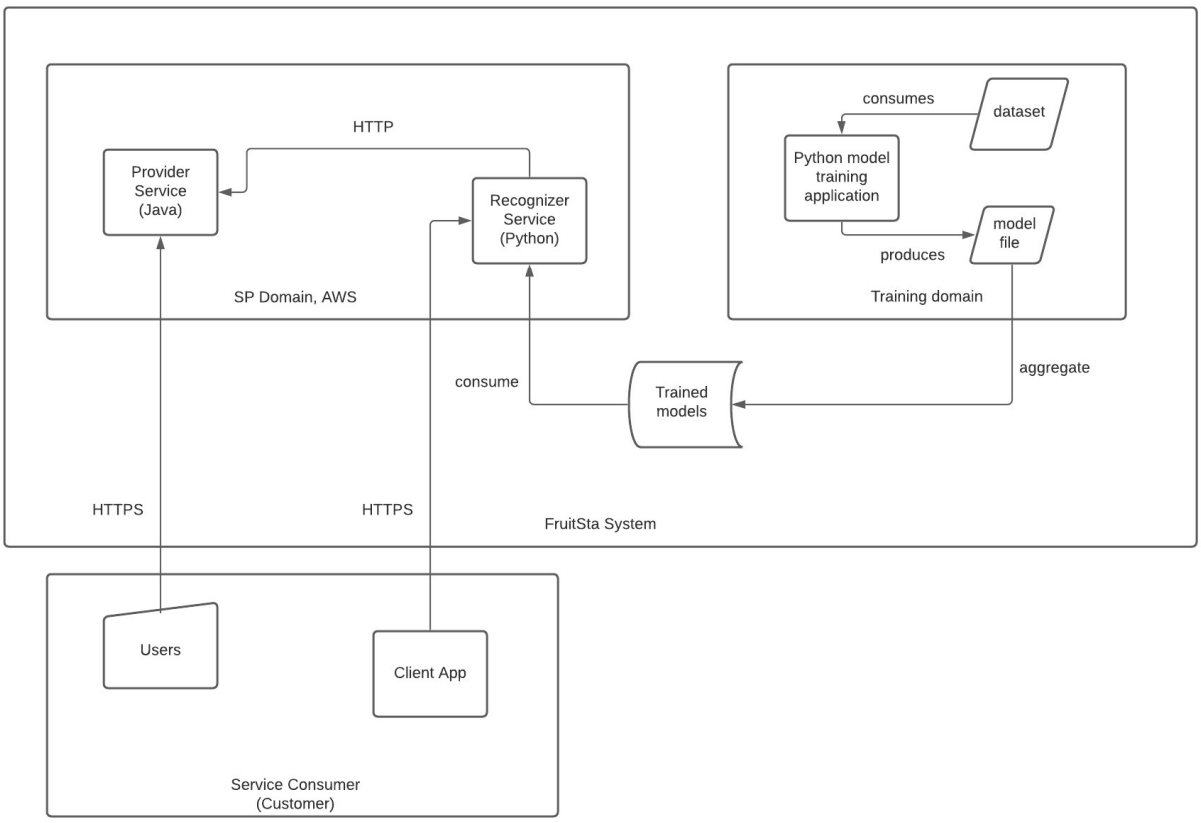


Рисунок 2.5 - Загальна архітектура системи

Інформаційні системи, розміщені в домені «SP Domain» відповідають підсистемам надання послуг та розпізнавання. Застосунок Provider Service, надає RESTful та Web UI інтерфейси для замовників та підсистем даного сервісу, веде облік підписок, платежів, користувачів і так далі. Інший додаток домену, Recognizer Service, обробляє запити від застосунків клієнта, перевіряючи наявність такої можливості за станом підписки клієнта. Якщо підписка є активною, про що має повідомити Provider Service підсистема, тоді зображення надіслане додатком клієнта аналізується і формується результат класифікації всіх знайдених підтримуваних фруктів з зображення.

Бізнес-домен «Training domain» (з англ. – домен тренування), це окреме Python середовище, що містить навчальний та тренувальний дата сети і код структури моделей. Цей застосунок навчає та перевіряє нейронні мережі, видаючи готові працюючі моделі в спільну площину доступу всієї системи.

Тоді Recognizer Service з «SP Domain» домену, споживає ці файли використовуючи їх при аналізі зображень.

Окремо варто відзначити, третій, зовнішній бізнес-домен, який містить репрезентацію користувачів та програмних застосунків клієнта. Цей домен визначає формат співпраці та інтеграції з інформаційною системою розпізнавання фруктів. Очікується, що вимоги щодо керування і адміністрування облікових записів клієнта і власне послуги розпізнавання виконуються окремими підсистемами домену «SP Domain».

Структура підсистеми надання послуг відповідає моделі даних системи, яка відображення на ER-діаграмі фізичної структури бази даних (рисунок 2.6). Незважаючи на ряд залежностей, які виводяться з моделі даних, класи підсистеми є максимально ізольованими та оптимізованими. Перш за все, це є результатом використання архітектурної моделі Dependency Injection, тобто ін’єкції залежностей. Таких підхід дає можливість інвертувати залежності й використовувати тільки абстракції у стабільних компонентах системи, забезпечуючи їх надійність й легкість зміни. Описані мною властивості можна побачити на діаграмі класів застосунку (рисунок 2.8), де чітко видно розмежування й мінімізація міжкомпонентних залежностей.

Структура підсистеми розпізнавання відображена на рисунку 2.7. Застосунок побудовано слідуючи архітектурному шаблону «каналів і фільтрів», що дозволяє визначити аналіз зображень як послідовність кроків – обробників та зв’язків передачі даних між ними.

Початковими кроками є попередня обробка зображення, критичною частиною якої є сегментаційна нейронна мережа базована на згортці та пірамідальному способі пропонування зон зацікавленості – Mask R-CNN [4]. Як результат цього кроку, отримується набір фруктів, їх координат та класів, які передаються на наступні кроки аналізу.

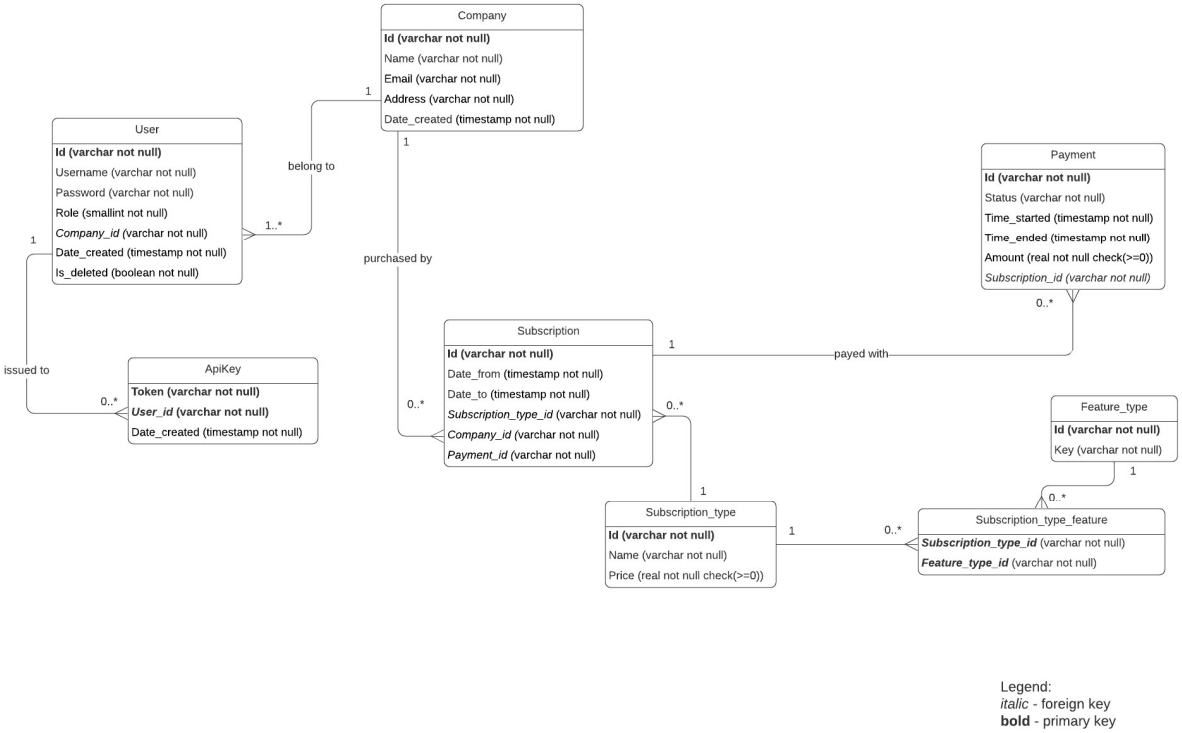


Рисунок 2.6 - Фізична модель бази даних

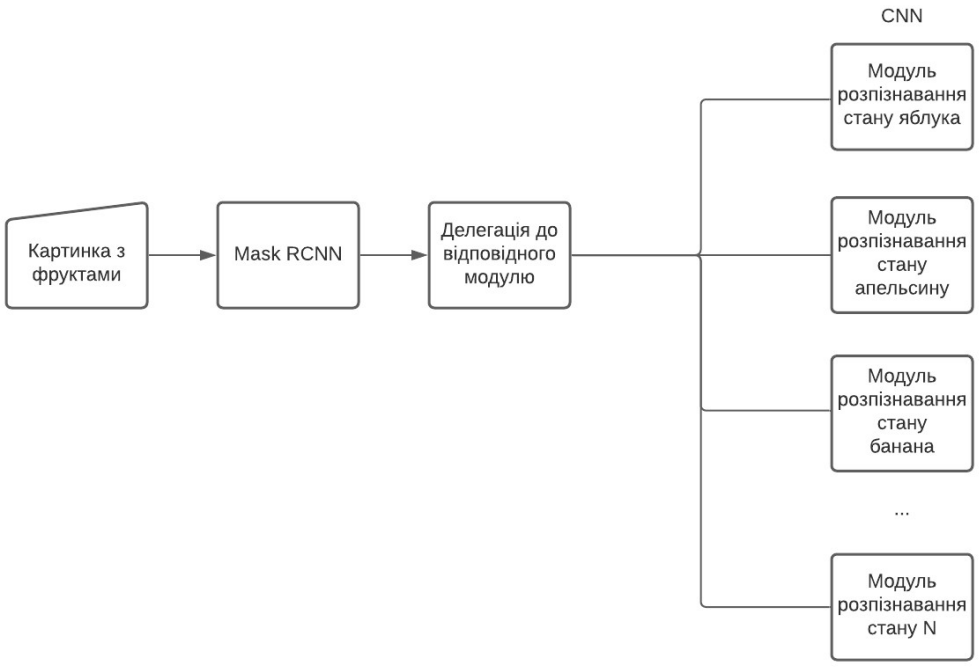


Рисунок 2.7 - Структура підсистеми розпізнавання

Вирішальною частиною процесу є крок делегації до конкретної класифікаційної мережі [6-8]. Важливість цього кроку полягає в забезпеченні можливості підтримки будь-якої кількості моделей з розпізнавання окремих типів фруктів. Розробники можуть швидко й без жодного ризику додавати нові моделі до підсистеми, маючи гарантію відсутності регресій.

В кінцевому результаті, вхідні дані користувача потрапляють до спеціалізованих моделей класифікації відповідно до типу зображеного фрукту.

Поведінкова модель програмного продукту передбачає постійну безпечну комунікацію між підсистемами надання послуг та розпізнавання. Обмін даними здійснюється через внутрішню мережу за протоколом HTTP. Підсистема надання послуг забезпечує такий функціонал відкриваючи RESTful точки доступу до серверу, які доступні тільки в межах AWS облікового запису системи.

Найяскравішим прикладом процесів роботи додатку, є процес аналізу зображень використовуючи Recognizer Service, який відображений на діаграмі послідовностей (рисунок 2.9). Як ми можемо спостерігати, перед власне обробкою запиту користувача, сервіс розпізнавання знаходить API ключ клієнта в заголовках запиту й відправляє ключ на перевірку до системи надання послуг через внутрішню мережу. Тільки при отриманні позитивної відповіді відбувається аналіз зображення й формування результату.

Рисунок 2.8 - Класи підсистеми надання послуг

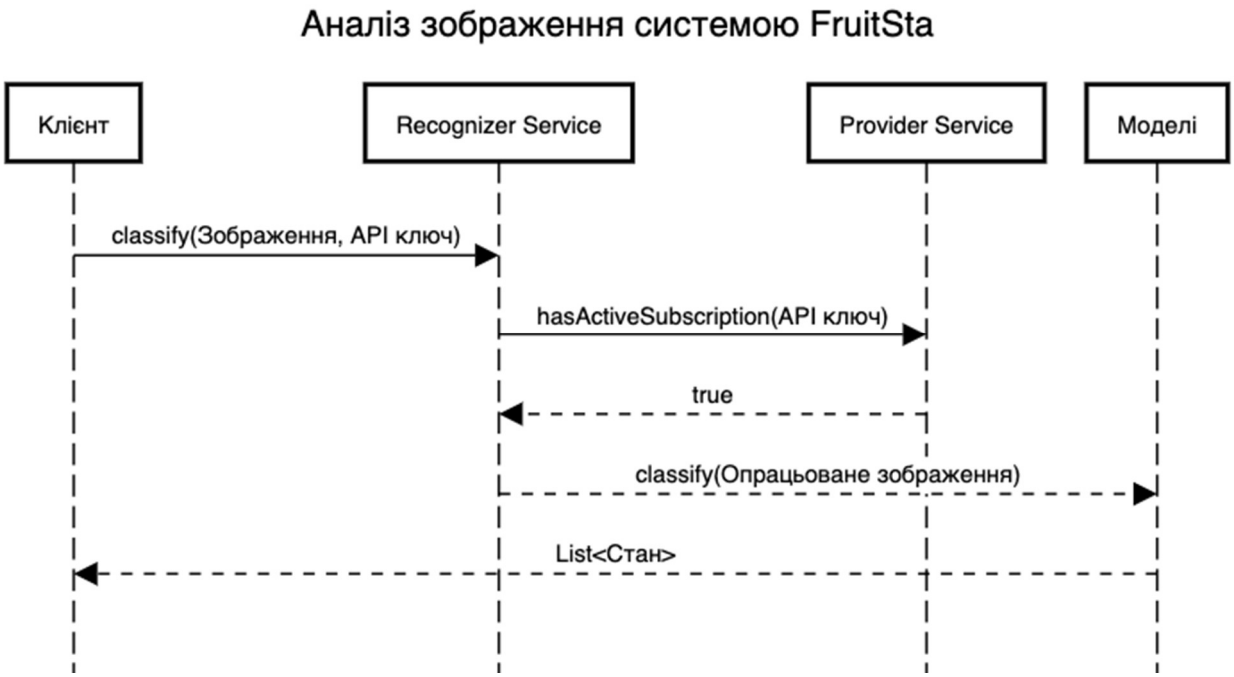


Рисунок 2.9 - Діаграма послідовностей класифікації зображення

2.2 Методи та засоби побудови системи

2.2.1 Підсистема надання послуг

Розробка підсистеми надання послуг здійснюється за моделями MVC (Model View Controller) розшарування та RESTful інтерфейсів. MVC дозволяє відділити бізнес-логіку застосунку, веб-інтерфейс (або REST інтерфейс) та сховище даних одне від одного, даючи можливість програмному продукту бути розширеним в подальшому до підтримки інших типів інтерфейсів та сховищ. RESTful підхід, у свою чергу, є стандартизованим способом комунікації додатків через мережу Інтернет, що забезпечує ефективну й просту інтеграцію підсистем та додатків замовників.

Задля реалізації інформаційної системи використовуються такі технології як:

- Java 11;
- Spring Web MVC;
- Spring Data JPA;
- Spring Security;
- Hibernate;
- Apache Freemarker;
- Caffeine Cache;
- Gradle;
- HTML;
- CSS;
- jQuery, Ajax;
- Bootstrap UI Library;
- jUnit;
- Mockito;
- Stripe SDK.

Як сховище даних буде використовуватись реляційна SQL СУБД PostgreSQL 10.

Початкове розгортання веб-сервісу здійснюватиметься в середовищі Heroku. Остаточним середовищем роботи системи буде хмарна платформа AWS, а саме екземпляри обчислювального сервісу EC2 з інтеграцією API Gateway проксі серверів.

Реалізація кодової бази кваліфікаційної роботи здійснюватиметься за методологією Test Driven Development для покращення якості та гнучкості коду проекту.

Інтерфейс вза'ємодії з користувачем базується на технології динамічної генерації HTML сторінок використовуючи мову Apache Freemarker. Такий підхід відповідає шаблону архітектури «тонкий клієнт – товстий сервер». Отже клієнт звільнюється від динамічного створення сторінок передаючи це завдання ресурсам сервера. В по'єднанні з мережами розподіленої доставки вмісту – CDN (Content Delivery Network) – відкривається можливість значної оптимізації комунікації сторінок.

Альтернативними способами побудови графічного інтерфейсу користувача є архітектури «товстий клієнт – тонкий сервер» та Single Page Architecture (з англ. – односторінкова архітектура). Для реалізації такого методу використовуються технології React, Angular, Vue.js. Проте в такому випадку перший візит веб-сайту клієнтом був б довгим, адже весь інтерфейс завантажувався б одразу й перевикористовувався при всіх наступних візитах. У випадку даної інформаційної системи, важливими є перші декілька запитів, вони повинні бути швидкими і ефективними. Тому було прийнято рішення відмовитися від даного способу на користь серверної генерації динамічних сторінок.

Вважаю за потрібне, звернути особливу увагу на метод інтеграції сповіщень про події від платіжного сервісу в запропонований програмний комплекс. Платіжний сервіс Stripe надсилає повідомлення до веб-хук точок доступу застосунку при завершенні процесу заповнення платіжних даних, під час якого створюється підписка та повідомлення про успішно або не успішно проведену оплату за підписку. Задля надійної роботи системи, надходження

цих подій має бути за строгим порядком. Адже якщо додаток отримає повідомлення про успішну оплату за підписку, про яку йому ще не відомо, дані про оплату будуть безповоротно втрачені, а отже, втрачені й гроші замовника.

Рішенням даної проблеми потенційної неузгодженості даних при інтеграції було обрано побудову in-memory черги повідомлень із фіксованою пріоритезацією за типом події. А саме, всі події які несуть інформацію щодо успішності або неуспішності певного платежу затримуються на 3 секунди, що дає можливість опрацювати всі події у правильному порядку, при цьому додаючи функціонал повторних спроб для надійності інтеграції.

2.2.2 Підсистема розпізнавання

А. Побудова моделей. Для класифікації станів фруктів побудовано багатошарові згорткові нейронні мережі для кожного типу фруктів. Структура мереж однакова й містить 3 згорткові шари, 2 шари підвибірки та 4 повнозв’язні шари. Всього використовуються дві активаційні функції – ReLu [5] для прихованих шарів та softmax для вихідного шару повнозв’язної підмережі, що формує результат із 3 чисел, кожне з яких відповідає ймовірності належності до кожного з класів станів фруктів. Кількість нейронів у шарах описана в таблиці 2.1. Вимоги до вхідних зображень кожного типу фруктів описані в таблиці 2.2.

Таблиця 2.1 – Структура нейронів класифікаційних мереж

Назва шару	Кількість нейронів (шт)
Початковий згортковий шар	128
Приховані згорткові шари	256
Вхідний повнозв’язний шар	256
Прихований повнозв’язний шар 1	512
Прихований повнозв’язний шар 2	128
Вихідний повнозв’язний шар	3

Таблиця 2.2 – Розмір вхідних зображень

Тип фруктів	Ширина (пікселі)	Висота (пікселі)
Банан	250	500
Яблуко	250	250
Апельсин	250	250

Б. Навчання моделей. Навчання моделей виконується методом зворотного розповсюдження помилки. При навчанні використовувались параметри відображені на таблиці 2.3. При цьому, задля покращення результатів навчання до частини дата сету додавався гаусовий шум, що сприяє запобіганню перенавчання та покращенню здібностей мережі до узагальнення.

Таблиця 2.3 – Параметри тренування нейронних мереж

Тип фруктів	Швидкість навчання	Кількість епох
Банан	0.0001	30
Яблуко	0.00001	20

В. Попередня обробка зображень. Завдання попередньої обробки зображень, а саме сегментації з частковою класифікацією виконано з застосуванням моделі з відкритим кодом Mask R-CNN. Це складний багатокроковий механізм, який спочатку виконує багатошарову згортку зображення, а потім, на основі мережі пропонування зон зацікавленості виділяє частини зображення, які подаються на вхід нейронним мережам для точного визначення положення й типу об’єкта. В даній кваліфікаційній роботі я використав попередньо натреновану модель на публічному дата сеті COCO. Детально етапи роботи мережі представлено на рисунку 2.10.

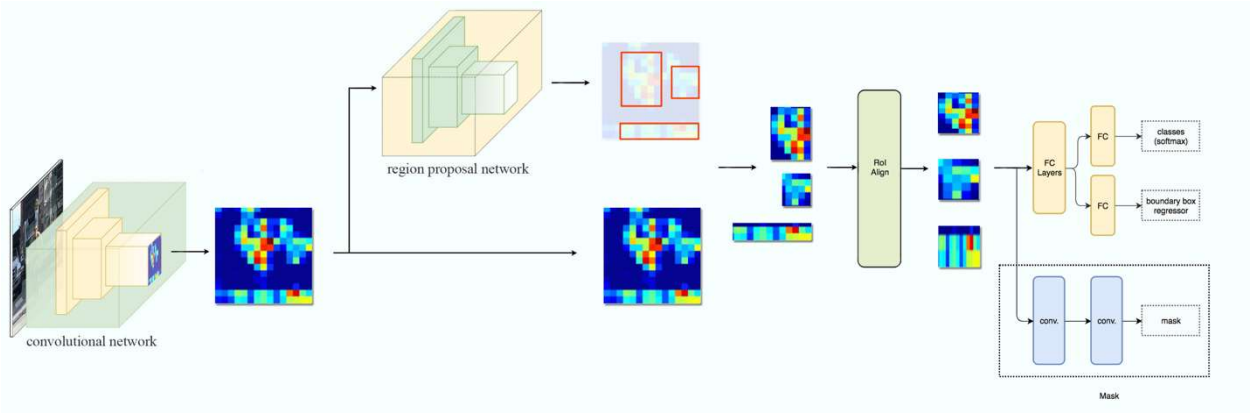


Рисунок 2.10 – Схема роботи Mask R-CNN

Висновки до розділу 2

У даному розділі мною були представлені альтернативи побудови системи та обрані моделі розробки. Проаналізувавши технічне завдання даної кваліфікаційної роботи, я вирішив реалізувати її використовуючи MVC архітектуру з «тонким» клієнтом та «товстим» сервером. Також в обсяг завдань включив побудову точок доступу RESTful API. Для інтеграції платіжних сервісів використовуватиметься методика веб-хуків.

Підсистема розпізнавання, у свою чергу, буде розроблена як набір згорткових нейронних мереж (CNN) в поєднанні з Mask R-CNN в ролі сегментатора цікавих областей зображення. Параметри навчання та структури моделі були обрані та представлені у відповідних таблицях.

РОЗДІЛ 3
ПРАКТИЧНА РЕАЛІЗАЦІЯ

3.1 Засоби розробки

Для виконання даної розробки я використовував такий інструментарій, як:

- IntelliJ IDEA Ultimate Edition;
- PyCharm;
- git cli;
- GitHub;
- TensorFlow;
- Keras;
- Python 3.8;
- JVM 8;
- Heroku;
- SonarQube.

При цьому використовувався апаратний засіб, у вигляді комп’ютера з характеристиками описаними в таблиці 3.1.

Таблиця 3.1 – Технічні характеристики ПК

Характеристика	Значення
Тактова частота процесора	2,60 GHz - 4,50 GHz
Об’єм оперативної пам’яті	16 GB
Обсяг накопичувача SSD	1 TB

3.2 Опис реалізації системи

Підсистема надання послуг реалізована як Java додаток з застосуванням Spring Framework. Програма компілюється в байткод та запускається сервлет контейнером Apache Tomcat для обробки запитів користувачів. Програма використовує реляційну базу даних PostgreSQL та підключається до неї використовуючи протокол JDBC. Проектування таблиць бази даних в об'єкти мови програмування здійснюється засобами фреймворку Hibernate. Також для кешування даних використовується бібліотека з відкритим кодом Caffeine. Основні безпекові функції, такі як автентифікація та авторизація реалізовані за використанням Spring Security та HTTP сесій.

Уривок коду представлений нижче реалізовує чергу подій платіжної системи з затримкою окремих елементів на 3 секунди для досягнення повної узгодженості даних системи незалежно від фактичного порядку прибуття подій до мережевої карти сервера.

```
package com.drofff.fruitsta.event.queue;

import com.github.benmanes.caffeine.cache.Cache;
import com.github.benmanes.caffeine.cache.Caffeine;
import org.springframework.stereotype.Component;
import java.time.Duration;
import java.time.LocalDateTime;
import java.util.List;
import java.util.Map;
import static java.util.stream.Collectors.toList;
import static java.util.stream.Collectors.toMap;

@Component
public class InMemoryEventQueue implements EventQueue {
```

```
    private final Map<QueueTopic, Cache<Object, LocalDateTime>> topics;

    public InMemoryEventQueue() {
        this.topics = QueueTopic.topics().map(t -> new Object() {
            final QueueTopic topic = t;
            final Cache<Object, LocalDateTime> events = initTopicEvents();
        }).collect(toMap(obj -> obj.topic, obj -> obj.events));
    }

    private Cache<Object, LocalDateTime> initTopicEvents() {
        return Caffeine.newBuilder()
                .expireAfterWrite(Duration.ofMinutes(20))
                .build();
    }

    @Override
    public void emit(QueueTopic topic, Object event, long delaySeconds) {
        Cache<Object, LocalDateTime> events = topics.get(topic);
        LocalDateTime visibleAfter =
LocalDateTime.now().plusSeconds(delaySeconds);
        events.put(event, visibleAfter);
    }

    @Override
    public List<Object> consume(QueueTopic topic) {
        LocalDateTime now = LocalDateTime.now();
        Cache<Object, LocalDateTime> events = topics.get(topic);
        List<Object> availableEvents = events.asMap().entrySet().stream()
                .filter(e -> isAvailableAt(e, now))
                .map(Map.Entry::getKey)
```

```
            .collect(toList());

        availableEvents.forEach(events::invalidate);
        return availableEvents;
    }

    private boolean isAvailableAt(Map.Entry<Object, LocalDateTime> event,
LocalDateTime time) {
        LocalDateTime visibleAfter = event.getValue();
        return time.isAfter(visibleAfter) || time.isEqual(visibleAfter);
    }
}
```

Перш за все, повідомлення що надходять до веб-хуків потрапляють до сприймачів, які здійснюють класифікацію події і передають її до черги описаної вище в потрібному типі мови програмування. Сприймачі реалізовані для інтеграції з платіжною системою Stripe використовують клієнтську бібліотеку Stripe SDK. Код даної реалізації надано нижче.

```
package com.drofff.fruitsta.stripe;

import com.drofff.fruitsta.event.CheckoutCompletedEvent;
import com.drofff.fruitsta.event.GatewaySubscriptionEvent;
import com.drofff.fruitsta.event.PaymentFailureEvent;
import com.drofff.fruitsta.event.PaymentSuccessEvent;
import com.drofff.fruitsta.event.consumer.GatewaySubscriptionEventConsumer;
import com.drofff.fruitsta.event.queue.EventQueue;
import com.drofff.fruitsta.event.queue.QueueTopic;
import com.drofff.fruitsta.exception.WebhookException;
import com.google.gson.Gson;
```

```
import com.google.gson.JsonObject;
import com.stripe.exception.SignatureVerificationException;
import com.stripe.model.Event;
import com.stripe.net.Webhook;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

@Component
public class StripeGatewaySubscriptionEventConsumer implements
GatewaySubscriptionEventConsumer {
   private static final long PROCESSING_DELAY_SECONDS = 3L;
   private static final String EVENT_CHECKOUT_COMPLETED =
"checkout.session.completed";
   private static final String EVENT_INVOICE_PAID = "invoice.paid";
   private static final String EVENT_INVOICE_FAILED =
"invoice.payment_failed";
   private static final String FIELD_ID = "id";
   private static final String FIELD_SUBSCRIPTION = "subscription";
   private final String webhookSecret;
   private final Gson gson;
   private final EventQueue eventQueue;

   public
StripeGatewaySubscriptionEventConsumer(@Value("${stripe.webhook.secret}")
String webhookSecret, Gson gson,
                                          EventQueue eventQueue) {
      this.webhookSecret = webhookSecret;
      this.gson = gson;
      this.eventQueue = eventQueue;
   }
```

```
    @Override
    public void consume(String body, String signature) {
        Event event = verifyEventSignature(body, signature);
        GatewaySubscriptionEvent gatewaySubscriptionEvent =
toGatewaySubscriptionEvent(event);
        eventQueue.emit(QueueTopic.GATEWAY_SUBSCRIPTION,
gatewaySubscriptionEvent, PROCESSING_DELAY_SECONDS);
    }

    private Event verifyEventSignature(String eventBody, String signature) {
        try {
            return Webhook.constructEvent(eventBody, signature, webhookSecret);
        } catch (SignatureVerificationException e) {
            throw new WebhookException("Invalid event signature", e);
        }
    }

    private GatewaySubscriptionEvent toGatewaySubscriptionEvent(Event event) {
        switch (event.getType()) {
            case EVENT_CHECKOUT_COMPLETED:
                return parseCheckoutCompletedEvent(event);
            case EVENT_INVOICE_PAID:
                return parsePaymentSuccessEvent(event);
            case EVENT_INVOICE_FAILED:
                return parsePaymentFailureEvent(event);
            default:
                throw new WebhookException("Unexpected event type: " +
event.getType());
        }
```

```
    }

    private GatewaySubscriptionEvent parseCheckoutCompletedEvent(Event event)
{
        JsonObject checkoutSessionJson = getEventPayload(event);
        String sessionId = checkoutSessionJson.get(FIELD_ID).getAsString();
        String                                subscriptionId                                =
checkoutSessionJson.get(FIELD_SUBSCRIPTION).getAsString();
        return new CheckoutCompletedEvent(sessionId, subscriptionId);
    }

    private GatewaySubscriptionEvent parsePaymentSuccessEvent(Event event) {
        JsonObject invoicePaidJson = getEventPayload(event);
        String                                subscriptionId                                =
invoicePaidJson.get(FIELD_SUBSCRIPTION).getAsString();
        return new PaymentSuccessEvent(subscriptionId);
    }

    private GatewaySubscriptionEvent parsePaymentFailureEvent(Event event) {
        JsonObject invoiceFailedJson = getEventPayload(event);
        String                                subscriptionId                                =
invoiceFailedJson.get(FIELD_SUBSCRIPTION).getAsString();
        return new PaymentFailureEvent(subscriptionId);
    }

    private JsonObject getEventPayload(Event event) {
        String dataJson = event.getDataObjectDeserializer().getRawJson();
        return gson.fromJson(dataJson, JsonObject.class);
    }
}
```

Підсистема розпізнавання розроблена як багатофункціональний Python додаток з використанням таких технологій як Keras, Numpy та Tensorflow. Побудова моделі здійснюється декларативним способом методами Keras фреймворку і представлена в уривках коду нижче для таких типів фруктів як банан та яблуко у відповідному порядку викладу.

```
def banana_model():
    model = models.Sequential()
    model.add(layers.Conv2D(CONV_FILTERS_L1_BANANA,
CONV_KERNEL_SIZE_BANANA, activation='relu',
                            input_shape=IMAGE_SHAPE_BANANA))
    model.add(layers.MaxPooling2D(MAX_POOLING_SIZE_BANANA))
    model.add(layers.Conv2D(CONV_FILTERS_LN_BANANA,
CONV_KERNEL_SIZE_BANANA, activation='relu'))
    model.add(layers.MaxPooling2D(MAX_POOLING_SIZE_BANANA))
    model.add(layers.Conv2D(CONV_FILTERS_LN_BANANA,
CONV_KERNEL_SIZE_BANANA, activation='relu'))

    model.add(layers.Flatten())
    model.add(layers.Dense(DENSE_L1_UNITS,
activation=ACTIVATION_FUNC))
    model.add(layers.Dense(DENSE_L2_UNITS,
activation=ACTIVATION_FUNC))
    model.add(layers.Dense(DENSE_L3_UNITS,
activation=ACTIVATION_FUNC))
    model.add(layers.Dense(OUTPUT_SITE))
    return model

def apple_model():
    model = models.Sequential()
```

```
    model.add(layers.Conv2D(CONV_FILTERS_L1_APPLE,
CONV_KERNEL_SIZE_APPLE, activation='relu',
                            input_shape=IMAGE_SHAPE_APPLE))
    model.add(layers.MaxPooling2D(MAX_POOLING_SIZE_APPLE))
    model.add(layers.Conv2D(CONV_FILTERS_LN_APPLE,
CONV_KERNEL_SIZE_APPLE, activation='relu'))
    model.add(layers.MaxPooling2D(MAX_POOLING_SIZE_APPLE))
    model.add(layers.Conv2D(CONV_FILTERS_LN_APPLE,
CONV_KERNEL_SIZE_APPLE, activation='relu'))

    model.add(layers.Flatten())
    model.add(layers.Dense(DENSE_L1_UNITS,
activation=ACTIVATION_FUNC))
    model.add(layers.Dense(DENSE_L2_UNITS,
activation=ACTIVATION_FUNC))
    model.add(layers.Dense(DENSE_L3_UNITS,
activation=ACTIVATION_FUNC))
    model.add(layers.Dense(OUTPUT_SITE))
    return model
```

Код, який виконує навчання нейронних мереж на основі зібраних тренувальному та тестувальному дата сетах реалізований засобами технології TensorFlow та представлений нижче.

```
import os
import sys

MR_CNN_DIR = os.path.abspath("../Mask_RCNN")

sys.path.append(MR_CNN_DIR)
```

```
from mrcnn import utils
import mrcnn.model as modellib

sys.path.append(MR_CNN_DIR + "/samples/coco/")
import coco

MODEL_DIR = "/home/drofff/models/frstar"
COCO_MODEL_PATH = "/home/drofff/models/coco"

RESULT_PARAM_ROIS = "rois"
RESULT_PARAM_MASKS = "masks"
RESULT_PARAM_CLASS_IDS = "class_ids"

RESULT_PARAM_ROIS = "rois"
RESULT_PARAM_MASK = "mask"
RESULT_PARAM_CLASS_ID = "class_id"

CLASS_ID_APPLE = 48
CLASS_ID_BANANA = 47
CLASS_ID_ORANGE = 50

CLASSES_TO_DETECT = [CLASS_ID_APPLE, CLASS_ID_BANANA,
CLASS_ID_ORANGE]

class DetectionConfig(coco.CocoConfig):
    GPU_COUNT = 1
    IMAGES_PER_GPU = 1

def init():
    if not os.path.exists(COCO_MODEL_PATH):
```

```
        utils.download_trained_weights(COCO_MODEL_PATH)

    config = DetectionConfig()
    config.display()

    model = modellib.MaskRCNN(mode="inference", model_dir=MODEL_DIR,
config=config)
    model.load_weights(COCO_MODEL_PATH, by_name=True)
    return model

def detect(img, model):
    results = model.detect([img])[0]

    detected_fruits = []
    for ri in range(len(results[RESULT_PARAM_ROIS])):
        class_id = results[RESULT_PARAM_CLASS_IDS][ri]
        if class_id in CLASSES_TO_DETECT:
            detected_fruits.append({
                RESULT_PARAM_CLASS_ID: class_id,
                RESULT_PARAM_ROIS: results[RESULT_PARAM_ROIS][ri],
                RESULT_PARAM_MASK: results[RESULT_PARAM_MASKS][:, :, ri],
            })
    return detected_fruits
```

Серверний застосунок підсистеми розпізнавання, який приймає зображення клієнтів та аналізує їх за допомогою нейромереж та сегментаційних інструментів містить код представлений нижче.

```
import base64
import json
```

```
import uuid
from http.server import BaseHTTPRequestHandler, HTTPServer
import skimage.io
import recognise

DIR_SERVER_FILES = 'tmp'

class RecognitionService(BaseHTTPRequestHandler):
    def do_POST(self):
        if self.path != '/api/v1/fruits':
            self.send_response(404)
            return
        try:
            data = self.rfile.read()
            image_dto = json.loads(data)
            image_base64 = image_dto['image']
            image_bytes = base64.decodebytes(image_base64)
            filename = DIR_SERVER_FILES + '/' + str(uuid.uuid4())
            with open(filename, 'w') as f:
                f.write(image_bytes)
                f.close()

            image = skimage.io.imread(filename)
            recognise.recognise(image)
            self.send_response(200)
        except Exception:
            self.send_response(500)
        self.end_headers()

if name == 'main':
```

```
server_address = ('', 5000)
httpd = HTTPServer(server_address, RecognitionService)
try:
    print('Server started. Listening port 5000')
    httpd.serve_forever()
except KeyboardInterrupt:
    pass
httpd.server_close()
```

3.3 Аналіз отриманих результатів

3.3.1 Контрольний приклад

А. Підсистема надання послуг. Додаток надання послуг виконує облік клієнтів та їх підписок. Основні сценарії використання застосунку відображені на рисунках 3.1-3.30. Такими сценаріями є:

- Початковий візит сайту, рисунки 3.1-3.2;
- Реєстрація та вхід у обліковий запис відповідного типу, рисунки 3.3-3.5;
- Відновлення паролю root користувача, рисунки 3.6-3.11;
- Створення та активація запису клієнта, рисунки 3.12-3.14;
- Оформлення підписки на послуги, рисунки 3.15-3.17;

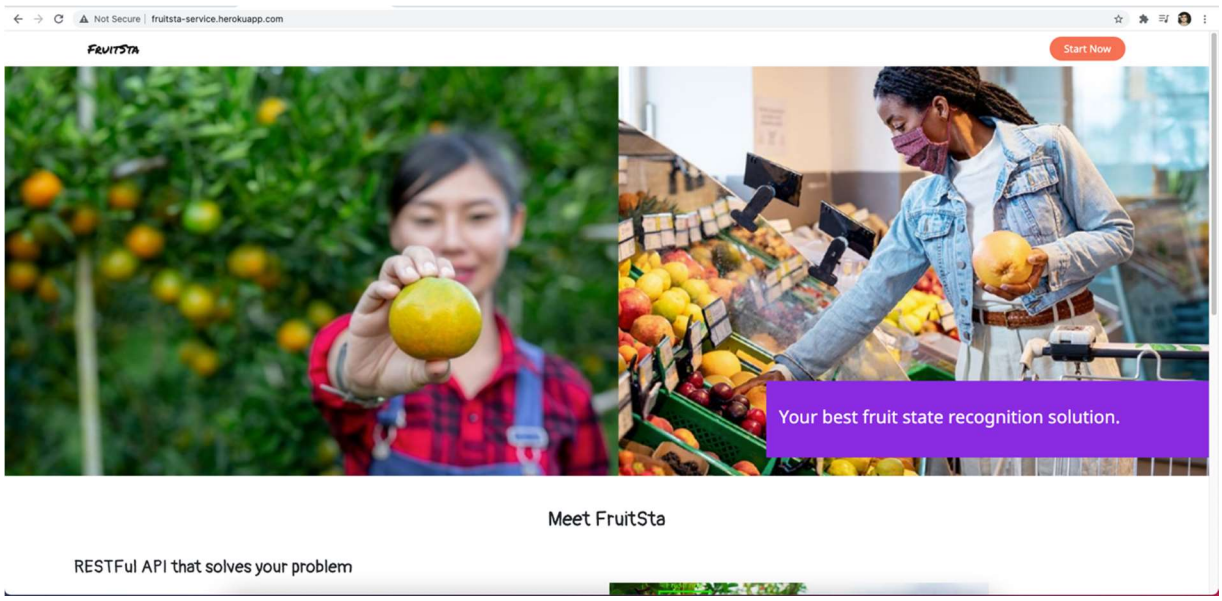


Рисунок 3.1 – Головна сторінка відвідувача, верхня частина

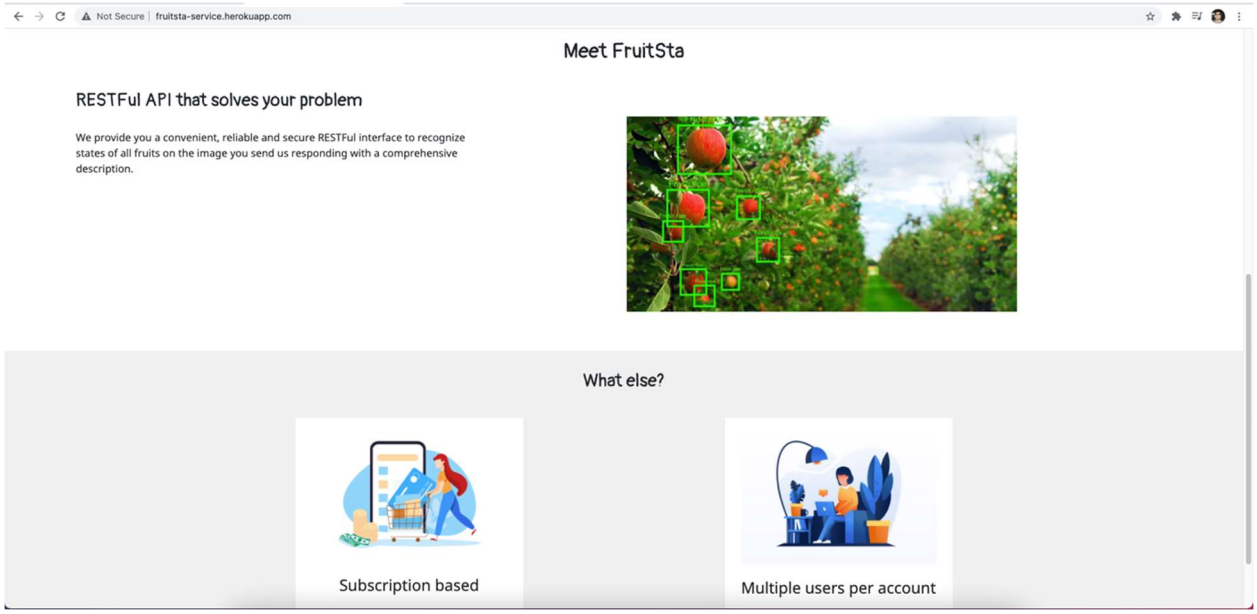


Рисунок 3.2 – Головна сторінка відвідувача, нижня частина

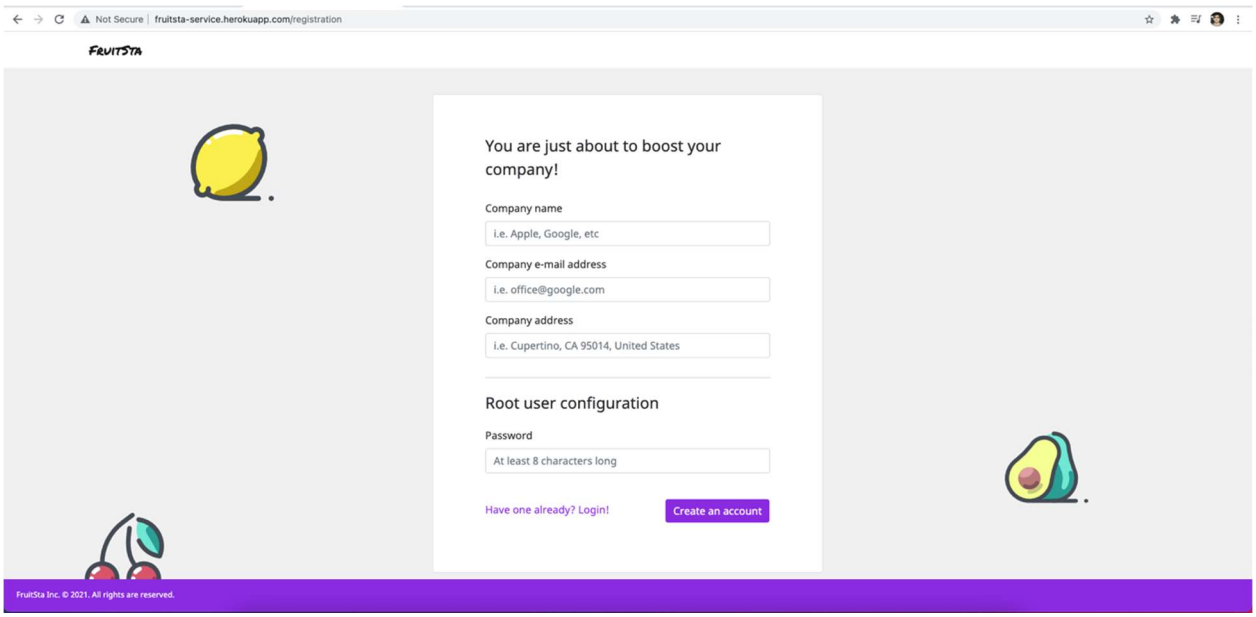


Рисунок 3.3 – Сторінка реєстрації клієнта

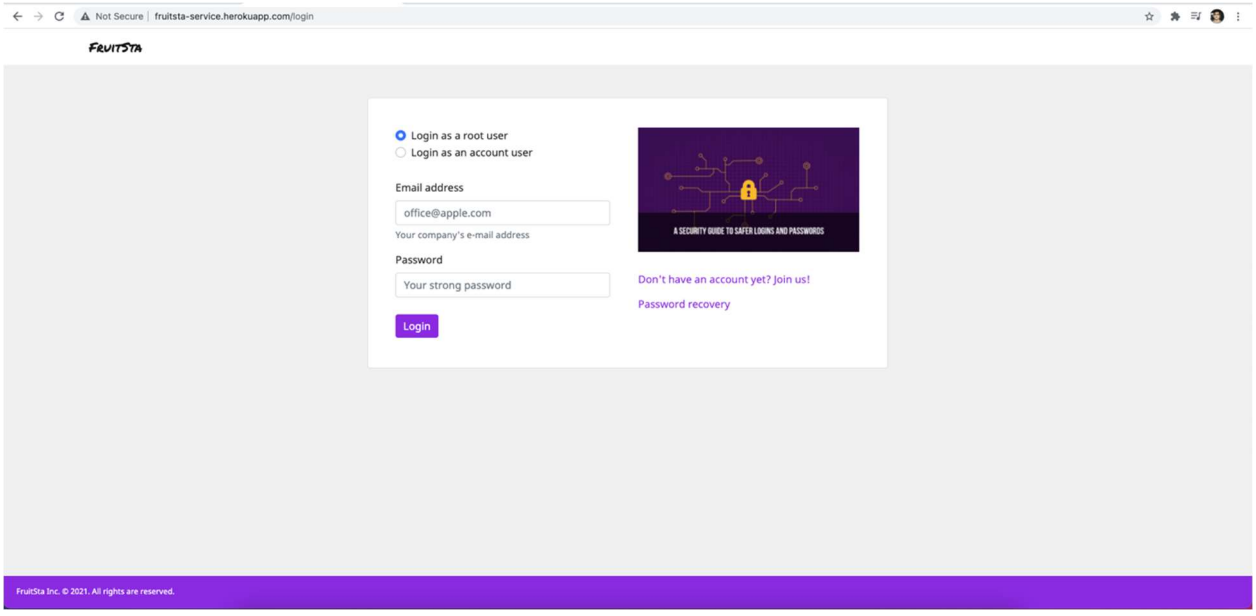


Рисунок 3.4 – Сторінка входу root користувача

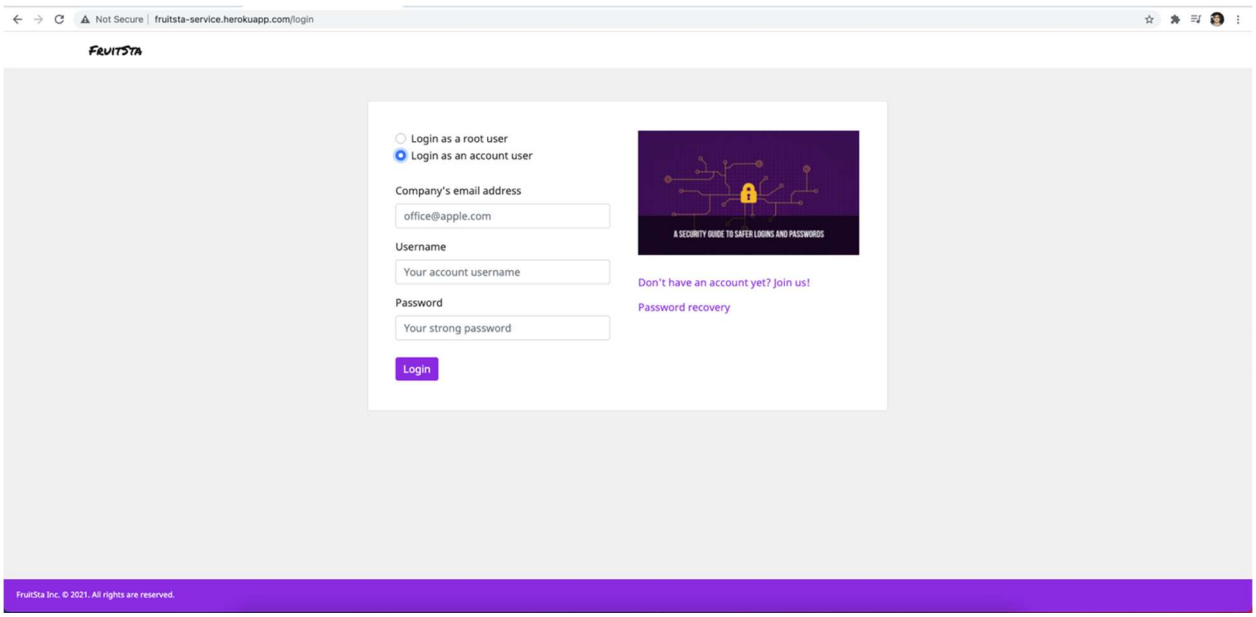


Рисунок 3.5 – Сторінка входу account користувача

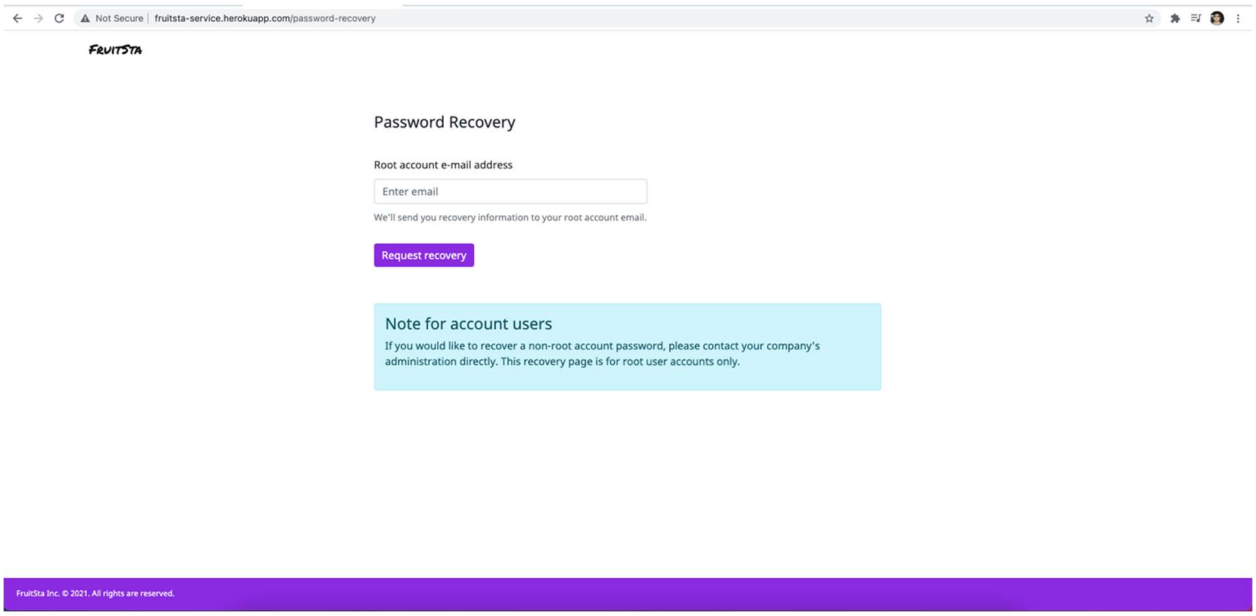


Рисунок 3.6 – Сторінка надсилання запиту на відновлення паролю

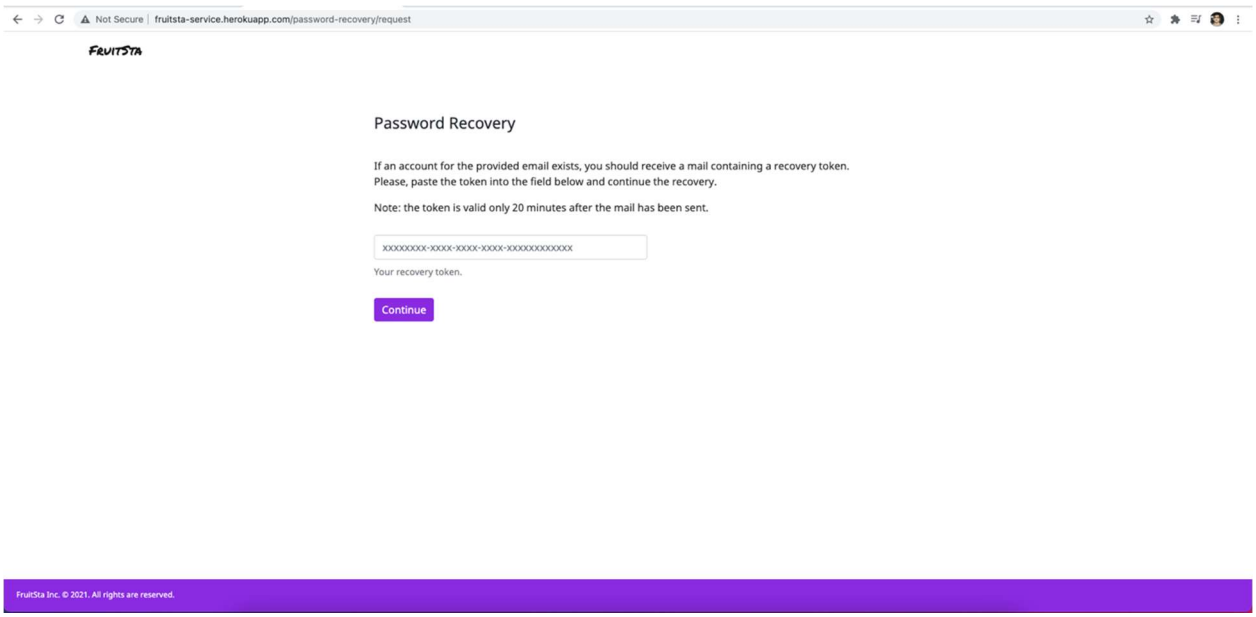


Рисунок 3.7 – Сторінка вводу коду відновлення

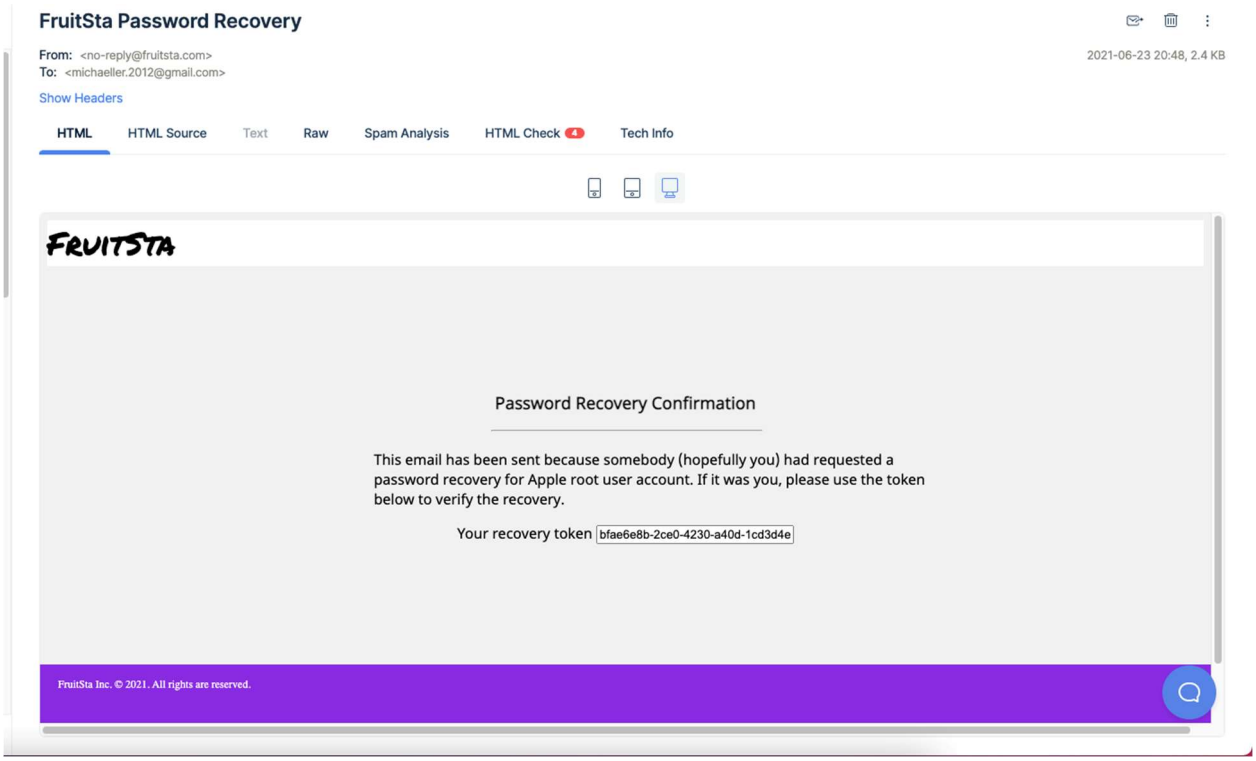


Рисунок 3.8 – Лист з кодом відновлення

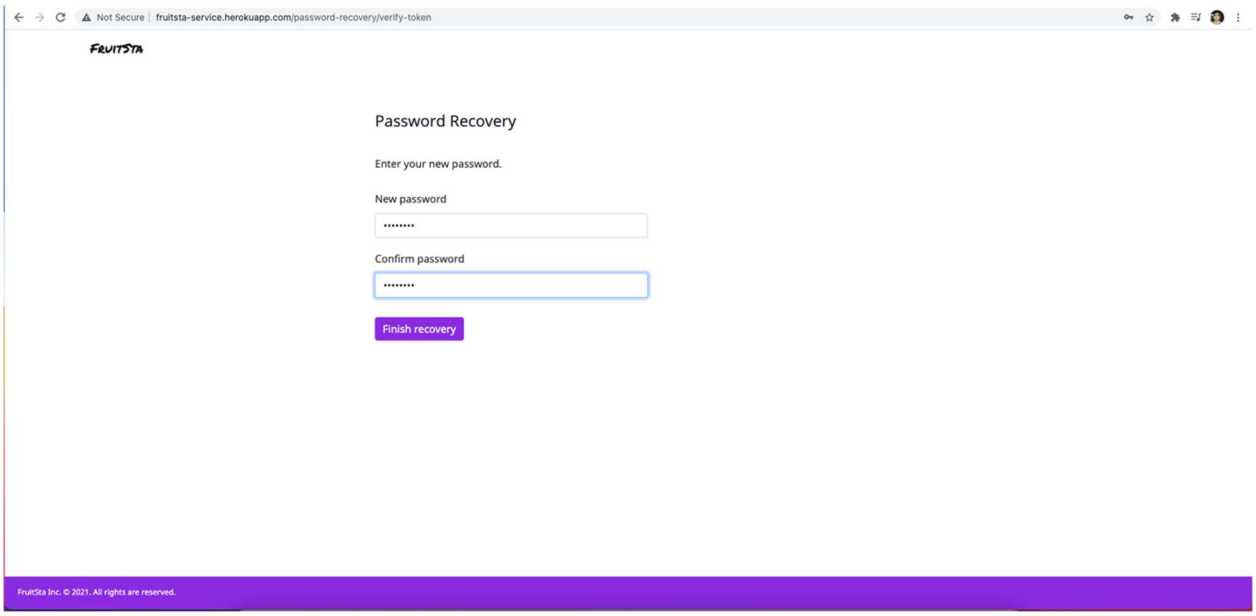


Рисунок 3.9 – Вікно вводу нового паролю

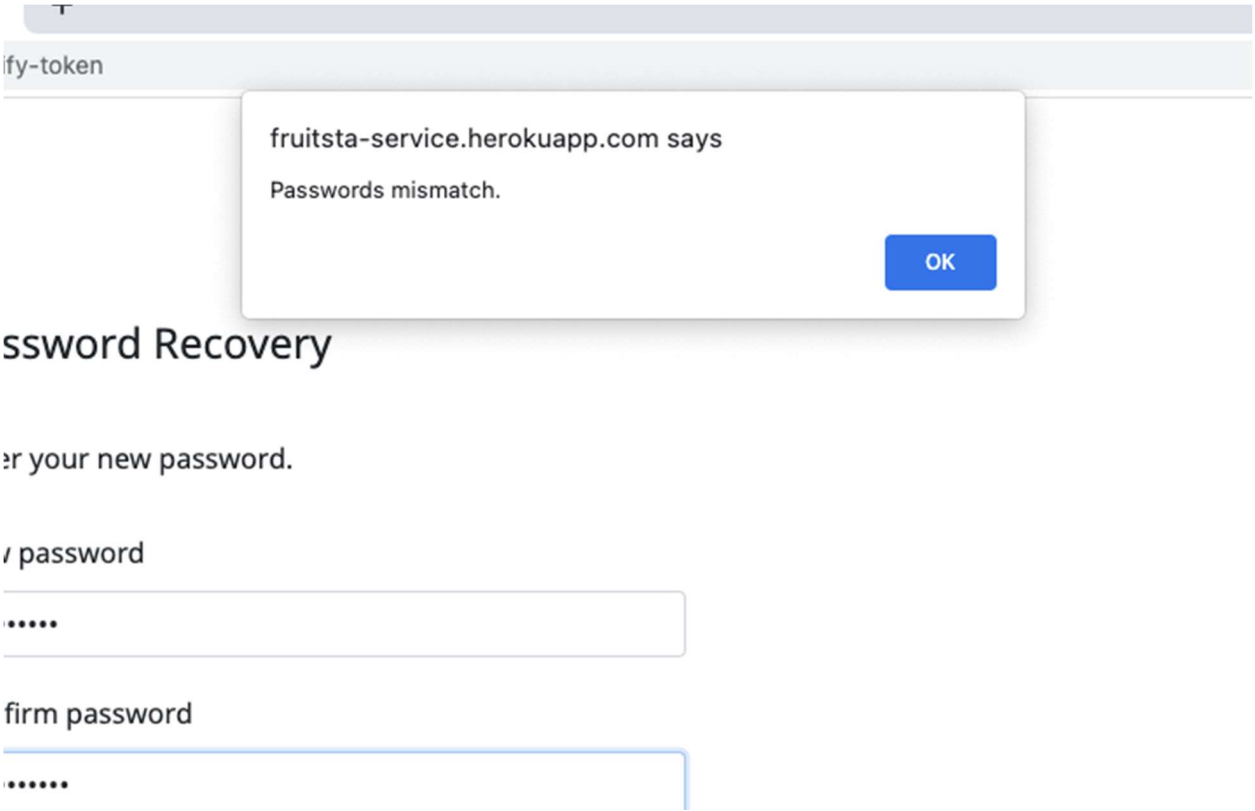


Рисунок 3.10 – Приклад невірного вводу нового паролю

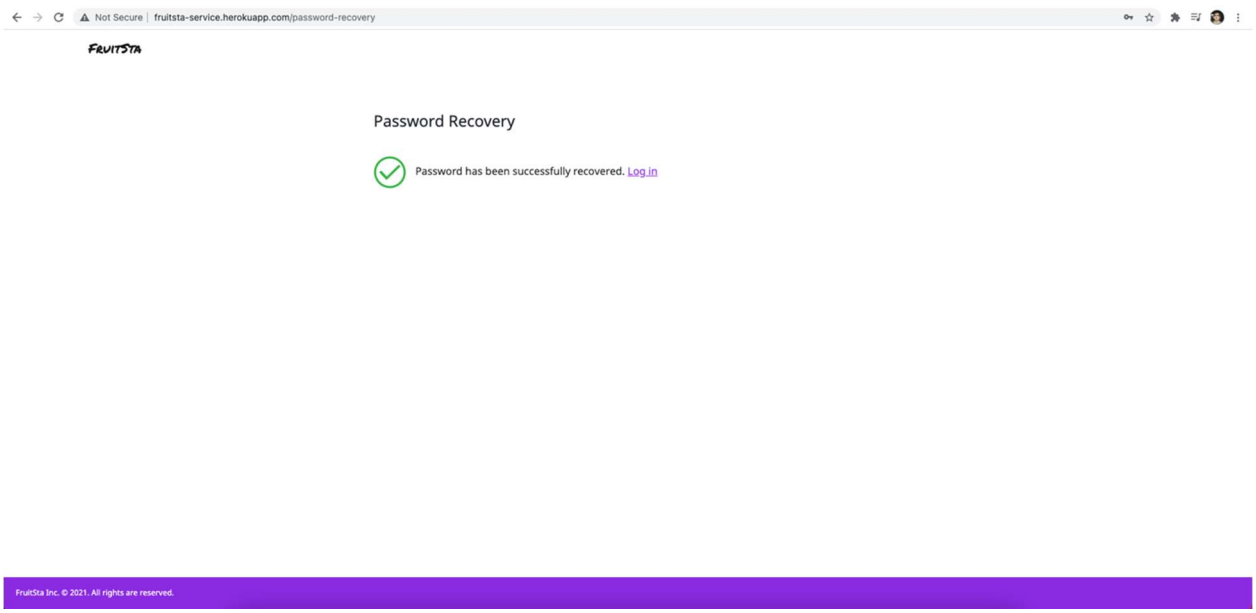


Рисунок 3.11 – Вікно успішного відновлення паролю

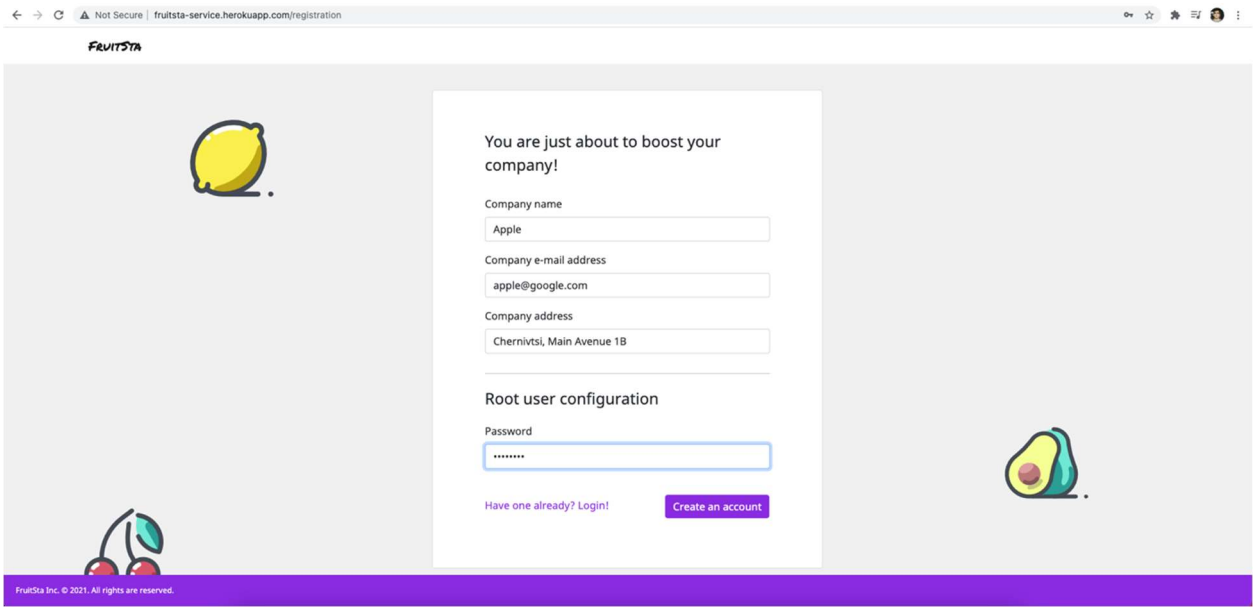


Рисунок 3.12 – Заповнення даних клієнта при реєстрації

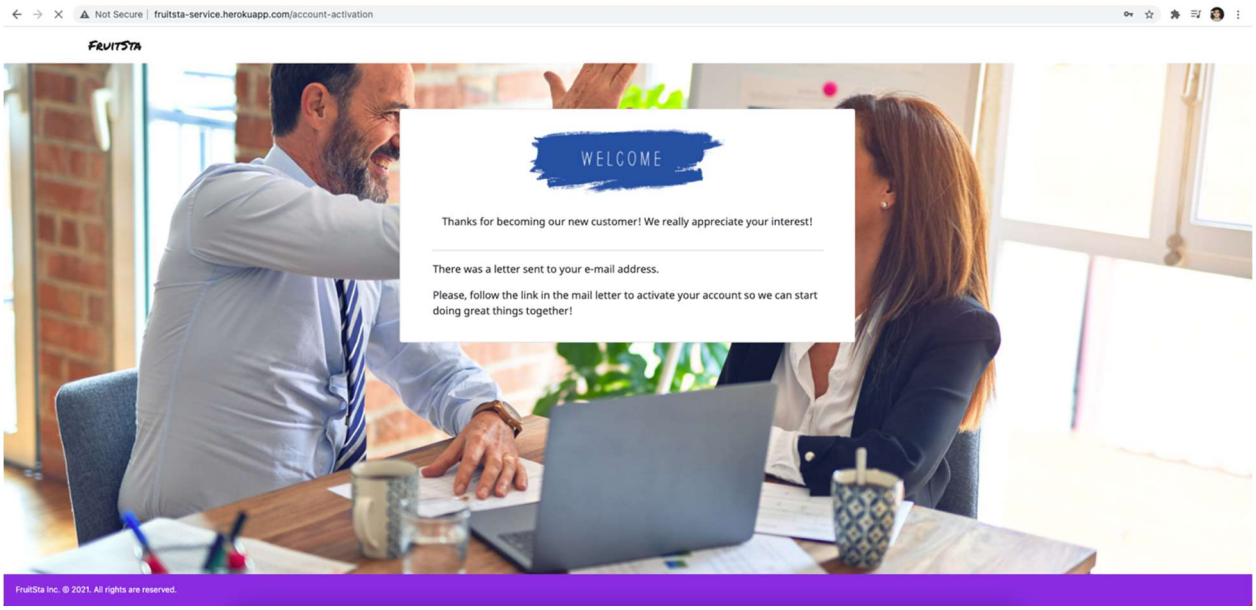


Рисунок 3.13 – Вікно повідомлення про потребу активації

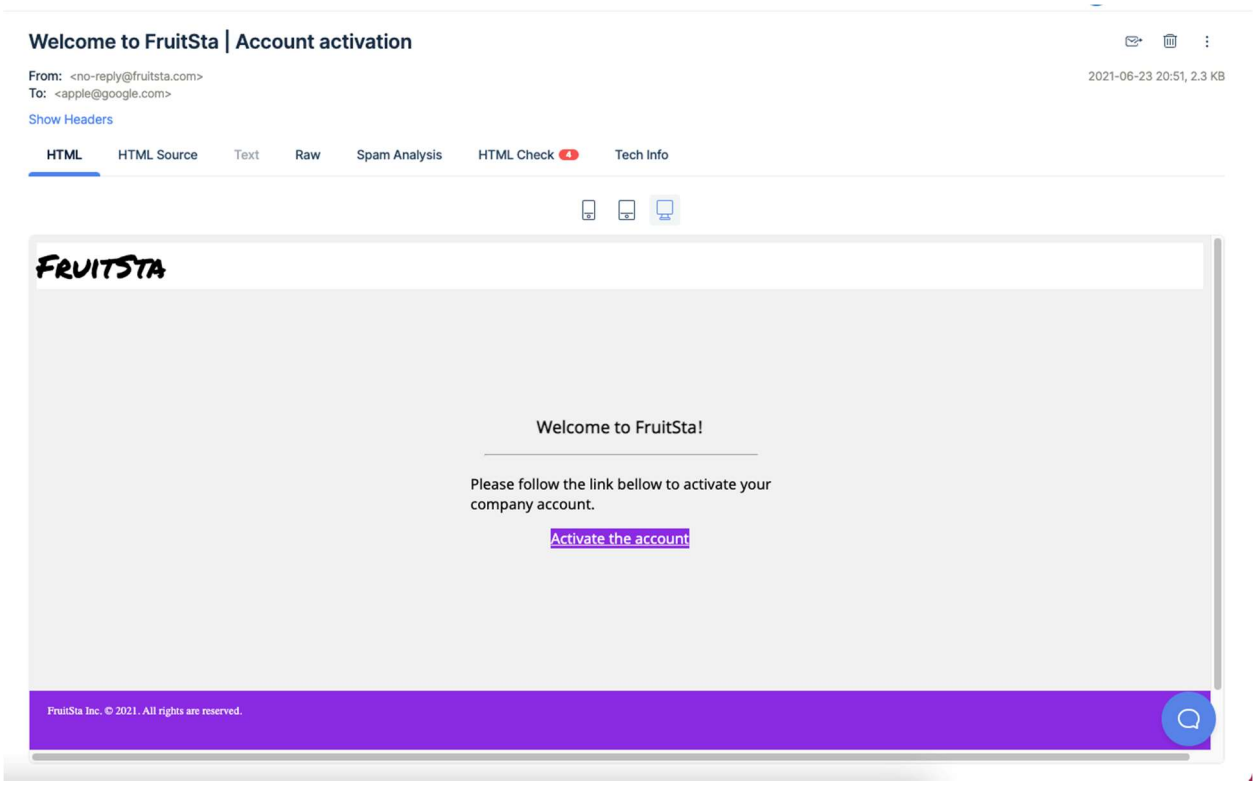


Рисунок 3.14 – Посилання на активацію облікового запису

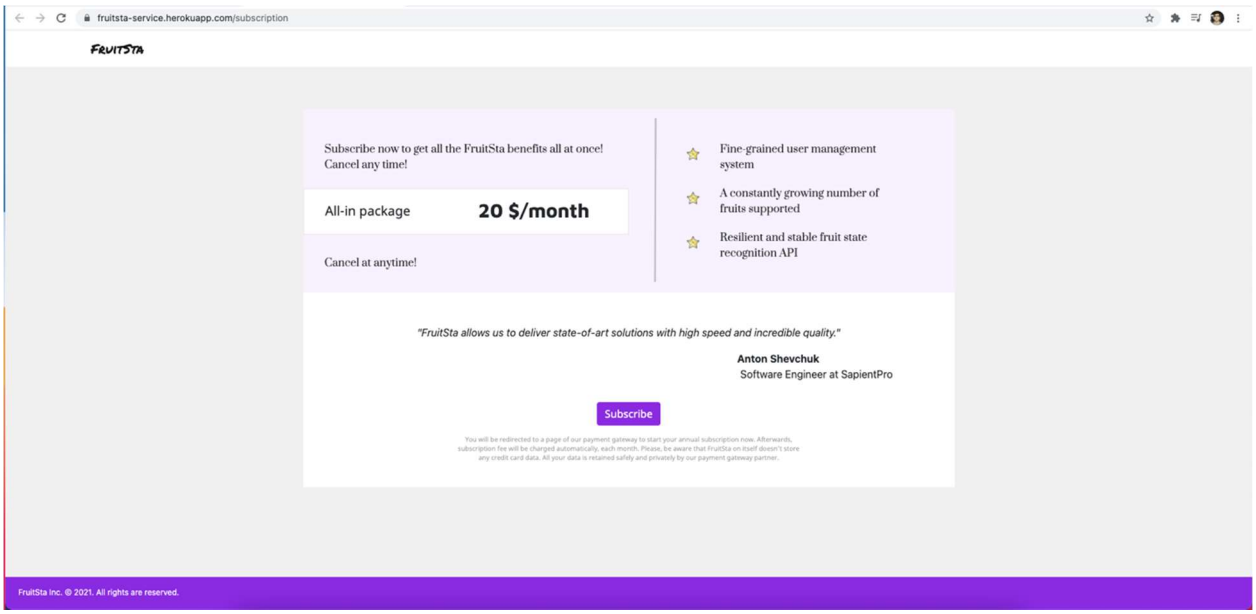


Рисунок 3.15 – Сторінки опису послуг підписки

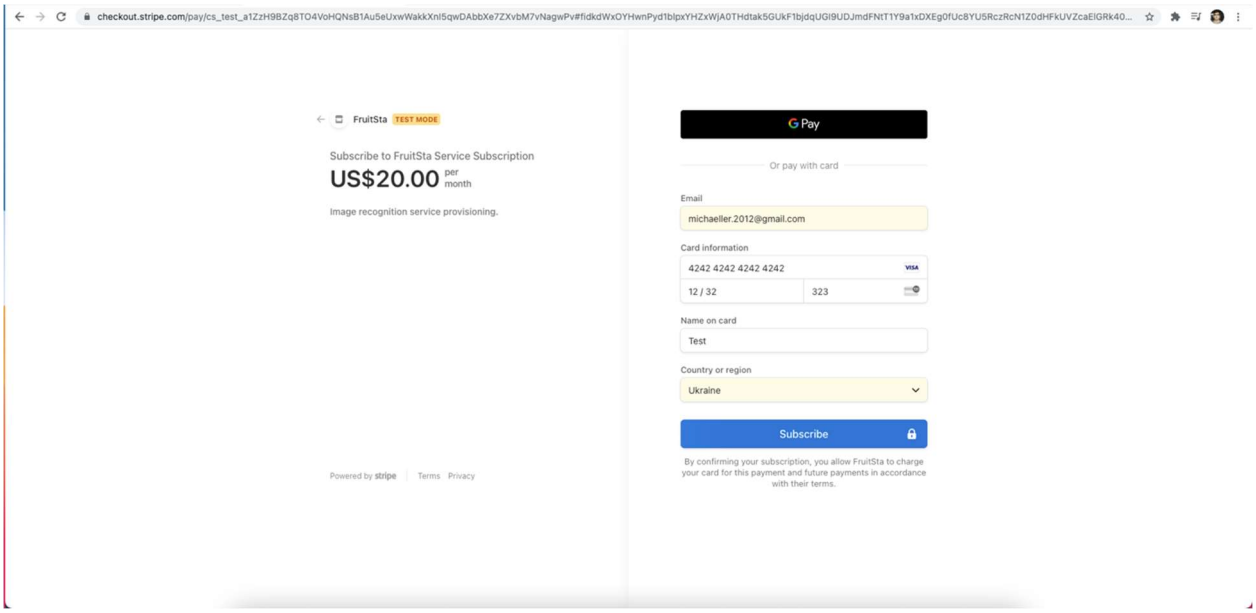


Рисунок 3.16 – Сторінка проведення оплати підписки

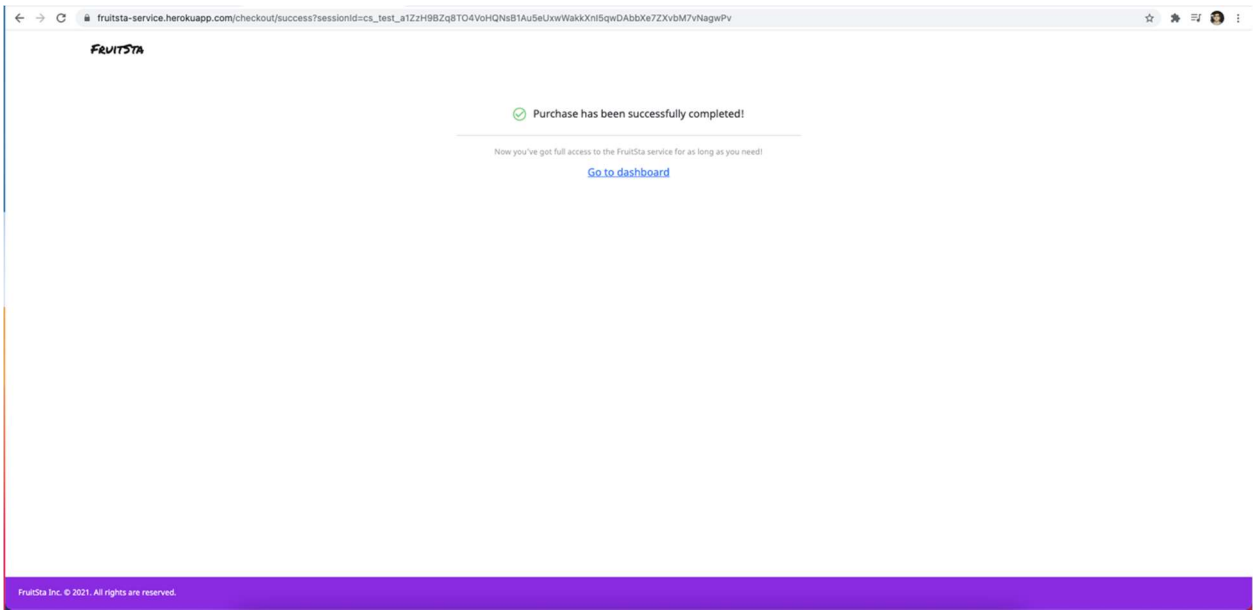


Рисунок 3.17 – Сторінка підтвердження первинної оплати підписки

Сценарії роботи root користувача передбачають можливості адміністрування account користувачів, що відображено на рисунках 3.18-3.22, та керування підпискою (рис. 3.24-3.26).

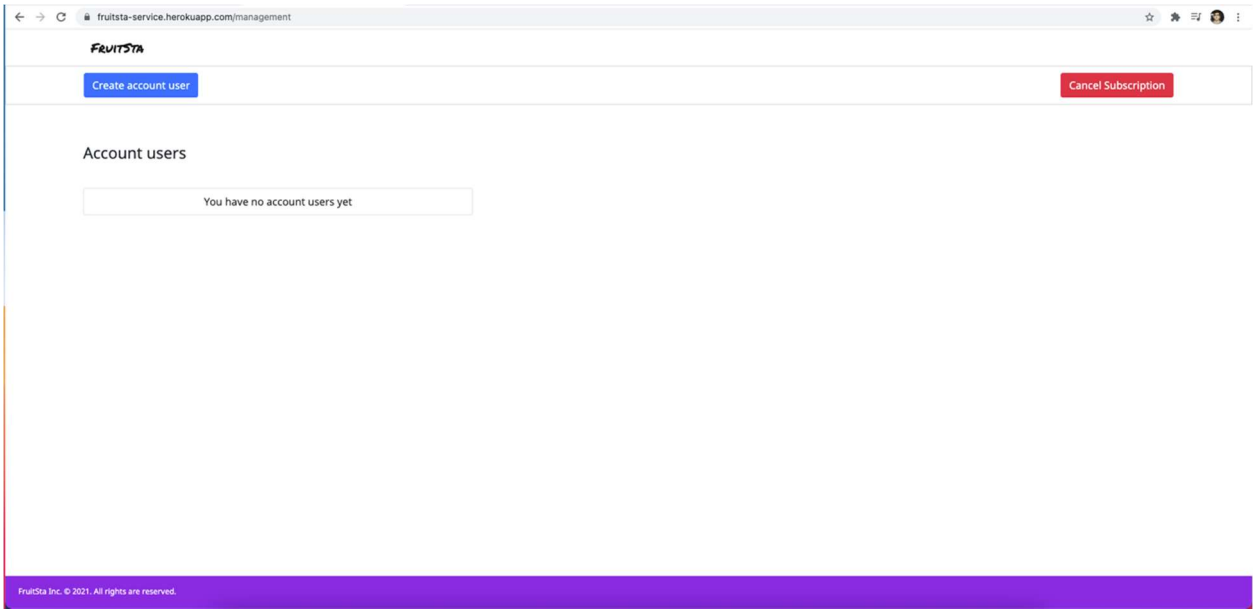


Рисунок 3.18 – Екран початку роботи root користувача

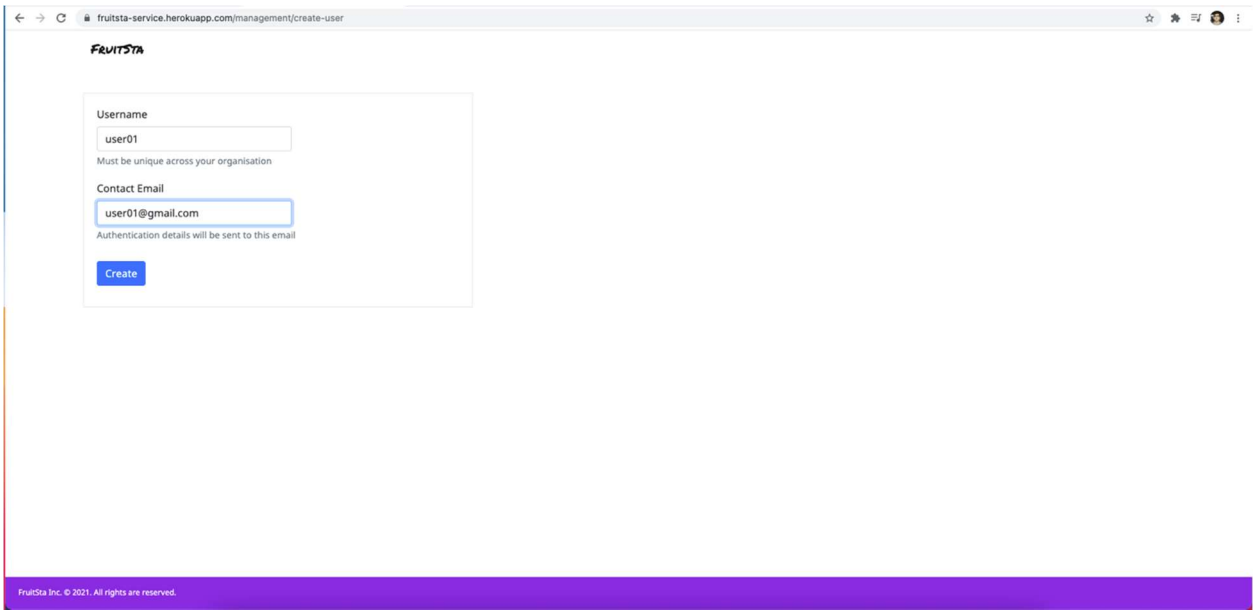


Рисунок 3.19 – Екран додавання account користувача

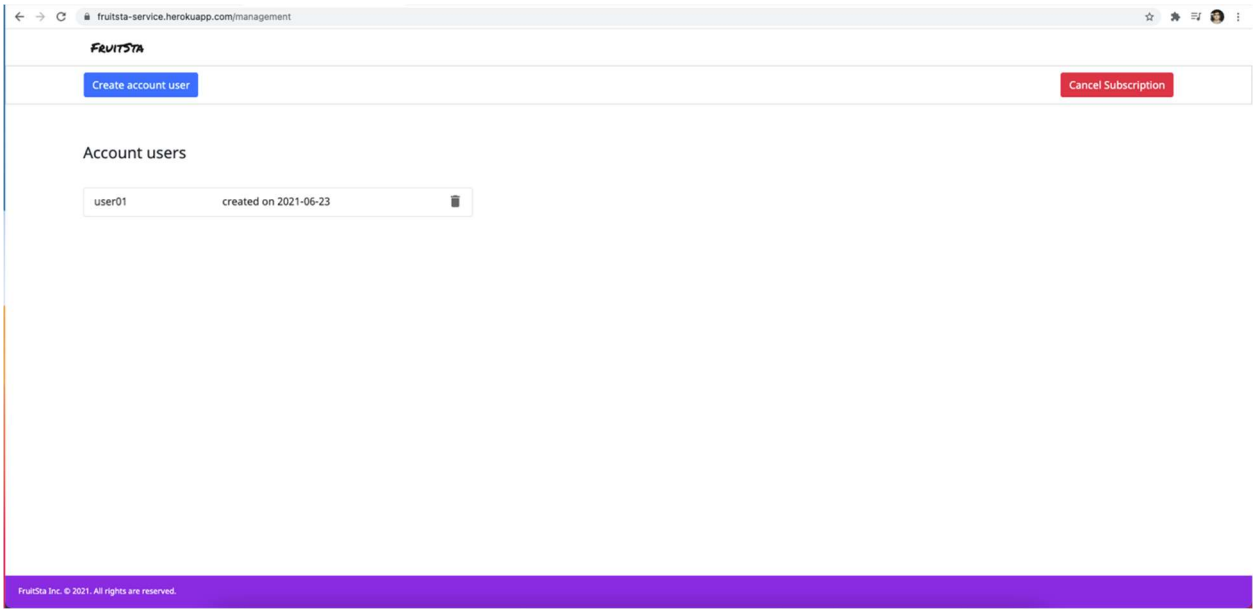


Рисунок 3.20 – Користувач з’явився у списку

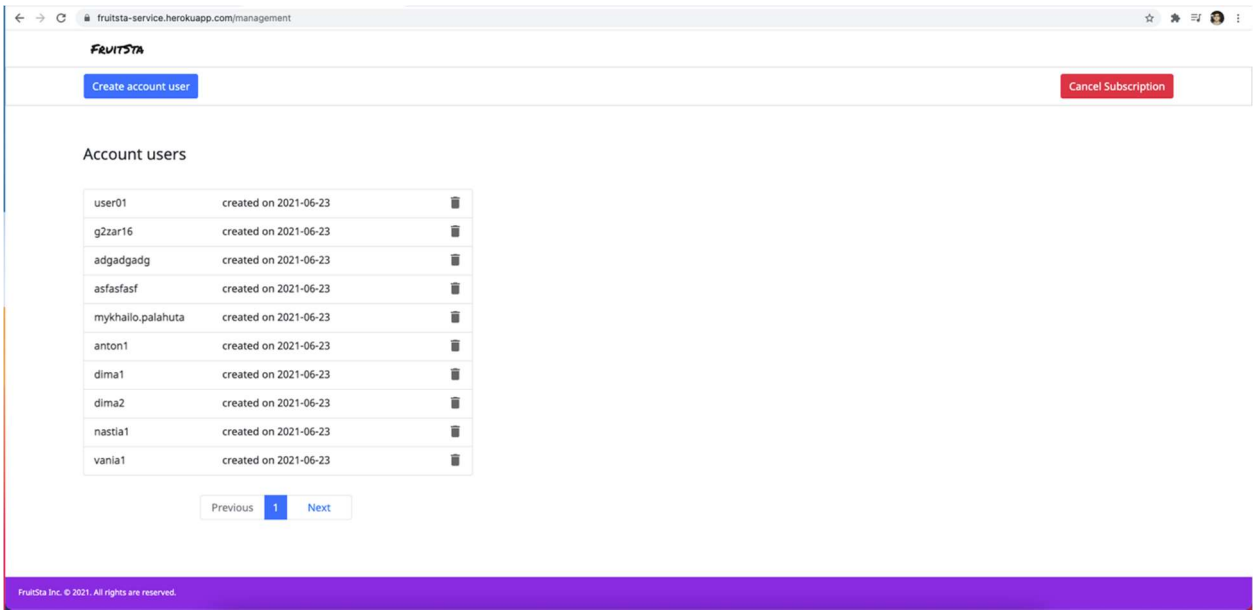


Рисунок 3.21 – Додано більше десяти користувачів

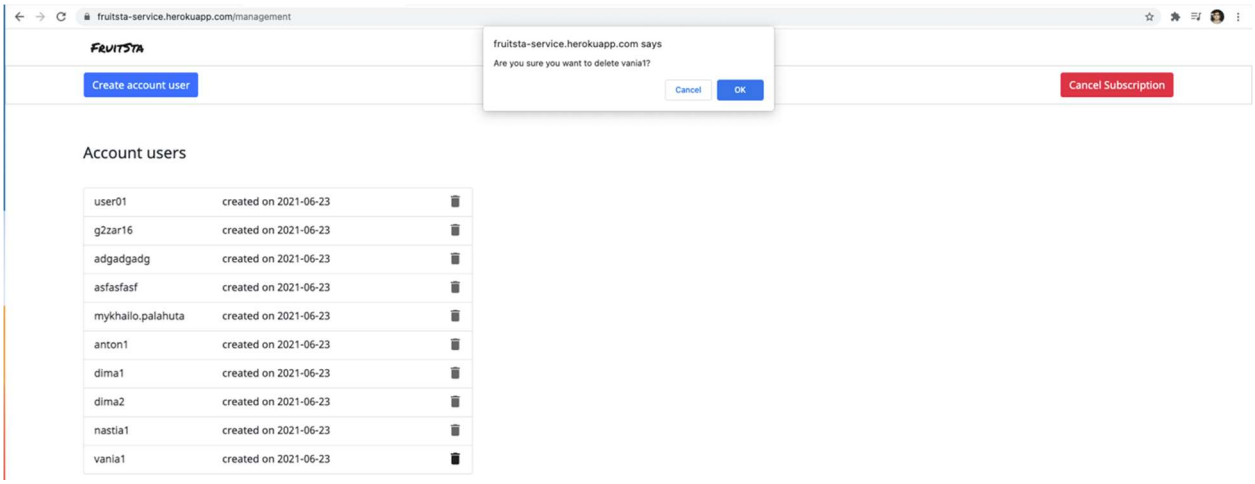


Рисунок 3.22 – Вікно видалення користувача

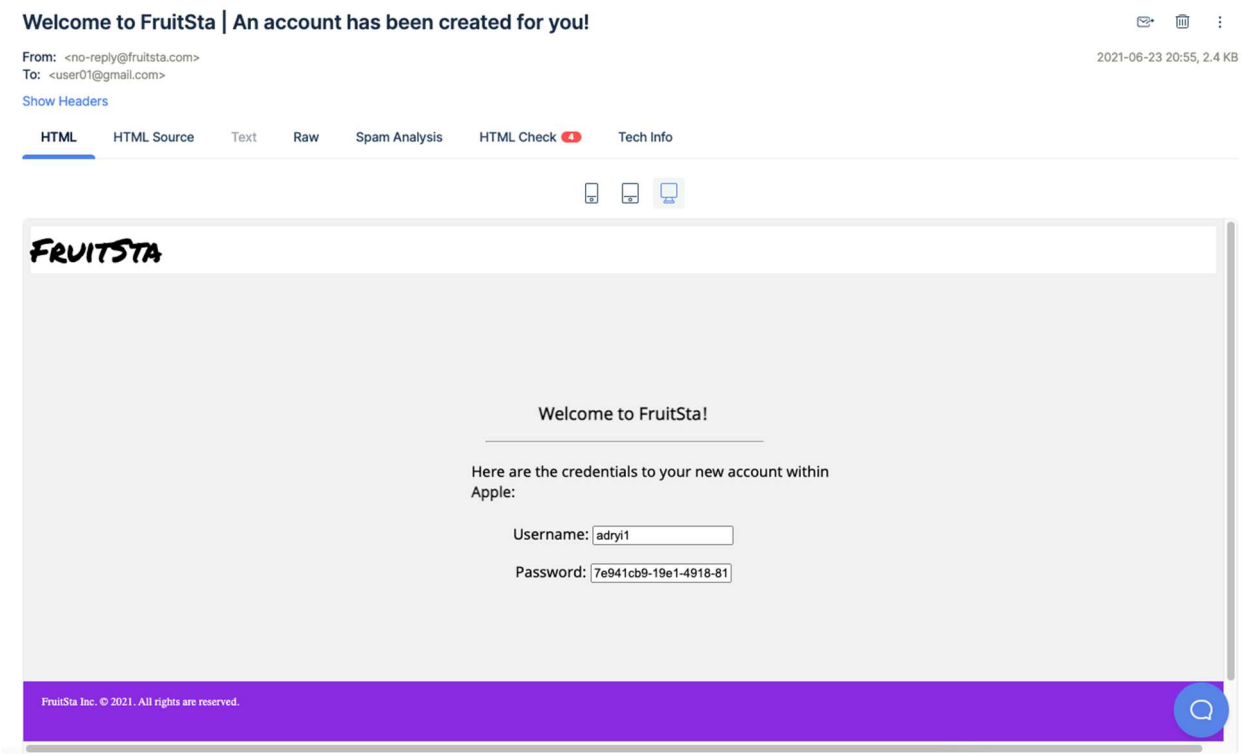


Рисунок 3.23 – Лист надісланий account користувачу

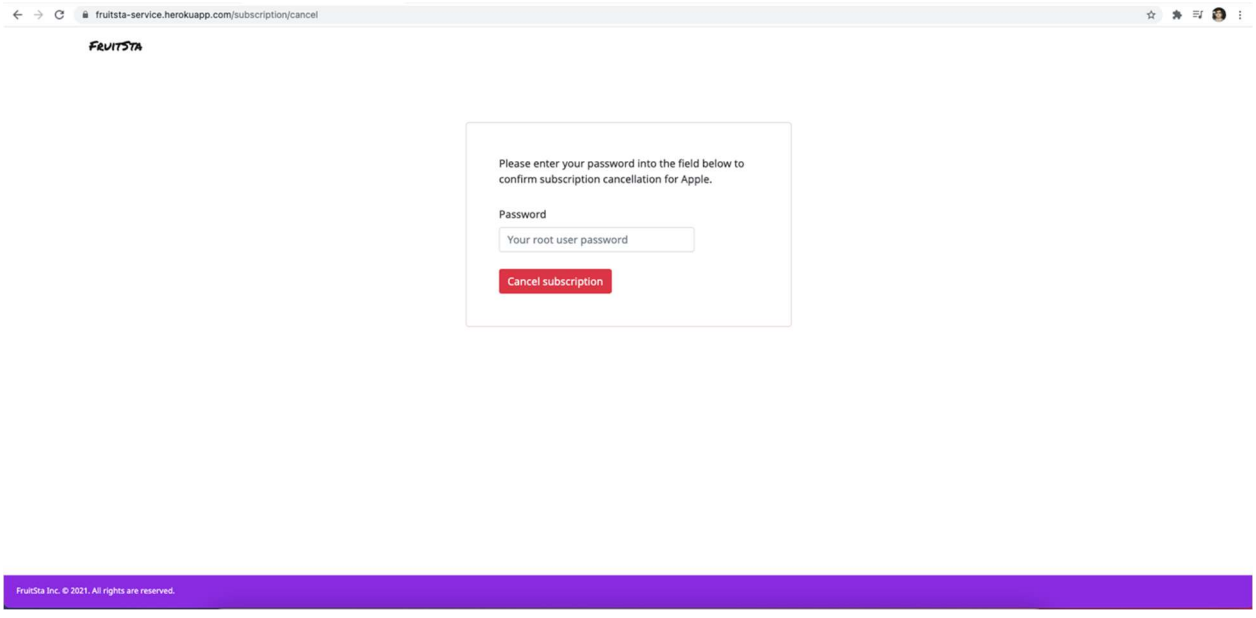


Рисунок 3.24 – Екран відміни підписки

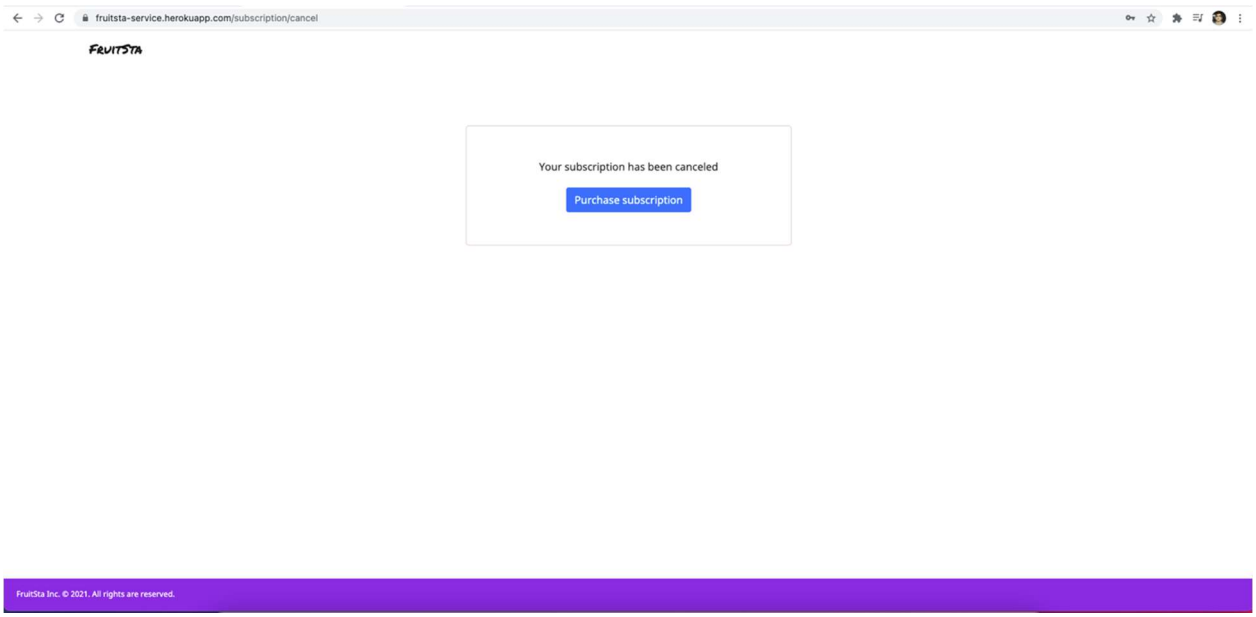


Рисунок 3.25 – Сторінка відображення успішної відміни підписки

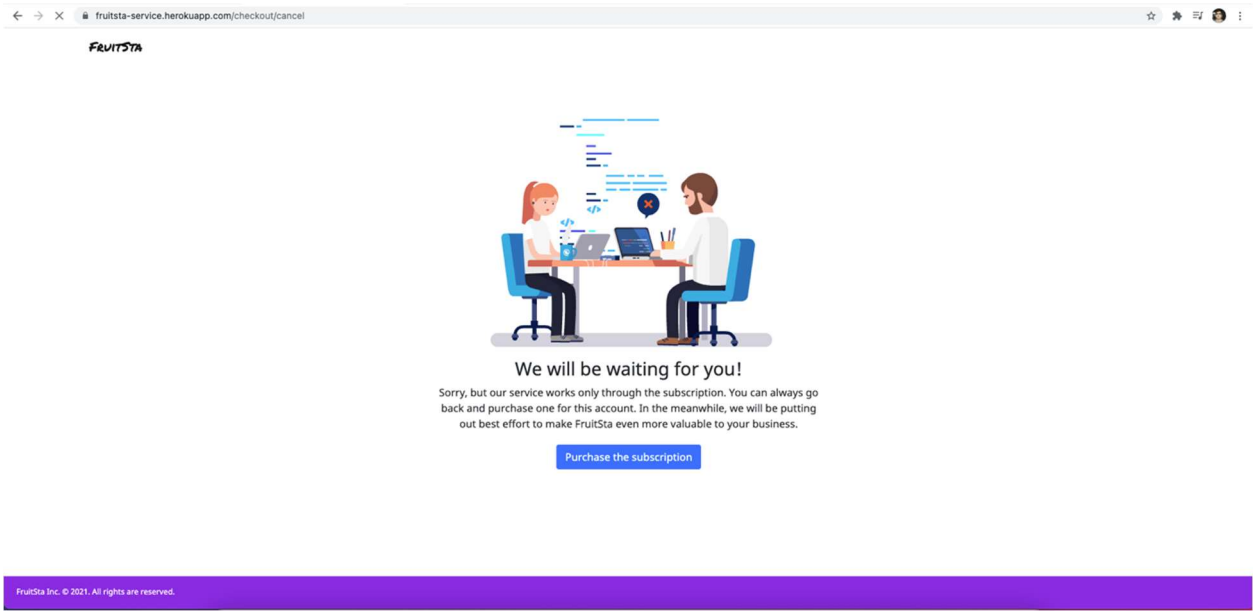


Рисунок 3.26 – Повідомлення клієнту, який відмінив першу оплату

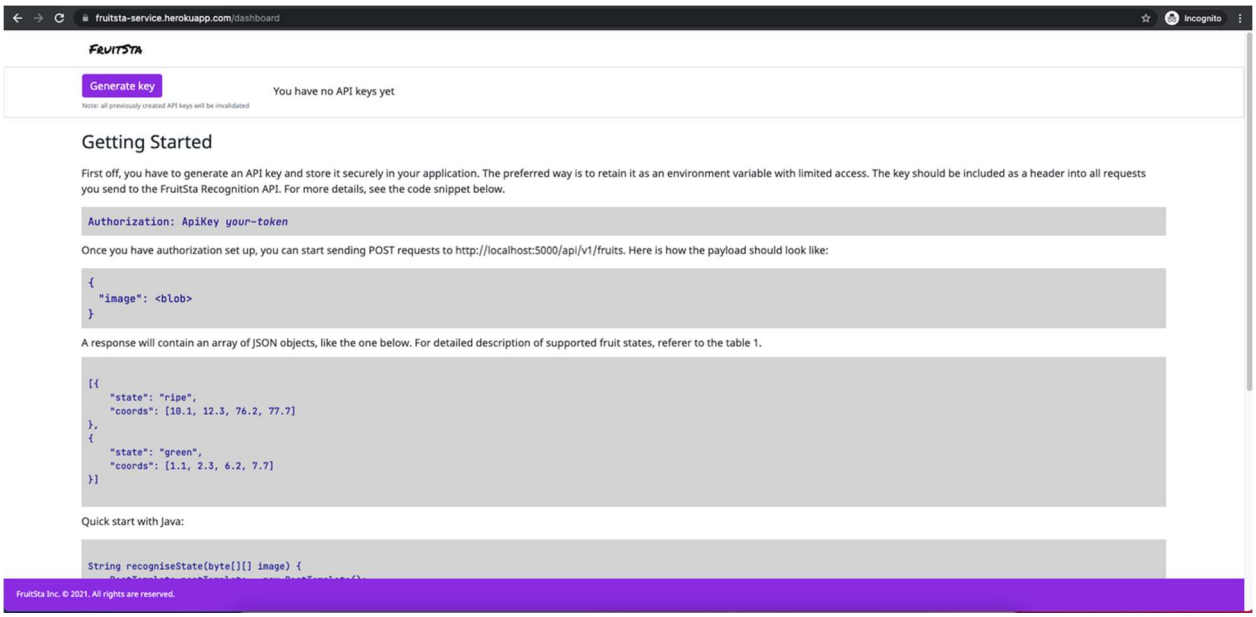


Рисунок 3.27 – Головна сторінка account користувача, верхня частина

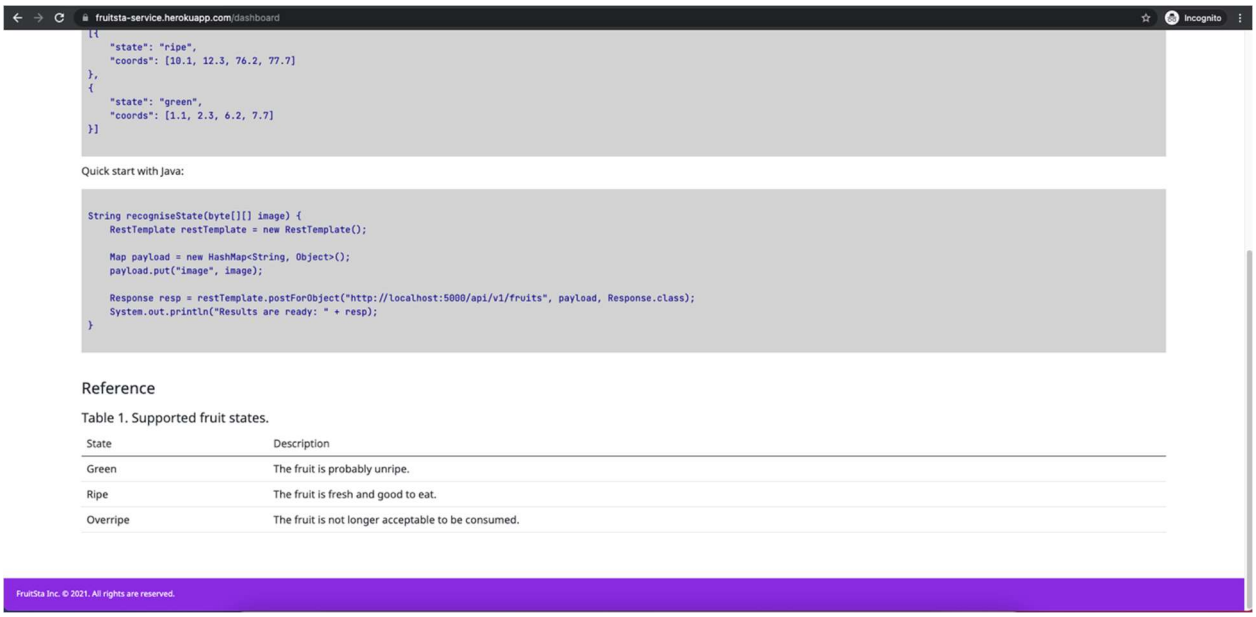
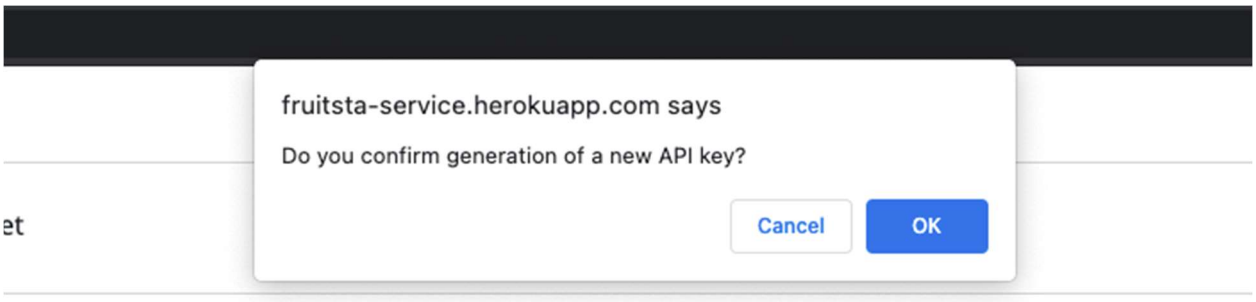


Рисунок 3.28 – Головна сторінка account користувача, нижня частина



in your application. The preferred way is to retain it as an environment variable with limited access
the code snippet below.

Рисунок 3.29 – Вікно підтвердження створення API ключа

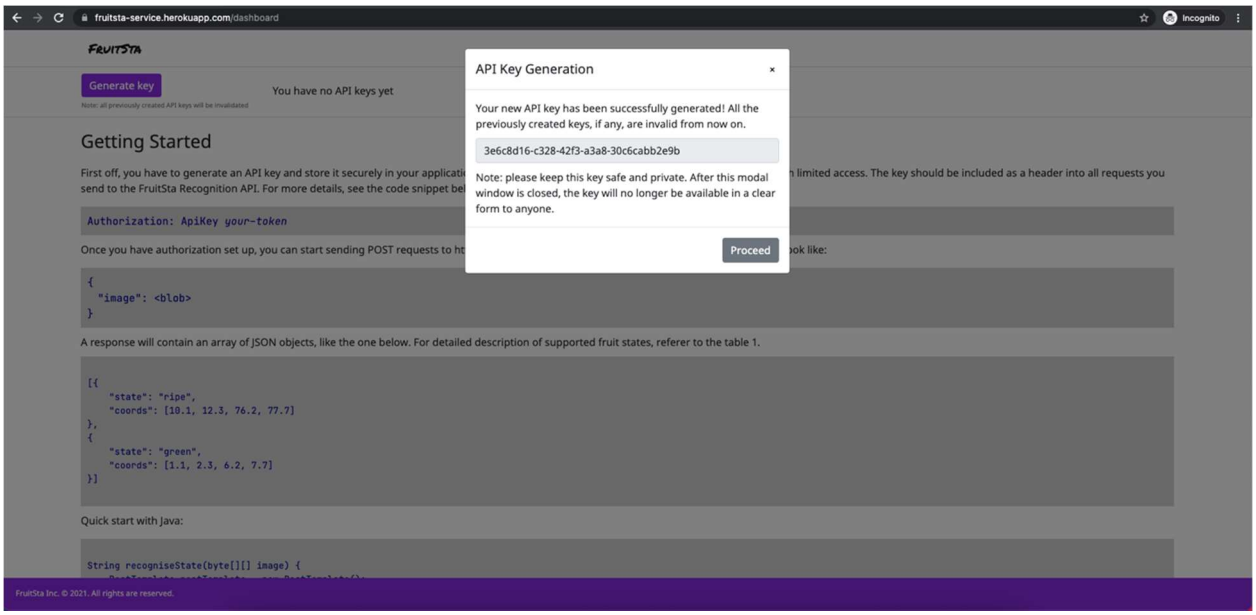


Рисунок 3.30 – Екран відображення згенерованого ключа

Інтерфейс account користувача надає функції генерування API ключів доступу (рисунки 3.29-3.30) та перегляду документації розробника (рисунки 3.27-3.28). Надходження даних входу для account користувачів відображено на рисунку 3.23.

Роботу підсистеми розпізнавання продемонструю на прикладі аналізу стану яблук на складному зображенні дерева з плодами. Вхідне зображення представлене на рисунку 3.31. Першим кроком, зображення опрацьовується сегментаційною мережею Mask R-CNN, яка видає проміжний результат роботи підсистеми (рис. 3.32). Далі кожен окремий сегмент, який відповідає окремому яблуку з зображення, передається CNN нейромережі для остаточної класифікації стану фруктів та формування результату, який відображений на рисунку 3.33. Звіривши стани та координати з результатів з вхідним зображенням ми побачимо, що результат класифікації є повністю коректним.

Рисунок 3.31 – Зображення зелених яблук на дереві

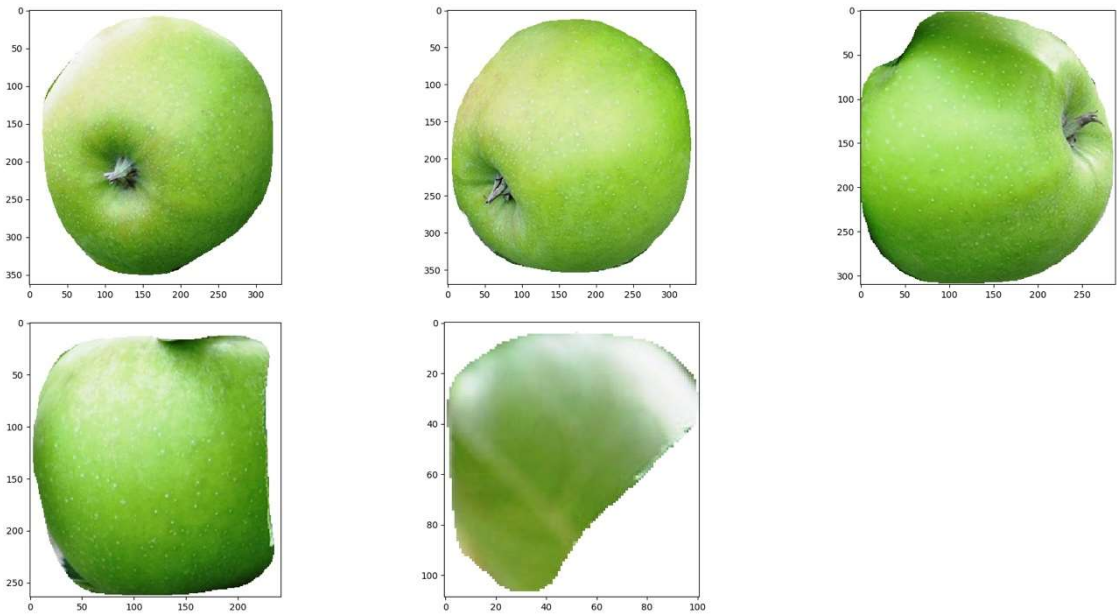


Рисунок 3.32 – Проміжний результат, набір знайдених яблук

```
2021-06-24 00:02:12.571112: W tensorflow/core/framework/cpu_allocator_impl.cc:81] Allocation of 792985600 exceeds 10% of free system memory.
2021-06-24 00:02:12.809919: W tensorflow/core/framework/cpu_allocator_impl.cc:81] Allocation of 792985600 exceeds 10% of free system memory.
2021-06-24 00:02:12.931489: W tensorflow/core/framework/cpu_allocator_impl.cc:81] Allocation of 792985600 exceeds 10% of free system memory.
2021-06-24 00:02:13.153683: W tensorflow/core/framework/cpu_allocator_impl.cc:81] Allocation of 792985600 exceeds 10% of free system memory.
2021-06-24 00:02:13.226857: W tensorflow/core/framework/cpu_allocator_impl.cc:81] Allocation of 792985600 exceeds 10% of free system memory.
Координати: y=67, x=989
Стан = green
Координати: y=932, x=955
Стан = green
Координати: y=856, x=1294
Стан = green
Координати: y=713, x=685
Стан = green
Координати: y=500, x=860
Стан = green

Process finished with exit code 0
```

Рисунок 3.33 – Результати аналізу вхідного зображення

3.3.2 Аналіз критеріїв якості

Головним показником якості системи є її відповідність до поставлених перед нею вимог. На мій погляд, всі функціональні та нефункціональні вимоги системи є задовільненими, адже система показує себе як надійна, відмовостійка та безпечна. Програмний продукт дозволяє контролювати та проводити аудит користувачів на рівні кожного окремого працівника компанії клієнта, реалізовує комплексну інтеграцію з платіжним сервісом Stripe та забезпечує повну узгодженість даних системи. Таким чином, підсистема надання послуг, серверний додаток продукту, однозначно заслуговує високу оцінку якості.

Розробка підсистеми розпізнавання ускладнилась наявністю проблеми різкого перепаду й сильної відмінності проявів станів фруктів. Так, наприклад, відрізнити зіпсоване яблуко від зеленого або червоного є набагато важче ніж відрізнити червоне яблуко від зеленого (рис. 3.34). А відрізнити зіпсований банан від іншого стану банану набагато легше ніж відрізнити зелений банан від спілого (рис. 3.35).

Рисунок 3.34 – Градієнт станів яблука

Рисунок 3.35 – Градієнт станів банана

Задля вирішення даних проблем, використовувалось додавання гаусового шуму до вхідних зображень та постійне розширення дата сету.

Як результат, точність класифікації зелених та червоних яблук виявилась близькою до максимальної, тоді як точність виявлення зіпсованих яблук досягла рівня 60-70%. Через розширення дата сету, класифікація бананів повністю позбавилась проблем неоднозначності й видає стабільну точність біля 90%.

Варто зазначити, що дана задача є новою й унікальною і не була роз’язана досі. Спільнота машинного навчання й штучного інтелекту має розв’язки визначення того чи деякі з фруктів є свіжими або зіпсованими, але точне визначення стану фрукту в межах трьох значень не було опублікованим в жодному з відомих мені джерел.

Враховуючи всі перспективи й погляди на рішення, я вважаю підсистему розпізнавання проектом високої якості, адже дана робота включила збір нового й ефективного дата сету станів фруктів, виявлення проблем та методів боротьби з ними та успішні результати для деяких типів фруктів, що визначає набір інструментів й способів подальшого розширення можливостей даної підсистеми.

Висновки до розділу 3

Побудова складних Інтернет сервісів з надання послуг аналізу зображень є непростою задачею й потребує рішення нестандартних проблем. В даному розділі, я продемонстрував свої результати – розподілену інформаційну системи, яка обслуговує як клієнтів так і їх додатки. З погляду поставлених перед проектом вимог, я оцінюю дану роботу як продукт високої якості. Наявність повного циклу розробки та процесів мануального й автоматичного тестування забезпечили надійність роботи всіх функцій системи. Важливу роль також відіграли Continues Delivery методологія в комбінації з хмарним сервісом Heroku, які стали основою для повноцінної інтеграції з платіжним сервісом Stripe.

ВИСНОВКИ

Метою даної кваліфікаційної роботи була побудова Інтернет платформи з надання послуг розпізнавання станів фруктів на зображеннях. Основними завданнями було визначено досягнення надійності й розширюваності інформаційної системи.

Засобами технологій розробки програмних продуктів Java та Spring Framework було розроблено та перевірено веб-додаток, сервер платформи, який обслуговує клієнтів та забезпечує інтеграцію послуг в їхні додатки. При цьому була розроблена повноцінна, стійка інтеграція з платіжним сервісом Stripe.

Завдання розпізнавання було вирішено за допомогою оригінальних рішень від початку й до кінця всіх етапів створення системи. В результаті, було зібрано ефективний та простий дата сет фруктів для тренування розпізнавання трьох станів – зелений, спілий та зіпсований. Також було розроблено методику побудови та навчання нейронних мереж призначених для задачі класифікації фруктів, яку було успішно застосовано. Для покращення результатів аналізу було залучено засоби попередньої обробки зображення, а саме, сегментації.

Отже кваліфікаційна робота «Інформаційна система з розпізнавання стану фруктів» пропонує унікальну основу для побудови глобального IT бізнесу, який сприятиме розвитку інноваційних вирішень в сферах робототехніки, автоматизації та соціальних сферах. Головною перевагою даної розробки є простота й дешевизна інтеграції в клієнтські системи без втрат надійності, якості та безпеки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Image Segmentation for Fruit Detection and Yield Estimation in Apple Orchards / Suchet Bargoti, James Underwood // Архів Університету Корнелла [сайт]. – Режим доступу: https://arxiv.org/abs/1610.08120, вільний. – Назва з екрану.
2. Train Neural Networks With Noise to Reduce Overfitting / Jason Brownlee // Machine Learning Mastery [електронний ресурс]. – Режим доступу: https://machinelearningmastery.com/train-neural-networks-with-noise-to-reduce-overfitting/, вільний. – Назва з екрану.
3. Computer Vision Tutorial: A Step-by-Step Introduction to Image Segmentation Techniques (Part 1) / Pulkit Sharma // Analytics Vidhya [електронний ресурс]. – Режим доступу: https://www.analyticsvidhya.com/blog/2019/04/introduction-image-segmentation-techniques-python/, вільний. – Назва з екрану.
4. Mask R-CNN for Object Detection and Segmentation / Waleed Abdulla // matterport [електронний ресурс]. – Режим доступу: https://github.com/matterport/Mask_RCNN, вільний. – Назва з екрану.
5. A Gentle Introduction to the Rectified Linear Unit (ReLU) / Jason Brownlee // Machine Learning Mastery [електронний ресурс]. – Режим доступу: https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/, вільний. – Назва з екрану.
6. Шапиро Л. Компьютерное зрение : [Пер. с англ.] / Л. Шапиро, Дж. Стокман. – М.: Бином. Лаборатория знаний, 2006. – 752 с.
7. Форсайт Д. Компьютерное зрение. Современный подход / Д. Форсайт, Ж. Понс. – М.: Вильямс, 2004. – 928 с.
8. Глибовець М. М. Штучний інтелект : підруч. [для студ. вищ. навч. закл.] / М. М. Глибовець, О. В. Олецький. – К. : КМ Академія, 2002. – 336 с.