**Пояснювальна записка  
до курсової роботи**

на тему: Система для визначення логотипів у зображеннях

КПІ.ІХ-0XXX.045440.02.81

Київ – 2022

Зміст

[Перелік умовних позначень 4](#_Toc115992741)

[Вступ 5](#_Toc115992742)

[1 АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 6](#_Toc115992743)

[1.1 Загальні положення 6](#_Toc115992744)

[1.2 Змістовний опис і аналіз предметної області 8](#_Toc115992745)

[1.3 Аналіз існуючих технологій та успішних IT-проєктів 10](#_Toc115992746)

[1.3.1 Аналіз відомих алгоритмічних та технічних рішень 11](#_Toc115992747)

[1.3.2 Аналіз допоміжних програмних засобів та засобів розробки 12](#_Toc115992748)

[1.3.3 Аналіз відомих програмних продуктів 14](#_Toc115992749)

[1.4 Аналіз вимог до програмного забезпечення 17](#_Toc115992750)

[1.4.1 Розроблення функціональних вимог 22](#_Toc115992751)

[1.4.2 Розроблення нефункціональних вимог 25](#_Toc115992752)

[1.5 Постановка задачі 26](#_Toc115992753)

[Висновки до розділу 28](#_Toc115992754)

[2 МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 29](#_Toc115992755)

[2.1 Моделювання та аналіз програмного забезпечення 29](#_Toc115992756)

[2.2 Архітектура програмного забезпечення 29](#_Toc115992757)

[2.3 Конструювання програмного забезпечення 29](#_Toc115992758)

[2.4 Аналіз безпеки даних 31](#_Toc115992759)

[Висновки до розділу 31](#_Toc115992760)

[3 АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 32](#_Toc115992761)

[3.1 Аналіз якості ПЗ 32](#_Toc115992762)

[3.2 Опис процесів тестування 32](#_Toc115992763)

[3.3 Опис контрольного прикладу 33](#_Toc115992764)

[Висновки до розділу 33](#_Toc115992765)

[4 ВПРОВАДЖЕННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 34](#_Toc115992766)

[4.1 Розгортання програмного забезпечення 34](#_Toc115992767)

[4.2 Підтримка програмного забезпечення 34](#_Toc115992768)

[Висновки до розділу 35](#_Toc115992769)

[ВИСНОВКИ 36](#_Toc115992770)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 37](#_Toc115992771)

Перелік умовних позначень

|  |  |  |
| --- | --- | --- |
| IDE | – | Integrated Development Environment – інтегроване середовище розробки. |
| API | – | Application programming interface, прикладний програмний Інтерфейс |
| SDK | – | Software development kit |
| IT | – | Інформаційні технології |
| ER | – | Entity-Relation diagram |
| OC | – | Операційна система. |
| БД | – | База даних. |
| CV | – | Computer Vision – Комп’ютерний зір |
| ШІ | – | Штучний інтелект |
| R-CNN |  |  |
| YOLO |  |  |
| SSD |  |  |
| FPS |  |  |

Вступ

<У вступі стисло викладають:

актуальність роботи та підстави для її виконання; світові тенденції розв’язання поставлених проблем і/або завдань; оцінку сучасного стану об’єкта розробки, розкриваючи практично розв’язані завдання провідними науковими установами та організаціями, а також провідними вченими й фахівцями певної галузі; можливі сфери застосування. 1 стр. чи більше>

# АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Загальні положення

В сучасному світі стрімко розвивається галузь програмного забезпечення, пов'язана з технологіями комп'ютерного зору (Computer Vision або СV), яка визначається своєрідною спробою надати комп'ютерам здатність аналізувати та розуміти зображення так, як це робить людський мозок. Комп'ютерний зір є галуззю штучного інтелекту і в основі його систем, як правило, лежать алгоритми, засновані на машинному навчанні, за допомогою яких вони навчаються розрізняти одні об'єкти від інших, виявляти патерни та закономірності. [1] По суті, завдання комп’ютерного зору стосуються того, щоб змусити комп’ютери розуміти цифрові зображення, а також візуальні дані з реального світу.

З початку розробки перших систем у галузі CV пройшло вже немало часу, тому дослідники умовно розподілили усі можливі задачі з розпізнавання на декілька ключових груп, що допомогло краще визначати проблеми та ефективно працювати над ними. Найпопулярніші завдання комп’ютерного зору включають: класифікацію, детекцію, сегментацію, відслідковування об’єктів, розпізнавання обличь та визначення пози. [2] Розробка систем для розпізнавання логотипів в сучасному світі ґрунтується на досягненнях у сфері детекції об'єктів, оскільки саме ці технології найкраще відповідають вирішенню цієї задачі.

Детекція об’єктів, як випливає з назви, відноситься до виявлення та локалізації предметів за допомогою обмежувальних рамок. Її головною метою є пошук об’єктів на зображенні чи відео та їх ідентифікація щоразу, коли вони з’являються. Об’єктами для розпізнавання може бути будь-що: автомобілі, тварини, люди та інше, на якому була навчена модель виявлення.

Моделі виявлення об'єктів потребують навчання. Це означає процес створення глибокої нейронної мережі, яка вивчає особливості зображення та об’єктів на ній. Ця нейронна мережа отримує на вхід зображення з об'єктами в різних сценаріях (тобто з різними фонами, об'єктами під різними кутами) та розміченими позначками, що відповідають об'єкту та його місцезнаходженню. Зображення для моделей можуть маркуються або вручну, використовуючи інструмент для маркування комп'ютерного зору, або за допомогою інших попередньо навчених рішень. Чим точніші анотації і чим більш релевантними є зображення для навчання нейромережі, тим краще модель виконуватиме завдання. Далі відбувається процес тренування, де модель запам’ятовує основні відмінності й критерії кожного окремого класу об’єкта. Ці особливості вивчаються та кодуються в "вагах" та "відхиленнях", які зберігаються після навчання для використання під час тестування та валідації. Після цього йде етап імплементації готової нейромережі в кінцевий продукт. [3]

Незважаючи на прогрес у цій області, існують деякі проблеми, з якими стикаються системи для визначення об’єктів. До найбільших відносяться низька якість та різноманітність зображень для початкового датасету та складність у роботі з великими масштабами даних та необхідність постійного оновлення моделей для врахування нових логотипів.

Загалом всі сучасні системи, що знаходять логотипи на зображеннях побудовані саме на цих моделях і їхня робота загалом визначається 2 основними критеріями: архітектурою підібраної глибокої нейронної мережі та якістю й відповідністю вимогам датасету.

Актуальність побудови систем для визначення логотипів виходить з необхідності в ефективному виявленні логотипів в різноманітних сценаріях, починаючи від аналізу соціальних мереж та медіа-застосунків, і закінчуючи перевіркою системами відеоспостереження. Найочевидніший випадок використання виявлення логотипів — це моніторинг бренду. Більшість користувачів соціальних мереж все більше віддають перевагу візуальному контенту на різних платформах, тому не лише рекомендується, але це абсолютно необхідно, щоб власники брендів могли аналізувати використання зображень та відео. З використанням таких систем менеджери бренду зможуть знайти набагато більше згадок бренду в соціальних мережах, якщо вони мають можливість сканувати та аналізувати візуальний контент зі своїми логотипами. Це допоможе краще керувати маркетинговою стратегією та управлінням репутації компанії. [4]

Також не менш важливим є завдання захисту авторських прав та розміщенню оголошень. Виявлення логотипів за допомогою комп’ютерного зору може відігравати важливу роль у запобіганні порушенням авторських прав на таких платформах. Власники брендів та торгових марок з використанням моделі для виявлення їхніх логотипів зможуть з легкістю відслідковувати використання їхньої символіки у відкритих джерелах та реагувати на її неправомірне використання. В даному випадку виявлення логотипів — єдиний спосіб досягнення цього.

Тим не менш, розпізнавання логотипів у поєднанні з можливістю аналізу відео в реальному часі зробили революцію в моніторингу спонсорства, особливо в сфері спортивного спонсорства. Сучасні спеціально розроблені API моніторингу спонсорства включає виявлення логотипів для надання точних даних про вплив появи брендів на спортивних подіях, особливо в поєднанні з можливістю визначення конкретних місць розміщення цих логотипів.

Ще один доволі цікавий випадок використання моделей для розпізнавання був застосований для подолання цифрового піратства. Раніше було майже неможливо впоратися зі зростанням кількості нелегальних потоків, доступних в Інтернеті, однак виявлення логотипів є однією з технологій, яка частково допомогла виявляти сайти з незаконно розміщеним контентом. Головним задумом була розробка моделі, яка зможе виявляти контент, що був позначений спеціальними водяними знаками чи іншими позначками. Таким чином це допомогло запобігти величезній втраті прибутків. [5]

## Змістовний опис і аналіз предметної області

Системи для визначення логотипів на зображеннях будуються за допомогою використання технологій розпізнавання об’єктів із застосуванням глибинного навчання. Як було зазначено вище основні критерії для таких систем являються якість початкового датасету та архітектура нейронної мережі, що використовується. Традиційно виявлення логотипу розглядається як задача виявлення об’єкта, у якій система навчається для попередньо визначеного набору класів логотипів. Очевидно, що такий підхід до побудови системи буде дуже точним, оскільки для кожного логотипу окремо будуть виділятися певні ознаки, за якими модель буде розпізнавати приналежність картинки до певного класу логотипу. Подібні детектори будують на базі таких архітектур як RCNN чи Faster R-CNN [6]. Проте не дивлячись на високу точність в таких рішеннях присутні суттєві недоліки:

* Обмежена кількість класів для розпізнавання. Модель такого типу не здатна розпізнавати ті об’єкти, які не були використані під час тренування.
* Низький рівень масштабування та розширення. Багато що в цьому пункті залежить від архітектури обраної мережі, оскільки вона впливає на максимальну кількість класів, які може розпізнавати модель одночасно. Зазвичай це число не перевищує 500 [7] класів, що інколи буває замало для певних задач.
* Складність збірки датасету для початкового навчання. Цей підхід потребує як мінімум 20-25 об’єктів одного класу на зображеннях з різними умовами й освітленістю. Звідси випливає, що необхідно близько 6000 унікальних зображень для того щоб навчити модель розпізнавати 300 логотипів, що в більшості являється дуже складним.
* Додаткове перетренування нейромережі для збільшення кількості класів розпізнавання
* Необхідність використання великого обсягу пам’яті та потужне обладнання для швидкого тренування

Для того щоб позбутися всіх цих недоліків відносно недавно впровадили новий підхід у розпізнаванні логотипів – «open-set» logo detection. [8] Головну мету цього методу було взяти з алгоритмів розпізнавання обличчя, а саме двоетапне розпізнавання. Перший етап - виявлення логотипу (рис. 1.1) без встановлення приналежності до класу за допомогою загального детектора логотипів, який повинен мати високий recall [9], але може мати низьку точність.

На другому етапі визначення логотипу, інша модель порівнює регіони кандидатів на логотип з набором всіх канонічних зображень логотипів для всіх класів логотипів, які повинні бути визнані. Модель на 2 етапі навчено призначати високий бал правильним збігам і низький бал хибним збігам. Виявлені області логотипів позначаються тим класом логотипу, який має найвищий бал.

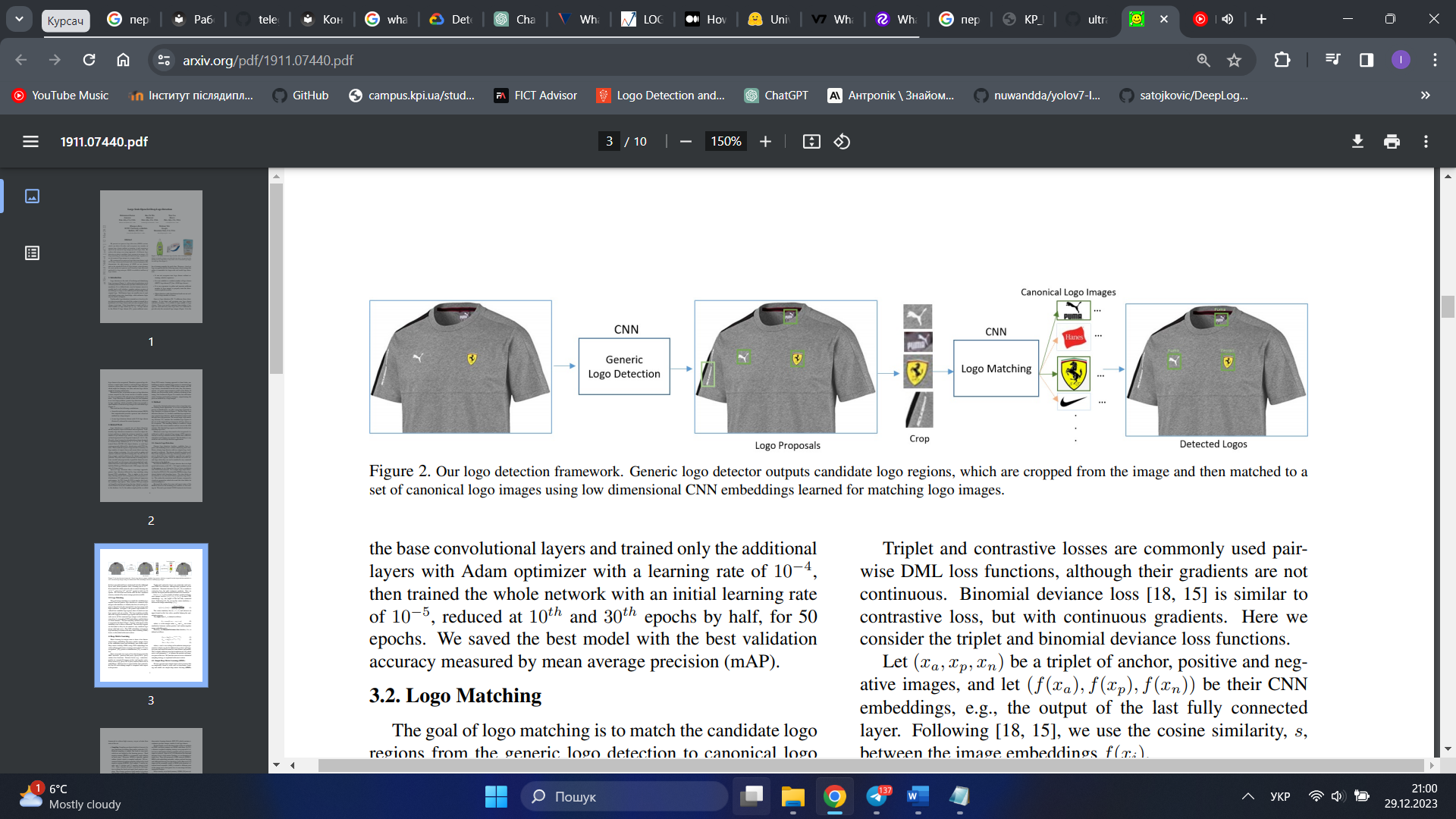


Рисунок 1.1. – Структура двоетапного детектора логотипів

Надалі кожного разу, коли потрібно визначити новий клас логотипу, достатньо додати його канонічні зображення до бази даних логотипів, і жодного подальшого збору даних, анотацій і тренування не потрібно.

Отже, «open-set» logo detection підхід дозволяє позбутися всіх недоліків, які були присутні в попередньому рішенні при цьому зберігаючи точність на прийнятному рівні.

В даній роботі буде реалізовано перший етап двоетапного детектора з деякими покращеннями, оскільки точно визначеного рішення для побудови згорткової моделі для 2 етапу не було знайдено.

## Аналіз існуючих технологій та успішних IT-проєктів

Проаналізуємо відоме на сьогодні алгоритмічне забезпечення у даній області та технічні рішення, що допоможуть у реалізації LogoSearchBot. Далі будуть розглянуті допоміжні програмні засоби, засоби розробки та готові програмні рішення.

### Аналіз відомих алгоритмічних та технічних рішень

Повністю готове рішення складається з 2 компонент: модель для розпізнавання логотипів та телеграм бот, для використання цієї моделі. Спочатку опишемо модель.

Для побудови загального детектору з розпізнавання логотипів теоретично можна використовувати будь-яку глибинну нейромережу, що призначена для розпізнавання об’єктів. На сьогоднішній день існує кілька популярних алгоритмів глибинного навчання для виявлення об'єктів, і вибір найкращого алгоритму може залежати від конкретного використання, обсягу даних, точності, швидкості та ресурсів. Короткий опис деяких з них:

1. YOLO (You Only Look Once) [10]: YOLO є відомим алгоритмом, який визначає об'єкти на зображеннях одноразовим переглядом. Він швидкий і ефективний, але може втрачати у точності для невеликих об'єктів.
2. SSD (Single Shot Multibox Detector) [11]: SSD є алгоритмом, який пропонує визначення об'єктів за один кадр. Він ефективний та має високу швидкість, але може втрачати в точності порівняно з більш складними алгоритмами.
3. Mask R-CNN [12]: Розширення Faster R-CNN, яке додає інформацію про маску до визначених об'єктів. Використовується для визначення областей об'єктів з точністю на рівні пікселів.
4. EfficientDet [13]: Це алгоритм, який об'єднує ефективність та точність, шляхом оптимізації моделі та розмірів зображення для обробки об'єктів.

В більшості дослідженнях й статтях для побудови загального детектора логотипів пропонується використовувати застарілі моделі такі як RFBNet [14] або Faster R-CNN [15], проте в даній роботі буде застосовано сучасну й швидку модель YoloV5 (you only look once) [16], що також дозволить включити додатковий функціонал до розробки, а саме пошук логотипів на відео. Незважаючи на те, що на даний момент існують новіші версії архітектури YOLO, наприклад YOLOV8 [17], модель 5 версії працює швидше при цьому не сильно поступаючись в точності. [18]

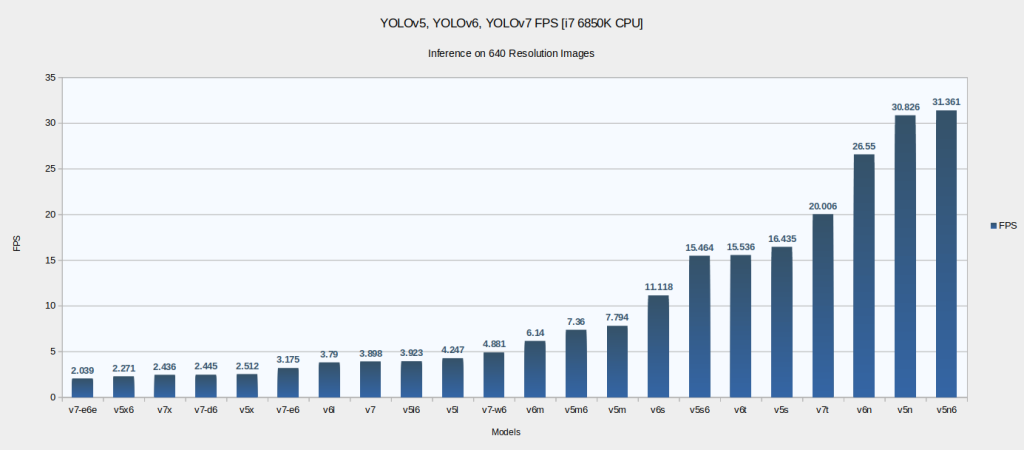


Рисунок 1.2 – Швидкіть (FPS) моделей YOLO

Для технічної реалізації чат-боту було вирішено використовувати саме Bot API від Telegram [19], оскільки серед всіх наявних рішень (Viber Python Bot API [20], WhatsApp API [21] та інші) він відомий своєю простотою та зручністю для побудови чат-ботів. Висока швидкість передачі повідомлень, можливість роботи з різноманітними типами медіа, вбудована підтримка клавіатур і оновлення чат-ботів, а також відсутність складних налаштувань роблять Telegram Bot API простим засобом для розробників. Також серед всіх конкурентів Telegram має найбільшу кількість програмних засобів та додаткових фреймворків, які використовуються розробниками для зручнішого налаштування ботів.

### Аналіз допоміжних програмних засобів та засобів розробки

Для реалізації програмного рішення використовується мова програмування Python, через велику кількість переваг над іншими мовами програмування в галузі розробки ШІ та написання чат-ботів. Python вирізняється своєю зручністю та читабельністю коду, простим і елегантним синтаксисом, що спрощує виразність ідеї розробника. Однією з ключових переваг Python є величезна спільнота та багата екосистема бібліотек саме для створення штучного інтелекту. Також безсумнівною перевагою є те, що yolov5 інтегрований разом з фремворком, що базується на Python, Pytorch [22], який дозволить дуже зручно налаштовувати та використовувати модель для власних цілей.

Серед існуючих можливих варіантів фреймворків для написання чат-боту в Telegram наразі існують такі найпопулярніші рішення:

1. python-telegram-bot [23]: Цей фреймворк має простий інтерфейс та розширений функціонал для взаємодії з Telegram Bot API. Він надає зручний доступ до функцій, таких як робота з повідомленнями, клавіатурами, фотографіями, аудіо та іншими мультимедійними елементами.
2. Aiogram [24]: Цей фреймворк розроблений для асинхронного програмування та дозволяє розробникам створювати Telegram ботів з використанням сучасних асинхронних підходів.
3. Telepot [25]: Telepot є простим у використанні фреймворком для роботи з Telegram Bot API. Він надає базовий функціонал для створення та управління ботами.

Найкращим серед цих всіх рішень виявився саме Aiogram. Це безкоштовний український проєкт, який зараз на піку своєї популярності й має дуже велику базу активних користувачів та цілий форум, де простому розробнику з легкістю допоможуть втілити свій проєкт. Тим не менш, цей фреймворк є асинхронним, що слугує просто величезним плюсом проти своїх інших конкурентів. З технічної частини наявність роутерів подій дозволяє ефективно структурувати код проекту. Вбудований кінцевий автомат полегшує реалізацію діалогів всередині чат-бота. Використання потужних магічних фільтрів забезпечує гнучку обробку та фільтрацію подій. Додатково, можливість використання мідлварів для обробки вхідних подій від API та для вихідних запитів до API дозволяє отримати додатковий рівень контролю та настроюваності. Наразі aiogram зарекомендував себе як надійний та простий засіб для розробки чат-ботів на базі Telegram Bot Api, який використовується як у великих комерційних цілях так і для звичайних розробок чат-боту.

Також бот буде мати зв’язок з БД PostgreSQL, тому для цього було обрано бібліотеку SQLАlchemy. [26] Дана бібліотека містить інструменти для використання об’єктно-реляційної моделі взаємодії з БД (скорочено ORM). Цей підхід дозволяє визначати звичайні об’єкти та методи Python і переводити їх у низькорівневі інструкції бази даних SQL. SQLAlchemy — це ORM, який вибирають для роботи з реляційними базами даних на python. Реалізувати ORM підхід через SQLAlchemy дуже просто, вона допомагає швидше розробити код і не потребує знання SQL, щоб почати роботу.

### Аналіз відомих програмних продуктів

У цьому розділі ми ретельно аналізуємо успішні IT-проекти, зосереджуючись на віртуальних файлових системах. Розглядаючи їх переваги та недоліки, ми визначимо ключові аспекти, які можуть визначати ефективні рішення в сучасному ІТ-середовищі.

Проаналізувавши відомі програмні продукти, що вирішують, або частково вирішують завдання пошуку логотипу на зображеннях вдалося виділити 3 основних ІТ-проєкти:

* Vision AI API від Google [27]
* YouScan [28]
* Logo Detection від SmartClick [29]

Кожен з цих програмних продуктів містить свої недоліки й переваги, які будуть висвітлені детальніше нище.

Vision AI – це нова розробка від компанії Google, яке надає API інтерфейс для створення застосунків з використанням комп'ютерного зору та отримання інсайтів з зображень та відео. За допомогою переднавчених API, AutoML та власних моделей він допоможе миттєво створювати застосунки для аналітики відео та зображень. Одним з функціональних можливостей даного продукту являється розпізнавання популярних логотипів на зображеннях передані до API. Зображення можуть бути завантаженні з локального комп’ютера або ж надане посилання на відкрите зображення. Даний функціонал можна отримати лише по платній загальний підписці на всі можливості Vision AI.

Наступним продуктом є YouScan. YouScan - це платформа для аналітики соціальних медіа на основі штучного інтелекту з найкращими у галузі можливостями аналізу зображень. Ця розробка розміщена як веб-застосунок, в якому можна створювати акаунт та активно відслідковувати соціальні мережі щодо певного бренду. YouScan надають ряд послуг, які корисні для маркетингового аналізу: відслідковування стану бренду, аналіз соцмереж, конкурентний аналіз, відслідковування ринку, аналізу аудиторіх та візуальні інсайти. Саме останній функціонал здатний розрізняти логотипи як власних компаній так і найпопулярніших у світі для аналізу ринку.

Logo Detection від SmartClick так само як і Vision AI надає свої послуги у вигляді API інтерфейсу, проте на відміну від вищезгаданої моделі, Logo Detection орієнтується сугубо на розпізнавання логотипів. Як запевняють розробники, технологія, яку вони використовують працює автоматично та виявляє будь-яку частину зображення, схожу на логотип. Також вони мають додатковий функціонал з додавання нових логотипів до існуючих. Проте наразі даний проєкт недійсний.

Подальше порівняння цих продуктів здійснено через таблицю

Таблиця 1.1 – Порівняння з аналогом

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерії порівняння | LogoSearchBot | YouScan | Vision AI | Logo Detection |
| Наявність платної підписки | Немає | Потрібно купувати місячну підписку на увесь продукт | Оплачується продукт за кількістю запитів зроблених до API | Купується підписка, яка може включати різну кількість запиті на міс. |

Продовження таблиці 1.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Доступність для звичайного користувача | Інтегрований до Telegram як чат-бот | Працює як веб-застосунок для користувача | Розрахований на використання розробниками ПЗ, незрозумілий для звичайного користувача | Аналогічно до Vision AI |
| Чи повністю вирішує проблему? | Так | Так | Частково. Невідомо чи можна знаходити будь-який логотип | Так |
| Підтримка сервісу й випущення нових оновлень | Легко інтегруються нові оновлення в модель | Проєкт активний, останнє оновлення відбулося в квітні 2022 | Великий проєкт з активною підтримкою, що постійно оновлюється | На даний момент проєкт недоступний і повністю закритий |

Продовження таблиці 1.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Можливість додавання відслідковування власних логотипів | Ні | Можливо, точно цього розробник не зазначав | Ні, використовує сталу кількість логотипів | Так, можна оновлювати модель за допомогою власних фото |
| Наявність інструкції користувача та зрозумілість | Наявна інструкція користувача, проте в силу своєї простоти він зрозумілий інтуєтивно | Розміщенні гайди по користуванню продуктом та сервісна підтримка | Містять детальну документацію | Розміщений приклад використання та опис продукту. За додатковим потрібно звертатися в підтримку |

Кожен із розглянутих сервісів має свої переваги та обмеження. LogoSearchBot є безкоштовним та простим для користування, тоді як YouScan та Vision AI надають більше функціоналу, але вимагають платної підписки. Залежно від потреб користувача, можна вибрати сервіс, що найкраще відповідає конкретним вимогам та рівню технічної експертизи.

## Аналіз вимог до програмного забезпечення

Головною функцією програмного забезпечення є визначення області, де знаходиться логотип на зображені, більше функцій можна побачити на рисунку 1.3.

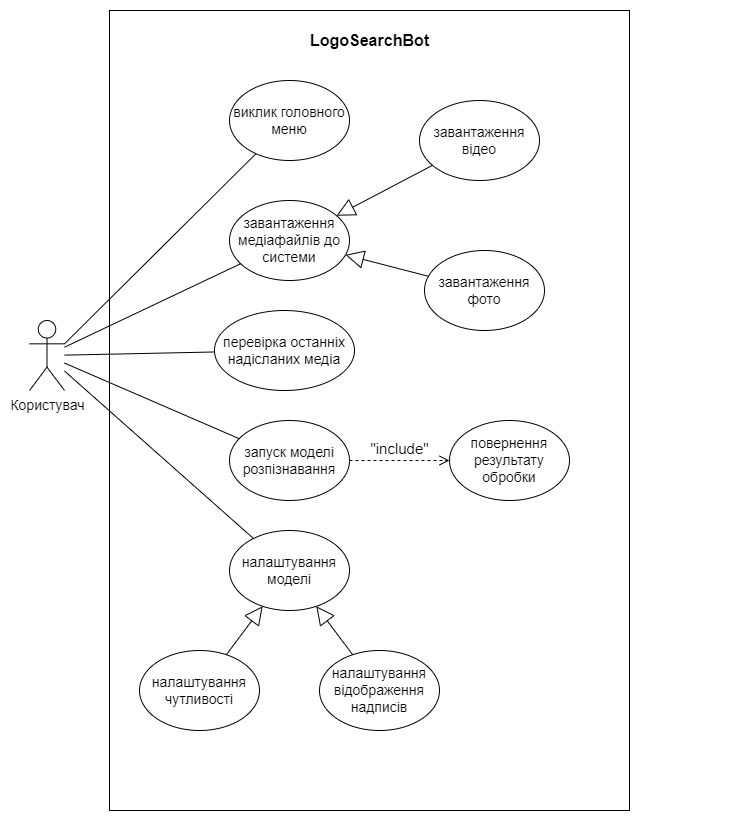


Рисунок 1.3 – Діаграма варіантів використання

В таблицях 1.2 - 1.9 наведені варіанти використання програмного забезпечення.

Таблиця 1.2 - Варіант використання UC-01

|  |  |
| --- | --- |
| Use case name | Виклик головного меню |
| Use case ID | UC-01 |
| Goals | Відображення головного меню для користувача |
| Actors | Користувач |
| Trigger | Користувач хоче отримати доступ до головного меню |
| Pre-conditions | Користувач не знаходиться в жодному з станів: надсилання медіафайлів, запуск моделі, налаштування моделі |
| Flow of Events | Користувач прописує команду /menu в чат та надсилає її. Також користувач може викликати її через меню команд в телеграмі. |
| Extension | Якщо користувач знаходиться у якомусь із станів, він отримує спливаюче вікно, яке повідомляє йому, що спочатку потрібно завершити поточну дію |
| Post-Condition | Відображення головного меню у діалозі |

Таблиця 1.3 - Варіант використання UC-02

|  |  |
| --- | --- |
| Use case name | Завантаження фото |
| Use case ID | UC-02 |
| Goals | Завантажити фото користувача для подальшого розпізнавання |
| Actors | Користувач |
| Trigger | Користувач хоче завантажити власні фото |
| Pre-conditions | Користувач натиснув на кнопку «Завантажити медіа» в головному меню |
| Flow of Events | Користувач надсилає до 5 власних фото на вибір через функціональні можливості телеграм. Коли всі фото завантажені користувач натискає скасувати завантаження. |
| Extension | Якщо користувач надсилає більше 5 фото він отримує попередження про максимальну кількість фото.  Якщо користувач надіслав фото не формату .jpg, .png, він отримує попередження. |
| Post-Condition | Система зберігає фото користувача |

Таблиця 1.4 - Варіант використання UC-03

|  |  |
| --- | --- |
| Use case name | Завантаження відео |
| Use case ID | UC-03 |
| Goals | Завантажити відео користувача для подальшого розпізнавання |
| Actors | Користувач |
| Trigger | Користувач хоче завантажити власні відео |
| Pre-conditions | Користувач натиснув на кнопку «Завантажити медіа» в головному меню |
| Flow of Events | Користувач надсилає до 5 власних відео на вибір через функціональні можливості телеграм. Коли всі фото завантажені користувач натискає скасувати завантаження. |
| Extension | Якщо користувач надсилає більше 5 фото він отримує попередження про максимальну кількість відео.  Якщо користувач надіслав фото не формату .mov, .mp4, він отримує попередження.  Якщо користувач надіслав відео довше 15 секунд, система попереджає про перевищення ліміту по довжині відео |
| Post-Condition | Система зберігає відео користувача |

Таблиця 1.5 - Варіант використання UC-04

|  |  |
| --- | --- |
| Use case name | Перевірка останніх надісланих медіа |
| Use case ID | UC-04 |
| Goals | Повернути користувачу медіа, які він завантажив |
| Actors | Користувач |
| Trigger | Користувач хоче дізнатися, що він завантажив |
| Pre-conditions | Користувач натиснув на кнопку «Отримати останнє завантажене медіа» в головному меню |
| Flow of Events | Користувач отримує всі попередньо завантаженні медіафайли |
| Extension | Якщо користувач не завантажував медіа, то система надішле спливаюче вікно про те, що спочатку потрібно завантажити медіа |
| Post-Condition | Система повернула надіслані медіа користувачем |

Таблиця 1.6 - Варіант використання UC-05

|  |  |
| --- | --- |
| Use case name | Запуск моделі розпізнавання |
| Use case ID | UC-05 |
| Goals | Почати роботу моделі з розпізнавання |
| Actors | Користувач |
| Trigger | Користувач хоче розпізнати власні медіа |
| Pre-conditions | Користувач натиснув на кнопку «Запустити модель» в головному меню |
| Flow of Events | Модель починає роботу з розпізнавання файлів |
| Extension | Якщо користувач не завантажував медіа, то система надішле спливаюче вікно про те, що спочатку потрібно завантажити медіа |
| Post-Condition | Система почала розпізнавання медіа |

Таблиця 1.7 - Варіант використання UC-06

|  |  |
| --- | --- |
| Use case name | Повернення результату обробки |
| Use case ID | UC-06 |
| Goals | Повернути користувачу результат роботи моделі |
| Actors | Користувач |
| Trigger | Користувач хоче отримати результат роботи моделі |
| Pre-conditions | Система почала роботу моделі з розпізнавання |
| Flow of Events | Після завершення роботи моделі, користувач отримує свої фото з розпізнаними логотипами на ньому |
| Extension | Якщо користувач скасовує дію, то результат не повертається |
| Post-Condition | Система повернула результат обробки користувачеві та меню |

Таблиця 1.8 - Варіант використання UC-07

|  |  |
| --- | --- |
| Use case name | Налаштування чутливості моделі |
| Use case ID | UC-07 |
| Goals | Змінити чутливість моделі для користувача |
| Actors | Користувач |
| Trigger | Користувач хоче змінити чутливість моделі |

Продовження таблиці 1.8

|  |  |
| --- | --- |
| Pre-conditions | Користувач натиснув на кнопку «Налаштувати модель», а потім «Налаштування чутливості» в головному меню |
| Flow of Events | Користувач надсилає в чат число яке буде позначати чутливість моделі. Якщо число коректне – діалог закривається |
| Extension | Якщо користувач ввів недопустиме значення для чутливості, система відповідає повідомлення з проханням ввести коректне число  Якщо користувач скасував дію, система закриває діалог |
| Post-Condition | Система змінила чутливість для користувача |

Таблиця 1.9 - Варіант використання UC-08

|  |  |
| --- | --- |
| Use case name | Налаштування відображення надписів |
| Use case ID | UC-08 |
| Goals | Змінити налаштування відображення надписів |
| Actors | Користувач |
| Trigger | Користувач хоче змінити налаштування надписів |
| Pre-conditions | Користувач натиснув на кнопку «Налаштувати модель», а потім «Налаштування відображення надписів» в головному меню |
| Flow of Events | Користувач обирає необхідне налаштування для себе на запропонованій клавіатурі |
| Extension | Якщо користувач скасував дію, система закриває діалог |
| Post-Condition | Система повернула надіслані медіа користувачем |

### Розроблення функціональних вимог

Програмне забезпечення умовно розділене на модулі: робота з медіа контентом, взаємодія з моделюю та налаштування моделі. Кожен модуль має свій певний набір функцій. Загальну модель вимог представлено у вигляді нумерованого списку, а в таблицях 1.10 – 1.16 наведений опис функціональних вимог до програмного забезпечення. Матрицю трасування вимог можна побачити на рисунку 1.4.

**Модель вимог:**

1. Робота з медіа контентом
   1. Надсилання медіафайлів
      1. Надсилання відео
      2. Надсилання фото
      3. Скасування надсилання медіа
   2. Отримання останніх надісланих медіа
   3. Збереження результатів розпізнавання на комп’ютері
2. Взаємодія з моделлю
   1. Робота моделі розпізнавання
      1. Запуск моделі
      2. Отримання користувачем результатів розпізнавання
      3. Скасування отримання результатів
   2. Повторне отримання результатів розпізнавання
3. Налаштування моделі
   1. Налаштування чутливості
   2. Налаштування відображень надписів

Таблиці функціональних вимог

Таблиця 1.10 – Функціональна вимога FR-1

|  |  |
| --- | --- |
| Назва | Надсилання медіафайлів |
| Опис | Система повинна надавати можливість користувачеві завантажувати до 5 власні фото або відео у форматах .jpg .png .mov .mp4. Функціональніть повинна також включати можливість скасувати надсилання медіафайлів. |

Таблиця 1.11 – Функціональна вимога FR-2

|  |  |
| --- | --- |
| Назва | Отримання останніх надісланих медіа |
| Опис | Система повинна забезпечити користувачу доступ до останніх завантажених ним медіафайлів за його запитом. |

Таблиця 1.12 – Функціональна вимога FR-3

|  |  |
| --- | --- |
| Назва | Збереження результатів розпізнавання на комп’ютері |
| Опис | Користувач повинен мати можливість завантажити фото оброблені моделлю з врахуванням усіх попередніх налаштувань локально собі на комп’ютер |

Таблиця 1.13 – Функціональна вимога FR-4

|  |  |
| --- | --- |
| Назва | Робота моделі розпізнавання |
| Опис | Модель з розпізнавання повинна забезпечувати розпізнавання логотипу на зображенні з врахування коефіцієнту чутливості. Користувач повинен мати можливість запускати модель та скасовувати її результати. Система повинна надіслати користувачеві результати роботи моделі |

Таблиця 1.14 – Функціональна вимога FR-5

|  |  |
| --- | --- |
| Назва | Повторне отримання результатів розпізнавання |
| Опис | Система повинна надавати можливість користувачеві повторно отримати результати останньої обробки моделі |

Таблиця 1.15 – Функціональна вимога FR-6

|  |  |
| --- | --- |
| Назва | Налаштування чутливості моделі |
| Опис | Система повинна надавати користувачеві функціонал для зміни чутливості моделі відповідно до допустимих значень |

Таблиця 1.16 – Функціональна вимога FR-7

|  |  |
| --- | --- |
| Назва | Налаштування відображення написів |
| Опис | Система повинна надавати користувачеві функціонал для зміни відображення результатів моделі розпізнавання |

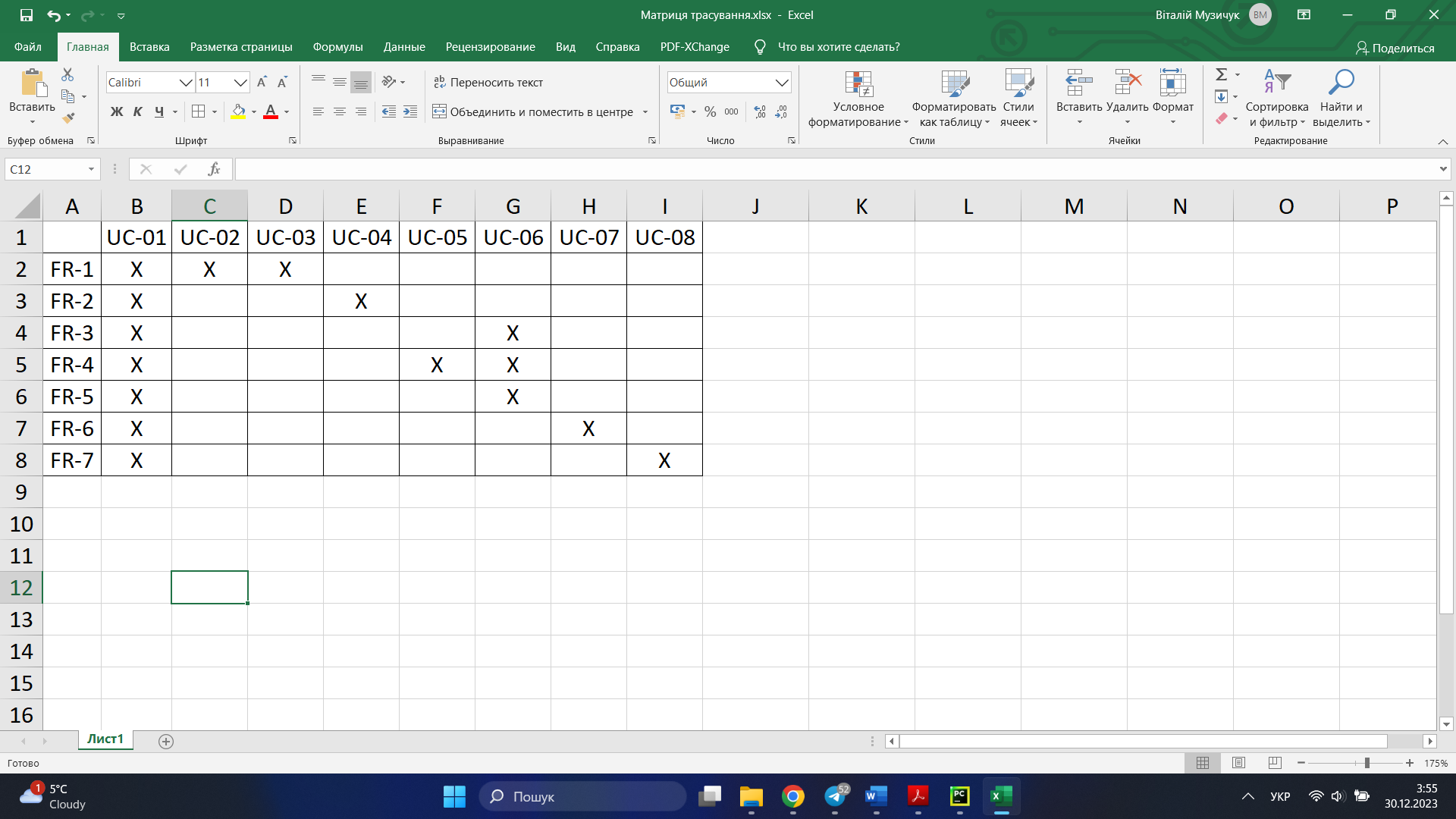


Рисунок 1.4 – Матриця трасування вимог

Важливо відзначити, що всі функціональні вимоги в таблиці трасування вимог для системи визначення логотипів завжди покриваються UC-01, оскільки цей варіант відповідає за виклик меню, через яке відбувається все управління програмою.

### Розроблення нефункціональних вимог

Забезпечення надійного контролю над введенням інформації та захист від некоректних дій користувача визначаються як важливі аспекти забезпечення безпеки та ефективної експлуатації системи. Зокрема, важливо передбачити механізми фільтрації та перевірки вхідних даних, щоб уникнути введення некоректної чи шкідливої інформації, що може вплинути на цілісність бази даних. Додатково, забезпечення конфіденційності переданої користувачем інформації виступає ключовим аспектом, що вимагає ретельного розгляду. Впровадження заходів шифрування, контролю доступу та інших технологій забезпечення конфіденційності стає необхідним для створення безпечного та довіреного середовища обробки інформації в системі фотографій.

Таблиця 1.17 – Нефункціональна вимога NFR-1

|  |  |
| --- | --- |
| Назва | Забезпечення надійного контролю |
| Опис | Забезпечити ефективний механізм контролю та захисту від некоректних дій користувача, спрямований на забезпечення безпеки системи. |

Таблиця 1.18 – Нефункціональна вимога NFR-2

|  |  |
| --- | --- |
| Назва | Перевірка вхідних даних |
| Опис | Розробити механізми фільтрації та перевірки вхідних даних для уникнення введення некоректної чи шкідливої інформації, яка може порушити цілісність бази даних. |

Таблиця 1.19 – Нефункціональна вимога NFR-3

|  |  |
| --- | --- |
| Назва | Забезпечення конфіденційності |
| Опис | Розглянути ключовий аспект забезпечення конфіденційності переданої користувачем інформації та розробити відповідні заходи, такі як шифрування та контроль доступу. |

## Постановка задачі

У результаті розробки програмного забезпечення на курсову роботу передбачається реалізація готового для кінцевого користувача продукту, який буде визначати логотипи на зображеннях.

Головним ядром цієї системи буде натренована на власному наборі даних модель глибинного навчання за архітектурою YoloV5. Особливістю тренування буде те, що модель буде навчатися не розрізняти логотипи окремих компаній, а просто намагатися передбачити область на зображенні де може знаходитися логотип без конкретизації його класу. Такий підхід в розробці дозволить побудувати систему, яка не потребуватиме постійного перетренування, оскільки модель визначать головні риси логотипів та їхні патерни і буде здатна навіть визначити той клас логотипів, які не були включені в початковий набір тренування моделі.

Далі, щоб надати можливість звичайному користувачеві скористатися можливістю цієї моделі розробиться чат-бот на базі Telegram Bot Api. Використання цієї платформи у розробці програмного забезпечення дозволить створити максимально зрозумілий та мобільний інтерфейс для користування розробкою моделі. Вагомою перевагою цього виду інтерфейсу є те, що користувач зможе виконувати запити та користуватися продуктом, як за допомогою мобільного пристрою так і ПК. Також ця «кросплатформеність» полегшить можливість завантаження власних фотографій до системи та збереження результатів обробки на власні пристрої.

Також для того, щоб забезпечити збереження даних користувацьких налаштувань розроблена система буде підключатися до бази даних PostgreSQL. Ця база даних міститиме інформацію про налаштування чутливості моделі та специфічні налаштування відображення результатів. Використання БД дозволить зберігати окремі налаштування навіть, якщо сервіс чат-боту буде тимчасово недоступний або матиме нестабільну роботу.

## Висновки до розділу

Отже, в даному розділі було розглянуто загальні положення для побудови систем для визначення логотипів, проаналізовано можливі рішення та вивчено сучасні підходи до вирішення цієї задачі. Також було розглянуто реальні рішення, які існують на даний час, визначено основні їхні переваги та недоліки й порівняно з завданням курсової. Загальна тенденція була така: всі програмні забезпечення є платними та недоступними або ж незрозумілими для звичайного пересічного користувача.

Далі була розроблена діаграма варіантів використання та табличний опис до неї, щоб визначити основні завдання, які буде вирішувати створена програма та які не були враховані у сучасних відкритих рішеннях. Після цього була сформована детальна модель вимог разом з табличним описом функціональних та нефункціональних вимог, які необхідні для розробки та втілення системи визначення логотипів. Це також дозволило отримати більш детальне уявлення про вимоги до мови програмування та технологій, які повинні бути використані. Матриця трасування вимог наочно показала, що кожна з функціональних вимог буде покрита юзкейсами, а в деяких випадках навіть декількома.

З технічної сторони для реалізації було вирішено використовувати мову програмування Python, фреймворк aiogram та бібліотеку SQLAlchemy, оскількі всі ці програмні засоби зручно поєднуються з використанням моделі yolo розробленій на фреймворці pytorch.

# МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Моделювання та аналіз програмного забезпечення

<У підрозділі викладають:

опис бізнес процесів вашої розробки та їх наочне представлення за допомогою засобів моделювання (BPMN та інші). Обсяг 2 сторінки, чи більше. >

Для опису бізнес процесу програмного забезпечення використовується BPMN модель (рисунок 2.1).

Опис послідовності створення облікового запису користувача:

* користувач переходить на сторінку реєстрації;
* користувач заповнює поля реєстрації;
* якщо введені поля, не відповідають шаблону заповнення на клієнтській стороні, відповідні поля підсвічуються помилкою;

## Архітектура програмного забезпечення

<У підрозділі викладають:

втілення архітектурного паттерну для вашого ПЗ та деталізація його компонентів за необхідності. Необхідно виконати повний опис архітектури (схеми, таблиці та ін.). Обсяг 2 сторінки, чи більше.>

## Конструювання програмного забезпечення

<У підрозділі викладають:

опис оригінальних алгоритмів чи модифікацій існуючих. Опис структур даних, програмних структур та ін. Опис бази даних з представленням концептуальної, логічної чи фізичної моделі та з описом сутностей чи таблиць. Опис утиліт, бібліотек та іншого стороннього програмного забезпечення, що використовується у розробці. Аналіз системних вимог. Обсяг 4 сторінки, чи більше.>

Бібліотека xpring, яка виконує запити в мережу блокчейн – однопоточна, тобто не може оброблювати паралельні запити. Через таку специфіку бібліотеки, довелось розробляти рішення яке б не блокувало запити з різних потоків, або інших частин коду. За основу ідеї алгоритму була взята багатопоточність самої мови програмування java. Java для вирішення проблем з потоками представляє ділянки коду які можна синхронізувати, що унеможливлює одночасний доступ до нього з різних потоків. Ця синхронізація проходить за допомогою передачі управління об’єкта-монітора. Алгоритм вирішення проблеми з доступом до однопоточної бібліотеки наведено на рисунку 2.3.

В якості системи управління базами даних використовується Postgres. База даних серверу призначена для зберігання користувачів, а також даних про їх …. Опис таблиць бази даних наведено у таблицях 2.11 - 2.14. Модель бази даних наведена на рисунку 2.12.

Таблиця 2.11 – Опис таблиці user

|  |  |  |  |
| --- | --- | --- | --- |
| Таблиця | Назва поля | Тип даних | Опис |
| user | id | serial | ідентифікаційний номер користувача |
| email | varchar | електронна пошта користувача |
| password\_id | int | посилання на запис у таблиці password, де зберігається пароль користувача |

Опис утиліт, бібліотек та іншого стороннього програмного забезпечення, що використовується у розробці наведено в таблиці 2.22.

Таблиця 2.22 – Опис утиліт

|  |  |  |
| --- | --- | --- |
| №  п/п | Назва утиліти | Опис застосування |
| 1 | IntelliJ IDEA | Головне середовище розробки програмного забезпечення серверної частини курсової роботи. |
| 2 | Postman | Програмне забезпечення необхідне для тестування rest запитів. Використовувалось для тестування API інтерфейсів, та клієнтських запитів. |
| 3 | MySQL Workbench | Програмне забезпечення яке надає легкий графічний інтерфейс для доступу до бази даних. |

## Аналіз безпеки даних

<У підрозділі викладають:

аналіз вразливостей ПЗ та будь-які питання пов’язані з безпекою даних. >

## Висновки до розділу

< Необхідно стисло описати усе, що було виконано у даному розділі. Обсяг 0,75-1 сторінка>

# АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Аналіз якості ПЗ

<У підрозділі викладають:

аналіз якості ПЗ за певними метриками. Обсяг 2 сторінки, чи більше.>

## Опис процесів тестування

<У підрозділі викладають:

опис процесів тестування та приклади тестів. Обсяг 4 сторінки, чи більше.>

Було виконане мануальне тестування програмного забезпечення, опис відповідних тестів наведено у таблицях 3.3 – 3.30.

Таблиця 3.3 – Тест 1.1

|  |  |
| --- | --- |
| Тест | Реєстрація користувача |
| Модуль | Реєстрація користувача |
| Номер тесту | 1.1 |
| Початковий стан системи | Користувач знаходиться на сторінці реєстрації |
| Вхідні данні | Електронна пошта, пароль, підтвердження паролю |
| Опис проведення тесту | У відповідні поля вводяться: коректна електронна пошта, яка до цього не була зареєстрована в системі, пароль від 10 до 64 символів, який містить хоча б з одну англійську літеру, одне число і один спеціальний символ, і який не входить у топ 10000 найпопулярніших паролей, підтвердження паролю, яке співпадає з раніше введеним паролем. Після цього … |
| Очікуваний результат | Реєстрація проходить успішно, користувач додається у систему і перенаправляється на сторінку авторизації. |
| Фактичний результат | Реєстрація проходить успішно, користувач додається у систему і перенаправляється на сторінку авторизації. |

## Опис контрольного прикладу

<У підрозділі викладають:

повний опис контрольного прикладу з усіма можливими розгалуженнями та особливостями. Кроки доповнюють ілюстраціями. Не обов’язковий розділ.>

## Висновки до розділу

< Необхідно стисло описати усе, що було виконано у даному розділі. Обсяг 0,75-1 сторінка>

# ВПРОВАДЖЕННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Розгортання програмного забезпечення

<У підрозділі викладають:

повний опис покрокового розгортання ПЗ. Кроки доповнюють ілюстраціями. Обсяг 2 сторінки, чи більше.>

Клієнтську і серверну частини програмного забезпечення було вирішено розгорнути на платформі Heroku. Для розгортання було використано сервіс GitHub Actions, який надає можливості для постійної інтеграції і розгортання [13].

Розгортання починається коли новий код застосунку доставляється у репозиторій у гілку main. Тоді у середовищі GitHub Actions створюється Docker image за допомогою Dockerfile, що знаходиться у проекті. Цей image розгортається у Heroku за допомогою пакету heroku-deploy. Інформацію про розгортання клієнтської і серверної частини проекту можна побачити на рисунках 4.1 і 4.2.

Рисунок 4.1 - Інформація про розгортання клієнту

## Підтримка програмного забезпечення

<У підрозділі викладають:

опис того, як буде виконуватись підтримка програмного забезпечення. Ілюстрації. Обсяг 1 сторінка, чи більше.>

Користувачі повинні мати можливість отримати нову версію консольного застосунку з кожною версією. До того ж кожна нова версія консольного застосунку повинна бути опублікована в npm. Для автоматизації цього процесу був використаний сервіс GitHub Actions.

Створення нового випуску починається, коли нова версія консольного застосунку доставляється у репозиторій у гілку main, тобто коли commit має tag формату “v\*.\*.\*.”, де замість “\*” знаходиться число. Тоді у середовищі GitHub Actions встановлюється NodeJS. Після цього для проекту встановлюються залежності і проект збирається. Bash скрипт за допомогою бібліотеки pkg генерує виконувані файли (executables) для Linux і для Windows, та пакує файл для Linux у .deb пакет. Після цього .deb пакет і файл для Windows архівуються,

## Висновки до розділу

< Необхідно стисло описати усе, що було виконано у даному розділі. Обсяг 0,75-1 сторінка>

# ВИСНОВКИ

У висновках викладають найважливіші наукові й практичні результати роботи та наводять:

* оцінку одержаних результатів і їх відповідність сучасному рівню наукових і технічних знань;
* ступінь впровадження та можливі галузі або сфери використання результатів роботи;
* наукову, науково-технічну, соціально-економічну значущість роботи;
* доцільність продовження досліджень за відповідною тематикою тощо.

Також у висновках необхідно відобразити стан вирішення усіх поставлених в курсовій роботі задач.

В результаті виконання курсової роботи було спроєктовано …

В якості середовища розробки обрано …

У якості БД використано …

Після реалізації застосунку він був протестований на пристроях з різними версіями Android, з різними розмірами екранів щоб переконатися, що додаток акуратно відображається на різних пристроях.

Наукова новизна роботи (якщо вона є) полягає в наступному (достатньо вказати щось одне).

Вперше:

* реалізовано можливість запитів від пацієнта до лікаря;
* використано те-то, що дозволило те-то.

Модифіковано:

* те-то, що дозволило те-то.

Набуло подальший розвиток:

* те-то, що дозволило те-то.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. [Оформлення бібліографії ДСТУ 7.1:2006](https://drive.google.com/file/d/1VUr7fwKHOk8t1u8sh-Sv2dlEc_6OBoIS/view?usp=sharing)
2. [Оформлення бібліографії ДСТУ 8302-2015](https://drive.google.com/file/d/17RYEYVnSXolL0S8D1k85VfReYd2yC6WO/view?usp=sharing) з прикладами