Вінницький національний технічний університет
Факультет інформаційних технологій та автоматизації
Кафедра комп'ютерних наук

Магістерська кваліфікаційна робота на тему:
«Інформаційна технологія розпізнавання об'єктів у мобільному застосунку»

Виконав: студент 2 курсу групи 1КН-22м
спеціальності 122 Комп'ютерні науки
_____ Вадим ВАКОЛЮК

Керівник: к. т. н., доцент каф. КН
_____ Сергій БАРАБАН
«04» 12 2023р.

Опонент: д. т. н., доцент, т.в.о.зав. каф.КСУ
_____ Марія ЮХИМЧУК
«07» 12 2023р.

Допущено до захисту
Завідувач кафедри КН
д. т. н., проф.
_____ Андрій ЯРОВИЙ
«02» 12 2023 р.

Вінниця ВНТУ – 2023 року

Вінницький національний технічний університет
Факультет інформаційних технологій та автоматизації
Кафедра комп'ютерних наук
Рівень вищої освіти II (магістерський)
Галузь знань – 12 Інформаційні технології
Спеціальність – 122 Комп'ютерні науки
Освітньо-професійна програма – Системи штучного інтелекту

ЗАТВЕРДЖУЮ
Завідувач кафедри КН,
д. т. н., проф.
_____Андрій ЯРОВИЙ
«29» 08 2023 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Ваколюку Вадиму Юрійовичу

1. Тема роботи: «Інформаційна технологія розпізнавання об'єктів у мобільному застосунку»
керівник роботи: Барабан Сергій Володимирович, к. т. н., доцент кафедри КН, затверджені наказом ректора ВНТУ від 18 вересня 2023 №247.

2. Строк подання студентом роботи: 13.11.23

3. Вихідні дані до роботи:
   - мова програмування Dart;
   - тип об'єктів для розпізнавання – об'єкти на зображеннях;
   - формат зображень;
   - навантажувальне тестування розробленого додатку.

4. Зміст текстової частини: Вступ. 1. Опис етапів побудови засобів розпізнавання об'єктів. 2.Засоби реалізації програмної системи розпізнавання об'єктів. 3.Опис програмної реалізації. 4.Методика роботи користувача з мобільним застосунком . 5. Економічна частина. Висновки. Перелік використаних джерел. Додатки.

5. Перелік ілюстративного матеріалу: Актуальність, мета та задачі МКР (плакат А4). Таблиця переліку розширених слів (плакат А4). Архітектура та перелік плагінів (плакат А4). Результати тестування відклику програми (плакат А4). Результати тестування на різних пристроях (плакат А4). Алгоритм роботи перекладача (плакат А4). Результати апаратного тестування (плакат А4). Результати тестування на сумісність (плакат А4). Алгоритм роботи програми (плакат А4). Результати тестування відмовостійкість (плакат А4). Алгоритм роботи налаштувань у додатку

(плакат А4). Алгоритм роботи за відсутності мережевого підключення (плакат А4). Висновки з виконаної роботи (плакат А4).

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Барабан С.В., к. т. н., доц. каф. КН		
2	Барабан С.В., к. т. н., доц. каф. КН		
3	Барабан С.В., к. т. н., доц. каф. КН		
4	Барабан С.В., к. т. н., доц. каф. КН		
5	Адлер О.О., к. т. н., доц. каф. ЕПВМ		

7. Дата видачі завдання 29.08.2023р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської дипломної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз завдання. Вступ	1.09.23 – 07.09.23	
2	Аналіз літературних джерел за напрямком магістерської кваліфікаційної роботи	08.09.23 – 20.09.23	
3	Розробка рішень	21.09.23 – 25.09.23	
4	Розробка модуля програмного засобу	25.09.23 – 01.10.23	
5	Практична реалізація, моделювання, експериментування, результати	01.10.23 – 15.10.23	
6	Розробка розділу тестування і обґрунтування доцільності розробки	15.10.23 – 01.11.23	
7	Висновки	01.11.23 – 06.11.23	
8	Оформлення пояснювальної записки	06.11.23 – 10.11.23	
9	Попередній захист та доопрацювання МКР	13.11.23	
10	Представлення МКР до захисту	13.12.23	
11	Захист МКР	19.12.23	

Студент _____ Вадим ВАКОЛЮК

Керівник роботи _____ Сергій БАРАБАН

АНОТАЦІЯ

УДК.004.8

Ваколюк В. Ю. Інформаційна технологія розпізнавання об’єктів у мобільному застосунку. Магістерська кваліфікаційна робота зі спеціальності 122 – комп’ютерні науки, освітня програма - комп’ютерні науки. Вінниця: ВНТУ, 2023. 129 с.

На укр. мові. Бібліогр.23: назв; рис.26:; табл.7.

Магістерська робота ставить перед собою завдання розглянути та вдосконалити етапи побудови засобів розпізнавання об'єктів. Перший розділ присвячений аналізу існуючих програмних рішень, вивченню алгоритмів розпізнавання та постановці задачі в контексті розпізнавання об'єктів. З цього виникає висновок, що стан сучасних технологій розпізнавання об'єктів потребує додаткового вдосконалення. У другому розділі розглядаються засоби реалізації програмної системи для розпізнавання об'єктів. Детально розглядаються мови програмування, такі як Dart та Kotlin, а також фреймворк Flutter і система управління базами даних Firebase Realtime Database. Третій розділ присвячений опису програмної реалізації, включаючи архітектуру системи, діаграму класів, структуру бази даних та алгоритм розпізнавання об'єктів. З цього розділу виходить висновок, що правильна архітектура та алгоритміка грають важливу роль у створенні ефективної програмної системи. Останній розділ розглядає методику роботи користувача з мобільним застосунком. Системні вимоги, встановлення та тестування, а також можливості використання програми детально розглядаються. Цей розділ вказує на важливість забезпечення зручного та інтуїтивно зрозумілого інтерфейсу для кінцевого користувача. Узагальнюючи, робота спрямована на розвиток та впровадження технологій розпізнавання об'єктів у мобільному застосунку з використанням сучасних інструментів та методів програмування.

Ключові слова: розпізнавання об'єктів, мобільні додатки, алгоритми, мови програмування, Dart, Kotlin, Flutter, Firebase Realtime Database, архітектура системи, діаграма класів, база даних, методика користувача, інтерфейс, технологічний аналіз.

ABSTRACT

Information technology for object recognition in a mobile application. Master's qualification work in specialty 122 - computer science, educational program - computer science. Vinnytsia: VNTU, 2023. 105 c.

In Ukrainian. Bibliogr.: titles; figs:; tables.

The master's thesis aims to consider and improve the stages of building real-time object recognition tools. The first chapter is devoted to the analysis of existing software solutions, the study of recognition algorithms, and the problem statement in the context of object recognition. This leads to the conclusion that the state of the art in object recognition technologies requires further improvement.

The second section discusses the means of implementing a software system for object recognition. Programming languages such as Dart and Kotlin, as well as the Flutter framework and the Firebase Realtime Database database management system are discussed in detail. The third chapter is devoted to the description of the software implementation, including the system architecture, class diagram, database structure, and object recognition algorithm. This chapter concludes that proper architecture and algorithmics play an important role in creating an effective software system.

The last section discusses how to use the mobile application. The system requirements, installation and testing, as well as the possibilities of using the application are discussed in detail. This section highlights the importance of providing a user-friendly and intuitive interface for the end user. To summarize, the work is aimed at developing and implementing object recognition technologies in a mobile application using modern programming tools and methods.

Keywords: object recognition, mobile applications, algorithms, programming languages, Dart, Kotlin, Flutter, Firebase Realtime Database, system architecture, class diagram, database, user methodology, interface, technological analysis.

ЗМІСТ

ВСТУП

Актуальність. З глобальним впровадженням системи Android, смартфони почали розвиваються набагато швидшими темпами та, дякуючи постійному доступу до Інтернету, інформація про місцезнаходження користувача може бути отримана будь–де та будь–коли. Завдяки цьому з’являється багато нових технологій, що є дуже корисним для користувачів, оскільки їх можна використовувати з вже існуючими технологіями та додатками в різних сферах.

Інформаційні технології постійно розвиваються, проникаючи в усі сфери нашого життя. Однією з найцікавіших галузей є розпізнавання об'єктів, що поєднує в собі комп'ютерний зор і штучний інтелект. [0]. Ця технологія дозволяє комп'ютерам аналізувати інформацію з відео або зображень у реальному часі та визначати різні об'єкти, людей, тварин, транспортні засоби і багато іншого. [0].

Завдяки швидкому розвитку обчислювальної потужності та алгоритмів машинного навчання, сьогодні ми маємо доступ до потужних систем розпізнавання об'єктів в реальному часі, які можуть працювати на пристроях з обмеженими обчислювальними можливостями, таких як смартфони або вбудовані системи. Вона використовується у відеоспостереженні, автономних автомобілях, системах безпеки, розпізнаванні осіб для біометричної ідентифікації, а також у віртуальній та доповненій реальності [**Помилка! Джерело посилання не знайдено.**].

Застосування систем розпізнавання об'єктів є ключовим елементом для вирішення завдань безпеки, медицини, транспорту, виробництва та багатьох інших галузей. Здатність миттєво визначати та аналізувати об'єкти навколишнього середовища відкриває безліч можливостей для ефективного управління та вдосконалення різноманітних процесів. Прискорений розвиток алгоритмів машинного навчання та штучного інтелекту сприяє вдосконаленню систем розпізнавання об'єктів, дозволяючи їм працювати ефективно та точно в режимі реального часу.

Сучасні алгоритми розпізнавання об'єктів базуються на глибокому навчанні та нейронних мережах, що дозволяє їм здійснювати точні та швидкі аналізи великої

кількості інформації. Наукова спільнота активно досліджує нові підходи до покращення алгоритмів розпізнавання об'єктів, використовуючи різноманітні моделі та техніки, такі як передові методи передбачення, генерації даних та адаптивного навчання.

Зв'язок роботи з науковими програмами, планами, темами. Магістерська робота виконана відповідно до напрямку наукових досліджень кафедри комп'ютерних наук Вінницького національного технічного університету 22 К1 «Розробка прикладних інтелектуальних інформаційних технологій та систем» та плану наукової та навчально-методичної роботи кафедри.

Мета і завдання досліджень. Метою магістерської кваліфікаційної роботи є розширення функціональних можливостей програмного забезпечення розпізнавання об'єктів у мобільному застосунку.

Для досягнення мети потрібно розв'язати такі задачі:

- Спроектувати архітектуру програмного забезпечення;
- Провести аналіз предметної області розпізнавання обєктів у мобільному застосунку
- Провести аналіз існуючих технологій області розпізнавання обєктів у мобільному застосунку
- Розробити інформаційну технологію розпізнавання обєктів у мобільному застосунку
- Розробити алгоритм розпізнавання обєктів у мобільному застосунку
- Здійснити програмну реалізацію розробленої інформаційної технології
- Здійснити тестування створеного програмного забезпечення та провести аналіз його роботи.
- Здійснити економічні розрахунки доцільності розробки нової інформаційної технології

Об'єкт дослідження – Процес розпізнавання об'єктів у мобільному застосунку

Предмет дослідження – Предметні засоби розпізнавання об'єктів у мобільному застосунку

Методи дослідження. У роботі використані наступні методи наукових досліджень: системного аналізу, теорії нейронних мереж для реалізації інформаційної технології, методи математичної статистики для розробки процесу розв'язання задачі розпізнавання рукописних цифр та обрахунків результатів експериментів із програмним засобом, об’єктно-орієнтованого програмування.

Наукова новизна одержаних результатів. Набула подальшого розвитку інформаційна технологія розпізнавання об’єктів у мобільному застосунку яка відрізняється від існуючих використанням згорткової нейроної мережі, що дозволело розширети функціональні можливості програмного забезпечення розпізнавання об’єктів у мобільному застосунку.

Практичне значення одержаних результатів полягає в тому, що на основі проведених досліджень розроблено програмне забезпечення розпізнавання об’єктів у мобільному застосунку.

Запропонована інформаційна технологія сприяє підвищенню достовірності програмних засобів розпізнавання об’єктів у мобільному застосунку.

- розроблено алгоритм розпізнавання об’єктів у мобільному застосунку.
- розроблено програмне забезпечення розпізнавання об’єктів у мобільному застосунку.

Достовірність теоретичних положень магістерської кваліфікаційної роботи підтверджується коректністю постановки завдання, коректністю використання математичного апарату методів дослідження, експериментальними дослідженнями тестування програмної реалізації інформаційної технології розпізнавання об’єктів у мобільному застосунку.

Особистий внесок здобувача. Усі результати, наведені у магістерській кваліфікаційній роботі, отримані самостійно. У працях, написаних у співавторстві, здобувачу належать: Аналіз розпізнавання обєктів у мобільному застосунку [1].

Апробація результатів роботи. Результати роботи були апробовані на Міжнародній науково-практичній Інтернет-конференції студентів, аспірантів та молодих науковців «МОЛОДЬ В НАУЦІ: ДОСЛІДЖЕННЯ, ПРОБЛЕМИ, ПЕРСПЕКТИВИ (МН-2024)» (м. Вінниця, Україна, 2023-2024) [1].

1 АНАЛІЙ ПРЕДМЕТНОЇ ОБЛАСТІ РОЗПІЗНАВАННЯ ОБ’ЄКТІВ

1.1 Аналіз існуючих програмних рішень розпізнавання об’єктів

Одним з етапів розробки програми для розпізнавання об’єктів є аналіз існуючих засобів для розпізнавання об’єктів. У аналізі засобів для розпізнавання об’єктів виділив декілька передових технологій які на сьогоднішній день вважаються передовими.

Глибоке навчання та нейронні мережі є визначальними компонентами в розвитку сучасних систем розпізнавання об'єктів у реальному часі. Використання конволюційних нейронних мереж (CNN), таких як YOLO (You Only Look Once) та Faster R-CNN, виявилося ключовим для досягнення високої точності та ефективності в роботі зображень [2]. Ці мережі здатні автоматично вивчати властивості об'єктів, що забезпечує вражаючі результати в умовах реального часу. Оптичний потік визначає напрямок та швидкість руху пікселів між кадрами, надаючи системі можливість відстежувати рух об'єктів у динамічних сценах. Алгоритми оптичного потоку виявляються невід'ємною частиною реального часу, забезпечуючи точність та надійність відстеження руху об'єктів, особливо у відеозаписах. Методи сегментації, що дозволяють системі визначати об'єкти на рівні пікселів та визначати їхні контури, покращують точність розпізнавання. Обробка зображень, така як уточнення контурів та фільтрація, є важливою для покращення якості вхідних даних та забезпечення стійкості системи до артефактів.

Ефективна реалізація алгоритмів на вбудованих системах стає ключовим фактором для успішного впровадження розпізнавання об'єктів у реальному часі. Використання легких архітектур, таких як MobileNet, дозволяє забезпечити високу продуктивність при обмежених ресурсах вбудованих пристроїв. Застосування передових архітектур, наприклад, EfficientNet, визначається їхньою високою ефективністю та здатністю до адаптації під різні умови. Їх використання дозволяє підвищити точність розпізнавання об'єктів при збереженні високої продуктивності.

Адаптивне навчання стає ключовою складовою для підтримки довгострокової ефективності систем. Методи адаптивного навчання можуть включати стратегії періодичного оновлення моделі, врахування нових даних та здатність до навчання на власних помилках, забезпечуючи системі гнучкість та високу адаптивність до змін у вихідних умовах. Це робить систему більш реактивною та надійною в умовах реального світу. Розглянемо більш детально моменти розпізнавання об’єктів на прикладі бібліотеки OpenCV.

OpenCV: OpenCV - це бібліотека з відкритим кодом для комп'ютерного зору та обробки зображень. Вона має модуль dnn, який дозволяє використовувати глибокі нейронні мережі для розпізнавання об'єктів. На рисунку 1.1 видно як проходить аналіз об’єктів програма аналізує ту інформацію яка передається через оптичну камеру.

Рисунок 1.1 - Аналіз об’єктів

Метод аналізу об'єктів з використанням OpenCV є ключовим елементом в розробці систем комп'ютерного зору та обробки зображень. Відповідно до конкретних вимог та завдань, розробники можуть використовувати різні алгоритми та підходи для досягнення оптимальних результатів. OpenCV дозволяє використовувати традиційні методи комп'ютерного зору, такі як виявлення

ключових точок та дескрипторів, що забезпечують роботу в умовах обмеженого обсягу даних чи обмеженої потужності обчислювальних пристроїв. У той же час, з введенням модуля DNN, OpenCV став можливим інструментом для використання глибоких нейронних мереж. Глибокі нейронні мережі, такі як YOLO та SSD, дозволяють ефективно розпізнавати об'єкти на зображеннях та відео в реальному часі.

Ці моделі навчаються визначати та локалізувати об'єкти, роблячи їх ідеальними для застосувань в областях безпеки, медицини, автономних транспортних засобів та багатьох інших [3]. Додатково, для роботи з об'єктами у мобільному застосунку, важливо враховувати вплив апаратних обмежень. OpenCV використовує оптимізовані алгоритми та можливості апаратного прискорення, такі як OpenVINO від Intel, для забезпечення високої швидкодії та продуктивності на різних пристроях. При розробці системи розпізнавання об'єктів, важливо також розглядати аспекти безпеки та конфіденційності даних, особливо коли застосовується глибоке навчання та обробка великих обсягів інформації. Забезпечення точності та надійності результатів, а також оптимізація під конкретні потреби, визначають успішність застосування методу аналізу об'єктів з використанням OpenCV в реальних умовах.

Плюси OpenCV: Відкритий код: Основний плюс OpenCV - це його відкритий код, що дозволяє розробникам вільно використовувати, модифікувати та розповсюджувати бібліотеку. Широкий функціонал: OpenCV надає широкий набір функцій для обробки зображень, включаючи фільтрацію, морфологічні операції, видалення шуму та роботу з кольорами. Підтримка різних мов програмування: OpenCV має інтерфейси для різних мов програмування, включаючи C++, Python та Java, що полегшує інтеграцію в різноманітні проекти. Активна спільнота: OpenCV має велику та активну спільноту розробників, яка активно співпрацює, обмінюється досвідом та надає підтримку. Підтримка різних платформ: Бібліотека підтримує різні операційні системи, включаючи Windows, Linux та macOS, а також може працювати на різних платформах, таких як x86 та ARM [4]. Ефективність та швидкість: OpenCV оптимізований для використання апаратних можливостей

пристроїв та здатний працювати з великими об'ємами даних, що робить його ефективним для великих обчислень. Розширені можливості глибокого навчання: Останні версії OpenCV мають модуль DNN (Deep Neural Networks), який надає можливості роботи з глибокими нейронними мережами.

Недоліки OpenCV: Складність для новачків: Для новачків у галузі комп'ютерного зору OpenCV може виглядати складно, оскільки вона вимагає розуміння основ та алгоритмів обробки зображень. Обмежена підтримка глибокого навчання: В порівнянні з іншими бібліотеками для глибокого навчання, підтримка OpenCV не така широка. Обмежена гнучкість при роботі з графічними інтерфейсами: Хоча OpenCV має підтримку графічних інтерфейсів, її гнучкість в цьому плані обмежена порівняно з іншими інструментами. Відсутність автоматизованої підтримки для деяких завдань: Деякі завдання, такі як розпізнавання облич, можуть вимагати додаткових бібліотек або моделей. Залежність від інших бібліотек: В деяких випадках використання OpenCV може вимагати інтеграції з іншими бібліотеками, зокрема для глибокого навчання або обробки медичних зображень. Потребує оптимізації для деяких завдань: Для деяких високорівневих завдань, таких як розпізнавання об'єктів у важких умовах освітлення, може знадобитися додаткова оптимізація або доповнення OpenCV іншими методами [5].

В етапі аналізу засобів розпізнавання обєктів виділив ще декілька сервісів та алгоритмів які проаналізував одним із них є YOLO (You Only Look Once): це інноваційний алгоритм розпізнавання об'єктів, який відзначається високою швидкістю та точністю. Розроблений компанією Joseph Redmon і Santosh Divvala, YOLO вперше був представлений у 2016 році і відзначений своєю здатністю виявляти та класифікувати об'єкти на зображенні в режимі реального часу.

Основним принципом YOLO є об'єднання завдань виявлення об'єктів та класифікації в одному проході, що робить його відмінним від багатьох інших методів, де ці завдання виконуються послідовно [6]. Алгоритм розділяє зображення

на сітку та використовує цю сітку для визначення потенційних областей, де можуть розташовуватися об'єкти. Основні етапи роботи YOLO рис. 1.2.

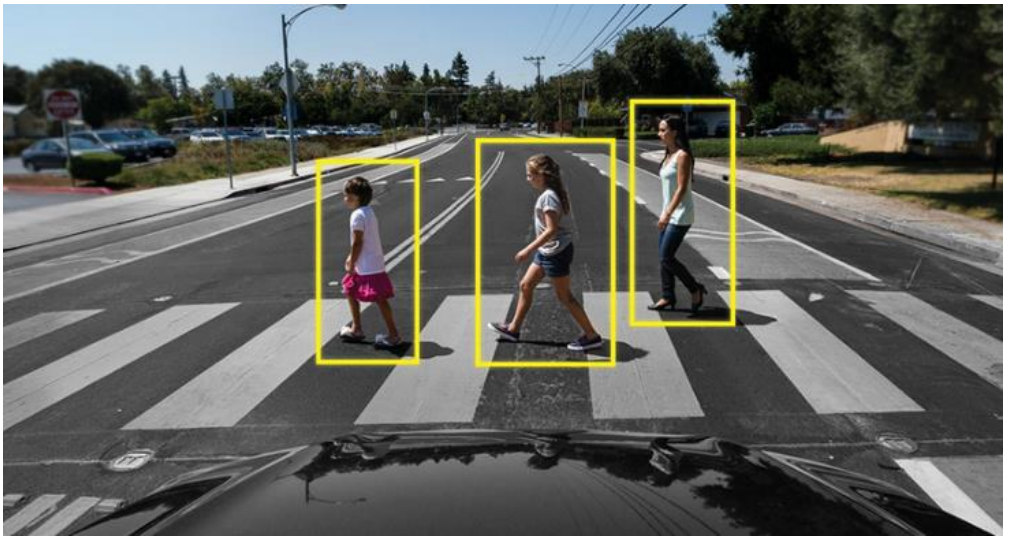


Рисунок 1.2 - розділення зображення

1. Сітка та якірі: Під час обробки зображень у моєму застосунку використовується важливий підхід, який включає в себе розділення зображення на сітку. Розмір цієї сітки може варіюватися в залежності від конфігурації моделі, що надає гнучкість та адаптивність системи до різних умов та завдань. Кожна клітинка сітки представляє собою окремий регіон на зображенні, і це визначає область, яка піддаватиметься подальшому аналізу та обробці. Такий підхід стає важливим елементом при роботі з різними типами та розмірами зображень, оскільки сітка дозволяє ефективно управляти та аналізувати різні області зображення. Ключовим аспектом є визначення якорів для кожної клітинки сітки [7]. Ці якорі функціонують як представники областей, де можуть розташовуватися об'єкти. Вони визначають ключові точки аналізу в межах кожного регіону та служать основою для подальшого виявлення та розпізнавання об'єктів на зображенні.

Такий підхід забезпечує важливий інструмент для точного та ефективного виявлення об'єктів на зображенні. Конфігурація моделі може визначати розмір сітки, що робить систему гнучкою та адаптованою до вимог конкретного завдання.

Застосування сітки та якорів стає особливо важливим у сферах комп'ютерного зору та обробки зображень, де потрібно ефективно вирішувати завдання розпізнавання та визначення об'єктів. Такий метод аналізу забезпечує необхідний рівень точності та розширює можливості системи у виявленні об'єктів різних форм та розмірів. Загалом, використання сітки та якорів стає ключовим компонентом в обробці зображень, що дозволяє оптимізувати та підвищувати ефективність системи при вирішенні складних завдань аналізу та розпізнавання [8].

2. Прогнози: Для кожної окремої клітинки та відповідного їй якоря моя модель виконує прогнози, спрямовані на визначення наявності об'єкта в цій конкретній області зображення. Окрім цього, модель також робить прогнози щодо конкретного типу об'єкта, який знаходиться в цьому регіоні, а також точних координат цього об'єкта на зображенні. Цей процес прогнозування включає в себе два основних аспекти: класифікацію та визначення області, де розташований конкретний об'єкт. Класифікація об'єкта дозволяє моделі визначити, до якої категорії об'єкт належить, що є важливою інформацією при вирішенні завдань розпізнавання. Далі, визначення області включає в себе точні координати та розміри області, де знаходиться об'єкт. Це важливий аспект для точного позначення місцезнаходження об'єкта на зображенні та надання інформації для подальших обчислень чи дій [9].

Ключовим етапом є здійснення прогнозів для кожної клітинки та якоря, що дозволяє моделі систематично проходити по всьому зображенню та визначати наявність об'єктів у різних його частинах. Це стає основою для розширених можливостей системи в аналізі та виявленні різних типів об'єктів на зображенні. Важливою частиною цього процесу є оптимізація та вдосконалення алгоритмів класифікації та визначення областей. Це дозволяє досягти високої точності прогнозів та забезпечити надійні результати в завданнях розпізнавання об'єктів на зображенні. Усі ці аспекти моделювання взаємодіють для створення повноцінної системи, яка забезпечує точний та ефективний аналіз зображень. Використання класифікації та визначення областей робить систему гнучкою та адаптованою до різних завдань та вимог застосування [10].

3. Фільтрація та об'єднання: Після отримання прогнозів від моделі, вони піддаються фільтрації за певним порогом впевненості. Цей поріг впевненості виступає як критерій відбору результатів, що дозволяє вибрати лише ті прогнози, які відповідають встановленому стандарту точності та надійності. Наступним етапом в процесі постпроцесингу є використання алгоритму об'єднання, відомого як non-maximum suppression (NMS). Цей алгоритм дозволяє ефективно вибрати найбільш точні та впевнені прогнози, враховуючи їх просторові взаємодії та усуваючи дублікати та менш впевнені результати. Механізм NMS базується на принципі вибору прогнозу з найвищим значенням впевненості в області, а потім пригніченні сусідніх прогнозів, які мають значення перекриття з обраною областю. Це дозволяє уникнути непотрібних дублікатів та забезпечити вибір лише найбільш точних та упевнених прогнозів [11].

Застосування NMS стає ключовим кроком в процесі обробки прогнозів, оскільки воно допомагає покращити точність та достовірність результатів. Цей алгоритм є важливим елементом у вирішенні завдань розпізнавання об'єктів, де важливо визначити лише найкращі та найбільш достовірні результати. Такий підхід дозволяє оптимізувати роботу системи та забезпечити високу якість виявлення об'єктів на зображенні. Використання порогу впевненості та алгоритму NMS стає ефективним механізмом відбору та уточнення результатів, що робить систему більш надійною та придатною до реальних умов використання [12].

4. Результат: Остаточний результат включає в себе інформацію про клас, ймовірність класифікації, координати та розміри об'єкта. Ця інформація може бути використана для відображення та взаємодії з розпізнаними об'єктами. Однією з ключових переваг YOLO є його висока швидкість роботи, оскільки він обробляє цілий образ одночасно. Це робить його ідеальним для застосувань у мобільних пристроях, таких як відеоспостереження, автономні транспортні засоби та інші. Важливою особливістю YOLO є його ефективність у виявленні маленьких об'єктів та здатність працювати з різними класами об'єктів [13].

Проте, для використання YOLO потрібні достатньо потужні обчислювальні ресурси, особливо для глибокого навчання. Загальною ідеєю YOLO є спрощення

процесу розпізнавання об'єктів та забезпечення ефективної та точної роботи в режимі реального часу.

Плюси YOLO (You Only Look Once): Швидкість та ефективність: Однією з головних переваг YOLO є його висока швидкість роботи. Алгоритм працює в режимі реального часу, обробляючи зображення в один прохід, що робить його ефективним для застосувань, де важлива швидкість виявлення об'єктів. Глобальний контекст: YOLO виявляє об'єкти на всьому зображенні, що дозволяє враховувати глобальний контекст. Це важливо для точного розпізнавання об'єктів та їх взаємодії на великих територіях. Висока точність виявлення об'єктів: YOLO забезпечує високу точність виявлення об'єктів, включаючи навіть маленькі об'єкти чи ті, що знаходяться далеко від камери. Це важливо для застосувань, де важлива деталізація об'єктів. Універсальність та робастність: YOLO працює добре з різними класами об'єктів і може бути успішно використаний в різних сценаріях, включаючи відеоспостереження, автономні транспортні засоби та інші області. Здатність розпізнавати різні класи: Модель YOLO може розпізнавати багато різних класів об'єктів, що робить його універсальним для застосувань в різних галузях.

Мінуси YOLO: Потребує значних обчислювальних ресурсів: Для ефективної роботи YOLO потребує значних обчислювальних ресурсів, особливо при використанні глибокого навчання. Це може становити виклик для використання на обмежених платформах. Вибір архітектури та гіперпараметрів: Вибір оптимальної архітектури та гіперпараметрів для YOLO може бути завданням, що вимагає експертного досвіду в галузі глибокого навчання.

Обмежена точність для малих об'єктів: Незважаючи на високу точність загалом, YOLO може мати обмежену точність виявлення дуже малих об'єктів або об'єктів з низькою контрастністю. Чутливість до областей з низькою якістю: Алгоритм може виявити вразливість до зображень низької якості або умов освітлення, що може вплинути на точність розпізнавання. Не підходить для обробки великої кількості об'єктів: У випадках, коли на зображенні присутні сотні чи тисячі об'єктів, YOLO може стати менш ефективним через обмеження розмірів сітки та якорів. Незважаючи на ці недоліки, YOLO залишається одним із визнаних та

використовуваних алгоритмів для завдань розпізнавання об'єктів. Розвиток архітектур та оптимізацій може вирішити деякі з цих питань та робить YOLO потужним інструментом для багатьох застосувань

1.2 Аналіз існуючих технологій розпізнавання об’єктів

На рисунку 1.3 наведено візуалізацію алгоритму розпізнавання об’єктів дорожнього трафіку.

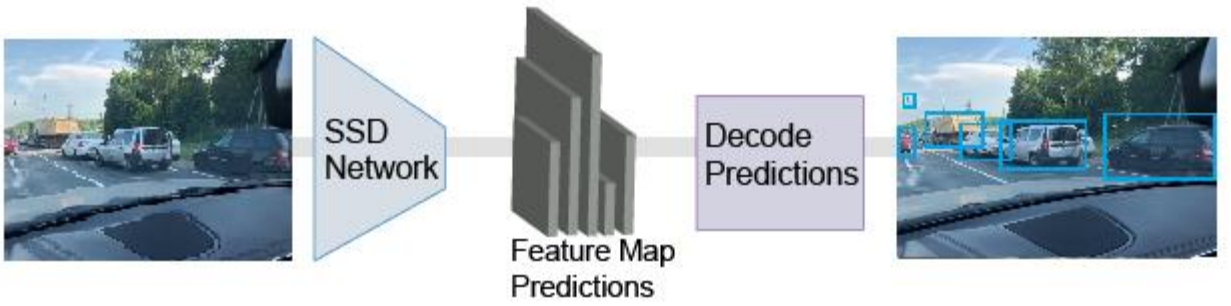


Рисунок 1.3 – Візуалізація алгоритму розпізнавання об’єктів

На вході ми маємо зображення з камери Ця модель, пройшла кропіткий процес навчання на великому наборі даних, щоб вона стала здатною розпізнавати об'єкти на зображеннях. В результаті вона генерує мапу передбачень, яка містить інформацію про розпізнані класи об'єктів та їхні місцезнаходження на зображенні. Ця мапа передбачень може бути великим масивом чисел чи ймовірностей, пов'язаних із кожним класом. Щоб ці дані стали зрозуміліми для подальшої обробки, їх необхідно розшифрувати і перетворити у зручний формат. Тут можуть використовуватися різні техніки, такі як порогові значення для визначення, які об'єкти вважаються розпізнаними, і які - ні. Отримані дані тепер можна використати для побудови нового зображення, яке відображає попереднє, але з виокремленими об'єктами, що були розпізнані на оригінальному зображенні.

Це може включати в себе виділення контурів об'єктів, можливо, навіть їхнє підсвічування чи розфарбовування для визначення їхньої ролі на зображенні. Цей

процес має потенціал для використання в різних областях, таких як комп'ютерне зорове сприйняття, розпізнавання об'єктів у великому масштабі, а також у вирішенні завдань, пов'язаних із сегментацією об'єктів на зображеннях. Він відкриває нові можливості для покращення автоматизації та аналізу великих обсягів візуальної інформації, що є особливо важливим у сучасному світі, насиченому величезними обсягами даних. Детальніше перейдемо до роботи нейронної мережі (рис. 1.4) .

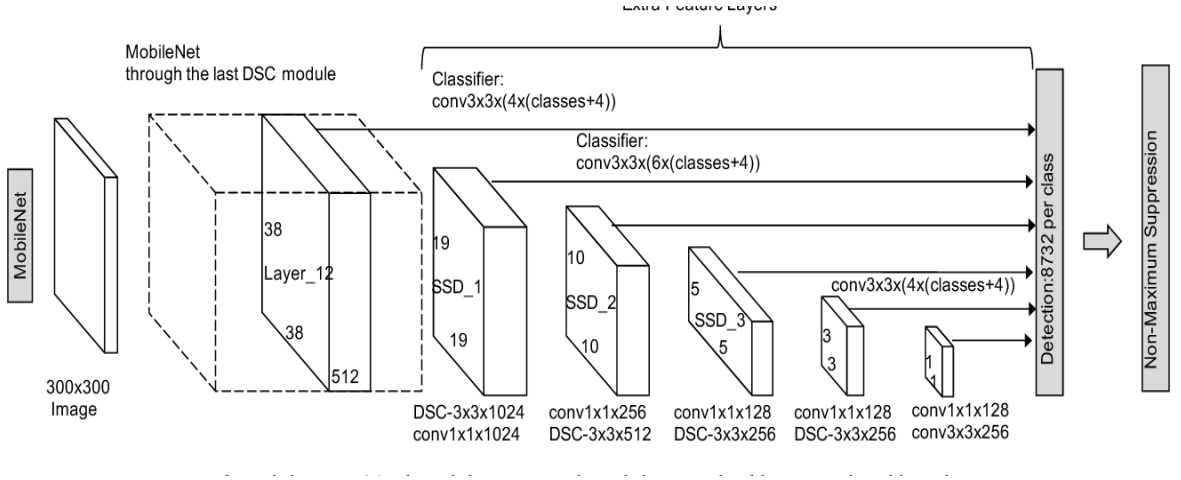


Рисунок 1.4 – Робота моделі

Зображення вже передано. Мережа додає йому глибини для різних ракурсів розпізнавання классу об’єктів. Після того зображення розбивається на пікселі. Мережа аналізує їх, порівнює з даними з датасету(матеріал по якому навчалась модель). Отримуються пікселі, які належать до певного класу з певною вірогідністю. Далі відбувається згортка(convolution) зображення, тобто воно зменшується в ширині та довжині, проте збільшується в глибині. Згорнуте зображення проходить через ті ж самі маніпуляції, описані вище. Ця операція виконується до тих пір, доки розмір згорнутого зображення не буде складати у розмірі 1 на 1 на z, де z - це максимальна глибина, до якої можна згорнути зображення. Після проведення вищезгаданих маніпуляцій та алгоритмів обробки, на виході отримуємо деталізований контур та точний клас об'єкта з вражаючою високою точністю. Зазначено на рисунку 1.4, що при використанні зображення розміром 300 на 300 пікселів можна здійснити розпізнавання об'єкта аж 8732 рази,

визначаючи його з різних сторін та ракурсів. Це вражаюче число вказує на велику точність та вірогідність системи в розпізнаванні об'єктів на зображенні.

Важливою перевагою є здатність системи проводити множинні розпізнавання об'єктів на одному та тому ж зображенні, враховуючи їхні різні аспекти та взаємодію з різних сторін. Такий підхід забезпечує високу ефективність та універсальність системи в різних сценаріях використання. Незалежно від розміру та орієнтації об'єкта на зображенні, система має можливість точного та повного розпізнавання, що робить її дуже потужною у сфері обробки зображень та комп'ютерного зору [14]. Крім того, здатність проводити множинне розпізнавання дозволяє системі працювати з складними та змішаними сценаріями, де присутні різні об'єкти різних типів та розмірів. Це розширює можливості системи та робить її гнучкою та адаптованою до різноманітних завдань розпізнавання. Узагальнюючи, ця висока точність та здатність проводити множинне розпізнавання об'єктів встановлюють систему в лідери в області розпізнавання зображень, де точність та надійність є ключовими факторами для досягнення успіху в різних видах застосування.

1.3 Постановка задачі

Метою магістерської кваліфікаційної роботи є розширення функціональних можливостей програмного забезпечення розпізнавання об’єктів у мобільному застосунку. Задачі, які потрібно розв’язати для досягнення мети:

1. Спроектувати архітектуру програмного забезпечення
2. Розробити функціонал
3. Створити та навчити модель машинного навчання для розпізнавання
4. Протестувати розроблену систему в реальних умовах

Постановка Кроків які потрібно виконати для створення програми в середовищі Flutter для інформаційної технології розпізнавання об'єктів у мобільному застосунку.

1. Налаштування середовища розробки: Встановелння Flutter SDK та налаштування редактор коду, такий як Visual Studio Code або Android Studio, для роботи з Flutter.
2. Створення проекту: Створення нового проеку Flutter за допомогою команди або інтерфейсу командного рядка, що включає в себе потрібні залежності.
3. Дизайн користувацького інтерфейсу: Flutter для створення ефективного та привабливого інтерфейсу користувача, що дозволить взаємодіяти з програмою.
4. Інтеграція з бібліотеками розпізнавання об'єктів: було використано підходящу бібліотеку для розпізнавання об'єктів в реальному часі, таку як TensorFlow або OpenCV, та інтегрування її з проектом Flutter.
5. Розробка алгоритмів розпізнавання: Розроблено алгоритми, що базуються на обраній бібліотеці, для розпізнавання об'єктів.
6. Інтеграція з камерою: функціональність для доступу до камери пристрою. Використання вбудованих можливості Flutter.
7. Обробка та візуалізація даних: Реалізація коду для обробки отриманих зображень та візуалізації результатів розпізнавання.
8. Тестування та налагодження: Перевірка роботи програми на різних сценаріях та відлагодження можливих помилок.
9. Впровадження: Після успішного тестування запуск програми та її встановлення на пристроях.

Загальний процес розробки програми в середовищі Flutter для інформаційної технології розпізнавання об'єктів у мобільному застосунку може бути складним і вимагати глибоких знань у сферах розпізнавання образів, машинного навчання та розробки програмного забезпечення. Техніка розпізнавання об'єктів, що використовується в комп'ютерному зорі, дозволяє впізнавати об'єкти на зображеннях або відео. Цей процес є невід'ємною частиною алгоритмів глибокого навчання та машинного навчання. При перегляді фотографій або відео люди легко

розпізнають і відмічають людей, предмети, сцени та візуальні деталі. На рисунку 1.5 зображено момент розпізнавання об’єктів за допомогою мобільного застосунку.

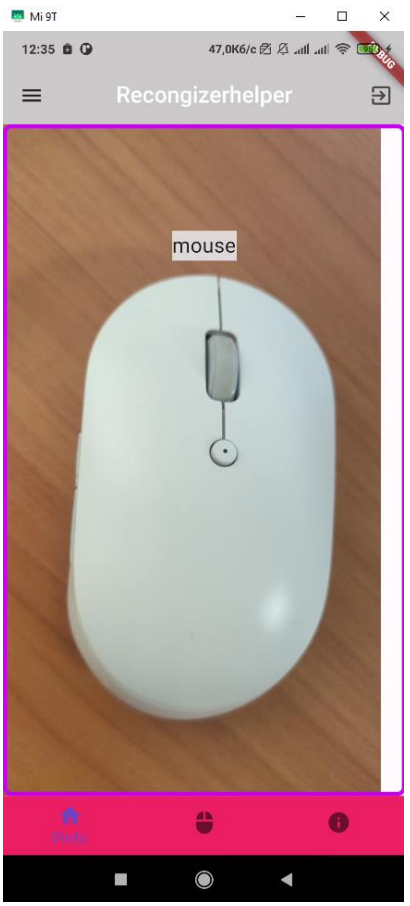


Рисунок 1.5 – Момент розпізнавання об’єкта

Механізм розпізнавання об'єктів базується на тому, як нейронні мережі вивчають репрезентації об'єктів через тренування на великому обсязі даних. Наприклад, для розпізнавання обличчя модель може бути навчена на тисячах фотографій людей з різних ракурсів та умов освітлення. Після тренування мережі вдається визначати унікальні ознаки, які характеризують обличчя, і застосовувати ці знання для розпізнавання на нових зображеннях. Важливим етапом у цьому процесі є попереднє оброблення вхідних даних, яке може включати в себе масштабування, нормалізацію та інші операції для забезпечення ефективного

функціонування мережі. Після цього вхідні дані подаються на вхід до різних шарів нейронної мережі, які відповідають за вивчення різних рівнів абстракцій.

1.4 Висновок до розділу 1

У даному розділі проведено докладний огляд передових мобільних додатків, спрямованих на розпізнавання об'єктів. Розглянуто не лише зовнішній вигляд цих додатків, але і поглиблено вивчено технічні деталі алгоритмів, які забезпечують їхню ефективну роботу. В огляді акцент було зроблено на передових мобільних додатках, які відрізняються високою ефективністю розпізнавання об'єктів. Ці додатки, засновані на передових алгоритмах нейронних мереж, відкривають нові можливості для користувачів у взаємодії з оточуючим світом через екран мобільного пристрою. Зокрема, вони надають можливість точно та швидко розпізнавати різноманітні об'єкти, включаючи предмети побутового вжитку, обличчя та тварин. Поруч із призначеними для користувачів додатками, увага приділяється інноваційним алгоритмам, які лежать в основі цих застосунків.

Розгляд технічних деталей дозволяє вглибитися в роботу нейронних мереж, розкриваючи, як вони навчаються розпізнавати патерни та особливості в зображеннях, а потім використовують ці знання для точної класифікації об'єктів. Детальний розгляд технічних аспектів включає в себе оптимізацію алгоритмів для ефективного використання ресурсів мобільних пристроїв, забезпечуючи високу швидкодію та точність розпізнавання. Аналіз інноваційних підходів до обробки зображень та вивчення великих обсягів даних розкриває динаміку розвитку цих технологій. Завершальний етап розділу присвячений глибокому дослідженню сучасних інновацій у галузі розпізнавання об'єктів та їхнього впливу на повсякденний досвід користувачів. Розгляд технічних взаємодій із додатками та розкриття глибинних механізмів роботи алгоритмів допомагає зрозуміти важливість та вплив цих технологій на сучасний розвиток області розпізнавання об'єктів

2 РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ У МОБІЛЬНОМУ ЗАСТОСУНКУ

2.1 Обгрунтування вибору мови програмування Dart

Dart, розроблена відомою компанією Google, вражає не лише своєю широкою популярністю, але й технічними характеристиками, які роблять її чудовим інструментом для розробників у багатьох областях. Почнемо з того, що Dart визначається своєю універсальністю. Ця мова програмування призначена для створення різноманітних програм, зокрема веб-додатків та мобільних додатків. Можливість компілювати програми на Dart для різних платформ, таких як Windows, Android та iOS, надає розробникам велику гнучкість та ефективність в роботі. Орієнтація на об'єктно-орієнтоване програмування вказує на те, що Dart спрямована на створення програм, де об'єкти грають ключову роль. Це рішення робить код більш структурованим, легким для розуміння та модифікації.

Крім того, вона забезпечує високу рівень абстракції, що полегшує розробку складних систем. Швидкість компіляції та компактність коду є ще однією особливістю Dart. Це дозволяє розробникам ефективно працювати над проектами, роблячи процес розробки швидшим та економічнішим. Акцент Google на створенні ефективної та компактної мови свідчить про їхню вдачливу стратегію в розробці Dart. Відкритість Dart вказує на те, що це не лише інструмент для конкретного вендора, але і мова, що відкрита для спільноти. Це дозволяє розробникам активно співпрацювати, доповнюючи та вдосконалюючи мову.

Залучення спільноти може призвести до появи нових ідей та покращень, збагачуючи екосистему Dart. Підтримка функціонального та реактивного програмування в Dart робить її більш гнучкою та адаптованою до різних стилів програмування [15]. Це важливо в умовах швидко змінюючихся технологій та вимог ринку. Розробники можуть вибирати підхід, який найкраще підходить для їхнього конкретного проекту. Однією з важливих переваг Dart є відсутність надмірної складності. Вона визначається як проста, сучасна та високоефективна мова. Це зробило Dart доступним для широкого кола розробників, навіть для тих, хто не має

глибокого досвіду в програмуванні. Компіляція Dart, схожа на C, робить мову значно швидшою в порівнянні з іншими мовами, зокрема Java. Це важливий аспект для розробників, які цінують продуктивність та ефективність своєї роботи. Типозахищеність Dart грає ключову роль у забезпеченні безпеки програм [16].

Dart допомагає уникнути багатьох типових помилок програмування, підвищуючи якість коду та сприяючи стабільності додатків. Надто важливою є можливість компіляції як з компіляторами AOT, так і з JIT. Це дозволяє розробникам вибирати стратегію оптимізації, яка найбільше відповідає їхнім потребам та характеристикам проекту. Гнучкість у виборі компілятора дозволяє оптимізувати продуктивність додатків для конкретного випадку використання.

У підсумку, Dart стає не лише мовою програмування, але і повноцінним інструментом для розробників, які прагнуть до продуктивної, ефективної та безпечної роботи. Її характеристики дозволяють легко адаптуватися до змін у технологій та вимог ринку, роблячи Dart однією з перспективних мов програмування в сучасному програмувальному середовищі.

2.2 Обгрунтування використання фреймворку Flutter

Flutter, потужний та інноваційний фреймворк розробки, який створений компанією Google, нині займає центральне місце в світі мобільної розробки. Його витоки можна простежити ще з 2015 року, коли під назвою "Sky" він розпочав свій шлях еволюції, перші кроки спрямовані були тільки на Android-додатки. Проте, з плином часу, з серією релізів, Flutter став доступним для розробки не лише під Android, але й під iOS та веб-додатків. Найбільший крок вперед був зроблений у версії 2.0, представленій у березні 2021 року, коли з'явилася підтримка створення десктопних додатків для різних операційних систем, включаючи Windows, macOS, Linux, і досить неочікувана експериментальна підтримка для Google Fuchsia. Однією з найвизначальніших особливостей Flutter є його вражаюча графічна продуктивність. Здатність рендерингу 120 кадрів в секунду робить його не тільки

високоефективним, але й забезпечує бездоганну плавність роботи мобільних додатків. Це особливо актуально в умовах сучасних вимог до візуального дизайну та продуктивності додатків, які постійно зростають. Важливим елементом архітектури Flutter є рушій, який в основному написаний на C++. Це надає йому високу швидкодію та ефективність. Для низькорівневого рендерингу використовується графічна бібліотека Google Skia, що гарантує високу якість візуалізації [17].

Надто, Flutter вміє взаємодіяти з платформозалежними SDK для Android та iOS, що дозволяє розробникам використовувати унікальні можливості кожної платформи для максимальної адаптації своїх додатків. Бібліотека Foundation, яка написана на мові Dart, відіграє ключову роль в системі Flutter. Вона містить основні класи та методи, необхідні для створення додатків на Flutter та ефективної взаємодії з рушієм Flutter. Це забезпечує розробникам необхідні інструменти для швидкої та продуктивної розробки додатків. Важливим аспектом Flutter є його підтримка віджетів, які визначають дизайн користувацького інтерфейсу. Віджети представляють собою незмінні об'єкти, які описують будь-яку частину інтерфейсу користувача, включаючи текст, форми та анімації.

Створення складних інтерфейсів здійснюється за допомогою комбінування простих віджетів, що дозволяє розробникам створювати унікальні та привабливі дизайни. Flutter поставляється з двома основними наборами віджетів: Material Design, який відповідає стилю Google, і Cupertino, який наслідує стиль Apple. Це дозволяє розробникам вибирати стиль і дизайн в залежності від їхніх власних вподобань та вимог конкретного проекту. З розвитком Flutter, його засоби розробки, такі як Flutter DevTools, стають ще потужнішими та більш розширеними. DevTools дозволяє розробникам використовувати різні інструменти для аналізу та оптимізації коду, що сприяє вдосконаленню процесу розробки та виявленню можливих проблем. В кінці кінців, Flutter визначається не тільки своєю технічною потужністю, але і гнучкістю використання, широким спектром підтримуваних платформ і стилів дизайну. Цей фреймворк стає все більш популярним серед розробників завдяки

своїм перевагам у швидкій розробці, високій продуктивності та універсальності у виборі платформ.

2.3 Обгрунтування вибору мови програмування Kotlin

Kotlin - це відносно молода мова від компанії JetBrains. З'явилась вона у 2011 році. На конференції Google I/O 2017 команда розробників Android повідомила, що Kotlin отримала офіційну підтримку для розробки Android-додатків. Мова названа на честь острова Котлін в Фінській затоці, на якому розташоване місто Кронштадт.

Як і Java, C і C ++, Kotlin - це статично типізований мова. Вона підтримує як об'єктно-орієнтоване, так і процедурне програмування.

Ось основні можливості та переваги Kotlin:

- мова дуже проста для вивчення;програми можуть використовувати всі існуючі Java-фреймворки і бібліотеки. Kotlin можна інтегрувати з Maven, Gradle і іншими системами збірки;
- компілюється в байткод JVM або в JavaScript;
- мова null-безпечна - докучливі NullPointerException залишилися в Java.
- вихідний код відкритий;
- в IntelliJ доступна автоматична конвертація Java-коду в Kotlin і навпаки;
- легко читається синтаксис не складе проблем при code review.

Давайте розглянемо деякі аспекти мови Kotlin більш детально. У своєму коді, коли ви спробуєте привласнити або повернути значення null, компілятор Kotlin не дозволить це зробити. Проте, Kotlin підтримує концепцію Nullable-типів, яка дозволяє створювати змінні або функції, які можуть приймати значення null. Це робиться додаванням символу "?" до типу. Крім того, Kotlin пропонує зручність у оголошенні простих функцій та структур. Вони можуть бути оголошені в одному рядку, що полегшує написання простого та лаконічного коду.

— Геттери і сеттери, які використовуються для інтероперабельності з Java-кодом, можуть бути задані "за лаштунками". Додавання анотації data до класу активує автоматичну генерацію різних шаблонів, таких як equals(), hashCode(), toString(), а також геттери і сеттери. У Kotlin були введені спеціальні класи, створені спеціально для зберігання даних. Ці класи автоматично генерують різні шаблони для полегшення рутинних завдань, таких як порівняння, отримання хеш-коду, та інші. Функції-розширення є ще однією потужною можливістю Kotlin, яка дозволяє розширювати функціональність існуючих класів без необхідності у наслідуванні [18]. Вони додають нові можливості до існуючих класів, при цьому зберігаючи чистоту коду та низьку залежність. Компілятор Kotlin вражає своєю розумінням контексту щодо приведення типів.

Оператор is використовується для автоматичного визначення типу, і часто не потрібно явно вказувати оператори приведення. Це полегшує роботу з типами та робить код більш зрозумілим. Однією з ключових рис Kotlin є його функціональна природа. Велика кількість корисних можливостей, таких як функції вищого порядку, лямбда-вирази, перевантаження операторів і "ледачі" обчислення логічних виразів, робить Kotlin привабливим вибором для фанатиків функціонального програмування. Коли створювалася мова Kotlin, її автори покладали за мету створити щось більш лаконічне і типобезпечне, ніж Java, та просте і легше для вивчення, ніж Scala. І вони справилися з цим завданням, забезпечивши більш швидку компіляцію та кращу підтримку в середовищі розробки.

Більш того, Kotlin є повністю сумісним з Java, що дозволяє поступово переходити від Java до Kotlin, а також вбудовується в Android, що забезпечує простий спосіб використання Kotlin для розширення функціональності існуючих Android-додатків. У заключенні, Kotlin вирізняється своєю лаконічністю, безпечністю типів, великою кількістю функціональних можливостей і плавною інтеграцією з іншими технологіями, забезпечуючи розробникам ефективні та продуктивні інструменти для творчості.

2.4 Обгрунтування використання СКБД Firebase Realtime Datebase

Firebase — це вражаюча платформа для розробки мобільних додатків, що перетворилася зі стартапу в інноваційний інструмент для розробників кросплатформених застосунків. Одна з найбільших переваг цієї платформи — можливість розробникам уникаючи створення бекенду, або серверної частини проекту, зосередитися на створенні високоякісних UX/UI для своїх додатків. Firebase пропонує широкий функціонал, який включає в себе різноманітні сервіси і можливості, що значно полегшують розробку та покращують досвід користувачів. Використання Firebase разом з фреймворком Flutter від Google стає ключовим аспектом для створення продуктивних та ефективних мобільних додатків для платформ Android і iOS. Firebase, у свою чергу, є представником категорії BaaS (Backend as a Service), що надає розробникам доступ до широкого спектру функцій. Це не тільки сервер і база даних, а й хостинг, аутентифікація та інші сервіси в єдиній інтегрованій платформі. Використання Firebase Realtime Database дозволяє синхронізувати дані між різними клієнтами та зберігати їх в хмарному сховищі [19].

При взаємодії додатку з базою даних Firebase використовується WebSocket для забезпечення ефективної синхронізації даних протягом усього сеансу. Це важливий момент, оскільки забезпечує стійку і швидку обмін інформацією між клієнтами та сервером. Крім того, Firebase пропонує різноманітні інструменти для аутентифікації, що робить процес управління користувачами безпечним та ефективним. Ідентифікація, авторизація та інші аспекти безпеки вирішуються на рівні платформи, дозволяючи розробникам уникнути великої частини зайвого коду та витрат на забезпечення безпеки свого додатку [20]. Firebase також надає сервіс хостингу, що дозволяє розміщувати веб-сайти та додатки в хмарі, забезпечуючи надійний і швидкий доступ до них для користувачів. Це важливо для покращення продуктивності та доступності додатків. Однією з ключових переваг Firebase є також можливість аналізу даних за допомогою Firebase Analytics. Збір та аналіз даних про використання додатку дозволяє розробникам зрозуміти поведінку

користувачів та оптимізувати додаток відповідно до їхніх потреб. Важливо відзначити, що Firebase і Flutter взаємодіють дуже ефективно, забезпечуючи розробникам інструменти для швидкої та продуктивної роботи. Ця інтеграція дозволяє розробникам миттєво взаємодіяти з хмарними сервісами, обробляти дані та забезпечувати користувачам безперебійний та надійний досвід. Firebase не лише спрощує розробку, але і впливає на загальну продуктивність команд та швидкість впровадження нових функцій та змін. Завдяки цій платформі розробники можуть концентруватися на тому, що їм подобається робити найбільше — створювати вражаючі інтерфейси та функціонал для користувачів. У сучасному світі, де темпи технологічного розвитку дуже високі, використання таких інтегрованих платформ, як Firebase, стає стратегічно важливим для ефективної та конкурентоздатної розробки мобільних додатків.

2.5 Обгрунтування вибору середовища розробки Visual Studio Code

Visual Studio Code — це безкоштовний текстовий редактор коду, розроблений компанією Microsoft для операційних систем Windows, Linux і macOS. Основні риси цього редактора включають підтримку налагодження, виокремлення синтаксису, інтелектуальне автозаповнення коду, використання сніпетів, можливість рефакторингу коду та вбудовану інтеграцію з системою контролю версій Git. Великою перевагою Visual Studio Code є його доступність для користувачів безкоштовно. Однією з особливостей цього редактора є можливість налаштовування теми оформлення, комбінацій клавіш, різноманітних налаштувань та встановлення розширень, що значно розширюють його базовий функціонал. У результатах опитування серед розробників Stack Overflow 2019 Visual Studio Code заслужено отримав статус найпопулярнішого інструменту середовища розробки. За результатами опитування, що взяли участь 87317 респондентів, понад половина, а саме 50,7%, визнали, що вони користуються цим редактором. Це свідчить про широке визнання та популярність серед розробників у всьому світі [21]. Visual

Studio Code надає зручний та ефективний інтерфейс для роботи з кодом, дозволяючи розробникам швидко та ефективно втілювати свої ідеї в програмному коді.

Інтеграція з Git робить процес контролю версій більш зручним та прозорим. Окрім базового набору функцій, користувачі можуть активно користуватися розширеннями, які додають різноманітні функції та інструменти до Visual Studio Code. Це дозволяє адаптувати редактор під конкретні потреби розробника та оптимізувати його для конкретного виду роботи. У світлі зростаючого інтересу до розробки програмного забезпечення, Visual Studio Code виступає як надійний інструмент для професіоналів у галузі інформаційних технологій. Його поєднання з високою функціональністю та простотою використання робить його привабливим вибором для широкого кола розробників.

2.6 Середовище розробки Visual Studio Code

Основні риси цього редактора включають підтримку налагодження, виокремлення синтаксису, інтелектуальне автозаповнення коду, використання сніпетів, можливість рефакторингу коду та вбудовану інтеграцію з системою контролю версій Git. Великою перевагою Visual Studio Code є його доступність для користувачів безкоштовно. Однією з особливостей цього редактора є можливість налаштовування теми оформлення, комбінацій клавіш, різноманітних налаштувань та встановлення розширень, що значно розширюють його базовий функціонал. У результатах опитування серед розробників Stack Overflow 2019 Visual Studio Code заслужено отримав статус найпопулярнішого інструменту середовища розробки. За результатами опитування, що взяли участь 87317 респондентів, понад половина, а саме 50,7%, визнали, що вони користуються цим редактором. Це свідчить про широке визнання та популярність серед розробників у всьому світі. Visual Studio Code надає зручний та ефективний інтерфейс для роботи з кодом, дозволяючи розробникам швидко та ефективно втілювати свої ідеї в програмному коді.

Інтеграція з Git робить процес контролю версій більш зручним та прозорим. Окрім базового набору функцій, користувачі можуть активно користуватися розширеннями, які додають різноманітні функції та інструменти до Visual Studio Code. Це дозволяє адаптувати редактор під конкретні потреби розробника та оптимізувати його для конкретного виду роботи. У світлі зростаючого інтересу до розробки програмного забезпечення, Visual Studio Code виступає як надійний інструмент для професіоналів у галузі інформаційних технологій. Його поєднання з високою функціональністю та простотою використання робить його привабливим вибором для широкого кола розробників.

2.7 Обгрунтування використання Google Teachable Machine

Google Teachable Machine відкриває широкі можливості для тренування моделей нейронної мережі, спрощуючи процес розпізнавання звуків та зображень для користувачів різного рівня технічної підготовки. Цей інноваційний сервіс вирізняється своєю доступністю та інтуїтивним інтерфейсом, що не вимагає глибоких знань мов програмування, зокрема Python. Основна функціональність Google Teachable Machine включає в себе можливість тренування моделей для класифікації зображень та звуків. Це стає вельми цінним інструментом для розробників, дослідників та навчальних закладів, які хочуть ефективно використовувати можливості нейронних мереж у своїх проектах. Сервіс працює у тісному взаємодії з фреймворками, що взаємодіють з TensorFlow, використовуючи моделі TensorFlow Lite. Це забезпечує високий рівень сумісності та дозволяє розробникам використовувати широкий спектр інструментів для подальшого аналізу та оптимізації їхніх моделей. Унікальною особливістю Google Teachable Machine є можливість експорту моделей у різноманітні формати. Це робить сервіс гнучким і відкритим для використання в різних середовищах розробки, таких як Coral та Arduino. Це особливо важливо для розробників, які ведуть проекти у галузі вбудованих систем та робототехніки.

Використання Teachable Machine може знайти широке застосування в різних сферах, включаючи обробку зображень, розпізнавання об'єктів, аудіокласифікацію, аналіз тексту та багато іншого. Від простих експериментів та навчання до застосувань в промисловості та науці, Google Teachable Machine відкриває двері для креативного використання машинного навчання. Важливо відзначити, що цей сервіс також може стати потужним інструментом для викладання основ машинного навчання та штучного інтелекту. Він дозволяє студентам, вчителям та дослідникам експериментувати з концепціями машинного навчання в невимушеному середовищі та розвивати свої навички у цьому сучасному напрямку технологій. У підсумку, Google Teachable Machine визначається не лише своєю потужністю та різноманітністю функцій, але і своєю легкістю використання та доступністю. Це інструмент, що відкриває нові можливості для творчого використання машинного навчання та демократизує доступ до цих технологій для широкого кола користувачів.

2.8 Висновок до розділу 2

У цьому розділі розглянемо різноманітні засоби розробки мобільних застосунків, зосереджуючись на передових технологіях та популярних інструментах, що забезпечують швидку та ефективну розробку нативних додатків. Однією з передових технологій в цьому контексті є Dart та Flutter. Dart - це мова програмування, розроблена компанією Google, яка стала ключовим елементом для створення мобільних додатків. У поєднанні з фреймворком Flutter, Dart надає зручний та продуктивний інструментарій для розробки платформонезалежних застосунків для Android і iOS. Flutter відзначається високою графічною продуктивністю та можливістю рендерингу 120 кадрів в секунду, роблячи його одним із найефективніших фреймворків. Крім того, для розробки Android-додатків використовується мова програмування Kotlin. Kotlin став популярним вибором серед розробників завдяки своїй експресивності та зручності в порівнянні з Java. Він

володіє чудовою сумісністю з Java, що дозволяє поступово переходити від однієї мови до іншої без значних зусиль. Коли мова йде про зберігання та обробку даних, вибір часто припадає на Firebase Realtime Database.

Це нереляційна база даних, яка надає розробникам API для синхронізації даних між клієнтами та зберігання їх у хмарному сховищі. Використання Firebase Realtime Database дозволяє позбутися необхідності вручну конфігурувати серверний код, що робить розробку більш швидкою та менш витратною.

Був проведений аналіз існуючих мобільних додатків, і виявлено, що розглянуті застосунки вирізняються значною різноманітністю та обширним спектром додаткового функціоналу. Однак слід відзначити, що виявилися обмеження у використанні технологій доповненої реальності та нейронних мереж, оскільки їхні можливості були майже невикористані чи застосовані обмежено. Під час проектування архітектури програмного забезпечення було акцентовано на створенні ефективних підсистем розпізнавання об'єктів. Розроблений функціонал використовував потужні інструментальні засоби TensorFlow, що забезпечило високу точність та швидкість розпізнавання. Кожна підсистема взаємодіє гармонійно, утворюючи вбудовану мережу для комплексного аналізу об'єктів. Такий підхід дозволяє створити потужне інструментаріум для розпізнавання різних об'єктів у реальному часі, що робить програмну систему надійним інструментом для вирішення завдань обробки зображень та взаємодії з оточенням. Також була додана можливість для навчання основам англійської мови для співробітників, використовуючи інформаційну систему розпізнавання об'єктів. Зокрема, база розпізнавання об'єктів у застосунку на основі TensorFlow була розширена для ефективного визначення різноманітних об'єктів. Після розробки системи вона пройшла етап тестування в реальних умовах для перевірки її працездатності та надійності.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ У МОБІЛЬНМУ ЗАСТОСУНКУ

3.1 Розробка архітектури програмного забезпечення розпізнавання об'єктів у мобільнму застосунку

Підсистема розпізнавання об'єктів використовує потужну модель машинного навчання TensorFlow Lite, а саме "SSD MobileNetV2", яка була натренована на різних датасетах для досягнення високої ефективності. Один із таких датасетів - Bosch Small Traffic Lights Dataset від Гейдельберзького Університету, включає понад 5 тисяч фотографій світлофорів, розташованих на значній відстані від камери смартфону. Додатково, використовуються crosswalk–dataset від Kaggle та Stanford Cars Dataset від Стенфордського університету, які складаються із більше ніж 4 тисяч зображень пішоходів та приблизно 8 тисяч зображень автомобілів відповідно.

Під час активації режиму розпізнавання, користувач вмикає камеру, яка працює у режимі ImageStream. Це означає, що велика кількість знімків фіксується та зберігається. Отримані фотографії поділяються на багато точок, які потім піддаються аналізу з використанням натренованої моделі. Результатом обробки стає інформація про detectedClass (назва розпізнаного об'єкта автомобільного трафіку), confidenceInClass (ймовірність правильності визначення) та масив POI (points of interest), який представляє собою координати точок, пов'язаних з визначеним об'єктом. Ці координати важливі для того, щоб система могла підсвічувати розпізнані об'єкти прямо на екрані камери, забезпечуючи користувачеві зручність і точність при взаємодії з інтерфейсом.

Такий підхід до розпізнавання об'єктів автомобільного трафіку дозволяє не лише надійно ідентифікувати елементи, а й взаємодіяти з ними в режимі реального часу, що робить цю підсистему ефективним інструментом для застосувань в сфері технологій безпеки та навігації [22].

3.2 UML діаграма класів

Діаграма класу UML - це графічна нотація, що використовується для побудови та візуалізації об'єктно-орієнтованих систем. Діаграма класів в Уніфікованій мові моделювання (UML) - це тип статичної структурної діаграми, що описує структуру системи, візуалізуючи її:

- класи,
- їх атрибути,
- операції (або методи),
- та взаємозв'язки між об'єктами.

Класи розробленого програмного продукту і взаємозв’язки між ними показано на рисунку 3.1.

Основні класи програми:

- Клас MyApp — містить функцію main(), що є точкою старту додатку.
- LoginScreen та SignUpScreen —класи, що містять відображення сторінок авторизації та реєстрації відповідно, а також методи для їх здійснення.
- HomeScreen — клас, що є головною сторінкою, і містить мапу для базової навігації.
- SearchScreen — клас, що містить вікно пошуку та вибору кінцевої точки маршруту, а також методи для їх функціонування.
- LiveCamera - клас, що має методи для розпізнавання за допомогою моделі нейронної мережі.

AppData — клас, що містить методи для відправлення запитів та отримання відповіді на/від хмарних API та їх направлення до різних класів програмного застосунку.

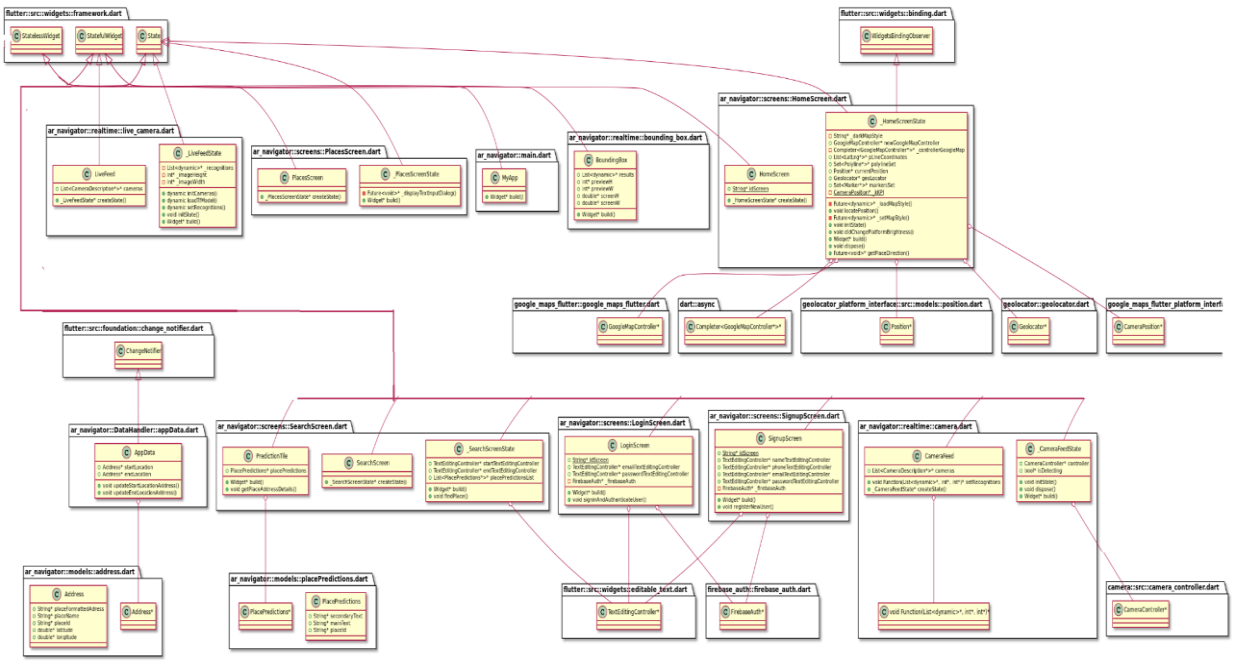


Рисунок 3.1 - Діаграма класів

Діаграма розгортання, що належить до родини діаграм UML (Unified Modeling Language), становить потужний інструмент для візуалізації архітектури виконання системи. Цей тип діаграми розглядає систему як сукупність взаємодіючих вузлів, які включають середовища виконання апаратного та програмного забезпечення, а також проміжне програмне забезпечення, яке забезпечує їхню взаємодію та співпрацю. Діаграми розгортання є ефективним інструментом для відображення фізичної структури системи, дозволяючи проектувальникам та розробникам отримати чітке уявлення про те, як система буде функціонувати та взаємодіяти на різних рівнях апаратного та програмного забезпечення.

У контексті діаграм розгортання важливо визначити основні елементи, які вони включають. По-перше, це вузли, які представляють апаратне або програмне обладнання. Апаратні вузли можуть включати сервери, комп'ютери, мобільні пристрої та інші фізичні пристрої, які використовуються для виконання системи. З іншого боку, програмні вузли вказують на програмне забезпечення, яке працює на цих апаратних пристроях. Другий важливий елемент - це зв'язки між вузлами. Ці зв'язки показують, як вузли взаємодіють та обмінюються даними. Вони можуть

визначати мережеві з'єднання, комунікаційні канали, а також інші аспекти взаємодії між різними елементами системи. Діаграми розгортання зазвичай використовуються для кількох ключових цілей. По-перше, вони допомагають зрозуміти фізичну структуру системи, включаючи розташування апаратного та програмного забезпечення. Це особливо корисно при проектуванні розподілених систем або систем, які використовують різні пристрої для виконання своїх функцій. Крім того, діаграми розгортання можуть слугувати інструментом для аналізу продуктивності системи. Вони дозволяють визначити, які частини системи можуть стати Eng bottlenecks та як можна оптимізувати їх продуктивність.

Масштабованості системи, вони надають можливість оцінити, як система може розширюватися або змінюватися у майбутньому, щоб відповідати зростаючим потребам та вимогам. У контексті проектування програмного забезпечення, діаграми розгортання UML служать важливим кроком у створенні прозорого та структурованого плану виконання системи [23]. Вони визначають, як компоненти системи фізично розташовані та взаємодіють один з одним для досягнення конкретних цілей. Це важливо, оскільки правильна архітектура розгортання може суттєво вплинути на ефективність, масштабованість та безпеку системи.

Однією з ключових переваг діаграм розгортання UML є їхня здатність передати інформацію візуально, що є особливо важливим при спілкуванні між різними членами команди проекту. Вони створюють єдиний мовний код, який зрозумілий як проектувальникам, так і розробникам, забезпечуючи єдність в сприйнятті архітектурних рішень. Діаграми розгортання дозволяють визначити, як фізично розташовані або розподілені різні елементи системи. Це може включати в себе сервери, комп'ютери, зовнішні сервіси, мережеві елементи та інші фізичні ресурси. Завдяки цьому вони стають ефективним інструментом для планування і оптимізації інфраструктури системи. При проектуванні систем, які мають велику кількість компонентів та взаємодіють між собою, діаграми розгортання UML виявляються особливо корисними. Вони надають зрозумілу та структуровану інформацію про конфігурацію системи, дозволяючи вдосконалити взаємодію між

різними елементами. Окрім того, діаграми розгортання UML дозволяють виявити можливі точки витоку, аналізуючи, як дані та інші ресурси переміщаються між різними частинами системи. Це допомагає покращити безпеку та ефективність системи, ідентифікуючи потенційні проблеми та забезпечуючи їх попередження. У світлі сучасних тенденцій у розробці програмного забезпечення, де міграція на хмарні рішення та використання мікросервісної архітектури стають все більш поширеними, діаграми розгортання UML стають необхідним інструментом для розуміння та управління розподіленою інфраструктурою системи. Взагалі, ці діаграми допомагають створювати єдиний стандарт для команди проекту, сприяючи легшому взаєморозумінню та співпраці. Їх використання не лише спрощує комунікацію, але і допомагає попереджати можливі проблеми на етапі проектування, що веде до більш ефективних та стабільних систем в подальшому.

Загалом, діаграми розгортання UML виступають як невід'ємна складова при створенні складних програмних систем, забезпечуючи чітку та візуально доступну інформацію. Вони є ключовим елементом успішного проектування та розгортання інформаційних систем, допомагаючи командам ефективно взаємодіяти та досягати високого рівня функціональності та безпеки. Використання діаграм розгортання UML дозволяє здійснювати оптимальний розподіл ресурсів. Архітектори систем можуть аналізувати фізичне розташування компонентів, забезпечуючи ефективне використання апаратного забезпечення та мережевих ресурсів. Це стає особливо важливим у великих системах, де оптимізація може значно підвищити продуктивність та швидкість роботи. Наприкінці діаграми розгортання UML є також потужним інструментом для документування системи. Вони створюють зручний та доступний довідник, який може використовуватися як для внутрішньої команди розробників, так і для зовнішніх сторін, що взаємодіють з системою.

На рисунку 3.2 зображено 4 основних вузли: смартфон, сервер бази даних (Firebase RealTime Database), 2 хмарних API та їх основні компоненти.

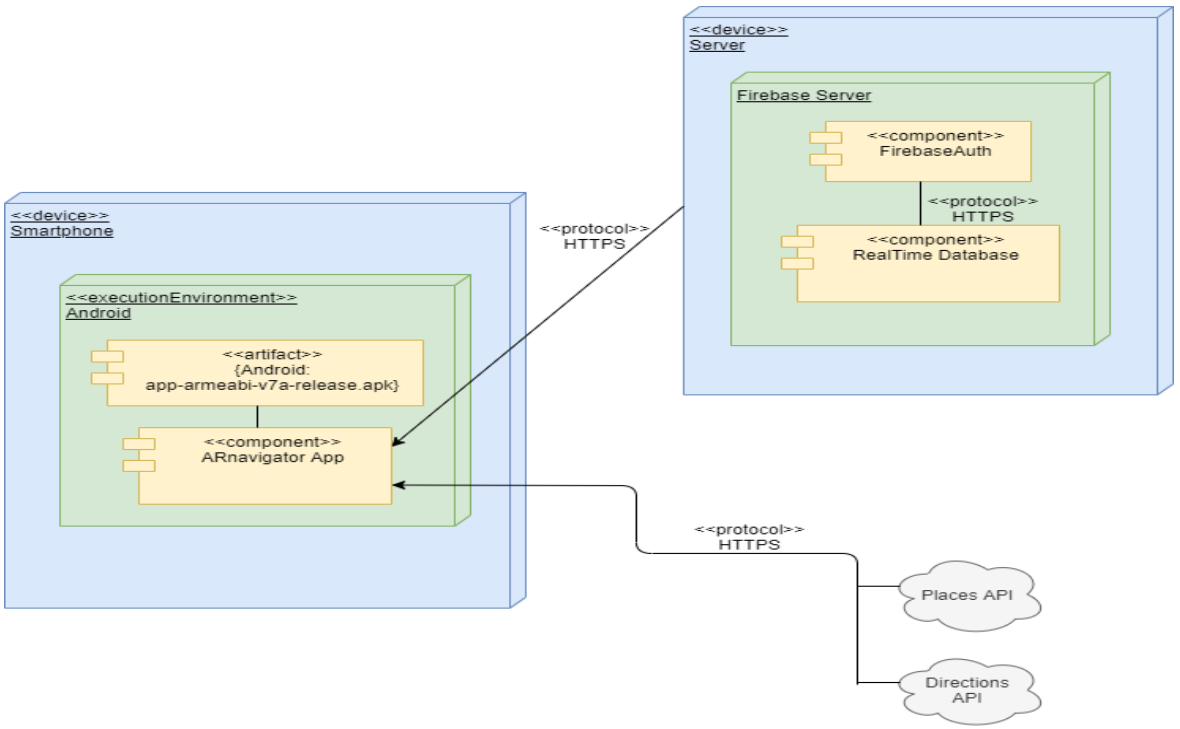


Рисунок 3.2 — Діаграма розгортання

Операційна система Android - середовище виконання мобільного додатку. Для економії пам’яті смартфону використовуються хмарні API - “Places API” та “Directions API”. Places API використовується для того, щоб повернути результати пошуку географічного місця за запитом від користувача, а Directions API - для того, щоб отримати всі проміжні точками між початком та кінцем маршруту користувача. З’єднання з базою даних та хмарними API відбувається за допомогою протоколу HTTPS — комбінації протоколів SSL/TLS і HTTP. Він забезпечує зашифровану і безпечну ідентифікацію мережевого сервера та з’єднання між додатком та сервером.

3.3 Опис структури бази даних програмного забезпечення розпізнавання об’єктів у мобільному застосунку

В архітектурі даного програмного продукту визначено наявність лише однієї бази даних, яка інкорпорована у систему для зберігання та обробки даних користувачів. Ця база даних створена за допомогою нереляційної системи

керування базами даних (СКБД) Firebase RealTime Database, яка володіє унікальними особливостями та можливостями для забезпечення високої ефективності та гнучкості обробки інформації. Структура цієї бази даних відображена на рисунку 3.3, де кожен елемент відіграє важливу роль у забезпеченні функціональності програмного продукту.

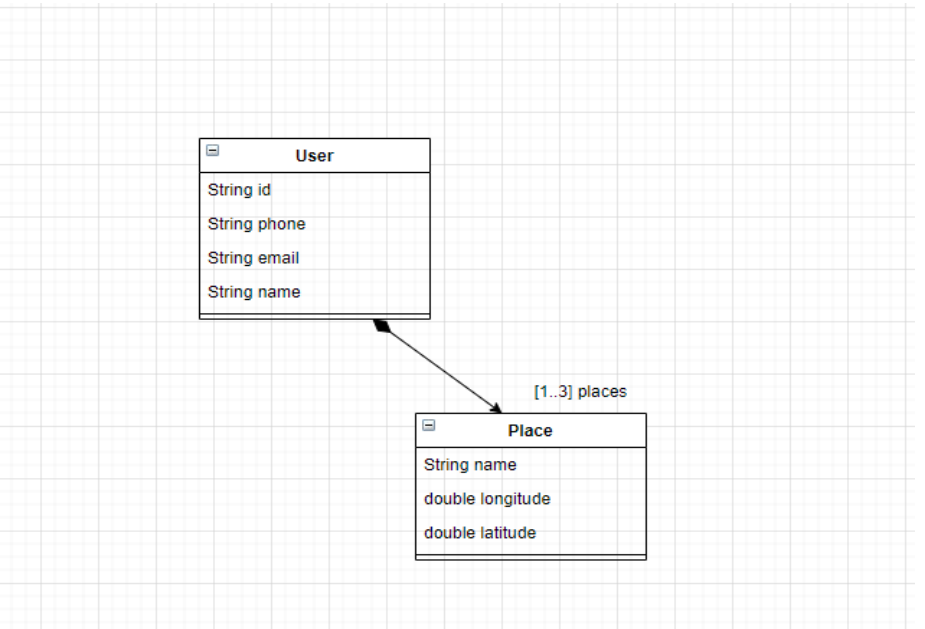


Рисунок 3.3 – Структура БД

Нереляційний характер цієї бази даних визначається її спрощеною структурою, орієнтованою на збереження та передачу даних у режимі реального часу, що є особливо важливим для інтерактивних та динамічних додатків [24]. Однією з ключових особливостей Firebase RealTime Database є можливість зберігання даних у форматі JSON, що сприяє простоті та легкості розробки, а також забезпечує динамічність та швидкий доступ до інформації. У цій базі даних визначені різні вузли, які відповідають конкретним аспектам користувачів та їхнім даним. Важливою частиною структури бази даних є взаємозв'язки між різними вузлами, які дозволяють ефективно організовувати та керувати даними. Відносини, визначені між різними елементами бази даних, створюють консистентність та забезпечують логічну організацію інформації. Одним із ключових аспектів бази

даних є здатність забезпечувати реальний час обмін даними. Firebase RealTime Database дозволяє програмному продукту миттєво отримувати та оновлювати дані, що робить його особливо підходящим для додатків, де актуальність інформації відіграє ключову роль. Окрім того, структура бази даних враховує особливості взаємодії з користувачами. Дані користувачів організовані відповідно до їхніх профілів та основних характеристик, забезпечуючи систематизацію та легкий доступ до важливих даних. Ця база даних є важливою складовою програмного продукту, оскільки вона не лише забезпечує зберігання інформації, а й дозволяє взаємодіяти з нею у режимі реального часу. Firebase RealTime Database стає надійною та ефективною основою для забезпечення функціональності продукту, що відповідає вимогам сучасних та високопродуктивних додатків.

При реєстрації в системі, кожен користувач представлений ім'ям, адресою електронної пошти та контактним номером телефону, що визначає його основні ідентифікаційні характеристики. Крім основних персональних даних, користувач також має можливість зберегти три ключові точки, які відображають його звичайні місця перебування: "Дім", "Робота" та "Навчання". Кожна з цих точок характеризується двома параметрами - довготою та широтою. Значення цих точок встановлюються за замовчуванням у 0 під час реєстрації, що вказує на їхню ініціалізацію. Такий підхід є зручним та логічним, оскільки на початковому етапі користувач може не мати конкретних координат для своїх збережених точок. Це також дозволяє відокремити новозареєстрованих користувачів від тих, які вже використовують систему та мають встановлені власні координати для ключових місць. Потрібно враховувати, що інформація про користувача, така як ім'я, пошта та телефон, є обов'язковою для реєстрації, і вона дозволяє ефективно ідентифікувати та взаємодіяти з кожним користувачем в межах системи. Водночас, можливість збереження та оновлення точок, таких як "Дім", "Робота" та "Навчання", розширює функціональність додатку, дозволяючи користувачам ефективно планувати свої переміщення та використовувати цю інформацію для різних функціональних можливостей. Важливим є і те, що обрані значення за замовчуванням (0) не тільки є практичними для ініціалізації, але й вони можуть бути подальше оновлені

користувачем, коли він встановить конкретні координати для своїх ключових місць. Це надає гнучкість та забезпечує можливість користувачеві актуалізувати інформацію у відповідності зі змінами в його повсякденному житті та рухомості. Кожна збережена точка, таким чином, стає не лише символічним позначенням, але й джерелом конкретних географічних координат, які можуть використовуватися для різних функцій у програмному продукті. Наприклад, це може включати в себе автоматичне визначення маршрутів або надання індивідуальних порад щодо поблизьких подій та послуг. Усі ці аспекти об'єднуються для створення комплексного та зручного середовища для користувачів, де основна інформація профілю поєднується з можливістю визначати та актуалізувати власні ключові точки для ефективного використання додатку у повсякденному житті.

3.4 Розробка алгоритму розпізнавання об’єктів у мобільному застосунку

У графічному зображенні, що представлене на рисунку 3.4., відображено блок-схему алгоритму, призначеного для ефективного розпізнавання об'єктів. Ця схема відображає послідовні етапи та кроки, які виконуються в процесі аналізу зображення з метою ідентифікації та класифікації різноманітних об'єктів. Ілюстрація надає компактний та зрозумілий огляд того, як весь алгоритм взаємодіє та функціонує для досягнення поставленої мети. На цьому графічному представленні виділені окремі блоки, кожен з яких відповідає конкретному етапу або операції в алгоритмі розпізнавання об'єктів. Це робить можливим аналіз кожного кроку і визначення його впливу на загальний процес. Блок-схема дозволяє зорганізувати логіку алгоритму у легкозрозумілий та структурований спосіб, сприяючи якісній взаємодії та зрозумінню для розробників та інших зацікавлених сторін. Кожен елемент блок-схеми виконує конкретну роль у розпізнаванні об'єктів, починаючи з вхідного зображення та завершуючи результуючим визначенням класів об'єктів. Такий візуальний підхід дозволяє оперативно виявити послідовність операцій та їх взаємозв'язок, що сприяє аналізу та оптимізації алгоритму. Не лише

етапи розпізнавання, але й можливі умови, рішення та переходи між ними представлені на графіці, надаючи повний огляд динаміки алгоритму. Завдяки візуальному оформленню, блок-схема стає зручним інструментом для обговорення та уточнення деталей алгоритму, що може сприяти подальшій оптимізації та удосконаленню.

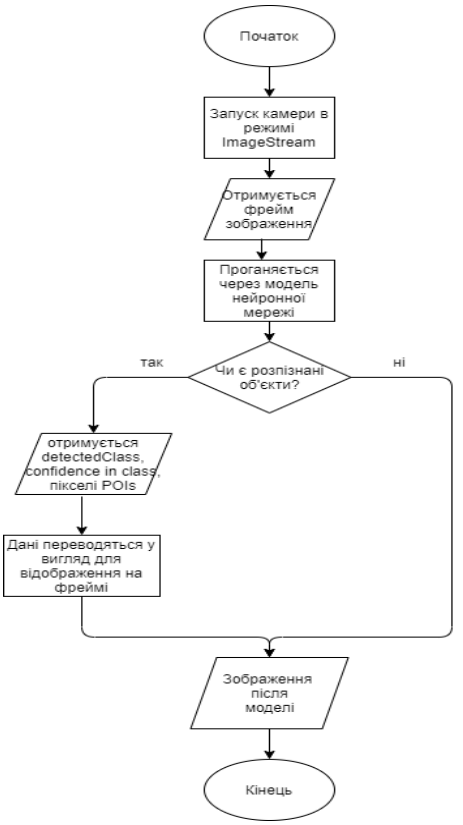


Рисунок 3.4 – Алгоритм розпізнавання об’єктів

При взаємодії з функціоналом програмного продукту, користувач має можливість запускати застосунок, спускаючись шляхом натискання на кнопку "Вільна подорож". Це відкриває нове вікно, яке інтегрує камеру смартфону та активує режим ImageStream. Основна ідея цього режиму полягає в розбитті відеопотоку на значну кількість фреймів, тобто окремих знімків з відео. Після отримання цих фреймів, кожен з них піддається аналізу за допомогою натренованої моделі машинного навчання. У випадку, коли модель розпізнає об'єкти на фреймі, вона проводить подальший аналіз та повертає важливу інформацію. Серед цих даних можна виділити клас розпізнаного об'єкта, рівень впевненості у точності

розпізнавання та координати пікселів, які відповідають даному класу об'єкта на зображенні. Отримані дані переходять до класу BoundingBox, де вони проходять подальше оброблення та відображаються на екрані смартфону. Важливою частиною цього процесу є представлення інформації у вигляді огороджених прямокутників (BoundingBox), які надають візуальне відображення розпізнаних об'єктів та їхніх меж. В разі, якщо модель машинного навчання не розпізнає об'єкти на певному фреймі, зображення просто відображається у додатку без будь-якої модифікації. Це створює плавний та природний досвід для користувача, де система реагує лише у випадку виявлення цікавих об'єктів.

Цей підхід до обробки зображень та розпізнавання об'єктів може бути особливо корисним в ситуаціях, коли користувач бажає отримати інформацію про навколишнє середовище навколо себе. Загалом, ця функціональність створює інноваційну можливість для користувачів не лише фотографувати світ навколо себе, але й автоматично отримувати інформацію та контекст, пов'язаний з розпізнаними об'єктами, що відкриває нові перспективи для інтерактивного та цікавого взаємодії з технологією.

3.5 Висновок до розділу 3

У цьому розділі висвітлено архітектуру мобільного застосунку, де розглядаються ключові компоненти системи. Надається діаграма класів з докладним описом основних класів, що є визначальними для функціональності системи. Крім того, приводиться діаграма прецедентів, яка відображає взаємодію користувачів з системою та обсяг їхніх можливостей. Також включена діаграма розгортання, яка надає уявлення про те, як компоненти системи фізично розташовані та взаємодіють один з одним. Окрім того, в цьому розділі розглядається структура бази даних, яка використовується в мобільному застосунку. Описуються таблиці та зв'язки між ними, що визначають спосіб організації та збереження даних.

4 ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ У МОБІЛЬНМУ ЗАСТОСУНКУ

4.1 Опис системних вимог

Для ефективного та безперебійного використання нашого мобільного додатку важливо враховувати ряд обов'язкових вимог під час його інсталяції та експлуатації. Застосунок спеціально розроблено для операційної системи Android, і його оптимальна робота гарантована лише на пристроях, що працюють під управлінням цієї платформи. Перш за все, переконайтеся, що ваш пристрій має версію операційної системи Android не нижче Android 9 Pie. Це є обов'язковим кроком для інсталяції, оскільки додаток використовує функціональність, яка підтримується тільки у більш нових версіях операційної системи. Додатково, переконайтеся, що на вашому пристрої є достатньо вільного місця, щоб забезпечити оптимальну роботу додатку. Рекомендується мати не менше 90 мегабайт вільного простору на вашому пристрої для ефективної інсталяції та подальшої експлуатації.

Це важливо враховувати, оскільки недостатній обсяг вільного простору може призвести до затримок у роботі додатку та нестабільності його функціонування. Окрім цього, необхідно забезпечити доступ до Інтернету для повного функціоналу додатку. Багато функцій, які надаються застосунком, базуються на актуальних даних з Інтернету, тому наявність стабільного підключення є важливим аспектом в коректній роботі додатку. Зазначте, що дотримання цих вимог не тільки гарантує оптимальну продуктивність додатку, але й забезпечує безперебійне та зручне користування ним в умовах повсякденного використання. Такий підхід дозволяє вам насолоджуватися всіма перевагами нашого застосунку, максимально використовуючи його потенціал.

4.2 Встановлення необхідних компонентів для мобільного застосунку

Щоб встановити додаток на ваш пристрій, виконайте кілька простих кроків. Завантажте файл застосунку за ім'ям app-recognizer-release.apk, а потім запустіть

його для початку процесу встановлення. Просто натискайте на кнопку "Встановити" або "Установить", яка зображена на рисунку 4.1. Цей етап легко розпізнається на графіці та визначає важливий момент встановлення. Варто відзначити, що увесь процес встановлення спрощений і легкий для розуміння. Вам лише потрібно виконати кілька простих дій, щоб отримати можливість користуватися додатком на вашому пристрої. Це включає в себе завантаження файлу застосунку, запуск інсталяції та натискання на кнопку "Встановити" чи "Установить" для початку процесу. Для цього було створино процес встановлення свого додатку максимально зручним і легким, щоб ви могли швидко та просто отримати доступ до всіх його функціональних можливостей.

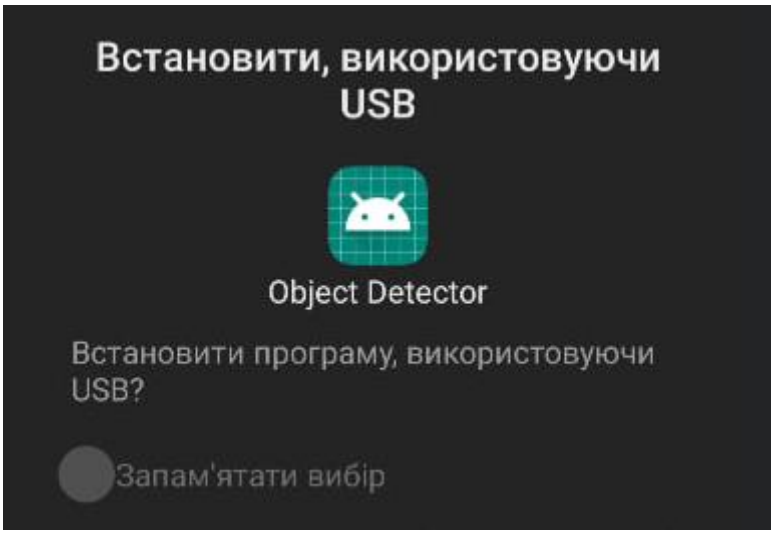


Рисунок 4.1 — Встановлення додатку пристрій

Тестування проектів в flutter чим більше функціоналу у додатка, тим складніше його перевіряти вручну. Однак завдяки автоматичним тестам ми можемо забезпечити надійність та ефективність програми перед її публікацією.

Автоматизовані тести дозволяють впевнитися, що кожна функція працює правильно, а вся програма в цілому відповідає очікуванням. Цей підхід дозволяє нам зберігати розширений функціонал, не компрометуючи якість. Модульні тести дозволяють перевірити окремі частини коду, забезпечуючи їх коректну роботу. Тестування віджетів або компонентів важливе для забезпечення зручності

користувача. Інтеграційні тести виконують роль остаточної перевірки, гарантуючи, що всі елементи взаємодіють без проблем. Такий комплексний підхід допомагає утримати стабільність програми та забезпечує швидкість виявлення та виправлення помилок перед їх впливом на користувачів. Автоматичне тестування поділяється на кілька категорій: Модульний тест перевіряє одну функцію, метод або клас. Тестування віджетів (в інших структурах інтерфейсу користувача називається тестуванням компонентів) перевіряє один віджет. Інтеграційний тест перевіряє програму в цілому або велику частину програми. Загалом, добре перевірена програма має багато модульних тестів і тестів віджетів, які відстежуються за покриттям коду , а також достатню кількість інтеграційних тестів, щоб охопити всі важливі випадки використання. Ця порада базується на тому факті, що існують компроміси між різними видами тестування.

Модульні тести спрямовані на перевірку правильності конкретної логічної одиниці, такої як функція, метод або клас. Основна мета таких тестів - перевірити, як ця логічна одиниця працює в різних умовах. Щоб забезпечити ізольованість, зазвичай імітуються зовнішні залежності тестованого блоку. Важливо відзначити, що модульні тести, як правило, не взаємодіють з файловою системою (не читають або не записують на диск), не відображають результати на екрані та не взаємодіють з користувачем поза контекстом тестового процесу. Для отримання додаткової інформації про модульні тести у контексті Flutter, можна скористатися вбудованим інструментом за допомогою команди flutter test --help у терміналі.

Тести віджетів щюоб розгорнути тему тестування віджетів або тестування компонентів у структурах інтерфейсу користувача, слід розглянути різні аспекти цього підходу. Тестування віджетів є важливою частиною розробки програмного забезпечення, оскільки визначає, наскільки ефективно взаємодіє інтерфейс користувача з користувачем. Однією з основних мет тестування віджетів є переконання, що інтерфейс користувача відображається та взаємодіє з користувачем належним чином. Це включає в себе перевірку візуального вигляду віджета, його поведінки та реакції на взаємодію користувача. Таке тестування допомагає виявляти та виправляти помилки, пов'язані з відображенням, анімацією

та іншими аспектами інтерфейсу. Окрім перевірки зовнішнього вигляду, тестування віджетів може включати перевірку внутрішньої логіки та взаємодії з іншими компонентами програми. Наприклад, важливо впевнитися, що віджет правильно реагує на зміни даних, передані йому з інших частин програми. Крім того, тестування віджетів може включати кілька класів тестів, кожен з яких спрямований на конкретний аспект віджета. Наприклад, один клас тестів може перевіряти візуальний аспект, інший - логіку взаємодії, а третій - реакцію на різні введення користувача. Для ефективного тестування віджетів необхідне налаштоване тестове середовище, яке враховує специфіку життєвого циклу віджета. Це може включати імітацію різних станів віджета та його взаємодію з іншими частинами програми. Такий комплексний підхід до тестування віджетів дозволяє впевнитися в їхній правильній роботі в різних умовах використання. В результаті ефективного тестування забезпечується висока якість інтерфейсу користувача та уникнення непередбачених проблем в процесі експлуатації програмного забезпечення.

Важливий аспект тестування віджетів полягає в його здатності ефективно взаємодіяти з користувачем та виконувати різноманітні завдання, такі як обробка дій та подій користувача, компонування та створення дочірніх віджетів. Такий підхід до тестування дозволяє забезпечити не лише правильність візуального вигляду віджета, а й його функціональність у контексті взаємодії з користувачем. Відмінною рисою тестування віджетів є його комплексність порівняно з модульними тестами. Віджет, що тестується, повинен бути здатний сприймати та відповідати на різні дії та події, які користувач може виконати. Це може включати в себе кліки, рухи миші, натискання клавіш та інші взаємодії. Також важливо враховувати можливість компонування віджетів, яке визначає, як вони спілкуються та розміщуються в інтерфейсі. Крім того, важливим аспектом є здатність віджета створювати та управляти дочірніми віджетами. Це може включати в себе динамічне додавання чи видалення віджетів, а також зміну їх параметрів та властивостей під час роботи програми. У той же час, слід зауважити, що середовище тестування віджетів може бути спрощеною реалізацією, яка не включає в себе всі аспекти повномасштабної системи інтерфейсу користувача. Такий підхід дозволяє

ефективно проводити тести в умовах, що розділені від реального середовища експлуатації, але при цьому забезпечуючи достатню кількість сценаріїв взаємодії. Загальний підхід до тестування віджетів є більш глибоким і включає в себе як аспекти функціональності, так і взаємодії з користувачем. Це робить його важливою частиною процесу розробки програмного забезпечення, оскільки забезпечує не тільки правильність відображення віджетів, але й їхню коректну роботу в різних умовах та сценаріях використання.

Інтеграційний тест – це ключовий етап в процесі тестування програмного забезпечення, оскільки він охоплює перевірку програми в цілому або значної її частини. Основна мета інтеграційного тестування полягає в переконанні, що всі компоненти, віджети та служби, які піддаються тестуванню, працюють спільно та взаємодіють належним чином. На відміну від модульних тестів, які фокусуються на перевірці окремих функцій або класів, інтеграційні тести охоплюють більші обсяги коду. Вони ставлять за мету виявлення проблем, які можуть виникнути при взаємодії різних компонентів програми. Це може бути особливо важливим у великих проектах, де різні частини програми можуть розроблятися незалежно та потім об'єднуватися.

Один з основних аспектів інтеграційного тестування – це переконання, що всі віджети та служби, які взаємодіють між собою, роблять це безсуперечно та без помилок. Це може включати в себе перевірку правильності передачі даних між різними частинами програми, взаємодію зі спільними ресурсами, а також взаємодію з базами даних та зовнішніми службами. Додатково інтеграційні тести можуть бути використані для оцінки продуктивності програми. Перевірка, як елементи програми ведуть себе під час великої кількості одночасних операцій або при великому об'ємі даних, є важливим етапом для забезпечення ефективності та швидкодії програми в реальному середовищі. Одним із важливих аспектів інтеграційного тестування є також тестування зовнішніх залежностей. Це може включати тестування інтеграції з іншими програмними продуктами, зовнішніми API, або навіть з іншими екосистемами програмного забезпечення. Важливо відзначити, що інтеграційні тести допомагають виявляти не тільки технічні проблеми, але й взаємодію між

різними частинами програми, що може призводити до виникнення непередбачених сценаріїв та помилок. Інтеграційне тестування взагалі визначається великою мірою контекстом використання програми та спрямовано на забезпечення її цілісності та працездатності в реальних умовах експлуатації. Такий підхід до тестування є ключовим елементом забезпечення високої якості програмного продукту.

Зазвичай, процес виконання тестів інтеграції включає в себе використання реальних пристроїв або емуляторів операційних систем, таких як симулятор iOS або емулятор Android. Цей підхід дозволяє тестувати програму в умовах, які максимально наближені до реального використання, що, в свою чергу, забезпечує надійні результати тестування. Використання реальних пристроїв або їх емуляторів дозволяє тестувати програму в різних умовах, таких як різні версії операційних систем, розміри екранів, архітектури процесорів та інші характеристики. Це особливо важливо в контексті розробки мобільних додатків, де різноманіття пристроїв є величезним. У багатьох випадках програма, що тестується, ізолюється від коду тестового драйвера. Це робиться для того, щоб уникнути спотворення результатів тестування та забезпечити об'єктивність та надійність тестів. Ізоляція програми від тестового драйвера дозволяє проводити тести в об'єктивних умовах та виявляти можливі проблеми або невідповідності у взаємодії компонентів програми. При використанні емуляторів операційних систем, які надають середовище для тестування, важливо враховувати, що це не тільки імітує реальні умови, але також дозволяє виконувати тести в контрольованому середовищі.

Це сприяє зручності та ефективності тестування, адже можна легко відтворити різні сценарії без необхідності використання реальних пристроїв. Забезпечення сумісності та оптимальної взаємодії програми на різних пристроях та операційних системах є ключовим завданням інтеграційного тестування. Реалізація тестів на реальних пристроях або їх емуляторах дозволяє виявляти потенційні проблеми та невідповідності, які можуть виникнути в реальному середовищі використання. Усі ці аспекти вказують на важливість правильного вибору інструментів та середовищ для виконання інтеграційних тестів. Забезпечення

правильної конфігурації тестового середовища допомагає не тільки підвищити ефективність тестування, але й забезпечити високу якість та стабільність програмного продукту в умовах реального використання.

При створенні першого тесту для віджету використовую тестовий пакет Flutter, починаю з написання коду в середовищі testWidgets(). Ця функція, яку надає пакет flutter_test, дозволяє визначити тест для віджетів і автоматично створює об'єкт WidgetTester для виконання різних перевірок. Коли я визначаю свій тест, я можу використовувати різні методи та функції, щоб перевірити правильність роботи віджету в різних умовах. Важливо використовувати WidgetTester для запуску віджетів та імітації взаємодії користувача, такої як тапання або прокручування. Також, ця функція надає зручний інтерфейс для створення тестових випадків та визначення очікуваних результатів. Використовуючи цей підхід, я можу переконатися, що мій віджет працює належним чином відповідно до очікуваних стандартів. Такий підхід до написання тестів для віджетів допомагає не лише перевірити їхню коректність, але й забезпечити стабільність та відповідність функціональним вимогам. Починаючи з першого тесту, я визначаю базову основу для подальшого розвитку тестового покриття та забезпечення надійності мого Flutter додатку.

Під час написання тесту для перевірки, чи мій віджет MyWidget відображає заданий заголовок і повідомлення, я враховую ряд аспектів для забезпечення точності та ефективності тестового випробування. Спершу, при визначенні цього тесту, важливо було створити відповідну структуру та функціональність для перевірки віджету. Я використовую функцію testWidgets(), що надається пакетом flutter_test, для визначення тестового випробування віджетів. Це забезпечує нам можливість створити тестовий випробувач та виконати перевірку віджета відповідно до наших очікувань. У цьому конкретному тесті, основною метою є перевірка правильного відображення заголовка та повідомлення в моєму віджеті. За допомогою методів та функцій, які надає WidgetTester, я можу імітувати взаємодію користувача та перевірити, чи віджет коректно відображає передані дані. Щоб

перевірити правильність відображення, я використовую метод expect(), який дозволяє визначити очікувані результати тесту. В цьому випадку, я перевіряю, чи віджет має правильний заголовок та повідомлення, які я передав йому для відображення. Важливо відзначити, що назва тесту також є ключовою. Вона повинна чітко відображати його основну мету, щоб легко зрозуміти, що саме перевіряється в даному тесті. Це дозволяє підтримувати легкість читання та розуміння коду та робити його доступним для інших розробників. Створюючи такий тест, я визначаю невід'ємний етап у процесі розробки програмного забезпечення, оскільки він дозволяє перевірити, чи віджет виконує свої функції належним чином. Цей підхід сприяє не лише забезпеченню коректності роботи програми, але й визначає основу для подальшого розвитку тестової бази та забезпечення стабільності та відповідності вимогам.

Під час створення тестового середовища для MyWidget, я використовую метод pumpWidget(), наданий об'єктом WidgetTester, щоб створити та відобразити віджет в тестовому середовищі. Цей етап є важливою частиною написання тесту, оскільки він забезпечує ініціалізацію та рендеринг віджета перед його перевіркою. За допомогою методу pumpWidget(), я можу передати екземпляр MyWidget, який я хочу перевірити, і WidgetTester візуалізує його в тестовому середовищі. Це дозволяє створити віртуальну обстановку, яка імітує відображення віджета в реальному додатку. Важливо враховувати, що pumpWidget() також відображає будь-які зміни або оновлення, які можуть бути виконані віджетом в процесі його роботи. Це забезпечує актуальність тесту, оскільки він враховує всі потенційні зміни, які можуть впливати на відображення віджета. У цьому етапі тестування важливо також враховувати взаємодію з іншими елементами середовища тестування, такими як зовнішні сервіси чи сторонні бібліотеки. Це забезпечить повноту перевірки та враховує можливі взаємодії в реальних умовах. Створюючи таке тестове середовище, я надаю можливість віртуально ініціалізувати та відобразити віджет, роблячи його готовим для подальшої перевірки та аналізу його стану. Це є

ключовим кроком у написанні ефективних тестів для віджетів та забезпеченні їхньої коректності в реальних умовах експлуатації.

Після виклику початкового pumpWidget() методу у моєму тестовому середовищі, WidgetTester надає додаткові можливості для перебудови того самого віджета. Ця функціональність стає особливо корисною при роботі з StatefulWidget або випадками, де виникає потреба в анімаціях та зміні стану віджета. Наприклад, у випадку, коли кнопка викликає setState(), яка змінює стан віджета, Flutter автоматично не перебудовує його в тестовому середовищі. Щоб вирішити цю проблему, можна скористатися одним з наступних методів, які надає WidgetTester, для того щоб попросити Flutter перебудувати віджет та відобразити оновлений стан.

Наприклад, метод pump() використовується для перебудови віджета один раз, враховуючи будь-які зміни в його стані. Це дозволяє емулювати ситуації, коли стан віджета змінюється під час взаємодії з користувачем чи відповідно до внутрішніх логік програми. Інші методи, такі як pumpAndSettle(), дозволяють вирішити асинхронність та очікувати, поки всі асинхронні завдання будуть завершені, перш ніж продовжити тестування. Це особливо важливо, коли ми маємо справу з анімаціями або іншими асинхронними подіями, які впливають на стан віджета. Такий підхід дозволяє забезпечити повноту тестування в умовах, коли віджет змінює свій стан через взаємодію користувача або інші динамічні процеси. Це надає додатковий рівень достовірності тестів та враховує реальні умови використання віджета.

4.3 Аналіз результатів роботи програмного забезпечення розпізнавання об’єктів у мобільному застосунку.

Для того, щоб розпочати використання застосунку, вам необхідно знайти на робочому столі піктограму додатку та виконати натискання на неї. Це дію можна виконати, використовуючи мишу або дотик на сенсорному екрані вашого пристрою. Яскравий момент цього етапу ілюструється на рисунку 4.2., де видно, як виглядає піктограма та як саме вона взаємодіє з користувачем на етапі запуску застосунку. Важливо відзначити, що цей процес також є простим та інтуїтивно зрозумілим.

Просто знайдіть відповідну іконку застосунку на вашому робочому столі та виконайте одне натискання, щоб запустити додаток. Це ефективний спосіб забезпечити швидкий доступ до функціоналу та початок користування.

Рисунок 4.2 – Іконка мобільного застосунку

При вході до мобільного додатку, користувач зазнає подвійного запиту, який відображається у формі запитань про використання камери (відображеного на Рисунку 4.3) та запитання щодо доступу до мікрофону (яке можна побачити на Рисунку 4.4). Обидва ці запити необхідні для належного функціонування системи розпізнавання об'єктів та оптимального використання функцій додатку. Важливо відзначити, що цей подвійний запит не є випадковим або нав'язливим, але є стратегічною частиною додатку, спрямованою на покращення користувацького досвіду та забезпечення ефективної роботи функцій розпізнавання об'єктів. Давайте розглянемо обидва аспекти цього запиту детальніше.

Запит про використання камери є необхідним елементом для забезпечення можливості додатку взаємодіяти з оточуючим середовищем через візуальний контекст. Камера використовується для захоплення зображення або відео, яке подалі буде аналізовано алгоритмами розпізнавання об'єктів. Це важливо для визначення та ідентифікації різних об'єктів, що може бути використано в різних сценаріях, від розпізнавання обличчя до ідентифікації предметів. З іншого боку, запит про використання мікрофону є ключовим елементом для забезпечення аудіального сприйняття додатком. Мікрофон використовується для запису звукових сигналів, що дозволяє додатку аналізувати аудіо-інформацію. Це може бути використано для різних завдань, включаючи аудіорозпізнавання, виявлення шумів, або навіть розпізнавання мови.

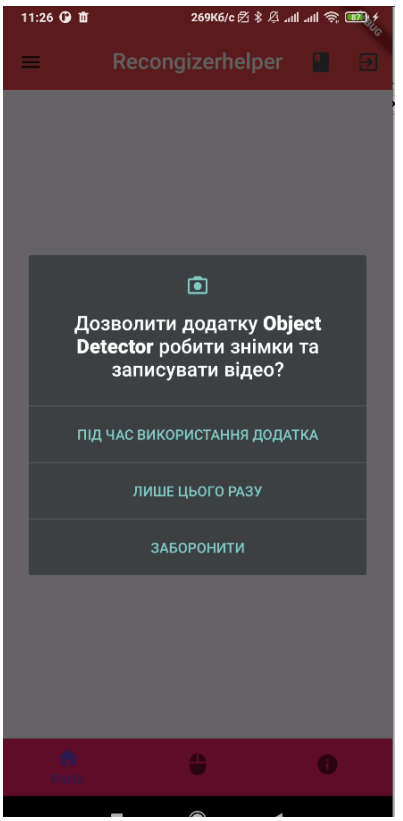


Рисунок 4.3 – Запит на використання камери

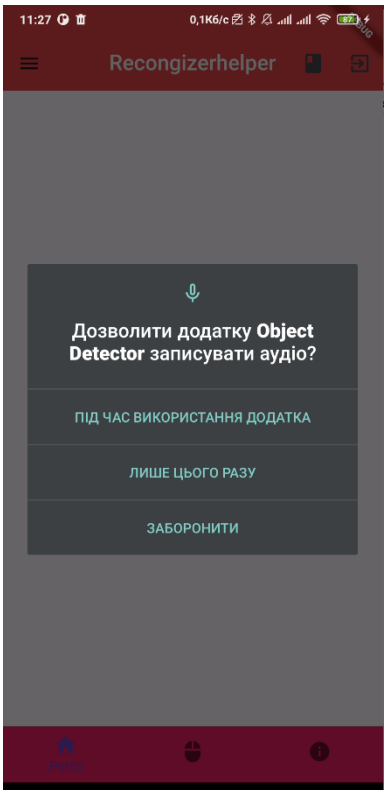


Рисунок 4.4 – запит на використання мікрофону

Після успішного отримання дозволу на використання камери та мікрофону, програма надалі ініціює етап сканування об'єктів, які відображені на екрані мобільного пристрою (як це можна бачити на Рисунку 4.5). Важливо відзначити, що надійність та точність цього процесу розпізнавання є вирішальними факторами для досягнення ефективної взаємодії з користувачем. Цей етап виконується з метою не лише визначення об'єктів, а й забезпечення їх правильного ідентифікування, що є ключовим для подальшого коректного виконання функцій додатку. Приглянемося зображеній на Рисунку 4.5 клавіатурі, яка слугує відмінним прикладом вдалого розпізнавання об'єктів. Цей елемент ілюструє високий рівень точності та стабільності в процесі сканування. Завдяки цьому успішному розпізнаванню, відкриваються широкі можливості для зручного введення тексту та команд користувачем. Використання камери для аналізу клавіатури надає додатку можливість "бачити" кожен окремий символ та визначати його положення. Це може бути використано для автоматичного визначення натискань користувачем на конкретні клавіші, сприяючи введенню тексту безпосередньо через зображення клавіатури на екрані. Такий підхід полегшує та прискорює введення інформації, покращуючи загальний комфорт взаємодії з додатком. Важливо також врахувати, що забезпечення стабільної роботи та високої точності розпізнавання об'єктів впливає на різноманітні аспекти користувацького досвіду. Наприклад, у випадку клавіатури це може визначити ефективність та швидкість введення тексту, а також попереджати можливі непорозуміння чи помилки при розпізнаванні. Отже, технологічна дбайливість про точність є ключем до забезпечення надійності та задоволення від використання додатку.

Крім того, успішне розпізнавання об'єктів через камеру розширює можливості додатку у взаємодії з різноманітними об'єктами та сценаріями. Це може включати в себе розпізнавання фотографій, ідентифікацію продуктів, чи навіть взаємодію з архітектурними елементами. Такий широкий функціонал відкриває безліч можливостей для розвитку та удосконалення додатку в майбутньому. Отже, етап сканування об'єктів є не лише технічним викликом для

програми, але й стратегічним рушієм для забезпечення високої якості взаємодії з користувачем та розширення функціональних можливостей додатку. Надійність та точність цього процесу визначають не лише функціональність, але й рівень задоволення користувача від використання застосунку

Рисунок 4.5 – Іконка мобільного застосунку

У верхньому лівому кутку екрана можна побачити кнопку яка відкриває при натисканні на яку відкривається контекстне меню(рисунок 4.6), що містить декілька важливих елементів програми:

1 Favorites objects

2 Report bugs

3 Request new functions

4 Settings

5 Exit

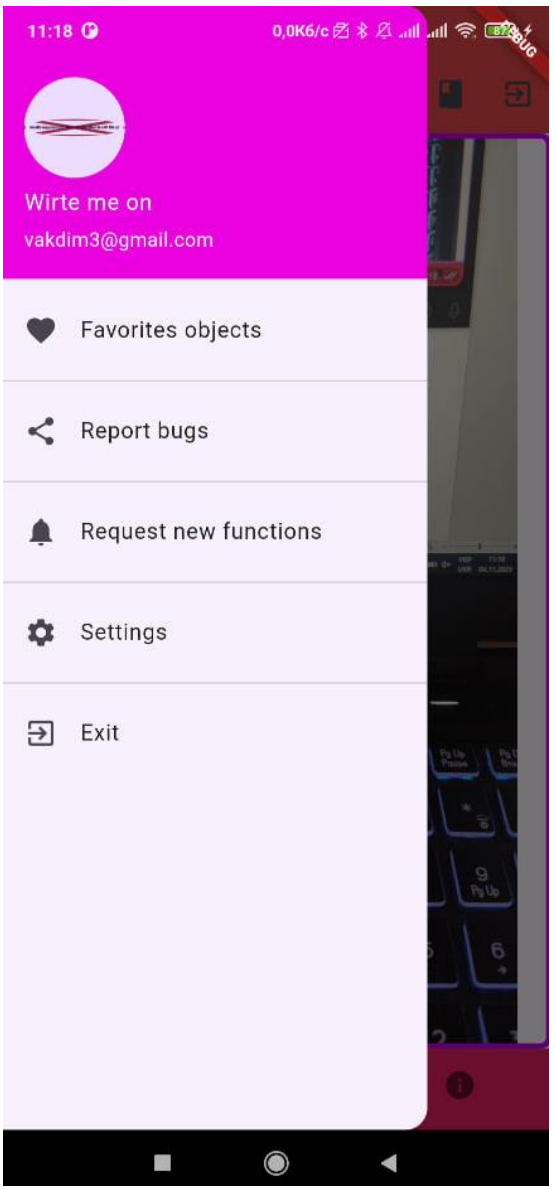


Рисунок 4.6 – Меню з додатковим функціоналом програми

1 Favorites objects

Ця нова сторінка відкриває перед користувачем можливість додавання та запам'ятовування нових об'єктів, розширюючи можливості вивчення англійської мови. Важливим етапом є можливість додавання слів англійською мовою, які користувач бажає вивчити чи навчитися. Це важливий аспект для індивідуального підходу та персоналізації процесу навчання, де користувач може визначити свої власні потреби та цілі. Додавані слова українською мовою стають важливим елементом для подальшого вивчення та використання. Це створює можливість навчання не лише англійської мови, але й її відповідників українською. Користувач може надати власні переклади, розширюючи свій словниковий запас і покращуючи навички вивчення обох мов.

На рисунку (Рисунок 4.7) надано приклад того, як додані слова англійською мовою виглядають у вигляді списку. Це забезпечує зручний та легкий доступ до доданих слів, дозволяючи користувачеві легко переглядати та вивчати їх у будь-який момент. Кожне слово супроводжується зображенням, що полегшує асоціацію та вивчення слова у контексті. У розділі (Рисунок 4.8) показано момент додання нового слова до переліку. Цей процес надає користувачеві можливість інтерактивно взаємодіяти з додаванням нових слів, спрощуючи процес та роблячи його цікавим та захоплюючим.

Крім того, відображення інтерфейсу на екрані надає візуальний контекст доданого слова та його перекладу, що полегшує зрозуміння та запам'ятовування нових термінів. Загальною метою цієї функції є створення зручного та ефективного інструменту для самостійного вивчення мови, який враховує індивідуальні потреби та вподобання користувача. Цей метод вивчення мов відкриває перед учнями можливість гнучкого контролю над власним навчальним процесом, дозволяючи досягати бажаних результатів у освоєнні англійської та української мов. Завдяки такому підходу, навчання стає індивідуалізованим та ефективним, а користувачі можуть самостійно керувати своїм власним шляхом до мовної компетентності. Один із основних принципів цього методу полягає в наданні можливості вибору власного темпу та ритму навчання. Кожен учень має

змогу пристосувати розклад та обсяги вивчення мови до свого власного графіка та потреб. Це створює комфортні умови для засвоєння матеріалу та підтримує індивідуальний підхід до навчання.

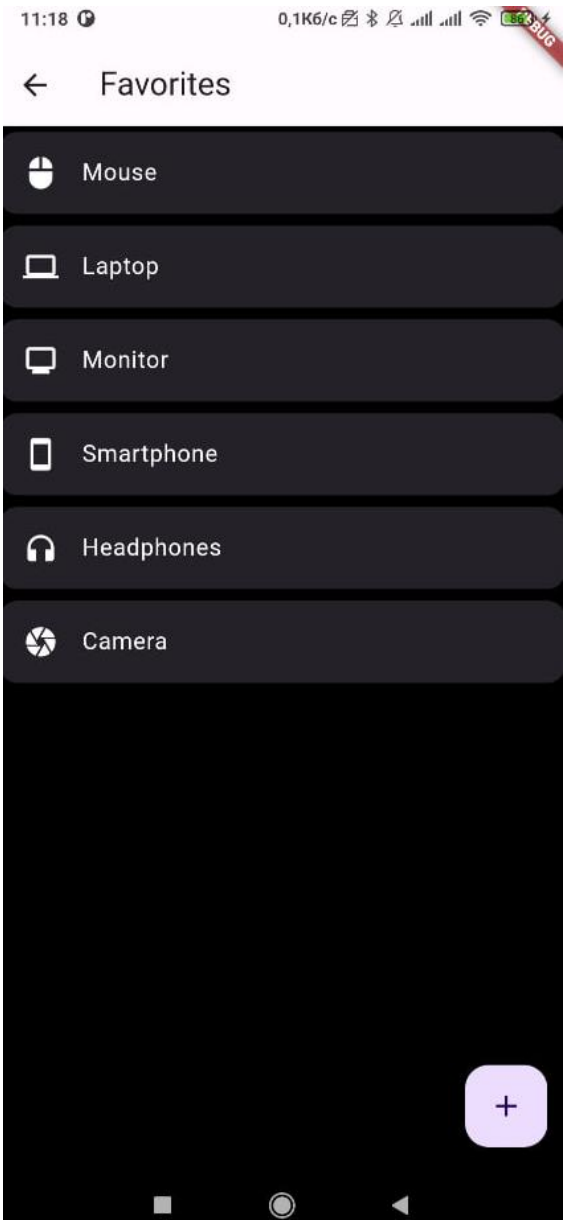


Рисунок 4.7 – список використаних обєктів

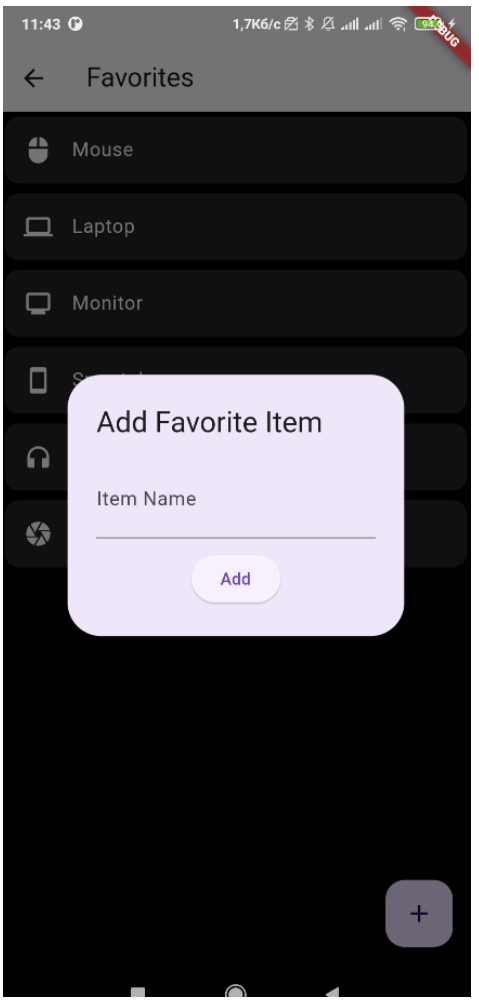


Рисунок 4.8 – додання нового слова

2 Report bugs

Створення нової сторінки для повідомлень про баги в додатку є ключовим етапом для поліпшення взаємодії з користувачами та підвищення якості його використання. На рисунку (Рисунок 4.9) ілюструється процес повідомлення про помилку, де користувач легко та зручно може передати відомості про виявлені недоліки чи проблеми у функціонуванні додатка. Механізм повідомлень про баги стає важливою ланкою для збору зворотного зв'язку від користувачів, що дозволяє розробникам ефективно реагувати на можливі проблеми та оперативно виправляти їх. Легкість опису виявлених недоліків, можливість додавання скріншотів або навіть аудіо- та відеозаписів забезпечує максимально точне передавання характеру проблем. Наголос на простоті та інтуїтивності процесу подачі повідомлень про баги робить цей інструмент доступним для користувачів навіть без спеціальних навичок

чи технічного досвіду. Прозорий та зручний інтерфейс сторінки повідомлень про баги робить взаємодію з ним приємною та ефективною для широкого кола користувачів. Розробники в даному випадку я отримую докладну та конкретну інформацію про виявлені помилки, що дозволить мені оперативно реагувати та вирішувати проблеми.

Це значно спрощує процес розробки та дозволяє швидко внести виправлення та покращення у функціонал додатка. Головною метою цієї сторінки є поліпшення якості та стабільності додатка через активну співпрацю з користувачами та урахування їхнього досвіду. Відкритий зворотний зв'язок визначає новий рівень взаємодії між розробниками та користувачами, спрямований на постійне удосконалення та оптимізацію додатка для задоволення потреб усіх його використовувачів.

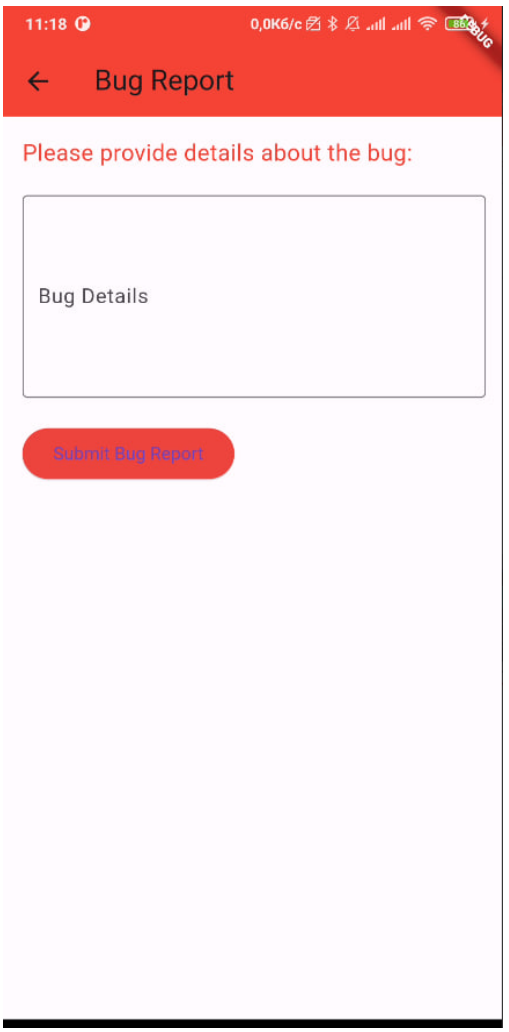


Рисунок 4.9 – Відправлення репорту проблеми

3 Request new functions

Введення в нову сторінку, що відповідає за взаємодію між розробником та користувачем, є ключовим кроком у створенні простору для обговорення та подання запитів на додавання нових функцій у додаток. На рисунку (Рисунок 4.10) проілюстровано процес подання запиту, де користувач може зручно та швидко висловити свої ідеї та очікування щодо подальшого розвитку додатка. Ця нова сторінка створює простір для відкритого спілкування, де користувачі можуть висловлювати свої ідеї, пропозиції та бажання щодо можливих нововведень у функціоналі додатка. Вона служить майданчиком для активної взаємодії та обміну думками між розробниками та користувачами, щоб спільно визначити напрямок розвитку. Процес подання запиту на додавання нових функцій відбувається легко та ефективно. Користувачі мають можливість чітко визначити свої потреби та внести конструктивні пропозиції, завдяки чому розробники отримують значний обсяг інформації для подальшого аналізу та впровадження. Важливим аспектом цього інструменту є його відкритість та доступність для широкого кола користувачів.

Прозорий механізм подання запитів сприяє активному включенню користувачів у формування функціоналу додатка, роблячи їхній голос вагомим елементом прийняття рішень. Розробники отримують можливість систематично аналізувати та враховувати запити користувачів, що сприяє створенню продукту, який відповідає реальним потребам та очікуванням аудиторії. Взаємодія на цій сторінці визначає динаміку розвитку додатка, надаючи йому гнучкість та адаптивність до змін у вимогах користувачів.

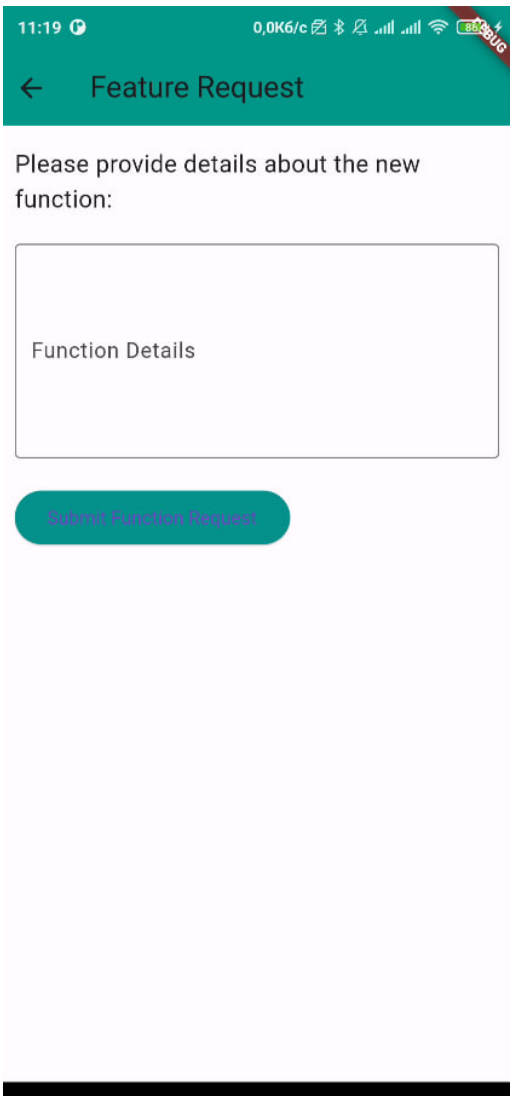


Рисунок 4.10 – Відправлення запиту на додання нових функцій

4 Settings

Перехід до нової сторінки, відповідальної за налаштування додатка, відкриває широкі можливості користувачам для налаштування та персоналізації свого власного досвіду використання. У цьому розділі доступні функції, такі як активація темного режиму, управління повідомленнями та зміна мови інтерфейсу.

Однією з ключових можливостей цього меню є змога активувати темний режим у додатку. Темний режим відмінно підходить для тих, хто віддає перевагу затемненому фону та приємному для очей світлу тексту. Ця функція не лише забезпечує комфортне читання в умовах слабкого освітлення, але й сприяє

збереженню заряду батареї на пристроях з екранами OLED. Ще однією корисною функцією в цьому меню є можливість керування повідомленнями.

Користувачі можуть вибирати, які повідомлення вони бажають отримувати та яким чином вони повідомлятимуться. Це дозволяє налаштовувати взаємодію з додатком відповідно до власних уподобань та потреб. Однією з особливостей цього меню є можливість зміни мови інтерфейсу додатка. Додаток підтримує кілька мов, таких як англійська, іспанська, французька, німецька та українська. Це робить додаток доступним та зручним для користувачів з різних країн та регіонів, надаючи можливість вибору мови, яка є найбільш комфортною для них.

На рисунках (Рисунок 4.11) та (Рисунок 4.12) зображено ілюстрації можливостей цього розділу меню. Користувач може з легкістю взаємодіяти з різними опціями та налаштуваннями, вибираючи ті, які відповідають його потребам та вподобанням. Це меню налаштувань виступає ключовим елементом, що дозволяє користувачам взаємодіяти з додатком на особистому рівні, пристосовуючи його до їхніх власних уподобань та вимог. Завдяки широкому спектру функцій та легкому використанню інтерфейсу, це меню стає не лише засобом управління, але й потужним інструментом для докладної настройки враження від користування додатком. Ця платформа налаштувань відкриває перед користувачем багатий функціонал, який дозволяє персоналізувати різноманітні аспекти додатка. Вона служить майданчиком для адаптації програми під індивідуальні потреби кожного користувача, забезпечуючи високий рівень гнучкості та контролю.

Однією з ключових переваг цього меню налаштувань є його відмінний баланс між розширеним функціоналом і простотою використання. Воно не лише надає користувачеві можливість впливати на роботу додатка в глибину, але й робить цей процес доступним і зрозумілим для широкого кола користувачів.

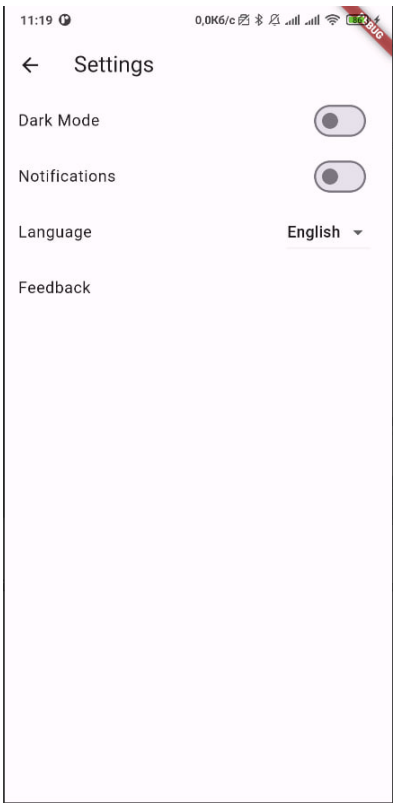


Рисунок 4.11 – Налаштування (Settings)

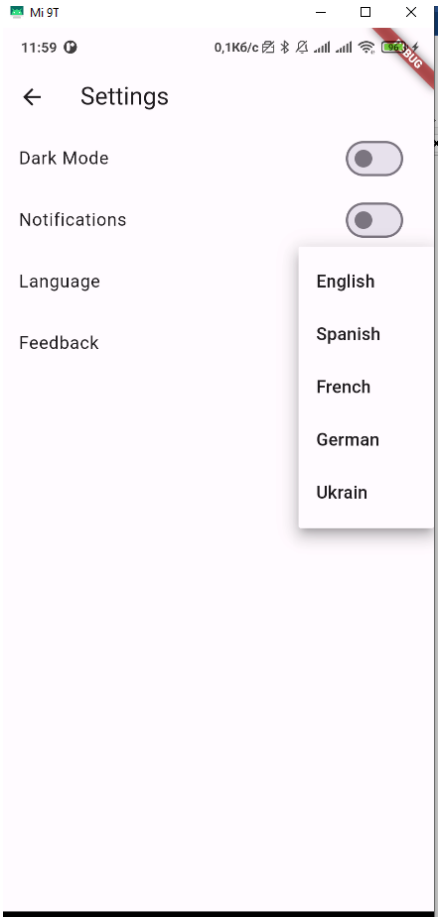


Рисунок 4.12 – перелік мов які на які переведено додаток

5 Exit

Це кнопка швидкого виходу з застосунку. Додаток пропонує важливу та корисну функцію - переклад слів з англійської мови на українську. Доступ до цієї функції забезпечено через кнопку, розташовану у верхньому лівому кутку екрану.

Рисунок 4.13 – Перекладач

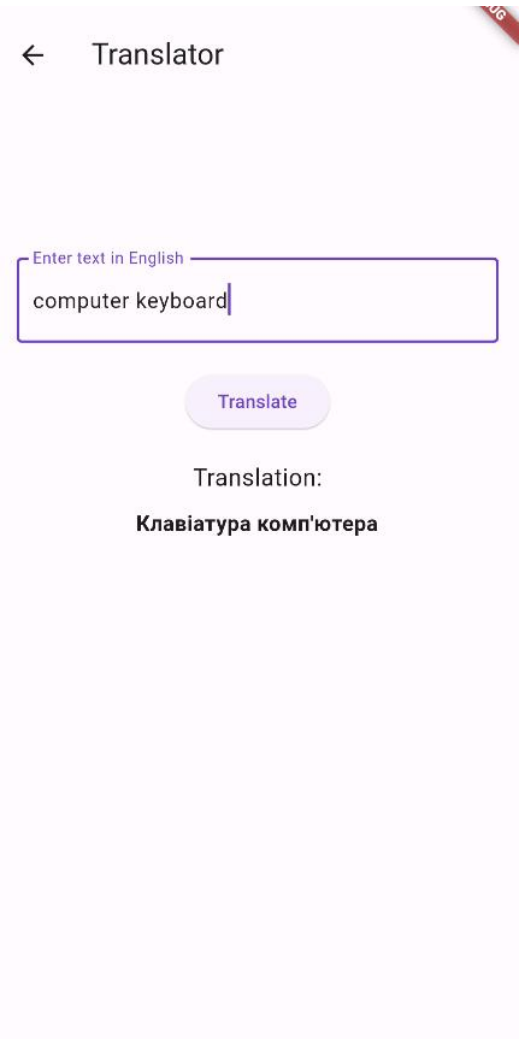


Рисунок 4.14 – Приклад перекладу слова

При натисканні на цю кнопку користувач відкриває перехід до перекладача, який легко доступний в додатку. На рисунках (Рисунок 4.13) та (Рисунок 4.14) можна спостерігати ілюстрації використання цієї функції. Перекладач представляє собою інтуїтивно зрозумілий інтерфейс, що дозволяє користувачам легко та ефективно перекладати розпізнані об'єкти та отримувати їх назви на українській мові.

Використання функції перекладу слів дозволяє полегшити сприйняття та розуміння об'єктів, які розпізнаються додатком. Вона стає невід'ємною частиною досвіду користувача, який може легко отримувати інформацію про назви об'єктів англійською мовою та перекладати їх на рідну українську. Ця функція особливо корисна для користувачів, які мають обмежений рівень англійської мови або хочуть

отримувати інформацію на мові, зрозумілій їм найкраще. Вона розширює можливості додатку та робить його більш доступним та корисним для різних аудиторій.

Функціонал перекладу слів є ще однією ступеню додатку в напрямку забезпечення повноти та гнучкості користувацького досвіду. Зручний та швидкий доступ до перекладача забезпечує користувачам простий спосіб отримання інформації на власній мові та покращує їхню взаємодію з додатком.

Ще у програмі є 3 кнопки переходи які розташовані в низу застосунку які відповідають за переходи до допоміжних частин застосунку (Рисунок 4.15) зображено ці кнопки переходу кожна з них відповідає за свою сторінку.

1 Pinfo – (Pronouns info)

2 Helpurl

3 Info

Рисунок 4.15 – 3 інформаційні кнопки

1 "Pinfo" або "Pronouns info" визначається як розділ застосунка, спрямований на надання користувачам інформації щодо займенників. У цьому розділі зібрана значна кількість слів, разом з їхніми перекладами. Цей функціонал виявляється корисним для тих, хто бажає вивчати та оволодіти всією базою слів, які може перекладати застосунок. На рисунку (Рисунок 4.16) представлено сам перелік слів, які доступні в розділі "Pinfo". Цей перелік становить цінний ресурс для користувачів, які прагнуть поглибленого вивчення мови та збагачення свого словникового запасу.

Функціонал "Pronouns info" сприяє структурованому та організованому навчанню, надаючи користувачам можливість систематично вивчати та відстежувати слова та їхні переклади. Це дозволяє користувачам зосередитися на конкретній групі слів та ефективно їх освоювати. Застосунок вносить вагомий внесок у навчальний процес, надаючи зручний та доступний інструмент для вивчення слів, особливо займенників. При цьому навчання стає більш інтерактивним та пристосованим до індивідуальних потреб користувачів. Загальна мета цього розділу - зробити навчання ефективним та приємним, сприяючи зростанню мовних навичок користувачів та допомагаючи їм досягати своїх мовних цілей.

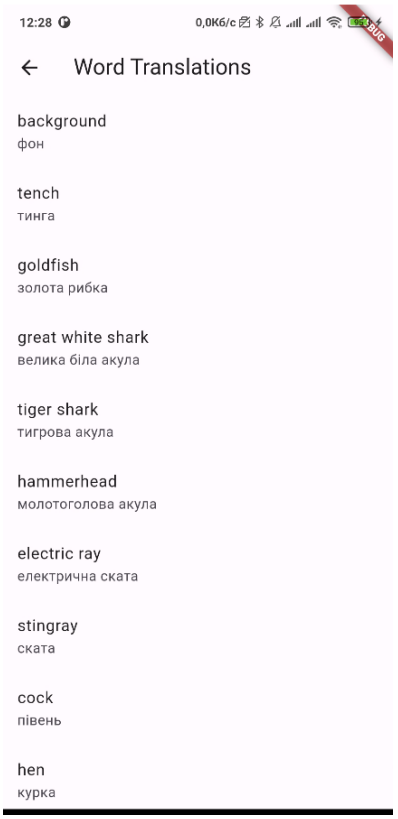


Рисунок 4.16 – Pinfo перехід до переліку слів

"Helpurl" є перехідом до веб-сторінки, де надано докладний огляд структури та відкритого коду застосунка. На рисунку (Рисунок 4.17) ви можете детально

ознайомитися з усіма аспектами, що стосуються функціональності та кодової бази додатка. Ця веб-сторінка служить як розширений довідник, де кожен елемент та модуль застосунка розглядається в розрізі. Інформація надається вичерпно та систематично, забезпечуючи користувачам повну картину про те, як працює застосунок та як його структуровано. Розділ "Helpurl" є важливим ресурсом для розробників, дослідників та тих, хто цікавиться вивченням внутрішньої архітектури програмного забезпечення. Ви знайдете високоякісні зображення, схеми та пояснення, що полегшують розуміння принципів роботи застосунка. Окрім цього, на сторінці "Helpurl" ви можете знайти відкритий код застосунка. Це важливий аспект для тих, хто бажає вивчати код та внести свій внесок у розвиток додатка. Відкритий код створює можливість співпраці та спільного вдосконалення програмного продукту.

Відвідування сторінки "Helpurl" може бути корисним кроком для тих, хто хоче глибше зрозуміти внутрішній світ застосунка, оцінити його розробку та взяти участь у спільноті, яка допомагає йому розвиватися.

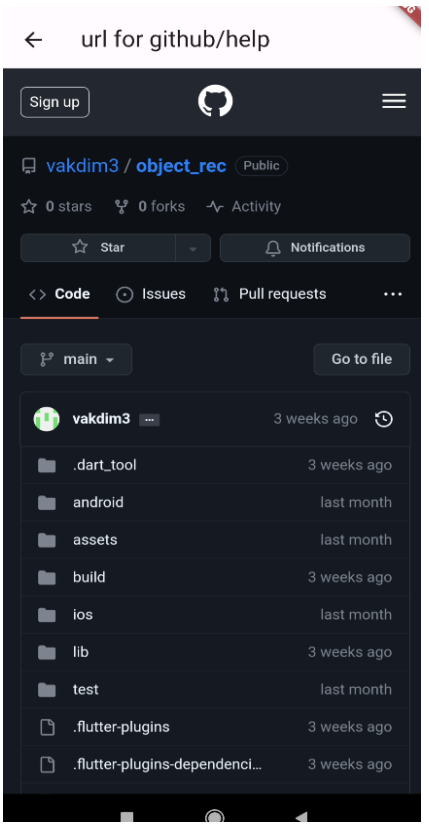


Рисунок 4.17 – Helpurl відкритий код застосунку

Розроблений додаток має функціонал, наведений в таблиці 4.1, який було порівняно із функціоналом аналогів Medi App, Recogn, Skyeyy.

Таблиця 4.1 – Порівняльна характеристика розробленого ПЗ

Назва програми	Точнічість розпізнавання	Функція перекладу	Браузер з можливостю перегляду коду додатку	Налаштування кількості об’єктів
Medi App	+/-	+	-	+/-
App	+	-	-	-
Skyeyy	-	+	-	-
Розроблений додаток	+	+	+	-

4.4 Висновок до розділу 4

Отже, розроблений додаток для розпізнавання об’єктів було протестовано на відповідність меті дослідження. Функціонал програмного забезпечення є розширеним по відношенню до існуючих сучасних рішень та відповідає завданню.

5. ЕКОНОМІЧНА ЧАСТИНА

5.1. Проведення комерційного та технологічного аудиту інформаційної технології розпізнавання об’єктів

Метою проведення комерційного і технологічного аудиту є оцінювання науково-технічного рівня та рівня комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності, тобто під час виконання магістерської кваліфікаційної роботи.

Для проведення комерційного та технологічного аудиту залучаємо 3-х незалежних експертів, якими є провідні викладачі випускової або спорідненої кафедри.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу здійснюємо із застосуванням п’ятибальної системи оцінювання за 12-ма критеріями, а результати зводимо до таблиці 1.

Таблиця 5.1 – Результати оцінювання науково-технічного рівня і комерційного потенціалу засобу поляриметричного аналізу оптично активних рідни

Критерії	Експерти		
	Озеранський 1	Арсенюк 2	Барабан 3
	Бали, виставлені експертами		
Технічна здійсненність концепції	2	2	3
Ринкові переваги (наявність аналогів)	3	3	2
Ринкові переваги (ціна продукту)	4	4	3
Ринкові переваги (технічні властивості)	4	2	3
Ринкові переваги (експлуатаційні витрати)	4	3	3
Ринкові перспективи (розмір ринку)	4	4	3
Ринкові перспективи (конкуренція)	3	3	2

Практична здійсненність (наявність фахівців)	3	3	3
Практична здійсненність (наявність фінансів)	3	4	4
Практична здійсненність (необхідність нових матеріалів)	2	3	3
Практична здійсненність (термін реалізації)	3	3	3
Практична здійсненність (розробка документів)	2	3	3
Сума балів	37	37	35
Середньоарифметична сума балів, СБ	36		

За результатами розрахунків, наведених в таблиці 5.1 робимо висновок про те, що науково-технічний рівень та комерційний потенціал інформаційної технології розпізнавання об’єктів у мобільному застосунку – вищий середнього.

5.2 Розрахунок витрат на здійснення науково-дослідної роботи

Витрати на оплату праці. Належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці, також будь-які види грошових і матеріальних доплат, які належать до елемента «Витрати на оплату праці».

Основна заробітна плата дослідників. Витрати на основну заробітну плату дослідників ($З_о$) розраховують відповідно до посадових окладів працівників, за формулою:

$$З_о = \sum_{i=1}^{к} \frac{М_{ni} \cdot t_i}{Т_р},$$

де к – кількість посад дослідників, залучених до процесу дослідження; $М_{ni}$ – місячний посадовий оклад конкретного розробника (інженера, дослідника, науковця тощо), грн.; $Т_р$ – число робочих днів в місяці; приблизно $Т_р$ = (21…23) дні; t_i – число робочих днів роботи розробника (дослідника).

Зроблені розрахунки зводимо до таблиці 5.2.

Таблиця 5.2 – Витрати на заробітну плату дослідників

Посада	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату, грн.
Керівник	25000	1190	60	71400
Розробник	15000	714	90	64260
Всього:	135660			

Основна заробітна плата робітників. Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт розраховують за формулою:

$$З_р = \sum_{i=1}^{n} C_i \cdot t_i,$$

де $С_i$ – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год; t_i – час роботи робітника на виконання певної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду С і можна визначити за формулою:

$$С_і = \frac{М_м \cdot К_і \cdot К_с}{Т_р \cdot t_{зм}},$$

де $М_м$ – розмір прожиткового мінімуму працездатної особи або мінімальної місячної заробітної плати (залежно від діючого законодавства), у 2023 році $М_м$=6700 грн; $К_і$ – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду; $К_с$ – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об'єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати; $Т_р$ – середня кількість робочих днів в місяці, приблизно $Т_р$ = 21…23 дні; $t_{зм}$ – тривалість зміни, год.

Таблиця 5.3 – Витрати на заробітну плату робітників

Найменування робіт	Трудомісткість, н-год.	Розряд роботи	Погодинна тарифна ставка	Тариф. коеф.	Величина, грн.
Створення архітектури	40	1	39,9	1	1596
Вдосконалення алгоритму	200	2	43,5	1,09	8700
Налаштування перекладача в додатку	7	3	47,1	1,18	330
Налаштування інтерфейсу та адаптування його до інших телефонів	8	4	50,6	1,27	405
Всього					11031

Додаткова заробітна плата. Додаткова заробітна плата $З_д$ всіх розробників та робітників, які брали участь у виконанні даного етапу роботи, розраховується як (10…12)% від суми основної заробітної плати всіх розробників та робітників, тобто:

$$З_д = 0{,}11 \cdot (З_о + З_р) = 0{,}11 \cdot (64260 + 11031) = 8282 \text{ грн.}$$

Відрахування на соціальні заходи. Нарахування на заробітну плату $Н_{зп}$ розробників та робітників, які брали участь у виконанні даного етапу роботи, розраховуються за формулою:

$$Н_{зп} = \beta \cdot (З_о + З_р + З_д) =$$
$$= 0{,}22 \cdot (64260 + 11031 + 8282) = 18386 \text{ грн.}$$

де $З_о$ – основна заробітна плата розробників, грн.; $З_р$ – основна заробітна плата робітників, грн.; $З_д$ – додаткова заробітна плата всіх розробників та робітників, грн.; β – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування, % (приймаємо для 1-го класу професійності ризику 22%).

Спецустаткування для наукових (експериментальних) робіт. Вартість спецустаткування визначається за прейскурантом гуртових цін або за даними базових підприємств за відпускними і договірними цінами.

$$В_{спец} = \sum_{1}^{к} Ц_і \cdot С_{пр.і} \cdot К_і,$$

де $Ц_і$ – ціна придбання спецустаткування і-го виду, грн.; $С_{пр.і}$ – кількість одиниць спецустаткування відповідного виду, шт.; $К_і$ – коефіцієнт транспортних витрат, $К_і$ = (1,1…1,15); n – кількість видів спецустаткування.

Таблиця 5.4 – Витрати на придбання спецустаткування

Найменування спецустаткування	Ціна за одиницю, грн.	Витрачено	Вартість спецустаткування, грн.
Монітори	8000	3	24000
Клавіатури	300	3	900
Миші	100	3	300
Всього, з врахуванням коефіцієнта транспортних витрат			27720

Програмне забезпечення. До балансової вартості програмного забезпечення входять витрати на його інсталяцію, тому ці витрати беруться додатково в розмірі 10…12% від вартості програмного забезпечення. Балансову вартість програмного забезпечення розраховують за формулою:

$$В_{прг} = \sum_{1}^{к} Ц_{іпрг} \cdot С_{прг.і} \cdot К_і,$$

де $Ц_{іпрг}$ – ціна придбання програмного забезпечення і-го виду, грн.; $С_{прг.і}$ – кількість одиниць програмного забезпечення відповідного виду, шт.; $К_і$ – коефіцієнт, що враховує інсталяцію, налагодження програмного забезпечення, $К_і = (1{,}1…1{,}12)$; к – кількість видів програмного забезпечення.

Таблиця 5.5 – Витрати на придбання програмного забезпечення

Найменування програмного забезпечення	Ціна за одиницю, грн.	Витрачено	Вартість програмного забезпечення, грн.
Windows	5000	2	10000
Flutter	0	3	0
Android	200	1	200
Всього, з врахуванням коефіцієнта інсталяції та налагодження			13200

Амортизація обладнання. Амортизація обладнання, комп’ютерів та приміщень, які використовувались під час (чи для) виконання даного етапу роботи.

У спрощеному вигляді амортизаційні відрахування А в цілому бути розраховані за формулою:

$$А = \frac{Ц_б}{Т_в} \cdot \frac{t}{12},$$

де $Ц_б$ – загальна балансова вартість всього обладнання, комп’ютерів, приміщень тощо, що використовувались для виконання даного етапу роботи, грн.; t – термін використання основного фонду, місяці; $Т_в$ – термін корисного використання основного фонду, роки.

Таблиця 5.6 – Амортизаційні відрахування за видами основних фондів

Найменування	Балансова вартість, грн.	Строк корисного використання, років	Термін використання, місяців	Сума амортизації, грн.
ПК	20000	5	1	333
Ноутбук	25000	5	1	417
Мобільний телефон	9000	3	1	250
Всього	1000			

Витрати на електроенергію для науково-виробничих цілей. Витрати на силову електроенергію $В_е$, якщо ця стаття має суттєве значення для виконання даного етапу роботи, розраховуються за формулою:

Таблиця 5.7 – Витрати на електроенергію

Найменування обладнання	Потужність, Вт	Тривалість годин роботи
ПК	700	255
Ноутбук	400	255
Мобільний телефон	100	255

$$Ве = \sum \frac{W_i \cdot t_i \cdot Ц_е \cdot К_{впі}}{ККД} = \frac{0{,}7 \cdot 255 \cdot 7{,}5 \cdot 0{,}9}{0{,}96} + \frac{0{,}4 \cdot 255 \cdot 7{,}5 \cdot 0{,}9}{0{,}96} + \frac{0{,}1 \cdot 255 \cdot 27 \cdot 0{,}9}{0{,}96} = 7746 \text{ грн.},$$

W_i – встановлена потужність обладнання, кВт; t_i – тривалість роботи обладнання на етапі дослідження, год.; Це – вартість 1 кВт електроенергії, грн.; $К_{впі}$ – коефіцієнт використання потужності; ККД – коефіцієнт корисної дії обладнання.

Інші витрати. До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуються як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$Ів = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%} = (64260 + 11031) \cdot \frac{50}{100} = 37646 \text{ грн.},$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати».

Накладні (загальновиробничі) витрати. До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов'язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов'язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов'язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуються як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$Внзв = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%} = (64260 + 11031) \cdot \frac{100}{100} = 75291 \text{ грн.},$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати».

Витрати на проведення науково-дослідної роботи. Витрати на проведення науково-дослідної роботи розраховуються як сума всіх попередніх статей витрат за формулою:

$$В_{заг} = З_о + З_р + З_{дод} + З_н + В_{спец} + В_{прг} + А_{обл} + В_е + І_в + В_{нзв} =$$
$$= 64260 + 11031 + 8282 + 18386 + 27720 + 13200 + 1000 + 7746 +$$
$$+37646 + 75291 = 264562 \text{ грн.}$$

Загальні витрати. Загальні витрати ЗВ на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховуються за формулою:

$$ЗВ = \frac{В_{заг}}{\eta} = \frac{264562}{0{,}2} = 1322810 \text{ грн.},$$

де η – коефіцієнт, що характеризує етап виконання науково-дослідної роботи. Оскільки, якщо науково-технічна розробка знаходиться на стадії технічного проектування, то η=0,2.

5.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

В даному випадку відбувається розробка засобу, тому основу майбутнього економічного ефекту буде формувати: ΔN – збільшення кількості споживачів, яким надається відповідна інформаційна послуга в аналізовані періоди часу; N – кількість споживачів, яким надавалась відповідна інформаційна послуга у році до впровадження результатів нової науково-технічної розробки; $Ц_б$ – вартість послуги у році до впровадження інформаційної системи; $\pm\Delta Ц_о$ – зміна вартості послуги (зростання чи зниження) від впровадження результатів науково-технічної розробки в аналізовані періоди часу.

Можливе збільшення чистого прибутку у потенційного інвестора ΔПі для кожного із років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховується за формулою:

$$\Delta П = (\pm\Delta Ц_о \cdot N + Ц_о \cdot \Delta N_і)_і \cdot \lambda \cdot \rho \cdot \left(1 - \frac{\vartheta}{100}\right),$$

де ±ΔЦ – зміна основного якісного показника від впровадження результатів науково-технічної розробки в аналізованому році. Зазвичай, таким показником може бути зміна ціни реалізації одиниці нової розробки в аналізованому році (відносно року до впровадження цієї розробки); $\pm\Delta Ц_о$ може мати як додатне, так і від'ємне значення (від'ємне – при зниженні ціни відносно року до впровадження цієї розробки, додатне – при зростанні ціни); N – основний кількісний показник, який визначає величину попиту на аналогічні чи подібні розробки у році до впровадження результатів нової науково-технічної розробки; $Ц_о$ – основний якісний показник, який визначає ціну реалізації нової науково-технічної розробки в аналізованому році; $Ц_б$ – основний якісний показник, який визначає ціну реалізації існуючої (базової) науково-технічної розробки у році до впровадження результатів; ΔN – зміна основного кількісного показника від впровадження результатів науково-технічної розробки в аналізованому році. Зазвичай таким показником може бути зростання попиту на науково-технічну розробку в аналізованому році (відносно року до впровадження цієї розробки); λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2023 році ставка податку на додану вартість становить 20%, а коефіцієнт λ =0,8333; ρ – коефіцієнт, який враховує рентабельність інноваційного продукту (послуги). Рекомендується брати ρ =0,2...0,5; ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2023 році ϑ =18%.

Очікуваний термін життєвого циклу розробки 1 рік, тому:

$$\Delta П = (5000 \cdot 1000 - 172 \cdot 25000) \cdot 0{,}8333 \cdot 0{,}3 \cdot \left(1 - \frac{18}{100}\right) = 13112605 \text{ грн.}$$

Далі розраховують приведену вартість збільшення всіх чистих прибутків ПП, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t} = \frac{13112605}{(1+0{,}1)^1} = 11920550 \text{ грн.},$$

де ΔП – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн.; Т – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки (приймаємо Т=1 рік); τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ =0,05…0,15; t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

Далі розраховують величину початкових інвестицій PV , які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки. Для цього можна використати формулу:

$$PV = k_{інв} \cdot ЗВ = 5 \cdot 1322810 = 6614050 \text{ грн.}$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію. Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо; зазвичай kінв=2...5, але може бути і більшим; ЗВ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, грн.

Тоді абсолютний економічний ефект $Е_{абс}$ або чистий приведений дохід для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$Е_{абс} = ПП - PV = 11920550 - 6614050 = 5306500 \text{ грн.},$$

де ПП – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, грн.; PV – теперішня вартість початкових інвестицій, грн.

Оскільки $Е_{абс}>0$, то можемо припустити про потенційну зацікавленість інвесторів у розробці.

Для остаточного прийняття рішення з цього питання необхідно розрахувати внутрішню економічну дохідність Ев або показник внутрішньої норми дохідності вкладених інвестицій та порівняти її з так званою бар'єрною ставкою дисконтування, яка визначає ту мінімальну внутрішню економічну дохідність, нижче якої інвестиції в будь-яку науково-технічну розробку вкладати буде економічно недоцільно.

Внутрішня економічна дохідність інвестицій Ев, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки, розраховується за формулою:

$$Е_в = \sqrt[Т_ж]{1 + \frac{Еабс}{PV}} = \sqrt{1 + \frac{5306500}{6614050}} = 1{,}34,$$

де $Т_ж$ – життєвий цикл розробки, роки.

Визначимо бар'єрну ставку дисконтування $\tau_{мін}$, тобто мінімальну внутрішню економічну дохідність інвестицій, нижче якої кошти у впровадження науково-технічної розробки та її комерціалізацію вкладатися не будуть.

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$ визначається за формулою:

$$\tau_{мін} = d + f = 0{,}9 + 0{,}1 = 1{,}0,$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2023 році в Україні d =0,9...0,12; f – показник, що характеризує ризикованість вкладення інвестицій; зазвичай величина f =0,05...0,5, але може бути і значно вищою.

Оскільки $E_в$=1,34>$\tau_{мін}$=0,95, то потенційний інвестор може бути зацікавлений у фінансуванні впровадження науково-технічної розробки та виведенні її на ринок, тобто в її комерціалізації.

Продовжимо подорож у світ інновацій та інвестицій, розглядаючи аспекти, пов'язані з комерційною привабливістю науково-технічної розробки та розрахунок періоду окупності інвестицій. Зануримося в деталі, щоб визначити, як ці елементи стають ключовими в залученні фінансування та реалізації ідеї на ринку $T_о$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_о = \frac{1}{E_в} = \frac{1}{1{,}34} = 0{,}75 \text{ року.}$$

Оскільки $T_о$=0,75<1…3-х років, то Зазирнемо глибше у фінансову лабіринтіну, де технології та інновації переплітаються з інвестиційними можливостями. Під То=0,75, яка витісняється уперед від прогнозного горизонту 1-3 років, виблискують не тільки технічні нововведення, але й великий комерційний потенціал науково-технічної розробки. Це не просто число, це світлофор для інвестора, який прагне знайти не тільки перспективні проекти, але й вигідні можливості в сучасному бізнес-середовищі.

5.4 Висновок до розділу 5

Зрештою, комерційна привабливість та період окупності взаємодіють, створюючи комплексний образ інвестиційної можливості. Потужний технічний старт, підкріплений ринковим попитом, разом із стратегіями ефективного управління ризиками і оптимізації окупності, роблять цей проект не лише цікавим, але й вигідним для потенційного інвестора.

ВИСНОВКИ

У роботі було розглянуто задачу розпізнавання об’єктів у мобільному застосунку. Розглянуто не лише зовнішній вигляд цих додатків, але і поглиблено вивчено технічні деталі алгоритмів, які забезпечують їхню ефективну роботу. В огляді акцент було зроблено на передових мобільних.

Був проведений аналіз існуючих мобільних додатків, і виявлено, що розглянуті застосунки вирізняються значною різноманітністю та обширним спектром додаткового функціоналу.

Надається діаграма класів з докладним описом основних класів, що є визначальними для функціональності системи. Крім того, приводиться діаграма прецедентів, яка відображає взаємодію користувачів з системою та обсяг їхніх можливостей.

Розроблений додаток для розпізнавання об’єктів було протестовано на відповідність меті дослідження. Функціонал програмного забезпечення є розширеним по відношенню до існуючих сучасних рішень та відповідає завданню.

Ця синергія технологічного та фінансового потенціалу стає ключем до інвестиційного успіху у світі невпинних змін та росту. профінансувати впровадження цієї розробки та виведення її на ринок.

Функціонал застосунку включав в себе можливість в режимі реального часу розпізнавати об'єкти, отримувати додаткову інформацію про них та відображати цю інформацію на екрані користувача. Зокрема, для покращення освітнього аспекту застосунок був налаштований на надання назв та додаткової інформації англійською мовою, щоб допомагати користувачам у вивченні нових слів та фраз.

Отже всі задачі виконані. Мету роботи досягнуто.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Vadim Vakoluk, Sergyi Baraban. USE OF INFORMATION TECHNOLOGY FOR REAL-TIME OBJECT RECOGNITION. December 2023. URL: https://conferences.vntu.edu.ua/index.php/mn/mn2024/author/submission/19621 (accessed: 06.12.2023)
2. Carmigniani, J., Furht, B., Anisetti, M. et al. Augmented reality technologies, systems and applications. Multimed Tools Appl 51, 2011 – pp.341–377
3. Benson E. R. et al. An evaluation of a geomagnetic direction sensor for vehicle guidance in precision agriculture applications //ASAE paper. – 1998. – T. 983203.
4. Buckett C. Dart in action / C. Buckett //Manning Publications, Shelter Island, NY, 2013 — P. 528.
5. B. Thomas, V. Demczuk, W. Piekarski, D. Hepworth and B. Gunther, "A wearable computer system with augmented reality to support terrestrial navigation," Digest of Papers. Second International Symposium on Wearable Computers (Cat. No.98EX215), 1998 — pp. 168-171
6. Hari Narayn. Get Online with SharePoint Online. September 2023 URL: https://link.springer.com/chapter/10.1007/978-1-4842-9726-1_1 (accessed: 15.10.2023)
7. Erica Mixon, Ivy Wigmore. What is Google Drive?. 2022. URL: https://www.techtarget.com/searchmobilecomputing/definition/Google-Drive (accessed: 15.10.2023)
8. Ski B. K. FIFO: First in, First Out. Argyll Productions, 2022.
9. Баришев Ю. В. Дискреційна модель та метод розмежування прав доступу до розподілених інформаційних ресурсів / Ю.В. Баришев, В.А. Каплун, К.В. Неуйміна // Наукові праці ВНТУ, 2017, № 2. – Вінниця, 2017.
10. Gomaa H. Designing concurrent, distributed, and real-time applications with UML //Proceedings of the 28th international conference on Software engineering. – 2006. – pp. 1059-1060.

11.Huaizhong Zhang, Mark Liptrott, Nik Bessis, Jianquan Cheng. Real–time Traffic Analysis Using Deep Learning Techniques And UAV Based Video[Електронний ресурс] – Режим доступу до ресурсу: https://core.ac.uk/download/pdf/286357582.pdf.

12.Linwood J. Displaying a Map with the Google Maps SDK //Build Location Apps on iOS with Swift. – Apress, Berkeley, CA, 2020. – pp. 89-105.

13.Jason Stagnitto. Dropbox Security 2023: The Good, the Bad and the Ugly. URL: https://www.cloudwards.net/dropbox-security/ (accessed: 15.10.2023)

14.Narzt, W., Pomberger, G., Ferscha, A. et al. Augmented reality navigation systems. Univ Access Inf Soc 4, 2006 – pp.177–187.

15.Oh J. A study on the correction of azimuthal angle by fusion of geomagnetic sensor and inertial sensor //2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN). – IEEE, 2019. – pp. 742-744.

16.Penker M. Business modeling with UML: business patterns at work. – John Wiley & Sons, 2000 – pp.13–42.

17.Rap Rayne. Beginning App Development with Flutter: Create Cross–Platform Mobile Apps 1st ed. Edition / Rap Rayne, 2019. – P.303.

18.Ronald T. Azuma; A Survey of Augmented Reality. Presence: Teleoperators and Virtual Environments 1997 – pp.355–385.

19.Wu W. React Native vs Flutter, Cross-platforms mobile application frameworks. – 2018.

20.Stroustrup B. C++ Programming Language. Addison Wesley, 2013. 1368 p.

21.Holmes D., Gosling J., Arnold K. Java Programming Language. Pearson Education, Limited, 2021. 992 p.

22.Boduch A. React and React Native. Packt Publishing - ebooks Account, 2017. 500 p.

23.Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа – Вінниця : ВНТУ, 2016. – 113 с.

ДОДАТКИ

ДОДАТОК А (Обов'язковий)

Протокол перевірки кваліфікаційної роботи на наявність текстових запозичень

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Інформаційна технологія розпізнавання об'єктів у мобільному застосунку

Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ кафедра комп'ютерних наук, ФІТА
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 95,09% Схожість 4,91%

Аналіз звіту подібності (відмітити потрібне):

✓ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

□ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ Ваколюк В.Ю.

Керівник роботи _____ Барабан С.В.

Опис прийнятого рішення

Магістерську кваліфікаційну роботу допущено до захисту

Особа, відповідальна за перевірку _____ Озеранський В.С.

ДОДАТОК Б (Обов’язковий)

Текст програми

main.dart

```
import 'package:flutter/material.dart';
import 'package:object_detection_flutter/buttonbar/help-url.dart';
import 'package:object_detection_flutter/buttonbar/info.dart';
import 'package:object_detection_flutter/buttonbar/pinfo.dart';
import 'package:object_detection_flutter/drawer_slides/Fav.dart';
import 'package:object_detection_flutter/drawer_slides/Report%20bugs.dart';
import 'package:object_detection_flutter/drawer_slides/Request%20new%20functions.dart';
import 'package:object_detection_flutter/drawer_slides/Settings.dart';
import 'package:object_detection_flutter/main/camera_view.dart';
// ignore: unused_import
import 'package:object_detection_flutter/main/navbar.dart';
import 'package:object_detection_flutter/main/Elevator.dart';

import 'package:flutter_translate/flutter_translate.dart';
import 'package:url_launcher/url_launcher.dart';
void main() {
  runApp(const MyApp());
}
class MyApp extends StatelessWidget {
  const MyApp({super.key});
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Start',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: const CameraView(),
      routes: {
        '/Helpurl' :(context) =>  HelpScreen (),
```

```
        '/Info' :(context) =>  InfoScreen (),
        '/Pinfo' :(context) =>  PTranslationPage (),
        '/elevator' :(context) =>  TranslationPage (),
        '/Fav' :(context) =>  FavoritesPage (),
        '/Report' :(context) =>  BugReportPage (),
        '/Request' :(context) =>  FeatureRequestPage (),
        '/Settings' :(context) =>  SettingsPage (),

      },
    );
  }
}
```

scan_controller.dart

```
import 'package:camera/camera.dart';
import 'package:flutter_tflite/flutter_tflite.dart';
import 'package:get/get.dart';
import 'package:permission_handler/permission_handler.dart';
class ScanController extends GetxController {
  @override
  void onInit() {
    super.onInit();
    initCamera();
    initTflite();
  }
  @override
  void dispose() {
    super.dispose();
    cameraController.dispose();
  }

  late CameraController cameraController;
  late List<CameraDescription> Cameras;

  var isCameraInitialized = false.obs;
  var cameraCount = 0;
  var x, y, w, h = 0.0;
```

```
var label = "";
var val = "";
initCamera() async {
  if (await Permission.camera.request().isGranted) {
    Cameras = await availableCameras();
    cameraController =
        await CameraController(Cameras[0], ResolutionPreset.max);
    await cameraController.initialize().then((value) {
      cameraCount = 0;
      cameraController.startImageStream((image) {
        cameraCount++;
        if (cameraCount % 10 == 0) {
          cameraCount = 0;
          objectDetector(image);
        }
        update();
      });
      update();
    });
    isCameraInitialized(true);
    update();
  } else {
    print("Permission denied");
  }
}
initTflite() async {
  await Tflite.loadModel(
    model: "assets/model.tflite",
    labels: "assets/label.txt",
    isAsset: true,
    numThreads: 1,
    useGpuDelegate: false,
  );
}
objectDetector(CameraImage image) async {
  var detector = await Tflite.runModelOnFrame(
    bytesList: image.planes.map((e) {
```

```
        return e.bytes;
      }).toList(),
      asynch: true,
      imageHeight: image.height,
      imageWidth: image.width,
      imageMean: 127.5,
      imageStd: 127.5,
      numResults: 1,
      rotation: 90,
      threshold: 0.4,
    );
    if (detector != null) {
      var OurDetectedObject = detector.first;
      print(

"**********************************************************************************************************result is $detector");
      val = detector[0]['label'];
      if (OurDetectedObject['confidence'] * 100 > 45) {
        label = OurDetectedObject['label'].toString();
        print(label);
        h = OurDetectedObject['rect'][0];
        w = OurDetectedObject['rect'][1];
        x = OurDetectedObject['rect'][2];
        y = OurDetectedObject['rect'][3];
      }
      update();
    }
  }
}
```

Elevator.dart

```
import 'package:flutter/material.dart';
import 'package:flutter_translate/flutter_translate.dart';
import 'package:translator/translator.dart';
class TranslationPage extends StatefulWidget {
  @override
```

```
  _TranslationPageState createState() => _TranslationPageState();
}
class _TranslationPageState extends State<TranslationPage> {
  final TextEditingController _inputController = TextEditingController();
  String _outputText = '';

  void _translate() async {
    final translator = GoogleTranslator();
    // Translating from English to Ukrainian
    Translation translation = await translator.translate(
      _inputController.text,
      from: 'en',
      to: 'uk',
    );
    setState(() {
      _outputText = translation.text;
    });
  }
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Translator'),
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            TextField(
              controller: _inputController,
              decoration: InputDecoration(
                labelText: 'Enter text in English',
                border: OutlineInputBorder(),
              ),
            ),
            SizedBox(height: 20),
```

```
            ElevatedButton(
              onPressed: () {
                _translate();
              },
              child: Text('Translate'),
            ),
            SizedBox(height: 20),
            Text(
              'Translation:',
              style: TextStyle(fontSize: 18),
            ),
            SizedBox(height: 10),
            Text(
              _outputText,
              style: TextStyle(fontSize: 16, fontWeight: FontWeight.bold),
            ),
          ],
        ),
      ),
    );
  }
}
```

camera_view.dart

```
import 'dart:io';
import 'package:camera/camera.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:get/get.dart';
import 'package:get/get_state_manager/get_state_manager.dart';
import 'package:get/get_state_manager/src/simple/get_state.dart';
import 'package:object_detection_flutter/recognizer-controller/scan_controller.dart';
import 'package:object_detection_flutter/main/navbar.dart';
// ignore: duplicate_import
import 'package:object_detection_flutter/main/navbar.dart';
class CameraView extends StatefulWidget {
  const CameraView({super.key});
```

```
 @override
 State<CameraView> createState() => _CameraViewState();
}
int myIndex = 0;
List<Widget> widgetList = const [
Text('Pinfo', style: TextStyle(fontSize: 30)),
Text('Helpurl', style: TextStyle(fontSize: 30)),
Text('Info', style: TextStyle(fontSize: 30)),
];
class _CameraViewState extends State<CameraView> {
 @override
 Widget build(BuildContext context) {
  return Scaffold(
   drawer: NavBar(),
   appBar: AppBar(
    title: const Text("Recongizerhelper",
      style: TextStyle(
       color: Color.fromARGB(255, 243, 243, 243),
       fontWeight: FontWeight.w500,
      )),
    backgroundColor: Color.fromARGB(255, 231, 71, 71),
    centerTitle: true,

    actions: <Widget>[
     Row(
      children: [
       IconButton(
        icon: const Icon(Icons.book),
        onPressed: (){
        Navigator.pushNamed(context, '/elevator');
        },
       ),
       IconButton(
      icon: const Icon(Icons.exit_to_app),
      onPressed: (){
       if (Platform.isAndroid){
        SystemNavigator.pop();
```

```
        } else{
          exit(0);
        }
      },
    )
      ],
    )
  ],
),
body: GetBuilder<ScanController>(
    init: ScanController(),
    builder: (controller) {
      return controller.isCameraInitialized.value
          ? Stack(
              children: [
                CameraPreview(controller.cameraController),
                Container(
                  width: double.infinity,
                  height: double.infinity,
                  decoration: BoxDecoration(
                    borderRadius: BorderRadius.circular(8),
                    border: Border.all(color: Color.fromARGB(255, 201, 1, 241), width: 4.0),
                  ),
                  child: Column(
                    mainAxisSize: MainAxisSize.min,
                    children: [
                      SizedBox(
                        width: 600,
                        height: 00
                      ),
                      Column(
                        mainAxisAlignment: MainAxisAlignment.start,
                        children: [
                          Container(
                              color: Color.fromARGB(255, 180, 170, 170),
                              child: Text(
                                controller.label,
```

```
                        style: TextStyle(
                          fontSize: 20,
                        ),
                      )),
                  ],
                ),
              ],
            ),
          ),
        ],
      )
          : Center(
            child: Text("Loading cam"),
          );
    },
  ),
bottomNavigationBar: BottomNavigationBar(
    //ignore showSelectedLabels: false
      showUnselectedLabels: false,
      backgroundColor: Colors.pink,
      type:BottomNavigationBarType.fixed,
      onTap: (index){
        setState(() {
          myIndex =index;
        });
              // Conditionally navigate based on the selected index
    if (index == 0) {
      // Pinfo
      // You can replace this with the appropriate navigation logic
      Navigator.pushNamed(context, '/Pinfo');
    } else if (index == 1) {
      // Helpurl
      Navigator.pushNamed(context, '/Helpurl');
    } else if (index == 2) {
      // Info
     Navigator.pushNamed(context, '/Info');
    }
```

```
        },
        currentIndex: myIndex,
              items: const [
                BottomNavigationBarItem(icon: Icon(Icons.home), label: 'Pinfo',
                backgroundColor: Colors.blue),
                BottomNavigationBarItem(icon: Icon(Icons.mouse), label: 'Helpurl',
                 backgroundColor: Colors.deepOrange),
                BottomNavigationBarItem(icon: Icon(Icons.info), label: 'Info',
                 backgroundColor: Colors.lightGreenAccent),
              ],
            ),
   );
  }
```

navbar.dart

```
import 'package:flutter/material.dart';
import 'package:get/get.dart';
class NavBar extends StatefulWidget {
  const NavBar({super.key});

 @override
 State<NavBar> createState() => _NavBarState();
}
class _NavBarState extends State<NavBar> {
 @override
 Widget build(BuildContext context) {
  return Drawer(
    child: ListView(
      padding: EdgeInsets.zero,
      children: [
       UserAccountsDrawerHeader(
        accountName: Text('Wirte me on'),
        accountEmail: Text('vakdim3@gmail.com'),
        currentAccountPicture: CircleAvatar(
          child: ClipOval(
            child: Image.network(
              '',
              fit: BoxFit.cover,
```

```
        width: 90,
        height: 90,
      ),
    ),
  ),
  decoration: BoxDecoration(
    color: Color.fromARGB(255, 235, 2, 223),
    image: DecorationImage(
        fit: BoxFit.fill,
        image: NetworkImage(
            '')),
  ),
),

ListTile(
  leading: Icon(Icons.favorite),
  title: Text('Favorites objects'),
  onTap: () => Navigator.pushNamed(context, '/Fav'),
),
Divider(),
ListTile(
  leading: Icon(Icons.share),
  title: Text('Report bugs'),
  onTap: () => Navigator.pushNamed(context, '/Report'),
),
Divider(),
ListTile(
  leading: Icon(Icons.notifications),
  title: Text('Request new functions'),
  onTap: () => Navigator.pushNamed(context, '/Request'),
),
Divider(),
ListTile(
  leading: Icon(Icons.settings),
  title: Text('Settings'),
  onTap: () => Navigator.pushNamed(context, '/Settings'),
),
```

```
        Divider(),
        ListTile(
          title: Text('Exit'),
          leading: Icon(Icons.exit_to_app),
          onTap: () {},
        ),
       ],
     ),
   );
  }
}
```

Fav.dart

```
import 'package:flutter/material.dart';


class FavoritesPage extends StatefulWidget {
  @override
  _FavoritesPageState createState() => _FavoritesPageState();
}

class _FavoritesPageState extends State<FavoritesPage> {
  List<FavoriteItem> favoriteItems = [
    FavoriteItem('Mouse', Icons.mouse),
    FavoriteItem('Laptop', Icons.laptop),
    FavoriteItem('Monitor', Icons.monitor),
    FavoriteItem('Smartphone', Icons.smartphone),
    FavoriteItem('Headphones', Icons.headphones),
    FavoriteItem('Camera', Icons.camera),
  ];

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Favorites'),
      ),
      backgroundColor: Colors.black, // Set background color to black
```

```
    body: ListView.builder(
      itemCount: favoriteItems.length,
      itemBuilder: (context, index) {
        return _buildFavoriteItem(
          favoriteItems[index].itemName,
          favoriteItems[index].iconData,
        );
      },
    ),
    floatingActionButton: FloatingActionButton(
      onPressed: () {
        _showAddFavoriteDialog();
      },
      child: Icon(Icons.add),
    ),
  );
}

Widget _buildFavoriteItem(String itemName, IconData iconData) {
  return Card(
    color: Colors.grey[900], // Set card color to dark grey
    child: ListTile(
      title: Text(
        itemName,
        style: TextStyle(color: Colors.white),
      ),
      leading: Icon(
        iconData,
        color: Colors.white,
      ),
    ),
  );
}

void _showAddFavoriteDialog() {
  showDialog(
    context: context,
```

```
      builder: (BuildContext context) {
        TextEditingController _itemNameController = TextEditingController();
        return AlertDialog(
          title: Text('Add Favorite Item'),
          content: Column(
            mainAxisSize: MainAxisSize.min,
            children: [
              TextField(
                controller: _itemNameController,
                decoration: InputDecoration(labelText: 'Item Name'),
              ),
              SizedBox(height: 10),
              ElevatedButton(
                onPressed: () {
                  if (_itemNameController.text.isNotEmpty) {
                    _addFavoriteItem(_itemNameController.text, Icons.star);
                    Navigator.pop(context);
                  }
                },
                child: Text('Add'),
              ),
            ],
          ),
        );
      },
    );
  }

  void _addFavoriteItem(String itemName, IconData iconData) {
    setState(() {
      favoriteItems.add(FavoriteItem(itemName, iconData));
    });
  }
}

class FavoriteItem {
  final String itemName;
```

```
  final IconData iconData;

  FavoriteItem(this.itemName, this.iconData);
}
```

Report bugs.dart

```
import 'package:flutter/material.dart';

class BugReportPage extends StatefulWidget {
  @override
  _BugReportPageState createState() => _BugReportPageState();
}

class _BugReportPageState extends State<BugReportPage> {
  final TextEditingController _bugDetailsController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Bug Report'),
        backgroundColor: Colors.red, // Set app bar color to red
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: <Widget>[
            Text(
              'Please provide details about the bug:',
              style: TextStyle(fontSize: 18, color: Colors.red), // Set text color to red
            ),
            SizedBox(height: 20),
            TextFormField(
              controller: _bugDetailsController,
              maxLines: 5,
              style: TextStyle(color: Colors.black), // Set text color to black
              decoration: InputDecoration(
```

```
            labelText: 'Bug Details',
            border: OutlineInputBorder(),
          ),
        ),
        SizedBox(height: 20),
        ElevatedButton(
          onPressed: () {
            _submitBugReport();
          },
          style: ElevatedButton.styleFrom(
            primary: Colors.red, // Set button color to red
            textStyle: TextStyle(color: Colors.white), // Set text color to white
          ),
          child: Text('Submit Bug Report'),
        ),
      ],
    ),
  ),
 );
}

void _submitBugReport() {
  String bugDetails = _bugDetailsController.text;

  print('Bug Details: $bugDetails');
}
}

void main() {
 runApp(MaterialApp(
  home: BugReportPage(),
 ));
}
```

Request new functions.dart

```
import 'package:flutter/material.dart';
class FeatureRequestPage extends StatefulWidget {
 @override
```

```
  _FeatureRequestPageState createState() => _FeatureRequestPageState();
}

class _FeatureRequestPageState extends State<FeatureRequestPage> {
  final TextEditingController _functionDetailsController =
      TextEditingController();
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Feature Request'),
        backgroundColor: Colors.teal,
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: <Widget>[
            Text(
              'Please provide details about the new function:',
              style: TextStyle(fontSize: 18),
            ),
            SizedBox(height: 20),
            TextFormField(
              controller: _functionDetailsController,
              maxLines: 5,
              style: TextStyle(color: Colors.black),
              decoration: InputDecoration(
                labelText: 'Function Details',
                border: OutlineInputBorder(),
              ),
            ),
            SizedBox(height: 20),
            ElevatedButton(
              onPressed: () {
                _submitFunctionRequest();
              },
```

```
            style: ElevatedButton.styleFrom(
              primary: Colors.teal,
              textStyle: TextStyle(color: Colors.white),
            ),
            child: Text('Submit Function Request'),
          ),
        ],
      ),
    ),
  );
 }

 void _submitFunctionRequest() {
  String functionDetails = _functionDetailsController.text;
  print('Function Details: $functionDetails');
 }
}
```

Settings.dart

```
import 'package:flutter/material.dart';

class SettingsPage extends StatefulWidget {
 @override
 _SettingsPageState createState() => _SettingsPageState();
}

class _SettingsPageState extends State<SettingsPage> {
 bool isDarkModeEnabled = false;
 bool areNotificationsEnabled = false;
 String selectedLanguage = 'English';

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: Text('Settings'),
   ),
```

```
body: ListView(
  children: <Widget>[
    ListTile(
      title: Text('Dark Mode'),
      trailing: Switch(
        value: isDarkModeEnabled,
        onChanged: (value) {
          setState(() {
            isDarkModeEnabled = value;
            _toggleDarkMode();
          });
        },
      ),
    ),
    ListTile(
      title: Text('Notifications'),
      trailing: Switch(
        value: areNotificationsEnabled,
        onChanged: (value) {
          setState(() {
            areNotificationsEnabled = value;
            // Handle notifications switch change
          });
        },
      ),
    ),
    ListTile(
      title: Text('Language'),
      trailing: DropdownButton<String>(
        value: selectedLanguage,
        onChanged: (String? newValue) {
          setState(() {
            selectedLanguage = newValue!;
            // Handle language change
          });
        },
        items: <String>['English', 'Spanish', 'French', 'German','Ukrain' ]
```

```
              .map<DropdownMenuItem<String>>((String value) {
            return DropdownMenuItem<String>(
              value: value,
              child: Text(value),
            );
          }).toList(),
        ),
      ),
      ListTile(
        title: Text('Feedback'),
        onTap: () {
          // Handle feedback button press
          _showFeedbackDialog();
        },
      ),
    ],
  ),
 );
}

void _toggleDarkMode() {
 // dark mode handling logic here
 if (isDarkModeEnabled) {
  // Enable dark mode
  MaterialApp(
   theme: ThemeData.dark(),
  );
 } else {
  // Disable dark mode
  MaterialApp(
   theme: ThemeData.light(),
  );
 }
}

void _showFeedbackDialog() {
 showDialog(
```

```
      context: context,
      builder: (BuildContext context) {
        return AlertDialog(
          title: Text('Feedback'),
          content: Text('Provide your feedback here.'),
          actions: <Widget>[
            TextButton(
              onPressed: () {
                Navigator.pop(context);
              },
              child: Text('Close'),
            ),
            TextButton(
              onPressed: () {
                Navigator.pop(context);
              },
              child: Text('Submit'),
            ),
          ],
        );
      },
    );
  }
}
```

help-url.dart

```
  import 'package:flutter/material.dart';
import 'package:webview_flutter/webview_flutter.dart';

class HelpScreen extends StatelessWidget {
  const HelpScreen({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('url for github/help'),
```

```
      ),
      body: WebView(
        initialUrl: 'https://github.com/vakdim3',
        javascriptMode: JavascriptMode.unrestricted,
        onWebViewCreated: (WebViewController webViewController) {
        },
        onPageFinished: (String url) {
        },
      ),
    );
  }
}
```

info.dart

```
import 'package:flutter/material.dart';
import 'package:url_launcher/url_launcher.dart';

class InfoScreen extends StatelessWidget {
  const InfoScreen({Key? key}) : super(key: key);

  // Function to open the default email app
  void _launchEmail() async {
    final Uri emailLaunchUri = Uri(
      scheme: 'mailto',
      path: 'info@gmail.com',
      queryParameters: {
        'subject': 'Feedback for Recongizerhelper App',
        'body': 'Hello, I would like to provide some feedback...',
      },
    );

    if (await canLaunch(emailLaunchUri.toString())) {
      await launch(emailLaunchUri.toString());
    } else {
      // Handle error
      print('Could not launch email');
    }
```

```
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('App Info'),
    ),
    body: Padding(
      padding: const EdgeInsets.all(16.0),
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text(
            'Recongizerhelper App',
            style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),
          ),
          SizedBox(height: 16),
          Text(
            'Version: 0.0.1',
            style: TextStyle(fontSize: 18),
          ),
          SizedBox(height: 16),
          Text(
            'Ласкаво просимо до майбутнього розпізнавання обє*ктів - нашого нового мобільного застосунку, який змінить ваш погляд на світ навколо. Неважливо, чи ви любите фотографувати, досліджувати або просто цікавитесь оточуючим середовищем, наш застосунок вам точно сподобається.',
            style: TextStyle(fontSize: 18),
          ),
          SizedBox(height: 32),
          ElevatedButton(
            onPressed: _launchEmail,
            child: Text('Contact Us'),
          ),
        ],
      ),
    ),
  );
```

```
  }
}
```

pinfo.dart

```
import 'package:flutter/material.dart';

class PTranslationPage extends StatelessWidget {
  final Map<String, String> wordTranslations = {
    'background': 'фон',
    'tench': 'тинга',
    'goldfish': 'золота рибка',
    'great white shark': 'велика біла акула',
    'tiger shark': 'тигрова акула',
    'hammerhead': 'молотоголова акула',
    'electric ray': 'електрична ската',
    'stingray': 'ската',
    'cock': 'півень',
    'hen': 'курка',
    'ostrich': 'страус',
    'brambling': 'зяблик',
  };

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Word Translations'),
      ),
      body: ListView.builder(
        itemCount: wordTranslations.length,
        itemBuilder: (context, index) {
          final word = wordTranslations.keys.elementAt(index);
          final translation = wordTranslations[word];

          return ListTile(
            title: Text(word),
            subtitle: Text(translation!),
          );
```

```
        },
      ),
    );
  }
}

void main() {
  runApp(MaterialApp(
    home: PTranslationPage(),
  ));
}
```

.flutter-plugins

```
camera=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\camera-0.10.5+4\\
camera_android=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\camera_android-0.10.8+10\\
camera_avfoundation=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\camera_avfoundation-0.9.13+5\\
camera_web=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\camera_web-0.3.2+3\\
file_selector_linux=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\file_selector_linux-0.9.2+1\\
file_selector_macos=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\file_selector_macos-0.9.3+3\\
file_selector_windows=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\file_selector_windows-0.9.3+1\\
flutter_plugin_android_lifecycle=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\flutter_plugin_android_
lifecycle-2.0.16\\
flutter_tflite=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\flutter_tflite-1.0.1\\
image_picker=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\image_picker-0.8.9\\
image_picker_android=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\image_picker_android-0.8.8+1\\
image_picker_for_web=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\image_picker_for_web-2.2.0\\
image_picker_ios=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\image_picker_ios-0.8.8+2\\
image_picker_linux=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\image_picker_linux-0.2.1+1\\
image_picker_macos=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\image_picker_macos-0.2.1+1\\
image_picker_windows=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\image_picker_windows-
0.2.1+1\\
permission_handler=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\permission_handler-10.4.5\\
permission_handler_android=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\permission_handler_androi
d-10.3.6\\
permission_handler_apple=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\permission_handler_apple-
9.1.4\\
permission_handler_windows=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\permission_handler_wind
ows-0.1.3\\
url_launcher=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\url_launcher-6.1.14\\
url_launcher_android=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\url_launcher_android-6.1.0\\
```

url_launcher_ios=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\url_launcher_ios-6.1.5\\

url_launcher_linux=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\url_launcher_linux-3.0.6\\

url_launcher_macos=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\url_launcher_macos-3.0.7\\

url_launcher_web=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\url_launcher_web-2.0.20\\

url_launcher_windows=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\url_launcher_windows-3.0.8\\

webview_flutter=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\webview_flutter-3.0.4\\

webview_flutter_android=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\webview_flutter_android-2.10.4\\

webview_flutter_wkwebview=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\webview_flutter_wkwebview-2.9.5\\

pubspec.yaml

```
name: object_detection_flutter
description: A new Flutter project.
publish_to: 'none'
version: 1.0.0+1
environment:
  sdk: '>=3.0.0 <4.0.0'
dependencies:
  flutter:
    sdk: flutter
  get: ^4.6.5
  camera: ^0.10.5+2
  permission_handler: ^10.3.0
  image: ^3.2.0
  flutter_tflite: ^1.0.1
  webview_flutter: ^3.0.0
  cupertino_icons: ^1.0.2
  flutter_translate: ^4.0.4
  translator: ^0.1.7
  url_launcher: ^6.1.14
dev_dependencies:
  flutter_test:
    sdk: flutter
  flutter_lints: ^2.0.0
```

Додаток В (Обов'язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

«ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ У МОБІЛЬНОМУ ЗАСТОСУНКУ»

Виконав: студент 2 курсу
групи 1КН-22м
спеціальності 122 Комп'ютерні науки
_____ Ваколюк В.Ю.

Керівник: к. т. н., доцент каф. КН
_____ Барабан С.В.
« 07 » 12 2023р.

Вінниця ВНТУ - 2023 рік

Актуальність, мета та задачі мкр.

Актуальність

Інформаційна технологія розпізнавання об'єктів в реальному часі стає визначальним елементом сучасного технологічного ландшафту. З кожним днем вона набуває все більшої актуальності, забезпечуючи інноваційні рішення в багатьох сферах життя. У сучасному світі, де дані стають новою валютою, системи розпізнавання об'єктів в реальному часі використовуються для вдосконалення безпеки, покращення транспортної інфраструктури, розвитку медичних технологій та багатьох інших сфер. Завдяки цим технологіям можливо автоматизувати процеси, що раніше вимагали великої кількості людських ресурсів. Застосування реального часу у розпізнаванні об'єктів відкриває нові можливості для ефективного контролю, моніторингу та аналізу. Це дозволяє реагувати на події миттєво, роблячи прийняття рішень швидшим та точнішим. Зростання обчислювальної потужності, розвиток штучного інтелекту та постійне удосконалення алгоритмів розпізнавання роблять цю технологію дедалі більш доступною та ефективною. Інформаційна технологія розпізнавання об'єктів в реальному часі — це не просто інструмент, вона стає кроком у майбутнє, де швидкість, точність та надійність є визначальними аспектами для досягнення успіху в будь-якій галузі.

Мета

Метою даної роботи є створення мобільного застосунку для сканування об’єктів у реальному часі з метою надання додаткової інформації про об’єкт та його назву та допомогу людям старшого віку на підприємствах вивчити англійську мову.

Задачі, які потрібно розв’язати для досягнення мети:

1 Спроектувати архітектуру програмного забезпечення

2 Розробити функціонал

3 Створити та навчити модель машинного навчання для розпізнавання

4 Протестувати розроблену систему в реальних умовах

Частина слів якими було розширено базу розпізнавання об’єктів.

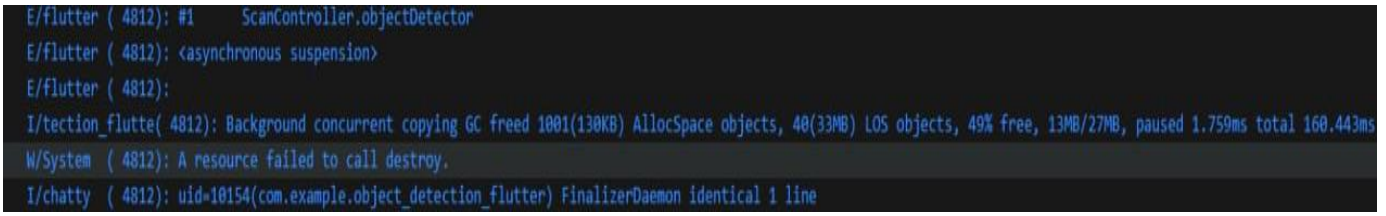
Назва	Переклад	Місце
background	фон	1
tench	линь	2
goldfish	золота рибка	3
great white shark	велика біла акула	4
tiger shark	тигрова акула	5
hammerhead	молотоголовий	6
electric ray	електричний промінь	7
stingray	скат	8
cock	півень	9
hen	курка.	10
ostrich	страус	11
brambling	бренькання	12

Архітектура та перелік плагінів

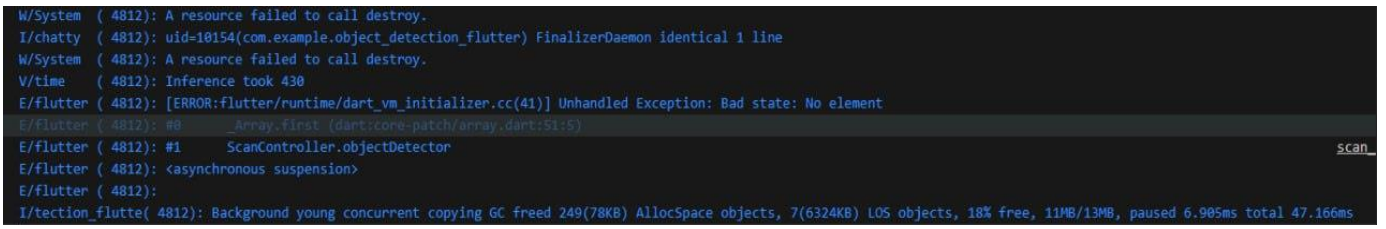
```
camera=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\camera-0.10.5+4\\
camera_android=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\camera_android-0.10.8+10\\
camera_avfoundation=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\camera_avfoundation-0.9.13+5\\
camera_web=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\camera_web-0.3.2+3\\
file_selector_linux=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\file_selector_linux-0.9.2+1\\
file_selector_macos=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\file_selector_macos-0.9.3+3\\
file_selector_windows=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\file_selector_windows-0.9.3+1\\
flutter_plugin_android_lifecycle=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\flutter_plugin_android_lifecycle-2.0.16\\
flutter_tflite=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\flutter_tflite-1.0.1\\
image_picker=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\image_picker-0.8.9\\
image_picker_android=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\image_picker_android-0.8.8+1\\
image_picker_for_web=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\image_picker_for_web-2.2.0\\
image_picker_ios=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\image_picker_ios-0.8.8+2\\
image_picker_linux=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\image_picker_linux-0.2.1+1\\
image_picker_macos=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\image_picker_macos-0.2.1+1\\
image_picker_windows=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\image_picker_windows-0.2.1+1\\
permission_handler=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\permission_handler-10.4.5\\
permission_handler_android=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\permission_handler_android-10.3.6\\
permission_handler_apple=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\permission_handler_apple-9.1.4\\
permission_handler_windows=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\permission_handler_windows-0.1.3\\
url_launcher=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\url_launcher-6.1.14\\
url_launcher_android=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\url_launcher_android-6.1.0\\
url_launcher_ios=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\url_launcher_ios-6.1.5\\
url_launcher_linux=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\url_launcher_linux-3.0.6\\
url_launcher_macos=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\url_launcher_macos-3.0.7\\
url_launcher_web=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\url_launcher_web-2.0.20\\
url_launcher_windows=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\url_launcher_windows-3.0.8\\
webview_flutter=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\webview_flutter-3.0.4\\
webview_flutter_android=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\webview_flutter_android-2.10.4\\
webview_flutter_wkwebview=C:\\Users\\Vadim\\AppData\\Local\\Pub\\Cache\\hosted\\pub.dev\\webview_flutter_wkwebview-2.9.5\\
```

Результати тестування відклику програми

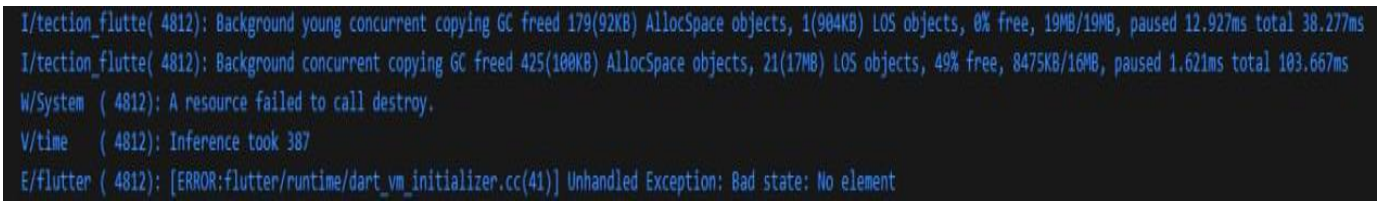
Перевірка швидкості розпізнавання



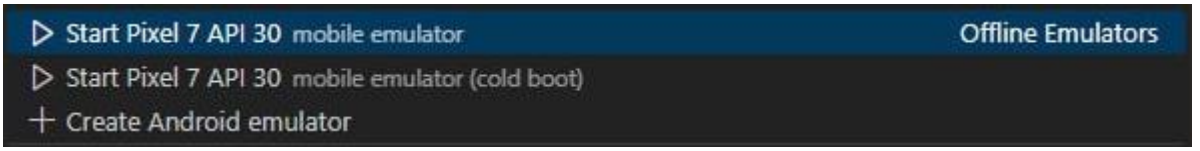
Перевірка швидкості оновлення обєктів у програмі



Перевірка швидкості оновлень



Результати тестування на різних пристроях



7.6" Fold-in with out...		7,59"	1768x2208	420dpi
7.4" Rollable		7,4"	1600x2428	420dpi
6.7" Horizontal Fold-in		6,7"	1080x2636	xxhdpi
5.4" FWVGA		5,4"	480x854	mdpi
5.1" WVGA		5,1"	480x800	mdpi
4.7" WXGA		4,7"	720x1280	xhdpi

Алгоритм роботи перекладача

Результати апаратного тестування

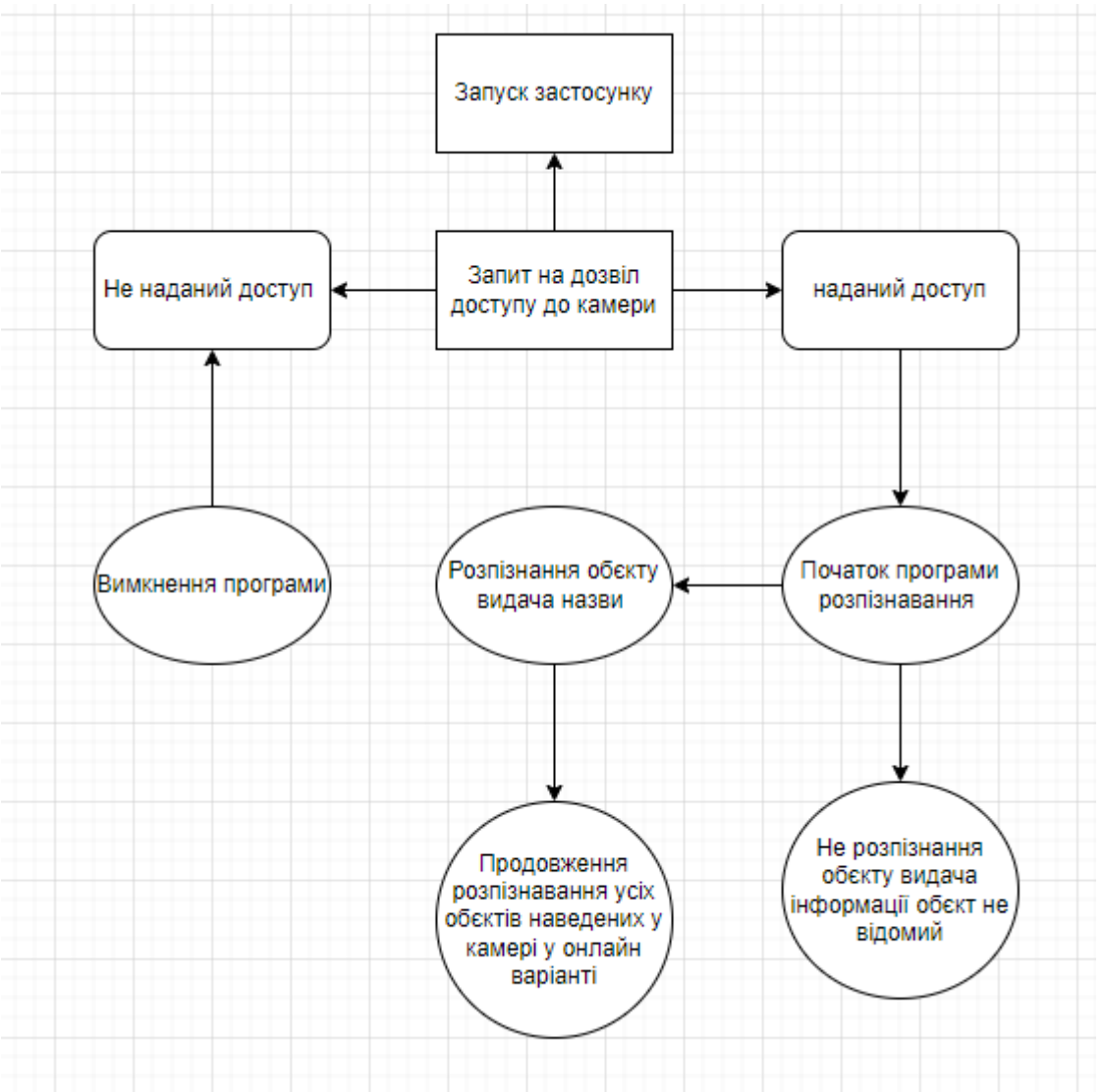
```
Running Gradle task 'assembleDebug'...

Running Gradle task 'assembleDebug'... Done                       267,0s (!)
✓ Built build/app/outputs/apk/debug/app-debug.apk.
Installing build/app/outputs/apk/app.apk...                        45,4s
D/FlutterActivity(20383): Using the launch theme as normal theme.
D/FlutterActivityAndFragmentDelegate(20383): Setting up FlutterEngine.
D/FlutterActivityAndFragmentDelegate(20383): No preferred FlutterEngine was provided. Creating a new FlutterEngine for this FlutterFragment.
```

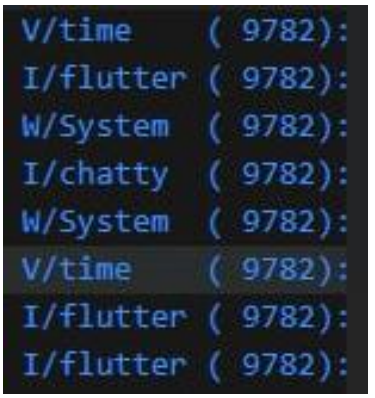
Результати тестування на сумісність

Resizable (Experimen...		6,0"	1080x2340	420dpi
Pixel XL		5,5"	1440x2560	560dpi
Pixel 7 Pro	▶	6,71"	1440x3120	560dpi
Pixel 7	▶	6,31"	1080x2400	420dpi
Pixel 6a	▶	6,13"	1080x2400	420dpi
Pixel 6 Pro		6,7"	1440x3120	560dpi
Pixel 6		6,4"	1080x2400	420dpi
Pixel 5		6,0"	1080x2340	440dpi
Pixel 4a		5,8"	1080x2340	440dpi
Pixel 4 XL		6,3"	1440x3040	560dpi
Pixel 4	▶	5,7"	1080x2280	440dpi
Pixel 3a XL		6,0"	1080x2160	400dpi

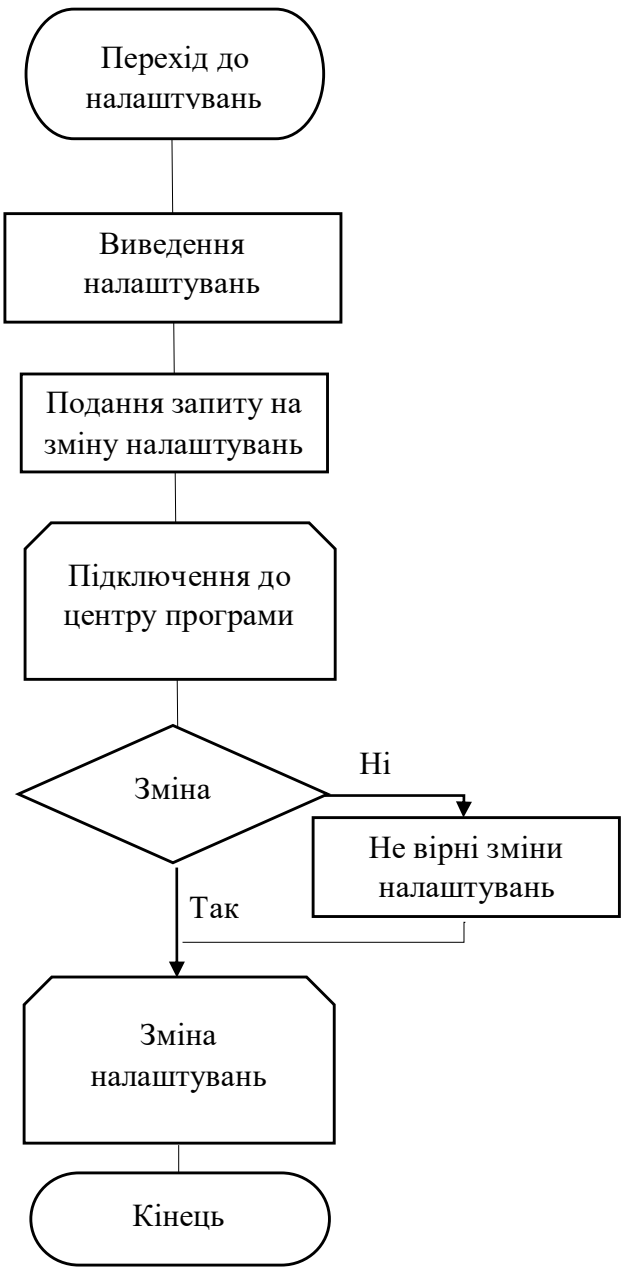
Алгоритм роботи програми



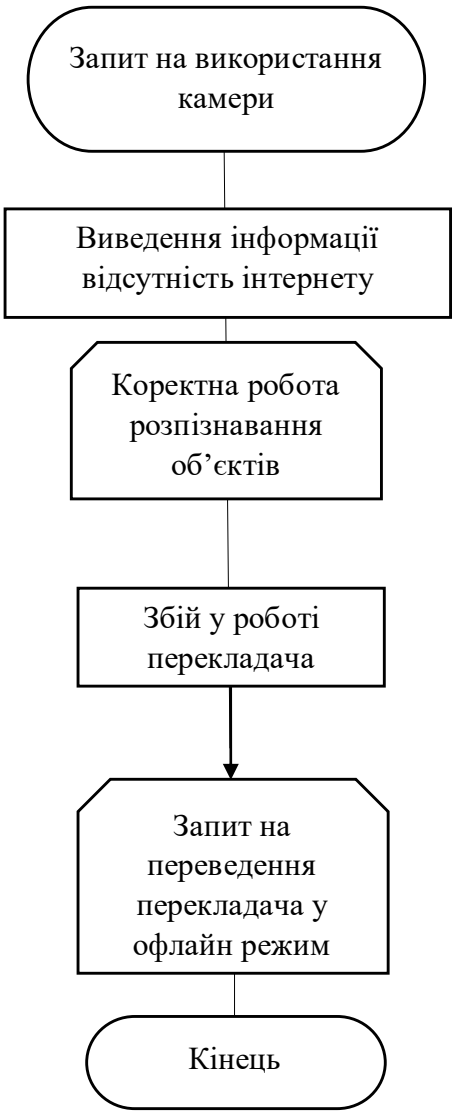
Результати тестування відмовостійкість



Алгоритм роботи налаштувань у додатку



Алгоритм роботи за відсутності мережевого підключення



Висновки з виконаної роботи

ВИСНОВОК

Інформаційна технологія розпізнавання об'єктів в реальному часі визначає нову еру технологічного розвитку, надаючи значущий вплив на різноманітні сфери людського життя. Її зростаюча актуальність свідчить про те, що ця технологія не просто відповідає сучасним викликам, але й активно формує майбутнє, визначаючи нові стандарти ефективності та інновацій. У цьому висновку розглянемо ключові аспекти її впливу на різні галузі та визначимо перспективи подальшого розвитку. Однією з визначальних характеристик інформаційної технології розпізнавання об'єктів в реальному часі є її роль у забезпеченні безпеки. Застосування систем розпізнавання облич, аналізу поведінки та виявлення непередбачуваних ситуацій дозволяє створити ефективні механізми контролю та реагування на потенційні загрози. Такі системи використовуються в аеропортах, на великих заходах, в об'єктах критичної інфраструктури, надаючи безпеку та спокій громадянам.

У рамках цієї дослідницької роботи було успішно реалізовано мету створення мобільного застосунку, спрямованого на сканування об'єктів у реальному часі з метою надання додаткової інформації про них, а також допомоги людям старшого віку у вивченні англійської мови на підприємствах. Мобільний застосунок був розроблений з урахуванням потреб користувачів, особливо з фокусом на представників покоління людей похилого віку. Його інтерфейс був спроектований з урахуванням простоти використання та зручності навігації, забезпечуючи комфортне користування для всіх категорій користувачів.

У реалізації цього проекту було успішно виконано низьку завдань, щоб досягти своєї основної мети — створення мобільного застосунку для сканування об'єктів у реальному часі. Від спроектування архітектури програмного забезпечення до розробки функціоналу та створення та навчання моделі машинного навчання для розпізнавання, кожен етап був виконаний відмінно

Додаток Г (Обов’язковий)

Інструкція користувача

Перш за все потрібно встановити додаток. Щоб встановити додаток на ваш пристрій, виконайте кілька простих кроків. Завантажте файл застосунку за ім'ям app-recognizer-release.apk, а потім запустіть його для початку процесу встановлення.

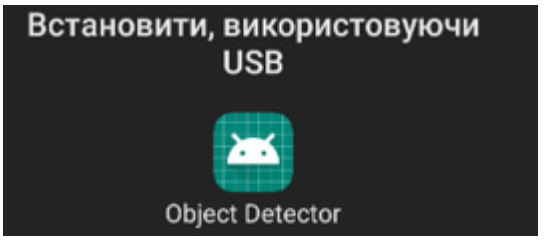


Рисунок Г. 1 – Встановлення додатку

Далі буде екран запиту на використання камери у додатку розпізнавання об’єктів у мобільному застосунку Рис. Г.2

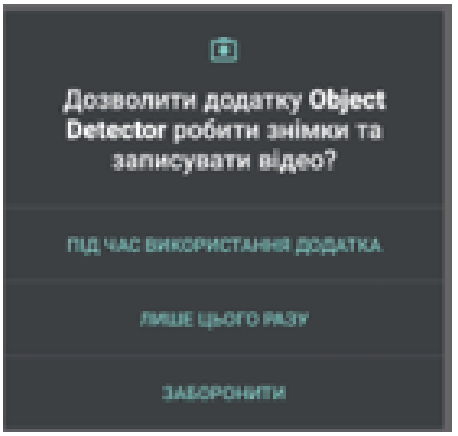


Рисунок Г. 2 – Встановлення додатку

Після підтвердження запиту на розпізнавання програма автоматично починає розпізнавати об’єкти які знаходяться перед камерою як продемонстровано на рис. Г. 3

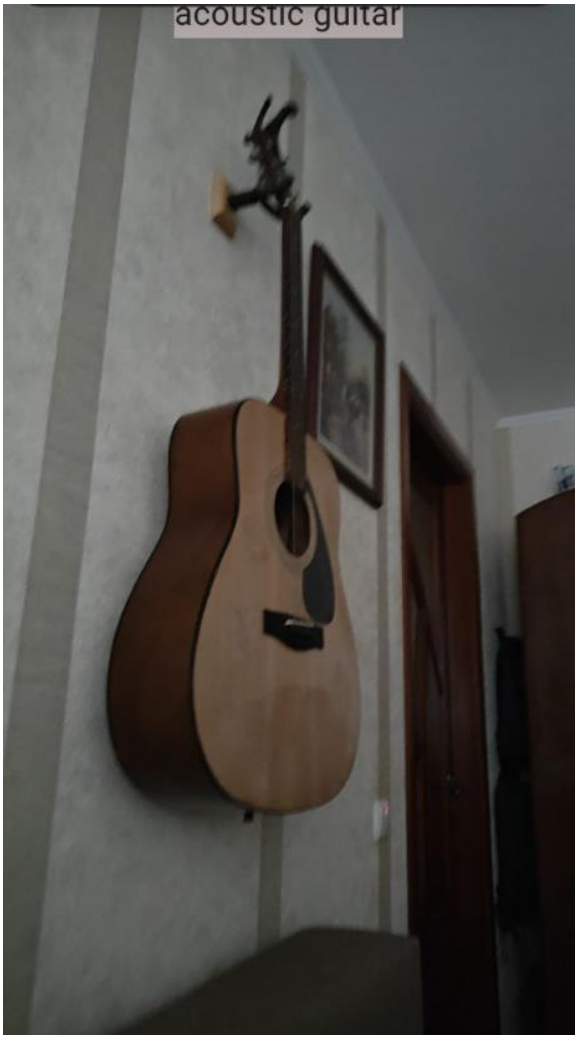


Рисунок Г. 3 – Розпізнавання об’єктів у мобільному застосунку

Після наведених дій програма повністю готова до використання.