МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп’ютерних наук

Кафедра комп'ютерної інженерії

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри,
д-р техн. наук, проф.

_____ І. М. Журавська
підпис
«___» _____2024 р.

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

Автоматизація планування подачі міського транспорту за даними IP-відеоспостереження системи «Розумне місто»

Спеціальність «Комп’ютерна інженерія»

123 – КМР.1 – 605.21810921

Студент _____І. А. Пастушок
підпис
«__» _____ 2024 р.

Керівник д-р техн. наук, професор _____І. М. Журавська
підпис
«__» _____ 2024 р.

Миколаїв – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет імені Петра Могили
Факультет комп'ютерних наук
Кафедра комп'ютерної інженерії

ЗАТВЕРДЖУЮ

Зав. кафедри

д-р техн. наук, проф.

_____ І. М. Журавська

« __ » _____ 2023 р.

ЗАВДАННЯ
на виконання кваліфікаційної магістерської роботи

Видано студенту групи 605м факультету комп'ютерних наук

Пастушок Ігор Анатолійович

(прізвище, ім'я, по батькові студента)

1. Тема кваліфікаційної роботи

Автоматизація планування подачі міського транспорту за даними IP-відеоспостереження системи «Розумне місто»

Затверджена наказом по ЧНУ ім. Петра Могили від 08.11.2023 № 226.

2. Строк представлення кваліфікаційної роботи «____» _____ 20__ р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні:
Початкові дані: предметно-орієнтований контент, функціональні вимоги до системи. Очікуваним результатом роботи є розпізнання та підрахунок людей на зображені

4. Перелік питань, що підлягають розробці: Визначення функцій системи, моделювання, створення бази даних системи, проєктування програмного застосунку, створення інтерфейсу користувача, кодування та тестування системи.

5. Перелік графічних матеріалів:
Слайди презентації.

6. Завдання до спеціальної частини:
Сформувати перелік вимог до робочого місця, організації та обладнання робочих місць, санітарно-гігієнічних вимог, вимоги щодо освітлення, електробезпеки та пожежної безпеки.

7. Консультанти:

Консультант	Кафедра (організація)	Частина роботи
Д-р біол. наук, професор Григор'єва Л. І.	Кафедра екології Медичного інституту ЧНУ ім. Петра Могили	Спеціальна частина з охорони праці

Керівник роботи д-р техн. наук, професор Журавська Ірина Миколаївна
(посада, прізвище, ім'я, по батькові)

_____
(підпис)

Завдання прийнято до виконання

Пастушок Ігор Анатолійович
(прізвище, ім'я, по батькові студента)

_____
(підпис)

Дата видачі завдання «____» _____ 20_____ р.

КАЛЕНДАРНИЙ ПЛАН

виконання кваліфікаційної магістерської роботи

Тема: Автоматизація планування подачі міського транспорту за даними IP-відеоспостереження системи «Розумне місто»

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Розробка та затвердження завдання на виконання КМР	01.09.2023	15.09.2023	Виконано
2.	Огляд літератури за темою роботи	15.09.2023	01.10.2023	Виконано
3.	Складання календарного плану КМР	01.10.2023	03.10.2023	Виконано
4.	Аналіз предметної області	05.10.2023	25.10.2023	Виконано
5.	Розробка проєктних рішень	25.10.2023	10.11.2023	Виконано
6.	Моделювання	10.11.2023	15.11.2023	Виконано
7.	Конструювання системи	15.11.2023	20.12.2023	Виконано
8.	Перевірка працездатності, тестування та апробація розробленої системи	20.12.2023	05.01.2024	Виконано
9.	Аналіз результатів тестування	05.01.2024	10.01.2024	Виконано
10.	Розробка керівництва користувача	10.01.2024	13.01.2024	Виконано
11.	Відгук керівника КМР	13.01.2024	05.02.2024	Виконано
12.	Оформлення КМР та презентації	07.02.2024	08.02.2024	Виконано
13.	Попередній захист	31.01.2024	11.02.2024	Виконано
14.	Рецензування	12.02.2024	18.02.2024	Виконано
15.	Захист кваліфікаційної роботи	27.02.2024	27.02.2024	Виконано

Розробив студент І. А. Пастушок _____ _____

(прізвище, ім'я, по батькові) *(підпис)*

«__» _____ 20__ р.

Керівник роботи зав. каф. КІ Журавська Ірина Миколаївна _____

(прізвище, ім'я, по батькові) *(підпис)*

«__» _____ 20__ р.

АНОТАЦІЯ

до кваліфікаційної магістерської роботи
«Автоматизація планування подачі міського транспорту за даними IP-відеоспостереження системи «Розумне місто»»
Студент 605м гр.: Пастушок Ігор Анатолійович
Керівник: д-р техн. наук, проф. Журавська Ірина Миколаївна

Актуальність теми кваліфікаційної роботи полягає в тому, що розвиток сучасних технологій та інфраструктури «Розумного міста» створює унікальну можливість для поліпшення системи планування подачі міського транспорту. Інтернет- та високотехнологічні рішення вже стали частиною повсякденного життя громадян, і це дозволяє впроваджувати нові підходи до оптимізації руху транспорту та поліпшення обслуговування пасажирів.

Метою кваліфікаційної роботи є розробка системи, яка дозволить автоматизувати моніторинг та планування подачі міського транспорту на основі даних, отриманих із IP-камер. Використовуючи цю систему, можна буде отримувати статистичну інформацію про кількість пасажирів на різних зупинках, що дозволить покращити якість обслуговування громадян та оптимізувати рух транспорту.

Для досягнення визначеної мети необхідно вирішити такі завдання:

1) дослідити об'єктну та предметну область;
2) розробити базу даних (БД) для зберігання інформації про кількість пасажирів на зупинці;
3) створити скрипт, який буде визначати кількість людей на зображенні;
4) розробити користувацький інтерфейс для роботи з даними;
5) розробити та протестувати застосунок з метою оцінки його працездатності.

Об’єктом кваліфікаційної роботи є процес автоматизації планування подачі міського транспорту.

Предметом кваліфікаційної роботи є система планування подачі міського транспорту на основі зображень з IP-камер на зупинках.

У **першому розділі** дипломної роботи розглянуто об’єкт та предмет досліджень, проведений аналіз схожих технічних рішень, сформовано завдання.

У **другому розділі** кваліфікаційної магістерської роботи проведено моделювання системи. Було проаналізовано архітектуру нейронної мережі «RetinaNet».

У **третьому розділі** розроблено дизайн користувацького інтерфейсу вебзастосунку, наведено основні технології для розробки системи кваліфікаційної роботи, проаналізовано характеристики IP-камер.

У **четвертому розділі** протестована робота системи.

КМР викладена на 84 с., містить 4 розділи, 4 табл., 59 рис., 17 джерел в переліку посилань, 3 додатки та спеціальну частину з охорони праці.

Ключові слова: *розумне місто, IP-відеоспостереження, підрахунок об’єктів на зображенні, нейронна мережа RetinaNet, вебзастосунок*

ANNOTATION

to the qualifying master's thesis
"Automation of urban transport planning based on IP video surveillance data of the Smart City system"
Student 605m gr.: Pastushok Ihor Anatoliiovych
Head: Dr. Sc. (Engin.), Prof. Iryna Mykolaivna Zhuravska

The urgency of the topic of qualifying work lies in the fact that the development of modern technologies and infrastructure of the "Smart City" creates a unique opportunity to improve the system of planning the supply of city transport. The Internet and high-tech solutions have already become part of the everyday life of citizens, and this allows the introduction of new approaches to optimizing traffic and improving passenger service.

The purpose of the qualification work is the development of a system that will allow for the automation of monitoring and planning of city transport services based on data obtained from IP cameras. Using this system, it will be possible to receive statistical information about the number of passengers at various stops, which will improve the quality of service to citizens and optimize traffic.

To achieve the specified goal, the following tasks must be solved:

1) research the object and subject area;

2) develop a database (DB) for storing information about the number of passengers at the stop.

3) create a script that will find the number of people in the image.

4) develop a user interface for working with data.

5) develop and test the application in order to assess its performance.

The object of research is

The subject of research is

In **the first section** of the diploma work, the object and subject of research is considered, the analysis of similar ones is carried out, and the tasks to be performed are formed.

In **the second section** of the qualifying master's thesis, the system was modeled. The architecture of the neural network "RetinaNet" was analyzed and the design of the user interface of the web application was developed, the main technologies for the development of the qualification work system were given, and the characteristics of IP cameras were analyzed.

In **the third section**, the design of the user interface of the web application is developed, the main technologies for the development of the qualification work system are given, and the characteristics of IP cameras are analyzed.

In **the fourth section**, the operation of the system is tested.

The master's thesis is laid out on 84 pages, it contains 4 sections, 59 figures, 4 tables, 17 sources in the list of references, 3 appendices and a special part on occupational safety.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

БД – база даних

ПЗ – програмне забезпечення

CMS – Content Management System

CNN –Convolutional Neural Network

CSS – Cascading Style Sheets

FPN – Feature Pyramid Net

FTP – File Transfer Protocol

HTML – HyperText Markup Language

IoT – Internet of Thing

IP – Internet Protocol

JS – JavaScript

MVVM – Model-View-ViewModel

PCA – Principal Component Analysis

SQL – Structured Query Language

UI – User Interface

ВСТУП

Однією з ключових проблем, що виникають у контексті розвитку сучасних міст і підвищення якості життя громадян, є забезпечення оптимального та ефективного планування подачі міського транспорту.

Поєднання високих технологій та інфраструктури «Розумного міста» відкриває нові можливості для автоматизації цього процесу та покращення якості транспортного обслуговування громадян.

Об’єктом кваліфікаційної роботи є система планування подачі міського транспорту.

Предметом кваліфікаційної роботи є система моніторингу, збору та аналізу інформації на транспортних зупинках.

Актуальність теми кваліфікаційної роботи полягає в тому, що розвиток сучасних технологій та інфраструктури «Розумного міста» створює унікальну можливість для поліпшення системи планування подачі міського транспорту. Інтернет та високотехнологічні рішення вже стали частиною повсякденного життя громадян, і це дозволяє впроваджувати нові підходи до оптимізації руху транспорту та поліпшення обслуговування пасажирів.

Метою кваліфікаційної роботи є автоматизувати моніторинг та планування подачі міського транспорту на основі даних, отриманих із IP-камер. Використовуючи цю систему, можна буде отримувати статистичну інформацію про кількість пасажирів на різних зупинках, що дозволить покращити якість обслуговування громадян та оптимізувати рух транспорту.

Для досягнення визначеної мети необхідно вирішити такі завдання:

6) дослідити об'єктну та предметну область;

7) розробити базу даних (БД) для зберігання інформації про кількість пасажирів на зупинці.

8) створити скрипт, який буде знаходити кількість людей на зображенні.

9) розробити користувацький інтерфейс для роботи з даними.

10) розробити та протестувати застосунок з метою оцінки його працездатності.

Практична цінність роботи полягає в тому, що вона відкриває нові можливості для покращення системи планування подачі міського транспорту. Завдяки впровадженню високих технологій та інфраструктури «Розумного міста», система забезпечить ефективний моніторинг та аналіз інформації на транспортних зупинках. Це дозволить оптимізувати рух транспорту, забезпечуючи точне визначення потреб пасажирів та вчасну реакцію на зміни в попиті.

Більше того, система має важливе значення для підвищення якості обслуговування громадян. Застосування IP-камер для збору даних про кількість пасажирів на різних зупинках дозволить адаптувати розклади та ресурси транспорту, спрямовуючи їх туди, де це необхідно найбільше. Це робить подорожі більш комфортними для громадян та сприяє покращенню їхнього загального досвіду.

Робота пройшла апробацію під час XXVI Всеукраїнської науково-практичної конференції молодих вчених, аспірантів і студентів «Інформаційні технології та інженерія», з 31 січня по 02 лютого 2024 року [1].

1 АНАЛІЗ СИСТЕМ ОБРОБКИ ЗОБРАЖЕНЬ ДЛЯ ПІДРАХУНКУ КІЛЬКОСТІ ПАСАЖИРІВ НА ЗУПИНКАХ

1.1 Порівняльний аналіз систем аналізу об’єктів на зображеннях

Для аналізу сучасного стану предметної області проведено аналіз існуючих застосунків зі схожим застосуванням.

Azure AI Vision – це одна з послуг обчислювального хмарного сервісу Microsoft Azure, спрямована на використання штучного інтелекту (ШІ) для роботи з обробкою зображень та комп'ютерним зором [2]. Основна мета цієї послуги – надання розширених можливостей аналізу та розпізнавання зображень для розробників і підприємств.

Переваги Azure AI Vision:

– Azure AI Vision надає вам можливість використовувати потужні алгоритми для роботи з зображеннями, включаючи їх аналіз, розпізнавання облич, виявлення об'єктів та інші операції;

– сервіс дозволяє розпізнавати та ідентифікувати обличчя на фотографіях, що може бути корисним для відомчих та комерційних застосувань;

– є можливість використання візуального пошуку для знаходження зображень, які мають схожий зміст або визначених об'єктів;

– Azure AI Vision надає API, яке розробники можуть використовувати для інтеграції функціоналу розпізнавання облич і обробки зображень у свої застосунки;

– відзначається різноманітністю можливостей, що дозволяють вам використовувати інтелектуальний аналіз зображень в різних галузях.

Недоліки Azure AI Vision:

– використання Azure AI Vision може бути дорогим, особливо для великих обсягів обробки зображень;

– деякі функції можуть вимагати активного Інтернет-з'єднання для коректної роботи, що може бути недоцільним у деяких випадках;

– для новачків у сфері обробки зображень та інтелектуального аналізу, Azure AI Vision може здатися складним у налаштуванні та використанні;

– як і в будь-якому обчислювальному хмарному сервісі, існують питання стосовно безпеки та обробки особистих даних, які можуть виникнути при використанні Azure AI Vision;

– деякі розробники можуть відчувати відсутність повного контролю над процесами обробки зображень, що може бути проблемою у випадку спеціалізованих вимог.

Інтерфейс сайту Azure AI Vision та приклад роботи можна побачити на рис. 1.1–1.2.

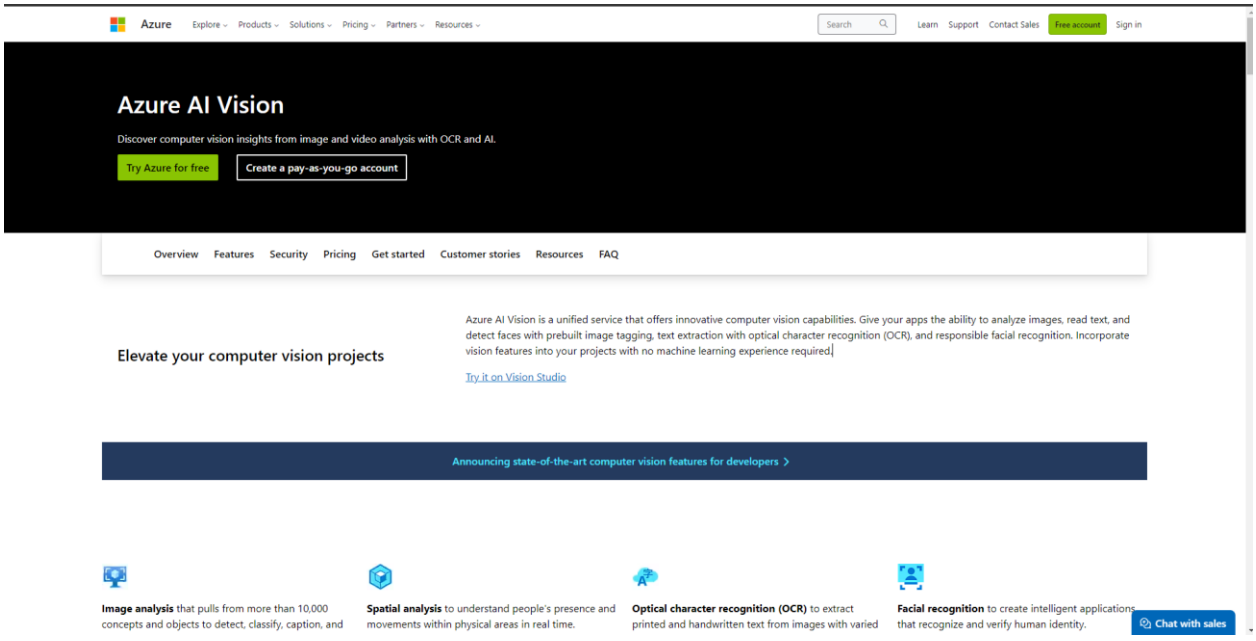


Рисунок 1.1 – Інтерфейс десктопного застосунку Azure AI Vision

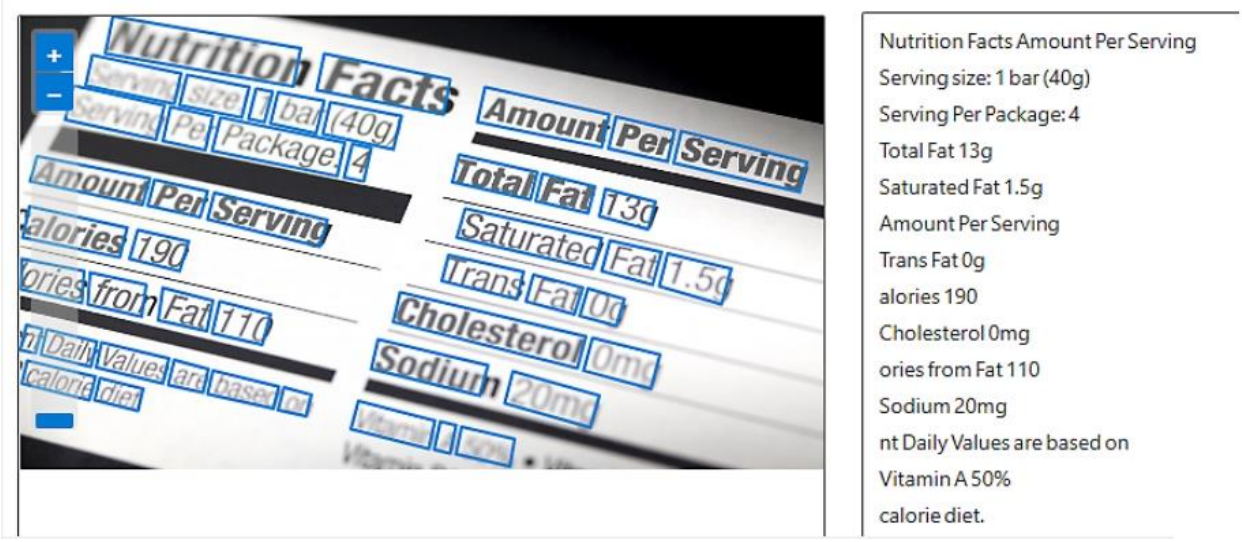


Рисунок 1.2 – Приклад роботи Azure AI Vision

Другим застосунком, який досліджувався, є EasyWay [3].

EasyWay – це сервіс, який надає повну інформацію про маршрути та зупинки громадського транспорту в 73 містах України, включаючи Київ, Харків, Миколаїв, а також в інших країнах, таких як Молдова, Болгарія, Узбекистан, Сербія, Хорватія, Казахстан, Польща, Греція та Туреччина.

Проєкт EasyWay також пропонує мобільний застосунок для платформ Android, IOS і WinPhone, а також надає доступ до API.

До основних переваг можна віднести:

– EasyWay славиться своєю зручністю та легкістю використання. Інтуїтивний інтерфейс дозволяє користувачам швидко знаходити інформацію про громадський транспорт та планувати маршрути без зайвих зусиль;

– однією з ключових переваг EasyWay є надання актуальної інформації про розклади громадського транспорту в режимі реального часу. Це дозволяє користувачам вчасно реагувати на зміни та планувати свої подорожі ефективно;

– платформа пропонує широкий спектр функціоналу, включаючи навігацію, виявлення та оновлення маршрутів, а також інші корисні можливості для полегшення використання громадського транспорту;

– партнерство з Google та Here свідчить про визнання та довіру, яку отримала EasyWay від провідних гравців у сфері картографії та навігації.

Недоліки EasyWay:

– завдання деяких функцій EasyWay може вимагати стабільного Інтернет-з'єднання, що може стати нецільовим у зонах з обмеженим доступом до мережі;

– збір та обробка особистої інформації користувачів може викликати питання щодо конфіденційності та приватності. Важливо, щоб EasyWay забезпечував високий рівень захисту особистих даних;

– інформація про громадський транспорт та функціонал EasyWay може бути неоднаково доступною в різних містах чи регіонах, що обмежує його користь для користувачів з певних локацій;

– для збереження актуальності та ефективності, EasyWay може вимагати регулярних оновлень, інакше інформація може стати застарілою, особливо з введенням нових маршрутів чи змінами в розкладі транспорту.

Інтерфейс застосунку EasyWay наведено на рис. 1.3–1.4.

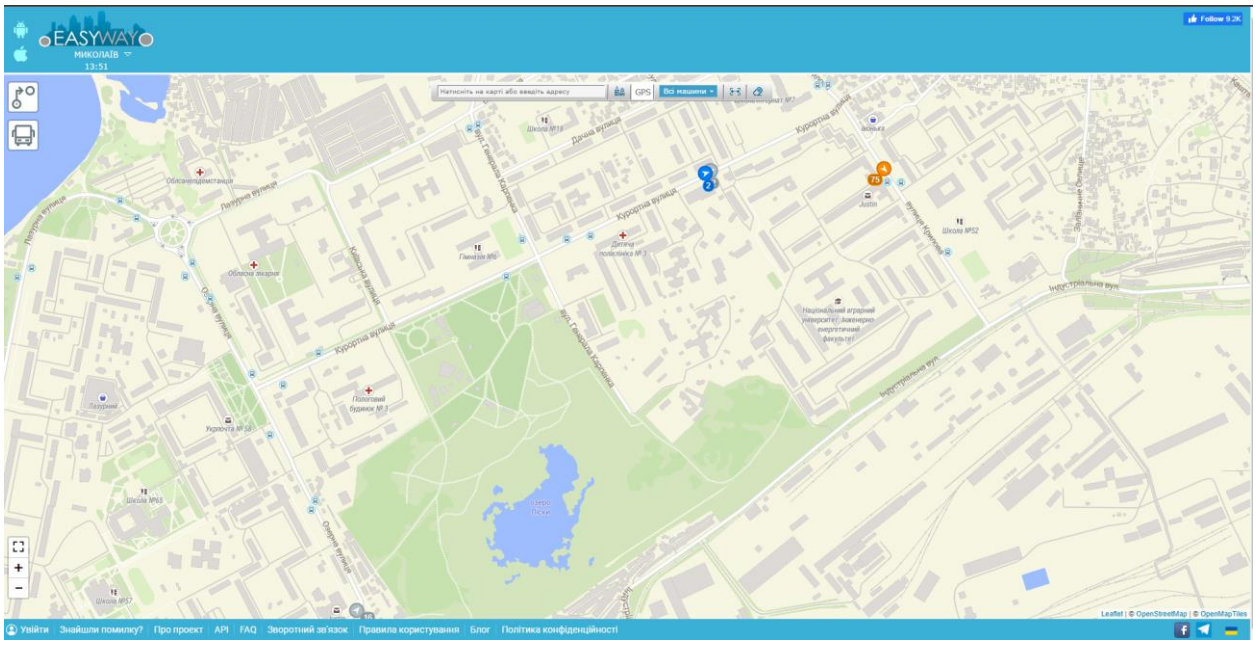


Рисунок 1.3 – Інтерфейс десктопного застосунку EasyWay

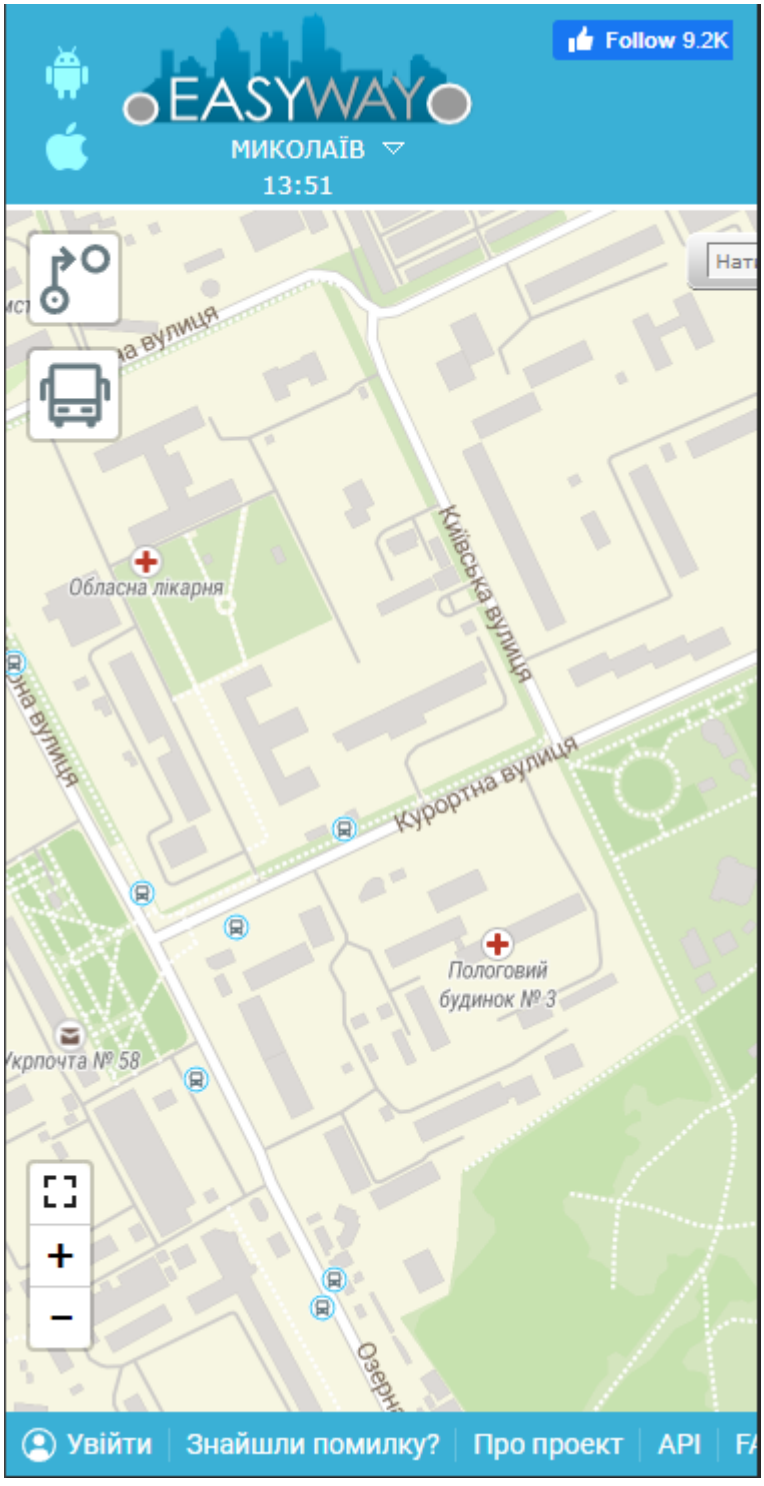


Рисунок 1.4 – Інтерфейс мобільного застосунку EasyWay

Останнім вебсервісом для аналізу став Amazon Rekognition [4].

Amazon Rekognition – це сервіс від Amazon Web Services (AWS), який надає можливості розпізнавання облич, аналізу об'єктів, детекції тексту та інших операцій з обробки зображень.

Інтерфейс сторінки Amazon Rekognition можна побачити на рис. 1.5.

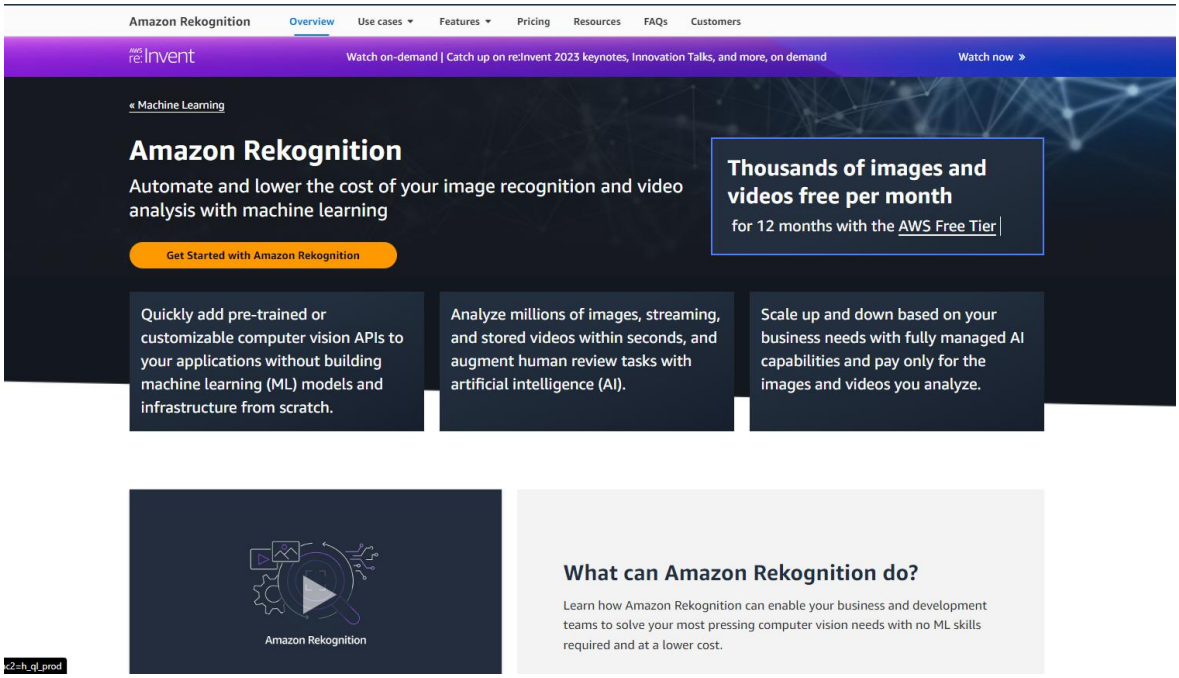


Рисунок 1.5 – Інтерфейс застосунку Amazon Rekognitio

Переваги:

– Amazon Rekognition пропонує широкий спектр функціональності, включаючи розпізнавання облич, виявлення та аналіз об'єктів, детекцію тексту, аналіз емоцій та інші опції, що робить його корисним для різних сценаріїв від розпізнавання осіб до обробки зображень;

– легка інтеграція Amazon Rekognition з іншими послугами Amazon Web Services (AWS) дозволяє розробникам легко впроваджувати функціональність розпізнавання облич в своїх застосунках та сервісах;

– Amazon Rekognition може легко масштабуватися в залежності від потреб користувача, а також має високий рівень надійності завдяки інфраструктурі AWS;

– система може використовувати глибоке навчання для розпізнавання складних структур та атрибутів, що дозволяє виявляти навіть деталі на зображеннях.

Недоліки:

– вартість використання Amazon Rekognition може бути високою, особливо для великих обсягів даних та викликів API. Важливо враховувати витрати для ефективного використання сервісу;

– збір та обробка великої кількості даних, зокрема особистих зображень, підносить питання щодо приватності та безпеки. Користувачам важливо враховувати ці аспекти при використанні сервісу;

– хоча Amazon Rekognition є потужним інструментом, точність його роботи може залежати від ряду факторів, таких як якість зображень та розпізнаваних об'єктів. Іноді можуть виникати помилки у роботі системи;

– у деяких випадках Amazon Rekognition може мати обмежені можливості в точності виявлення та виділення об'єктів на зображеннях порівняно з іншими спеціалізованими системами;

– ефективність сервісу може бути залежати від зовнішніх факторів, таких як освітлення або якість камери, що може вплинути на його роботу в певних умовах.

Приклад роботи Amazon Rekognition наведений на рис. 1.6.

Рисунок 1.6 – Приклад роботи Amazon Rekognition

В табл. 1.1 приведено порівняння систем.

Таблиця 1.1 – Порівняння систем

Показники	**Azure AI Vision**	**EasyWay**	**Amazon Rekognition**
Функціональність	+	+	+
Застосування	+	+	+
Легкість використання	–	+	-
Масштабованість	+	–	+
Вартість	–	+	–
Залежність від інтернет-з'єднання	+	+	+
Сумарний показник	4	5	4

1.2 Основні методи обробки зображень

1.2.1 Алгоритм Віоли-Джонса

Алгоритм Віоли-Джонса, розроблений Полом Віолою і Майклом Джонсом у 2001 році [5], представляє собою систему розпізнавання об'єктів, яка здатна виявляти елементи на зображеннях в реальному часі. Навіть не зважаючи на те, що цей метод є дещо застарілим, він відзначається потужністю і залишається використовуваним в системах розпізнавання об'єктів.

Процес виявлення осіб за допомогою методу Віоли-Джонса включає такі етапи:

1) зображення представляються у формі інтегрального вигляду для швидкого виконання розрахунків;

2) пошук об'єктів на зображеннях здійснюється за допомогою аналізу ознак Хаара;

3) для вибору найбільш підходящих ознак застосовується метод посилення слабких класифікаторів, відомий як метод бустингу;

4) для прийняття рішень використовуються прості бінарні класифікатори з двома значеннями – «Істина» та «Брехня»;

5) для швидкого відкидання вікон, де осіб не знайдено, використовуються каскади ознак.

Метод використовується за допомогою скануючого вікна, яке переміщується з кроком по зображенню. На кожному кроці застосовується набір фільтрів розпізнавання осіб. Якщо один фільтр дає позитивну відповідь, особа виявляється в поточній вікні. Кожен фільтр має набір каскадних класифікаторів, які дивляться на прямокутну підмножину скануючого вікна та визначають, чи виглядає вона як обличчя. Якщо всі класифікатори дають

позитивну відповідь, фільтр також видає позитивний результат, і особа розпізнається. В іншому випадку застосовується наступний фільтр з набору.

Кожен класифікатор складається з екстракторів ознак Хаара, які є слабкими класифікаторами. Кожна ознака Хаара представляє собою зважену суму двовимірних інтегралів невеликих прямокутних областей, прикріплених один до одного. Ваги можуть мати значення ± 1. Класифікатор визначає, до якого класу належить об'єкт, і при позитивному результаті подальший аналіз продовжується. На рис. 1.7 показані приклади ознак Хаара.

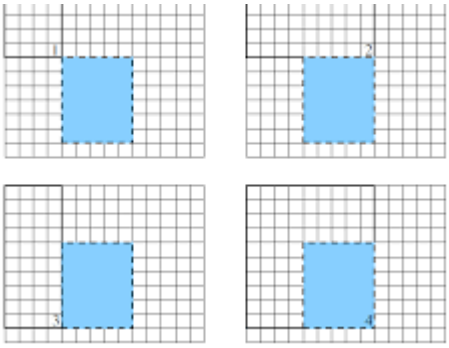


Рисунок 1.7 – Приклад ознак Хаара

Так, справжній успіх алгоритму Віоли-Джонса полягає не тільки у розпізнаванні, а й у передній частині навчання алгоритму, яке передує його використанню в реальному часі. Процес тренування включає в себе навчання алгоритму на основі певної БД, яка складається з різних ознак тестових зображень.

Алгоритм тренується на великій кількості даних з зображеннями облич, де кожне зображення має відзначені ознаки. Під час тренування алгоритм стискає зображення до розміру 24 × 24 пікселів і застосовує створену БД для пошуку вказаних ознак на зображенні.

Важливо мати велику БД з різними формами облич, щоб машина могла адаптуватися до різних варіантів та форм обличчя. У роботі Пола Віоло та Майкла Джонса використовувалася БД, яка включала 4960 зображень, кожне

з яких було позначено вручну. Це дозволило їм створити навчальний набір, на основі якого алгоритм міг ефективно розпізнавати обличчя в реальному часі.

Отже даний метод має такі переваги:

– можливість виявлення більше ніж одного обличчя на зображенні;

– висока швидкість розпізнавання за допомогою класифікаторів;

– можливість використання методу у відеопотоці;

– початково алгоритм був розроблений для розпізнання тільки облич, але його можна використовувати і для розпізнання інших об'єктів.

До недоліків можна віднести:

– низьку швидкість алгоритму навчання;

– необхідність великої кількості даних для навчання.

1.2.2 Модель TensorFlow

TensorFlow пропонує різноманітні архітектури моделей для виявлення об'єктів. Розглянемо деякі з них:

1) Faster R-CNN (Region-based Convolutional Neural Network) – Ця архітектура пропонує двоетапний процес, який поділяє завдання на передбачення областей з об'єктами та їх класифікацію. Faster R-CNN відома своєю точністю, але може вимагати більше обчислювальних ресурсів;

2) SSD (Single Shot Multibox Detector) – SSD надає баланс між точністю та швидкістю, пропонуючи однокроковий процес виявлення об'єктів. Ця архітектура підходить для застосунків, де потрібна висока швидкість обробки;

3) YOLO (You Only Look Once) – YOLO пропонує однокроковий підхід та забезпечує здатність виявляти об'єкти в реальному часі. YOLO може бути кращим вибором за необхідності високої швидкості роботи;

4) RetinaNet – вводить інноваційну архітектуру з використанням фокусованих втрат та Feature Pyramid Network (FPN). RetinaNet демонструє

здатність ефективного виявлення об'єктів, включаючи людину, за наявності різних масштабів об'єктів на зображенні.

Якість навчальних даних грає ключову роль успішному навчанні моделей виявлення людини на фотографіях. Важливо мати широкий та різноманітний набір даних, який охоплює різні умови освітлення, погодні умови, фони та пози людини. Це допомагає моделі узагальнювати, а не перенавчати на конкретних сценах.

Якісна розмітка: Чітка та точна розмітка даних є фундаментом для успішного навчання. Особливу увагу слід приділяти коректному маркуванню областей, що містять людину, щоб модель могла точно виявляти об'єкти інтересу.

Управління дисбалансом: Забезпечення рівномірного представництва класів, включаючи людину та фон, допомагає уникнути зміщення у навчальних даних та підвищує стійкість моделі.

Додатковий спосіб покращення якості навчальних даних – використання аугментації:

– варіювання масштабу, повороти та відображення зображень допомагають створити різні варіанти того самого об'єкта, що сприяє навчанню моделі на різноманітних даних;

– внесення невеликих штучних шумів у зображення може зробити модель стійкішою до реальних умов, де зображення можуть бути неідеальними;

– зміна характеристик кольору зображень допомагає моделі бути стійкою до змін освітлення.

TensorFlow надає передбачені моделі, які можуть бути використані як відправна точка для виявлення людини. Це прискорює процес навчання та може бути особливо корисним при обмежених ресурсах.

1.2.3 Виявлення на зображеннях за допомогою Model Builder

Виявлення об'єктів стає особливо важливим, коли на зображеннях є різні об'єкти різних типів. Model Builder – це інструмент, який значно спрощує процес розробки та навчання моделей виявлення об'єктів [6]. Дозволяючи легко налаштовувати та навчати власні моделі, Model Builder робить цей складний процес доступним для новачків у галузі машинного навчання.

Деякі варіанти застосування виявлення об'єктів:

Автомобілі з автономним керуванням: Виявлення об'єктів є важливим для систем автономного керування, де необхідно точно розпізнавати різні елементи навколишнього середовища, такі як інші транспортні засоби, пішоходи та дорожні знаки.

Робототехніка: В області робототехніки виявлення об'єктів дозволяє роботам взаємодіяти з навколишнім середовищем, визначати розташування предметів та передбачати їхній рух.

Виявлення осіб: Системи забезпечення безпеки та відеоспостереження активно використовують виявлення осіб для ідентифікації та відстеження осіб.

Техніка безпеки: Виявлення об'єктів застосовується для забезпечення безпеки у різних контекстах, включаючи аеропорти, транспортні вузли та громадські місця.

Підрахунок об'єктів: У роздрібній торгівлі чи виробництві виявлення об'єктів може використовуватися автоматичного підрахунку товарів чи компонентів.

Розпізнавання активності: У охороні здоров'я та фітнес-індустрії виявлення об'єктів може бути застосоване для розпізнавання та аналізу рухової активності людей.

Приклад роботи Model Builder наведений на рис. 1.8.

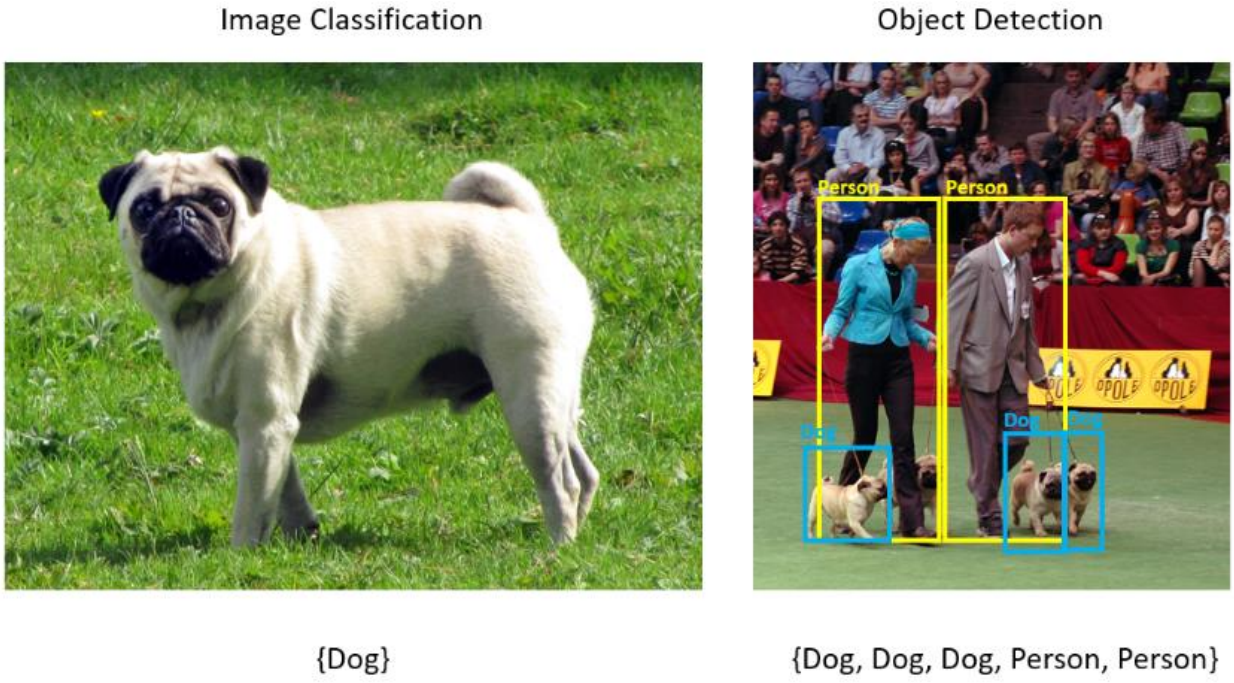


Рисунок 1.8 – Приклад знаходження об’єктів

З використанням Model Builder можна створювати та навчати моделі виявлення об'єктів, що сприяє розвитку навичок у галузі машинного навчання та комп'ютерного зору. Цей інструмент відкриває нові можливості для дослідження та застосування виявлення об'єктів у різних галузях, що робить його незамінним інструментом у сучасній науці та технологіях.

1.3 Специфікація вимог до ПЗ

На основі аналізу предметної області та дослідження існуючих систем сформовані вимоги до власної системи, що розроблюється, відповідно до мети та сформовано специфікацію вимог, що представляє собою опис поведінки застосунку, що розроблюється.

ПРИЗНАЧЕННЯ ПРОЄКТУ

Застосунок буде являтися системою для підрахунком кількості людей на зображені, для майбутнього аналізу та відправки, до мобільного застосунку водія громадського транспорту.

ФУНКЦІЇ СИСТЕМИ

Функція системи 1:

Створення скриншотів з відео.

Функціональні вимоги

Розробити скрипт, що буде робити скриншоти з відео, що транслює камера.

Функція системи 2:

Знаходження на зображенні людей.

Функціональні вимоги

На зображенні, що було отворено, порахувати точну кількість людей.

Функція системи 3:

Збереження інформації за датою та часом.

Функціональні вимоги

Після аналізу зображення, дані зберігаються та відправляються до БД.

Функція системи 4:

Вивід інформації.

Функціональні вимоги

Після збереження даних у БД, має бути можливість переглянути дані у вебзастосунку.

ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Мова і технологія розробки

Для розробки буде використано Python з бібліотеками OpenCV, tensorflow, Для розробки вебчастини проєкту буде застосовано Vue js та UI бібліотека Vuetify.

Вимоги:

– доступ до інтернету;

– комп’ютер чи ноутбук ;

ВЛАСТИВОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Практичність:

– застосунок має бути простим у використанні;
– дизайн застосунку має бути максимально зручним для використання.

1.4 Розумне місто

Урбанізація – найважливіший та найбільш впливовий процес сучасного світу. Зростаюча міграція населення з сільських районів в міста призводить до трансформації скромних поселень у мегаполіси. Цей процес має як позитивні, так і негативні наслідки.

З одного боку, урбанізація сприяє комерційному розвитку, співпраці та економічному зростанню. Міста стають центрами інновацій та можливостей для своїх мешканців. Проте з іншого боку, цей процес супроводжується заторами, зростанням рівня злочинності та проблемами утилізації відходів.

З розвитком урбанізації стає все важливішим питання безпеки та якості життя у містах. На жаль, безпека не обмежується лише фізичними аспектами, вона також стосується психологічного та соціально-економічного благополуччя громадян. Таким чином, виникає потреба в розумних стратегіях міського розвитку, які б враховували не лише фізичну безпеку, а й ефективність, інклюзивність та стійкість міста.

Розумне місто – це не просто концепція, а відповідь на виклики сучасного урбанізованого світу. Шляхом інтеграції технологій Інтернету речей та аналітики даних, воно спрямоване на оптимізацію всіх сфер міського життя - від транспорту та енергетики до громадської безпеки та управління відходами.

Високотехнологічні рішення, використання даних у реальному часі та інтелектуальні системи дозволяють розумним містам ефективно вирішувати проблеми, з якими вони стикаються. За допомогою інноваційних підходів до громадської безпеки, руху та екології, вони не лише полегшують життя мешканців, а й зроблять міста більш стійкими та ресурсоємними.

Таким чином, розумне місто є не лише відповіддю на виклики урбанізації, а й ключовим елементом сучасного міського розвитку. Воно покликане створити сприятливі умови для життя та розвитку людей у містах, забезпечуючи їхню безпеку, комфорт та стабільність.

Пілотні проєкти міст представлені у табл. 1.2.

Таблиця 1.2 – Пілотні проєкти в Європі

Місто або країна	**Опис**
Барселона	Модель Barcelona City City охоплює чотири ключові області: розумне управління, розумна економіка, розумний спосіб життя та розумні люди. Барселона відіграє центральну роль у сфері інновацій та економічного розвитку, оскільки малі та середні підприємства використовують цей регіон для тестування нових технологій. Наразі немає інформації про стандарти, прийняті в Барселоні.
Румунія	Румунська асоціація розумного міста є відомим лідером у галузі розвитку індустрії розумних міст в Румунії. Вона об'єднує професіоналів та експертів з різних галузей, а також має підтримку понад 200 національних та міжнародних партнерів. Головна мета асоціації - створення та підтримка креативно-інтелектуальних спільнот у Румунії за допомогою розвитку екосистеми «Розумного міста». Крім цього, асоціація активно сприяє впровадженню в законодавство 8 міжнародних стандартів ISO.
Великобританія	Британський інститут стандартизації (BSI), у співпраці з ISO, активно працює над стандартами для розумних міст та міських показників. BSI є провідною організацією у розробці таких стандартів в Великобританії. Специфікація PAS 180 надає детальний огляд узгоджених визначень та термінів, що стосуються розумних міст у Великобританії.

Місто або країна	Опис
	Це сприяє розробці основи для майбутньої стандартизації та передової практики. Крім того, PAS 180 сприяє покращенню розуміння концепції розумних міст, створюючи загальну мову для дизайнерів, розробників, клієнтів та виробників.

Згідно з дослідженнями компанії IHS Markit, яка є провідним постачальником аналітичних досліджень та експертних оцінок сучасного бізнесу у світі, кількість камер видеоспостереження у світі у 2022 р. досягла 1 млрд шт. Тільки в США ця кількість становить біля 85 млн та зростатиме кожного року майже на 10 млн IP-камер [14].

Такі вуличні камери (рис. 1.9, а) використовуються у системі «Розумне місто» («Smart City) та можуть допомогти визначити навантажені (рис. 1.9, б) та напівпусті зупинки (рис. 1.9, в). На підставі даних з IP-камер відеоспостереження можливо вагомо знизити рівень завантаженості зупинок пасажирами, які очікують міський транспорт, особливо у найпопулярніших частинах міст.

а)

б)

в)

Рисунок 1.9 – Зупинка міського транспорту «розумного міста»:
а – розташування IP-камери; б – навантажена пасажирами зупинка;
в – напівпуста зупинка

Вже протягом десятиліть камери відеоспостереження використовуються правоохоронними та комунальними службами «розумних міст» для спостереження за районами, що представляють інтерес у місті. Але диспетчерам доводиться постійно стежити за відеостіною, на якій відображаються відеопотоки з декількох камер (рис. 1.10).

Рисунок 1.10 – Перегляд відеопотоку з IP-камер «розумного міста»

Тягар виявлення гострих місць лягає на діспетчера, тому відеоспостереження могло бути настільки ефективним, наскільки це дозволяє концентрація уваги людини.

Ще одна проблема полягає в тому, що обробка великих обсягів записаного відео займає багато часу. Це обмежує ефективність та швидкість прийняття рішень для забезпечення мешканців міста якісними комунальними послугами, у т. ч. міським транспортом.

Штучний інтелект та нейромережі є рушійною технологією, що стоїть за змінами у розвитку «розумних міст». Інтелектуальна відео аналітика здатна навчитися класифікувати нормальний і перенавантажений стан ділянок міста, може позначати відповідні кадри для перегляду відповідальними співробітниками міських служб. Це скорочує години роботи до декількох хвилин, дозволяє персоналу діяти швидше. Відео аналітика фокусує увагу людей на тому, що найбільш важливо, швидко доводячи важливі події та ідеї до відома співробітника, що економить час в ситуаціях підвищеної напруги на певних ділянках міста.

Наприклад, система «Смарт Сіті Київ» базується на IP-камерах фірми Hikvision. З 2020 року відбувається впровадження нових систем керування, які підвищать інтелектуальний рівень розумного міста, будуть використовуватись для прийняття рішень у різних ситуаціях і галузях. З урахуванням світового досвіду, в останні роки почалося впровадження інструментів «розумного міста» в таких містах України, як Івано-Франківськ, Львів, Запоріжжя, Дрогобич, Полтава, Миколаїв.

У Миколаєві у 2023 р. встановлено 150 камер за 7,5 млн грн у майже 90 точках Центрального та Заводського районів міста. 2017 році. До цього часу з 2017 року у місті вже було встановлено 48 камер за 1 млн 100 тис. гривень [15]. Ще 10 камер відеоспостереження для системи «Безпечне місто» Миколаєву передали з румунського міста Брашова.

Зараз також будується волоконно-оптична мережа для під'єднання камер. За підрахунками, одна точка з камерою обходиться місту приблизно в 50 тис. гривень. Це розумні системи зі штучним інтелектом, які мають змогу

розпізнавати обличчя, номерні знаки, марку, модель, колір транспортного засобу в потоці, відслідковувати інтенсивність автомобільного трафіку, виявляти певні події та об'єкти, предмети, людей, які стоять, сидять або навіть лежать.

Стандарти Smart City можна розділити на три основні рівні: стратегічний, процесний і технічний. Стандарти стратегічного рівня надають керівництву міста поради щодо розробки стратегії Smart City, визначення пріоритетів, розробки дорожньої карти і контролю за її виконанням. Стандарти процесного рівня пов'язані з закупівлями та управлінням проектами, об'єднуючи різні сфери відповідальності міських організацій. Вони пропонують найкращі практики і вказівки. Стандарти технічного рівня описують технічні специфікації для впровадження продуктів і послуг Smart City. Міжнародні організації, такі як ISO, ITU і IEC, розробляють найбільш відомі стандарти у цій області.

У Європі стандарти Smart City розробляються та узгоджуються трьома офіційно визнаними європейськими організаціями: Європейським комітетом стандартизації (CEN), Європейським комітетом електротехнічної стандартизації (CENELEC) і Європейським інститутом телекомунікаційних стандартів (ETSI). Для спільних стандартів використовується абревіатура CEN/CENELEC/ETSI. Крім того, в провідних країнах світу розроблено власні системи стандартів Smart City, які в основному базуються на стандартах міжнародних організацій. Найбільш розвиненою є серія рекомендацій Міжнародної організації зі стандартизації ISO. На весну 2023 року ISO розробила 13 спеціальних документів, що стосуються впровадження технології Smart City (табл. 1.3). Десятки інших стандартів знаходяться у процесі розробки. Крім того, виділено окрему групу стандартів Smart City, які стосуються транспорту та сталої мобільності.

Таблиця 1.3 – Стандарти Smart City, розроблені Міжнародною організацією зі стандартизації (ISO) [16]

№ з/п	Нумерація стандарту	Назва стандарту
1	ISO/TR 6030:2022	Розумні інфраструктури громад – Зменшення ризику стихійних лих – Результати опитування та аналіз прогалин
2	ISO/TR 37150: 2014	Розумна інфраструктура спільноти − огляд існуючих дій, пов’язаних із показниками
3	ISO/TS 37151:2015	Розумна інфраструктура спільноти−принципи та вимоги до показників ефективності
4	ISO/TR 37152: 2016	Розумна інфраструктура спільноти − спільна основа для розвитку та функціонування
5	ISO 37153:2017	Розумна інфраструктура спільноти − модель зрілості для оцінки та вдосконалення
6	SO 37155-1:2020	Структура для інтеграції та функціонування розумних інфраструктур спільнот. Частина 1: Рекомендації щодо розгляду можливостей і проблем, пов’язаних із взаємодією в розумних інфраструктурах спільнот з відповідних аспектів впродовж життєвого циклу
7	ISO 37155-2:2021	Структура для інтеграції та функціонування розумних інфраструктур спільнот. Частина 2: Цілісний підхід і стратегія розвитку, експлуатації та підтримки розумних інфраструктур громад.
8	ISO 37156:2020	Розумні інфраструктури спільнот − Рекомендації щодо обміну даними та спільного використання для розумних інфраструктур спільнот
9	ISO 37160:2020	Розумна громадська інфраструктура. Електроенергетична інфраструктура. Методи вимірювання якості теплоенергетичної інфраструктури та вимоги до роботи та управління станцією

№ з/п	Нумерація стандарту	Назва стандарту
10	ISO 37166:2022	Розумна інфраструктура спільноти − структура інтеграції міських даних для розумного міського планування (SCP)
11	ISO 37170:2022	Розумна інфраструктура спільноти − структура даних для управління інфраструктурою на основі цифрових технологій у розумних містах
12	ISO/TR 37171:2020	Звіт про пілотне тестування щодо застосування стандартів інтелектуальної інфраструктури спільноти ISO
13	ISO/TS 37172:2022	Розумна інфраструктура спільноти−обмін даними та спільне використання для інфраструктури спільноти на основі географічної інформації

На початку 2023 року Міжнародна Спілка електрозв'язку (ITU) розробила одинадцять спеціальних рекомендаційних документів, що стосуються телекомунікаційних та комп'ютерних протоколів, пов'язаних з впровадженням технології Smart City (табл. 1.4). Ці документи готує окремий відділ, який має абревіатуру ITU-T (ITU Telecommunication Standardization Sector – Сектор стандартизації телекомунікації в рамках ITU).

Таблиця 1.4 – Стандарти Smart City, розроблені Міжнародною Спілкою електрозв’язку (ITU) [17]

№ з/п	Нумерація стандарту	Назва стандарту
1	ITU-T Y.4900/L.1600	Огляд ключових показників ефективності розумних сталих міст
2	ITU-T Y.4901/L.1601	Ключові показники ефективності, пов'язані з використанням інформаційно-комунікаційних технологій у розумних сталих містах
3	ITU-T Y.4902/L.1602	Ключові показники ефективності, пов'язані зі стійким впливом інформаційно-комунікаційних технологій у розумних містах

№ з/п	Нумерація стандарту	Назва стандарту
4	ITU-T Y.4903/L.1603	Ключові показники діяльності розумних сталих міст для оцінки досягнення цілей у сфері сталого розвитку
5	ITU-T Y.4904	Модель зрілості розумних сталих міст
6	ITU-T Y.4905	Оцінка впливу розумного сталого міста
7	ITU-T Y.4906	Рамки оцінки цифрової трансформації секторів у розумних містах
8	ITU-T Y.4907	Еталонна архітектура уніфікованого управління даними KPI на основі блокчейну для розумних стійких міст
9	ITU-T Y.4908	Структури оцінки ефективності електронних систем охорони здоров'я в Інтернеті речей
10	ITU-T Y.4909	Структура оцінки якості відстеження IoT
11	ITU-T Y.4910	Модель зрілості цифрового ланцюга поставок для розумних сталих міст

Висновок до розділу 1

У результаті проведеного аналізу визначено, що створення системи для розпізнавання кількості людей на зображені є актуальним, тому що дане завдання допоможе з аналізом, створенням або переробкою існуючого розкладу громадського транспортного міста.

Проведено аналіз схожих систем та застосунків та визначено необхідність створення даного застосунку. На основі проведеного дослідження предметної області сформовано вимоги до програмного забезпечення, що розроблюється.

2 МОДЕЛЮВАННЯ ТА ПРОЄКТУВАННЯ СИСТЕМИ ВИЗНАЧЕННЯ КІЛЬКОСТІ ВИЯВЛЕНИХ ЛЮДЕЙ

2.1 Архітектура нейронної мережі RetinaNet

Архітектура згорткової нейронної мережі (англ. Convolutional Neural Network, CNN) RetinaNet складається з 4 основних частин, кожна з яких має своє призначення:

Backbone – основна (базова) мережа, що служить для вилучення ознак з зображення, що надходить на вхід. Ця частина мережі є варіативною і до її основи можуть входити класифікаційні нейромережі, такі як ResNet, VGG, EfficientNet та інші;

Feature Pyramid Net (FPN) – згорткова нейронна мережа, побудована у вигляді піраміди, що служить для поєднання переваг карт ознак нижніх і верхніх рівнів мережі, перші мають високу роздільну здатність, але низьку семантичну, узагальнюючу здатність; другі - навпаки;

Classification Subnet – підмережа, що витягує з FPN інформацію про класи об'єктів, вирішуючи завдання класифікації;

Regression Subnet – підмережа, що витягує з FPN інформацію про координати об'єктів на зображенні, вирішуючи завдання регресії.

На рис. 2.1 зображена архітектура RetinaNet з ResNet нейромережею як backbone.

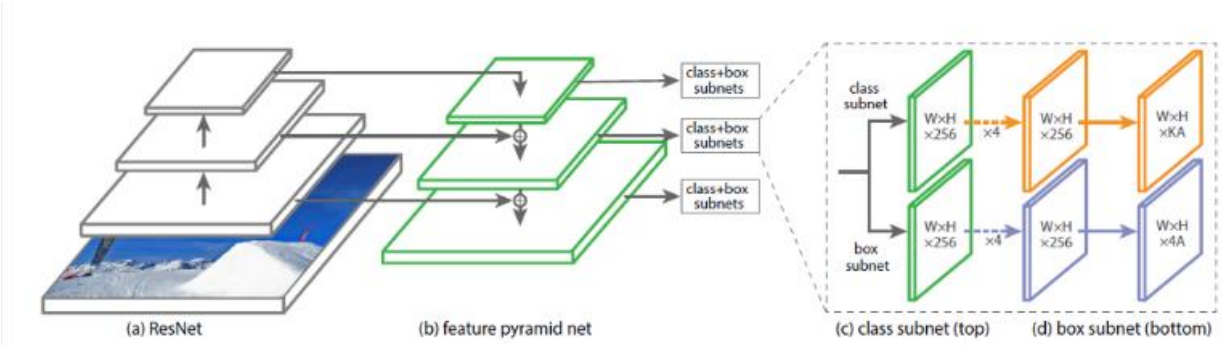


Рисунок 2.1 – Архітектура RetinaNet із backbone-мережею ResNet

Нещодавні дослідження з оптимізації CNN дозволили розробити класифікаційні моделі, які випередили всі раніше розроблені архітектури з кращими показниками точності на датасеті ImageNet при покращенні ефективності в 10 разів. Ці мережі отримали назву EfficientNet-B(0–7). Показники сімейства нових мереж представлені на рис. 2.2.

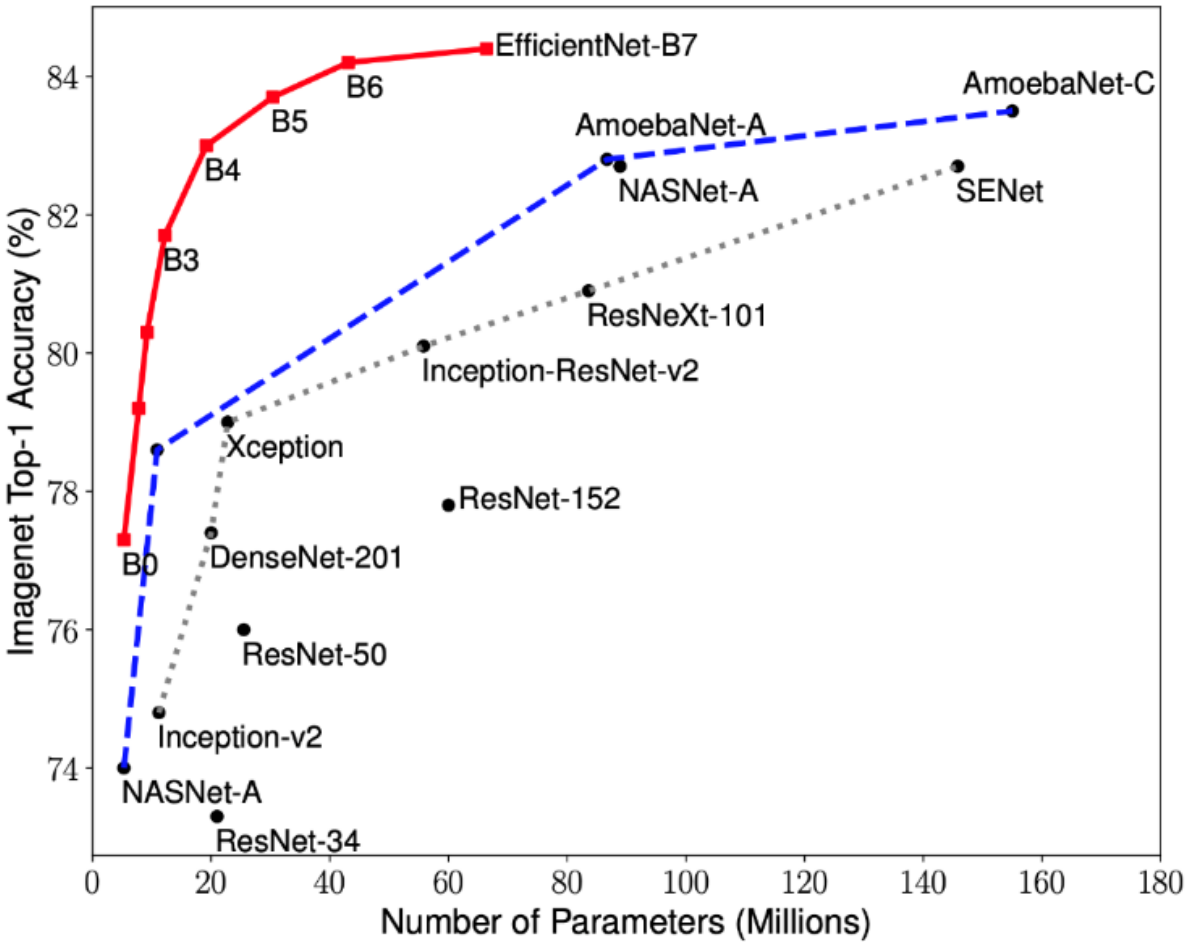


Рисунок 2.2 – Графік залежності найбільшого показника точності від кількості ваги мережі для різних архітектур

Feature Pyramid Network складається з трьох основних частин: висхідний шлях (Bottom-Up Pathway), низхідний шлях (Top-Down Pathway) та бічні з'єднання (Lateral Connections).

Висхідний шлях є якоюсь ієрархічною «пірамідою» – послідовністю згорткових шарів з меншою розмірністю, у нашому випадку – backbone

мережа. Верхні шари згорткової мережі мають більше семантичне значення, але меншу роздільну здатність, а нижні навпаки (рис. 2.3). Bottom-Up Pathway має вразливість при добуванні ознак – втрата важливої інформації про об'єкт, наприклад через зашумлення невеликого, але значущого, об'єкта тлом, оскільки до кінця мережі інформація сильно стиснута та узагальнена.

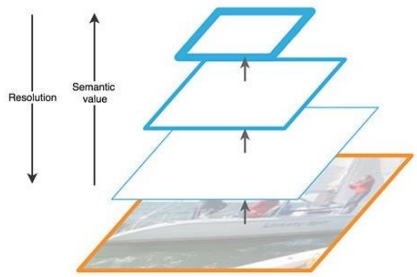


Рисунок 2.3 – Особливості карт ознак на різних рівнях нейромережі

Східний шлях також є «пірамідою». Карти ознак верхнього шару цієї піраміди мають розмір карт ознак верхнього шару Bottom-Up піраміди і збільшуються вдвічі методом найближчого сусіда у напрямку донизу.

Таким чином, у Top-Down мережі кожна карта ознак вищележачого шару збільшується до розмірів карти нижчележачого. Крім цього, FPN присутні бічні з'єднання, це означає, що карти ознак відповідних шарів Bottom-Up і Top-Down пірамід поелементно складаються, причому карти з bottom-up проходять згортку 1 × 1. Цей процес схематично представлено на рис. 2.4.

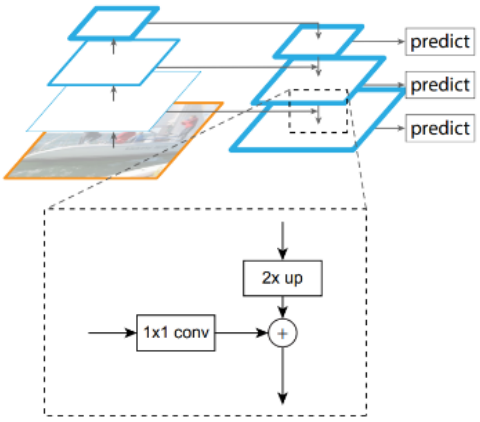


Рисунок 2.4 – Влаштування піраміди ознак

Бічні з'єднання вирішують проблему загасання важливих сигналів у процесі проходу шарами, поєднуючи семантично важливу інформацію, отриману до кінця першої піраміди і більш детальну інформацію, отриману в ній раніше.

Далі, кожен з отриманих шарів у Top-Down піраміді обробляється двома підмережами.

Третьою частиною архітектури RetinaNet є дві підмережі: класифікаційна та регресійна (рис. 2.5). Кожна з цих підмереж утворює на виході відповідь про клас об'єкта та його розташування на зображенні. Розглянемо принцип роботи кожної їх.

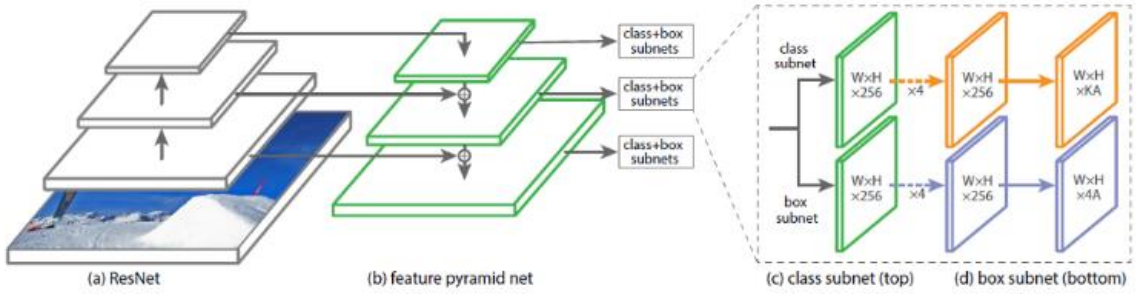


Рисунок 2.5 – Підмережі RetinaNet

Різниця в принципах роботи блоків (підмереж), що розглядаються, не відрізняється до останнього шару. Кожен із них складається з 4 шарів згорткових мереж. У шарі формуються 256 карт ознак. На п'ятому шарі кількість карт ознак змінюється: регресійна підмережа має 4 × A карт ознак, класифікаційна – K × A карт ознак, де A – кількість якірних рамок (докладний опис якірних рамок наведено нижче), K – кількість класів об'єктів.

В останньому, шостому шарі кожна карта ознак перетворюється на набір векторів. Регресійна модель на виході має для кожної якірної рамки вектор із 4 значень, що вказують на зсув цільової рамки (англ. Ground-Truth Box) щодо якірної. Класифікаційна модель має на виході для кожної якірної рамки one-hot

вектор довжиною K, в якому індекс значення 1 відповідає номеру класу, який нейромережа привласнила об'єкту.

Якірна рамка (англ. Anchor Box) – гіперпараметр нейромереж-детекторів, заздалегідь визначений прямокутник, що обмежує, щодо якого працює мережа.

Допустимо, мережа має на виході карту ознак розміром 3 × 3. У RetinaNet кожна з осередків має 9 якірних рамок, кожна з яких має різний розмір та співвідношення сторін (рис. 2.6). Під час навчання кожній цільовій рамці підбираються у відповідність якірні рамки. Якщо їх показник IoU має значення від 0,5 – то якірна рамка призначається цільовою, якщо значення менше 0,4 – вона вважається тлом, в інших випадках якірна рамка буде проігнорована для навчання. Класифікаційна мережа навчається щодо виконаного призначення (клас об'єкта чи фон), регресійна мережа навчається щодо координат якірної рамки (важливо зазначити, що помилка обчислюється щодо якірної, але з цільової рамки).

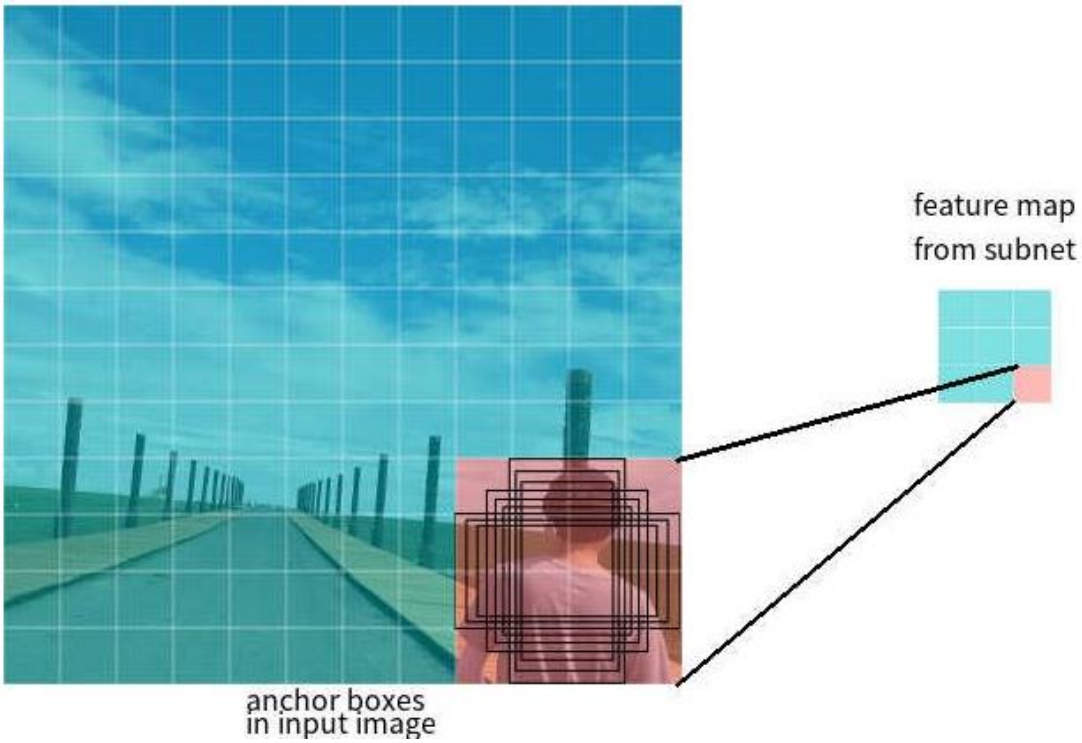


Рисунок 2.6 – Якірні рамки для одного осередку карти ознак розміром 3 × 3

2.2 Структура мережі

На рис. 2.7 можна побачити намальовану структуру мережі проєкту.

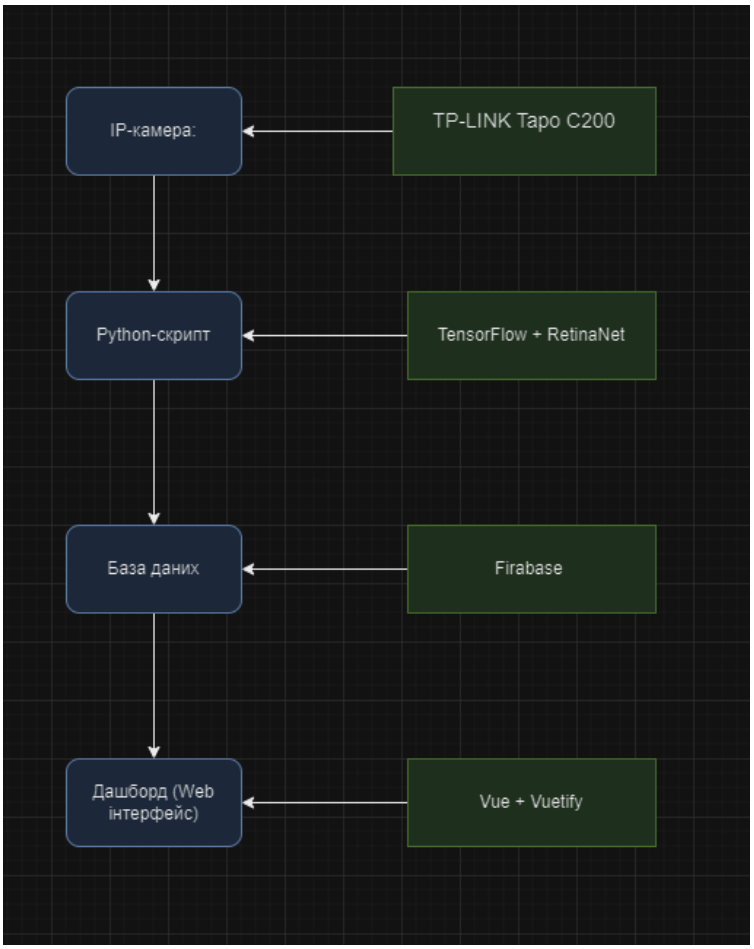


Рисунок 2.7 – Структура мережі

Пояснення структури:

IP-камера: Це пристрій, який надає відеопотік.

Python-скрипт: Скрипт написаний на Python, який підключається до IP-камери, використовує бібліотеку TensorFlow з моделлю RetinaNet для аналізу кожного кадру кожні 15 секунд і зберігає результати в БД.

База даних: Зберігає інформацію про кількість виявлених людей на кожному скріншоті, зробленому скриптом.

Дашборд (вебінтерфейс): Вебінтерфейс, який надає користувачеві доступ до даних БД, відображаючи загальну інформацію.

2.3 Моделювання прототипу користувацького інтерфейсу

Моделювання зовнішнього вигляду є дуже важливою частиною в проєктуванні застосунку. Зручний і інтуїтивно зрозумілий користувацький дуже спрощує роботу із застосунком. Розташування елементів повинне бути зручним і інтуїтивно зрозумілим будь-якому користувачу.

Для розробки прототипів користувацького інтерфейсу обрано інструмент Figma. Figma простий та потужний інструмент, який має необхідний функціонал для розробки прототипу дизайну мобільного застосунку. Інтерфейс інструменту можна побачити на рис. 2.8.

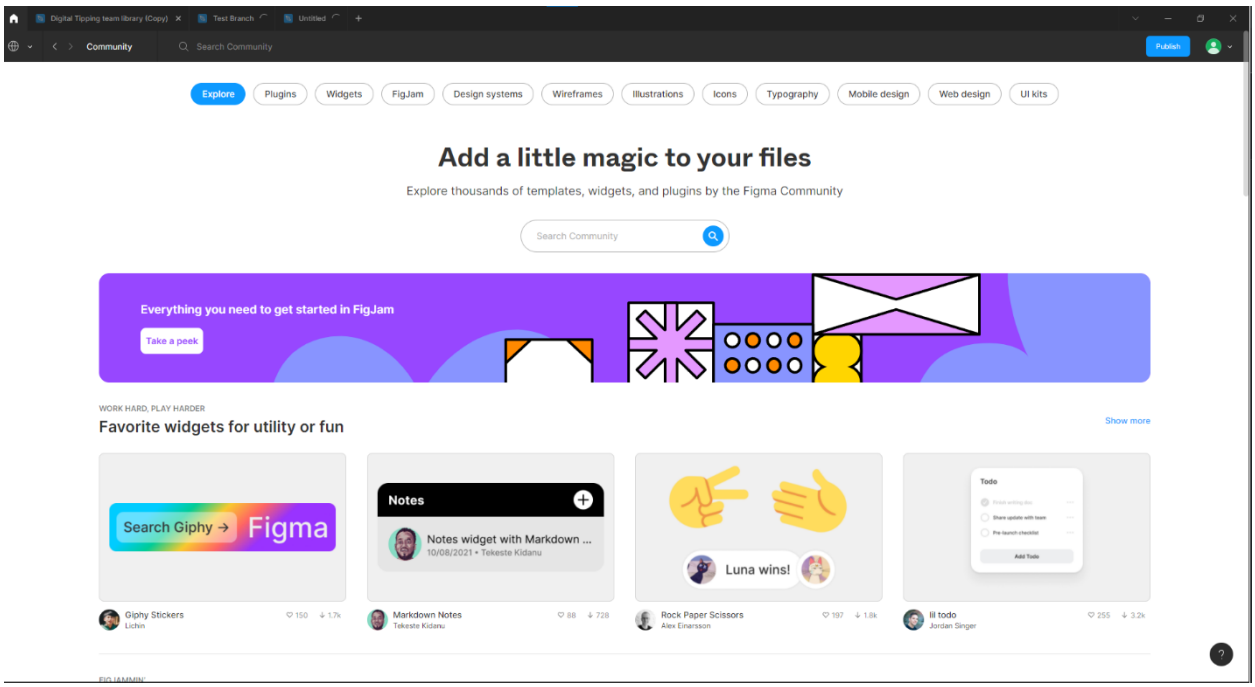


Рисунок 2.8 – Інтерфейс Figma

В першу чергу створено mockup інтерфейсу основної сторінки застосунку (рис. 2.9). На головній сторінці у центрі знаходиться таблиця, де буде виводитися список наявних камер у місті. Таблиця буде приймати поля id, model, address та status. Справа від таблиці буде зображено два графіка. Перший графік відображає кількість камер, що працюють та не працюють,

другий же графік виводить загальну кількість камер, що знаходяться у окремому районі міста. Зліва від таблиця навігаційна панель, де є 2 сторінки.

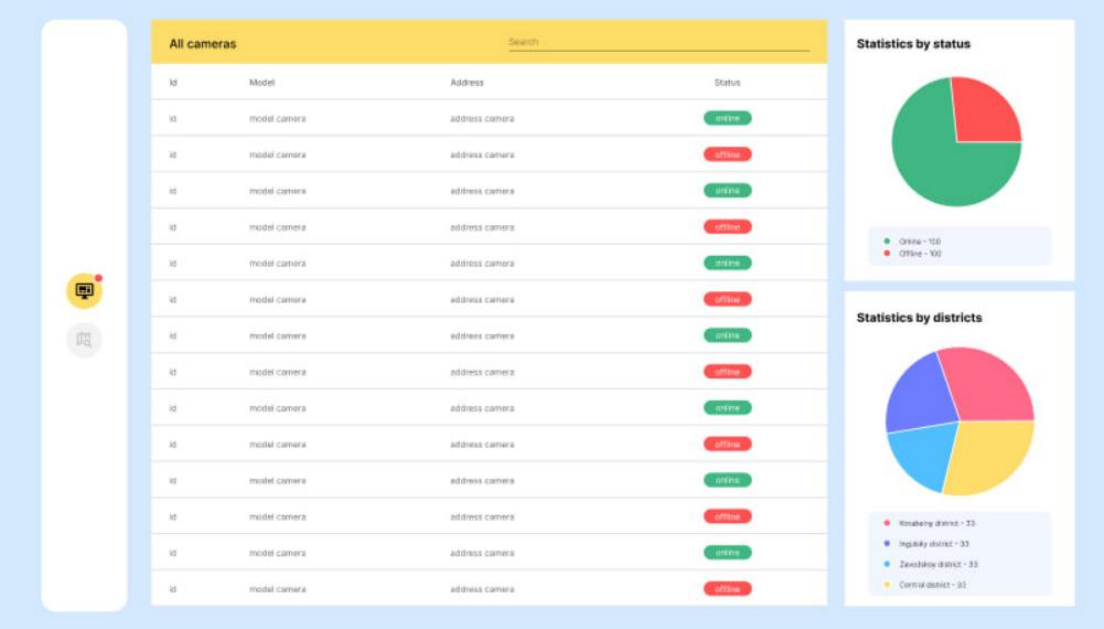


Рисунок 2.9 – Головна сторінка

Щоб переглянути, скільки було максимально людей на зупинці у камері, було створено модальне вікно (рис. 2.10), де буде розташована діаграма, на якій буде видно максимум людей погодинно, наприклад, с 06:00 по 07:00 максимум був 40 осіб, то ця кількість і буде виведена у діаграму (рис. 2.10).

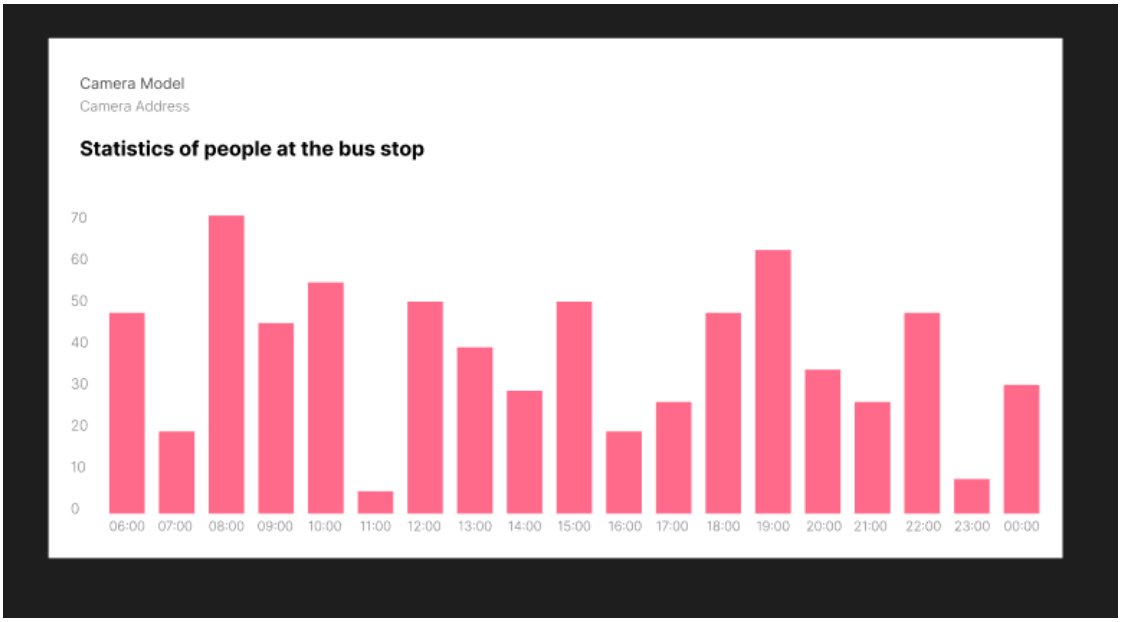


Рисунок 2.10 – Модальне вікно

На другій сторінці можна побачити мапу, яка буде мати можливість приближати та віддаляти у необхідній точці. На самій мапі розташовані іконки камер за адресою у таблиці. Сторінка буде мати допоміжну функцію, щоб користувачу можна будо побачити візуально, де саме вона розташована. Дизайн сторінки наведено на рис. 2.11.

Рисунок 2.11 – Друга сторінка

При натиснення на камеру має випливати те саме модальне вікно, що було описано вище.

2.3 Обгрунтування вибору технологій розробки

2.3.1 Webstorm

WebStorm є інструментом для розробки вебсайтів і редагування HTML, CSS і JavaScript коду [7]. Рішення забезпечує швидку навігацію по файлах і генерує повідомлення про проблеми в коді в режимі реального часу. JetBrains WebStorm дозволяє додавати розмітку HTML-документів або елементів SQL

безпосередньо до JavaScript. JetBrains WebStorm здійснює розгортання та синхронізацію проєктів через протокол FTP.

Використовуючи можливості коду HTML/XHTML та XML, WebStorm забезпечує автоматичне завершення стилів, посилань, атрибутів та інших елементів коду. Під час роботи з CSS здійснюється завершення коду класів, HTML-номерів, ключових слів тощо. WebStorm пропонує автоматичне вирішення таких проблем, як вибір формату, властивостей, класів, посилань на файли та інших атрибутів CSS.

Інтерфейс редактора можна побачити на рис. 2.12.

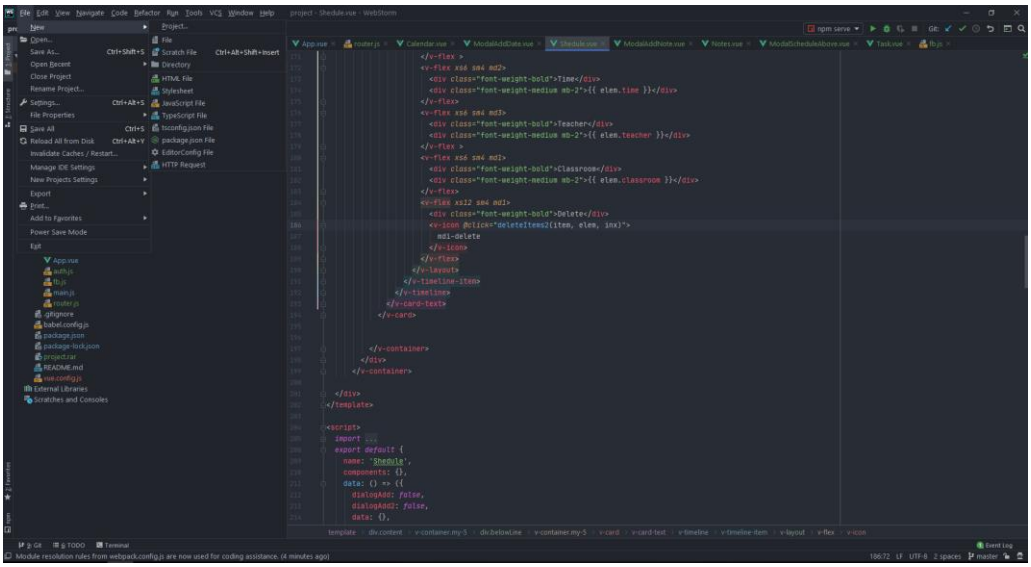


Рисунок 2.12 – Інтерфейс WebStorm

2.3.2 Vue

Vue.js [8] – це JavaScript бібліотека для створення вебінтерфейсів з використанням шаблону архітектури MVVM (Model-View-ViewModel). Оскільки Vue працює тільки на «рівні представлення» і не використовується для проміжного програмного забезпечення та бекенд, він може легко інтегруватися з іншими проєктами та бібліотеками. Vue.js містить широку функціональність для рівня вистав і може використовуватися для створення потужних односторінкових вебзастосунків.

Головну сторінку сайту можна побачити на рис. 2.15.

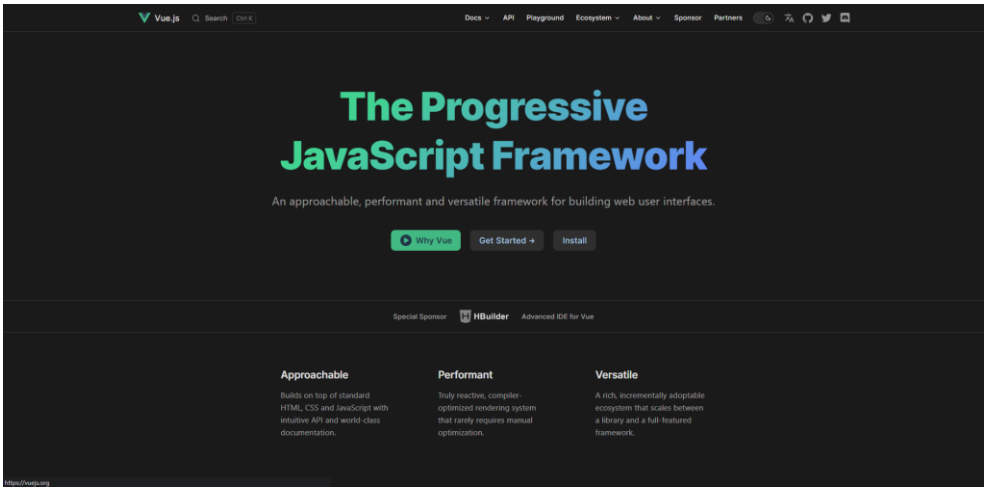


Рисунок 2.15 – Головна сторінка сайту Vue

Функції Vue.js:

– реактивні інтерфейси;

– декларативний рендеринг;

– зв'язування даних;

– директиви (усі директиви мають префікс «V-». У директиву передається значення стану, а як аргументи використовуються html атрибути або Vue JS події); логіка шаблонів;

– компоненти;

– опрацювання подій;

– властивості;

– переходи та анімація CSS;

– фільтри.

Основна бібліотека Vue.js 2 дуже мала (всього 17 кбайт). Це гарантує, що навантаження на проєкт, реалізований за допомогою Vue.js, мінімальне, а сайт швидко завантажуватиметься.

Vue підходить для невеликих проєктів, яким необхідно додати трохи реактивності, представити форму за допомогою AJAX, відобразити значення

під час введення даних користувачем, авторизація або інші аналогічні завдання. Vue легко масштабується та добре підходить для об'ємних проєктів, тому його називають прогресивним фреймворком.

Vue також підходить для великих односторінкових застосунків завдяки своїм основним компонентам, таким як Router та Vuex. З Vue можна використовувати загальнодоступні API для створення програм, так і реалізовувати виконувані сервером програми. Але Vue найкраще підходить для розробки рішень, які використовують зовнішні API для обробки даних.

За допомогою Vue також можна створювати frontend-блог на популярних CMS. Vue.js відмінно підходить і для розробки динамічних інтерфейсів, що адаптуються до користувача.

Vue-Router [9] – це пакет JavaScript, який дозволяє налаштувати маршрутизацію для односторінкових програм (SPA).

SPA відноситься до вебзастосунку, який обслуговує тільки одну index.html і динамічно відображає контент, що, швидше за все, є способом налаштування сучасних JavaScript-фреймворків, таких як Vue.js.

Використання SPA має безліч переваг, але одним із основних недоліків є те, що всі компоненти вебсторінки доставляються, додаються або видаляються за допомогою JavaScript без завантаження додаткових HTML-сторінок з сервера. У цьому суть SPA, але головна проблема полягає в можливості пересуватися по сторінках, до яких користувачі звикли на більшості вебсайтів. Ось тут і допомагає Vue-Router.

2.3.3 Vuetify

В даний час вебзастосунки вже важко уявити без складних елементів керування, таких як інтерактивні таблиці з даними, діалогові вікна, випадають меню та ієрархічні дерева. Писати весь цей набір компонентів самостійно, щоб

вирішити бізнес завдання – довго. Але, на щастя, існують уже готові рішення, які беруть на себе створення такої інфраструктури. Одне з них – Vuetify [10].

Немає необхідності додавати таблиці, форми, меню та діалогові вікна у вигляді окремих пакетів у проєкт, а потім приводити їх усіх до загального стилю оформлення. Компоненти Vuetify вже виконані в одноманітному стилі та дуже добре налаштовуються та розширюються під потреби користувача.

Сторінка сайту продемонстрована на рис. 2.16.

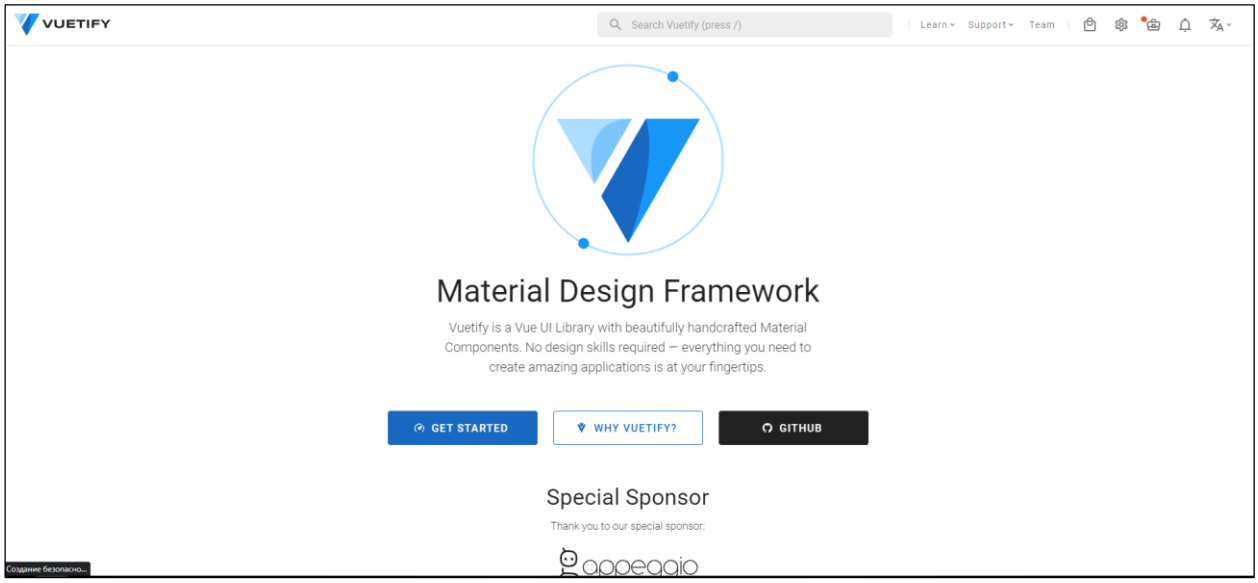


Рисунок 2.16 – Головна сторінка сайту Vuetify

Vuetify розроблений точно відповідно до специфікації Material Design. Кожен компонент розроблений вручну, щоб надати інструменти інтерфейсу користувача для наступного застосунку. Розробка не зупиняється на основних компонентах, описаних у специфікації Google. Завдяки підтримці членів спільноти та спонсорів, додаткові компоненти будуть розроблені та доступні для всіх.

2.3.4 Firebase

Firebase [11] – це платформа для розробки мобільних застосунків від компанії Google, в якій є найсучасніші функції для розробки, перекомпонування та покращення застосунків.

Firebase – це, по суті, набір інструментів, які розробники можуть використовувати, створюючи та змінюючи програми залежно від своєї потреби.

Мета Firebase полягає у вирішенні трьох основних проблем розробників:

– швидко створити програму;

– випустити та забезпечити надійний моніторинг працездатності;

– залучити користувачів.

Головну сторінку можна переглянути на рис. 2.17.

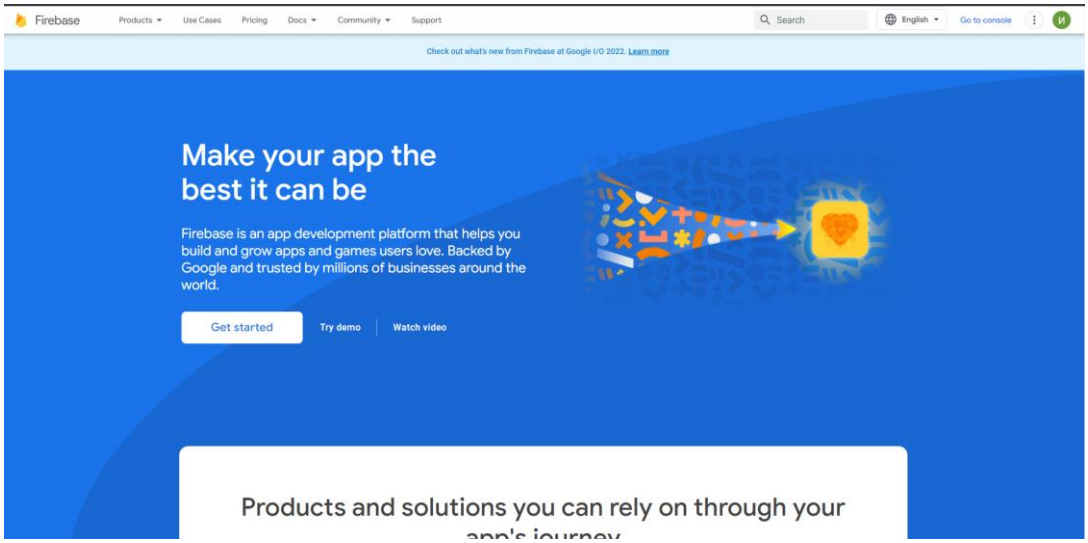


Рисунок 2.17 – Головна сторінка сайту Firebase

Розробники, що використовують цю платформу, отримують доступ до сервісів, за допомогою яких вони можуть розробляти свої продукти, і це дозволяє їм зосередитися безпосередньо на наданні якісного продукту.

Деякі з найпопулярніших функцій платформи Google Firebase включають БД, автентифікацію, push-сповіщення, аналітику, зберігання файлів і багато іншого.

2.3.5 PyCharm

PyCharm [12] – це інтегроване середовище розробки (IDE) для мови програмування Python. Воно надає розширений набір інструментів для розробки програм, що дозволяє програмістам ефективно створювати, тестувати та налагоджувати свої проєкти. Основна мета PyCharm – спростити робочий процес програміста, забезпечуючи йому зручний інтерфейс та потужні функціональні можливості.

У програмі є різні інструменти для автоматичного завершення коду, підказок, аналізу коду та відлагодження. PyCharm підтримує використання віртуальних середовищ, що дозволяє ізолювати залежності проєкту. Він також інтегрований із системами контролю версій, такими як Git, щоб допомогти ефективно керувати змінами в коді.

PyCharm має версії для роботи як з відкритим, так і з комерційним (Professional) функціоналом. Версія Community надає базовий функціонал для роботи з Python, в той час як версія Professional має додаткові інструменти, такі як підтримка вебфреймворків, інструменти для візуального редагування БД та інші розширені можливості.

Інтерфейс редактора можна побачити на рис. 2.18.

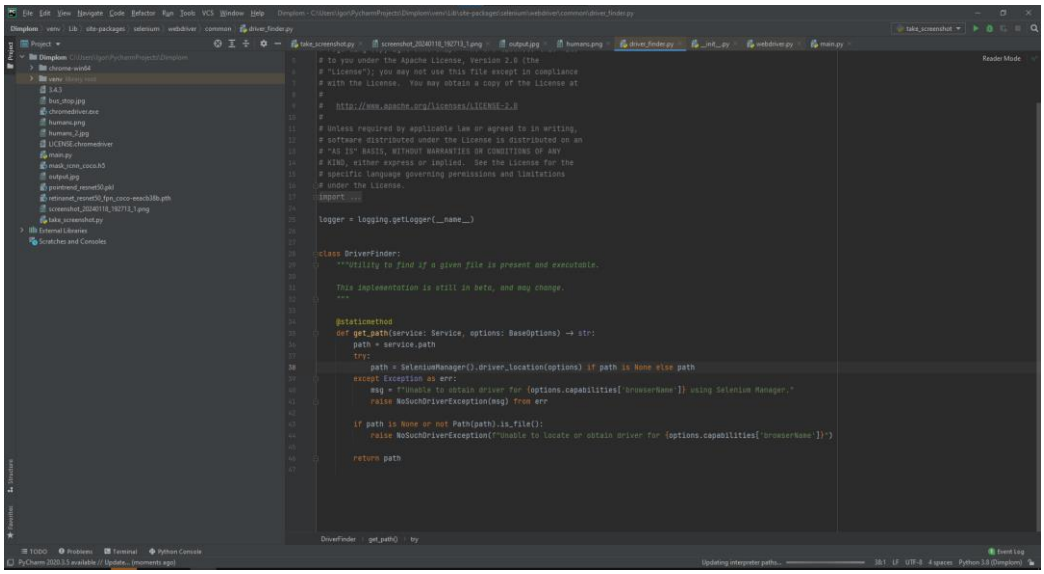


Рисунок 2.18 – Інтерфейс PyCharm

2.3.6 Python

Мова програмування Python [13] є високорівневою, інтерпретованою мовою загального призначення. Вона була створена Гвідо ван Россумом і вперше випущена у 1991 році. Python вирізняється простотою синтаксису та акцентом на читабельності коду, що робить його відмінним вибором для початківців та експертів програмування.

Основні особливості мови Python включають:

– читабельний синтаксис: Python використовує відступи для визначення блоків коду, що робить код більш зрозумілим та читабельним;

– динамічна типізація: Змінні в Python не потребують явного вказівання типу, їхній тип визначається автоматично в процесі виконання;

– багатий набір бібліотек: Python має велику кількість стандартних бібліотек, які дозволяють ефективно виконувати різноманітні завдання без необхідності написання коду «з нуля»;

– спрощена робота із змінними: Python дозволяє легко взаємодіяти зі списками, словниками та іншими структурами даних, що полегшує роботу з інформацією;

– широкий спектр застосувань: Python застосовується у багатьох областях, таких як веброзробка, наукове моделювання, штучний інтелект, обробка даних, автоматизація систем і багато іншого;

– активна спільнота: Python має велику та активну спільноту розробників, яка активно підтримує і розвиває мову, а також створює різноманітні сторонні бібліотеки та інструменти.

2.4 Аналіз характеристик IP-камер для відеоспостереження

Проводові та безпроводові IP-камери – це два типи камер для відеоспостереження, кожен з яких має свої переваги та недоліки.

До переваг проводових камер можна навести:

– стійкість сигналу: Проводові камери працюють на основі фізичного підключення до мережі, тому не піддаються впливу перешкод, які можуть виникнути у безпровідних мережах;

– більш стабільний потік даних: Проводові камери зазвичай мають стабільніший потік даних і менше можливостей для переривань зв'язку, що важливо для відеоспостереження в реальному часі;

– більш висока якість зображення: Оскільки проводове підключення може передавати більше даних, камери можуть забезпечувати вищу якість зображення та відео.

До недоліків цих видів камер, можна навести:

– складніше встановлення: Проводові камери потребують прокладання кабелю до кожної камери, що може бути складним у великих будівлях або на відкритих територіях;

– обмежена мобільність: Камера зазвичай обмежена розташуванням кабелю, тому її складніше переміщувати або перевстановлювати.

Щодо безпроводових камер, то вони мають наступні переваги:

– простота встановлення: Бездротові камери не потребують прокладання кабелю, тому їх легко встановити там, де це необхідно;

– більша мобільність: Бездротові камери можуть бути легко переміщені або перевстановлені без необхідності перекладати кабель;

– більша гнучкість: Ці камери можуть бути розміщені в місцях, де важко або неможливо провести кабельне підключення.

Недоліки:

– вразливість до перешкод: Сигнал з бездротових камер може бути втрачений або зіпсований через перешкоди, такі як стіни або інші об'єкти;

– нижча стабільність сигналу: Бездротові камери можуть мати менш стабільний зв'язок і видаляти або зважати на якість зображення.

IP-камера TP-LINK Tapo C200 є сучасним пристроєм відеоспостереження, розробленим провідним виробником мережевого обладнання TP-LINK. Ця камера має кілька важливих характеристик, які сприяють її ефективному використанню в системах безпеки та відеоспостереження.

Однією з ключових особливостей є висока якість зображення, що забезпечується високою роздільною здатністю відеопотоку. Це дозволяє отримувати чіткі та деталізовані знімки, що важливо для точного моніторингу того, що відбувається в просторі, що спостерігається.

Широкий кут огляду об'єктива також є значною характеристикою. Ця особливість забезпечує широке охоплення площі, зменшуючи кількість камер, необхідні повного покриття об'єкта відеоспостереження.

Однак, не лише візуальні дані є важливими. IP-камера TP-LINK Tapo C200 підтримує двосторонній аудіозв'язок. Це дозволяє користувачам не лише бачити, а й чути звуки у місці встановлення камери. Така функція розширює можливості в галузі безпеки та забезпечує повнішу взаємодію з навколишнім середовищем.

Зовнішній вигляд камери наведено на рис. 2.19.

Рисунок 2.19 – IP-камера TP-LINK Tapo C200

Висновок до розділу 2

У другому розділі роботи було проведено аналіз ключових аспектів моделювання та проєктування системи для визначення кількості виявлених людей. Було здійснено ретельний аналіз архітектури нейронної мережі RetinaNet, яка є важливим етапом у розробці системи відеоспостереження.

Розроблений дизайн для користувацького інтерфейсу з урахуванням зручності та інтуїтивності використання, що є важливим для успіху системи.

Було проведено обґрунтування вибору конкретних технологій розробки з метою забезпечення ефективності та масштабованості. Також виконано аналіз характеристик IP-камер для відеоспостереження з метою обрати оптимальну модель, яка відповідає вимогам системи та забезпечує необхідну якість відеозапису для подальшого аналізу.

3 РОЗРОБКА СИСТЕМИ ПІДРАХУНКУ КІЛЬКОСТІ ЛЮДЕЙ НА ЗУПИНЦІ

3.1 Налаштування середовища для розробки

Спочатку перед розробкою, потрібно налаштувати середовище для програмування. Як було описано раніше, в якості середовища для розробки буде використано WebStorm. Налаштування у даному застосунку досить просте, все починається з першого запуску де відкривається головне вікно редактора з підказками до гарячих клавіш (рис. 3.1). Далі середовище запропонує завантажити додаткові плагіни для більш зручного програмування, так як будуть з’являтись підказки чи спрацьовувати автопідстановку тексту.

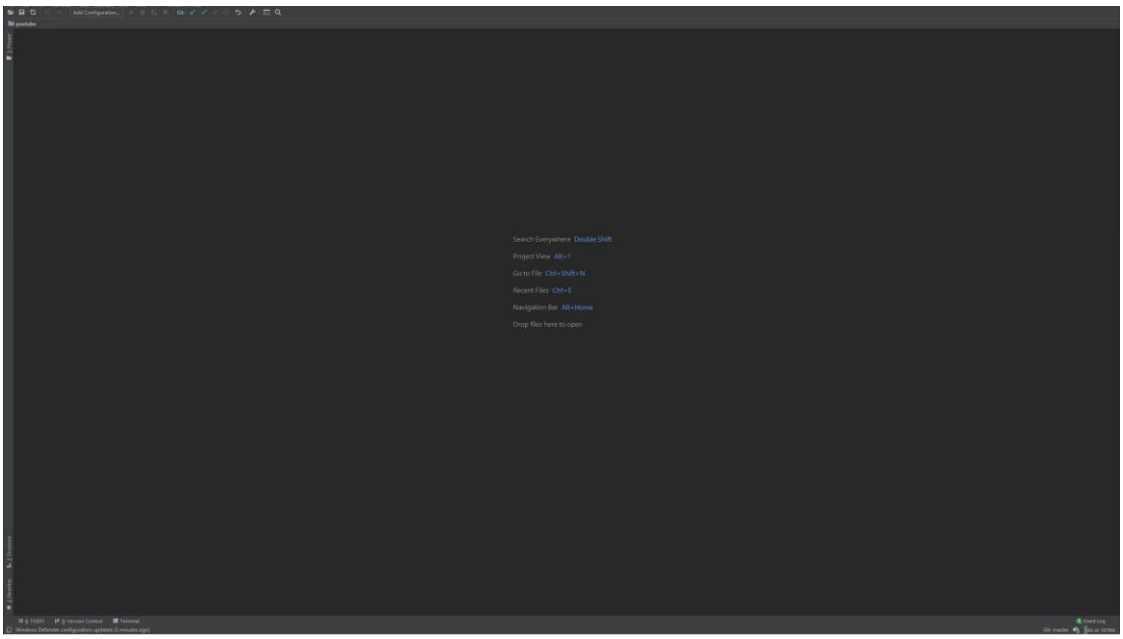


Рисунок 3.1 – Головна сторінка редактору

Наступним кроком налаштування є створення проєкту в редакторі. Завдяки плагінам середовище надає можливість вибору вебфреймворків прямо під час створення проєкту. Ми обираємо необхідний нам фреймворк, наприклад, Vue.js, та натискаємо кнопку «Створити». Після створення нового проєкту, якщо обраний фреймворк, він автоматично почне завантажуватись та

встановлюватись до проєкту. Потім необхідно обирати місце розташування для проєкту та натиснути відповідну кнопку (рис. 3.2).

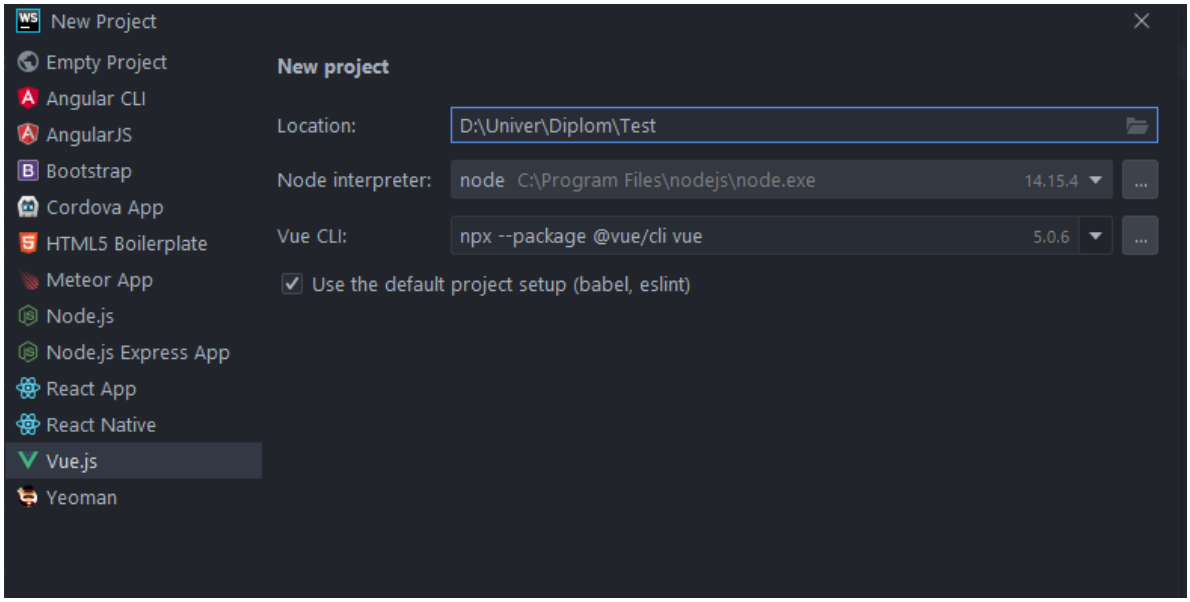


Рисунок 3.2 – Створення нового проєкту з фреймворком Vue

Перед початком розробки вебзастосунку потрібно налаштувати середовище та встановити всі необхідні залежності, бібліотеки та пакети. Це можна зробити за допомогою командного рядка, який має безліч альтернатив.

У якості командного рядка можна використовувати звичайну консоль, PowerShell, термінал PhpStorm або консоль OpenServer. Першим кроком для розробки клієнтської частини є завантаження та встановлення фреймворку. Для цього використовується пакетний менеджер npm, який однією командою завантажує всі необхідні залежності та встановлює їх.

NPM є менеджером, який використовується багатьма JS-фреймворками для завантаження додаткових модулів та компонентів.

Для виконання цих дій у відкритій консолі редактора вводиться відповідна команда (рис. 3.3).

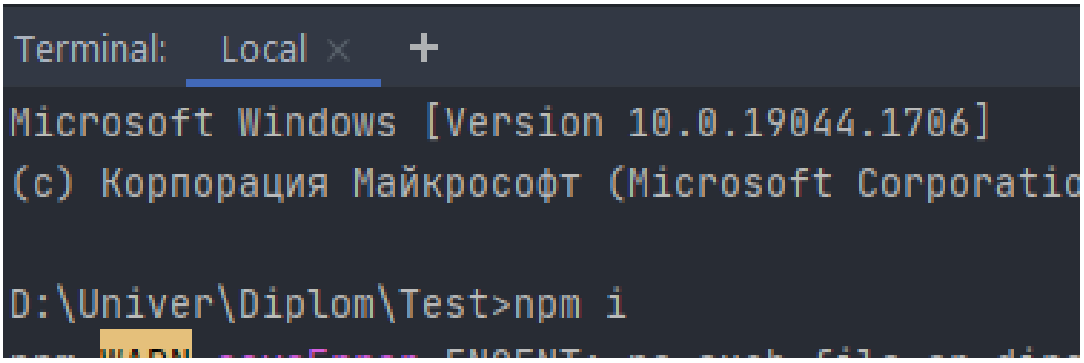


Рисунок 3.3 – Встановлення додаткових бібліотек та оновлення старих

Тепер, встановивши фреймворк та всі потрібні залежності, проєкт готовий до написання власного коду.

3.2 Розробка бази даних

Для початку, на головній сторінці Firebase потрібно створити новий проєкт. Це можна побачити на рис. 3.4

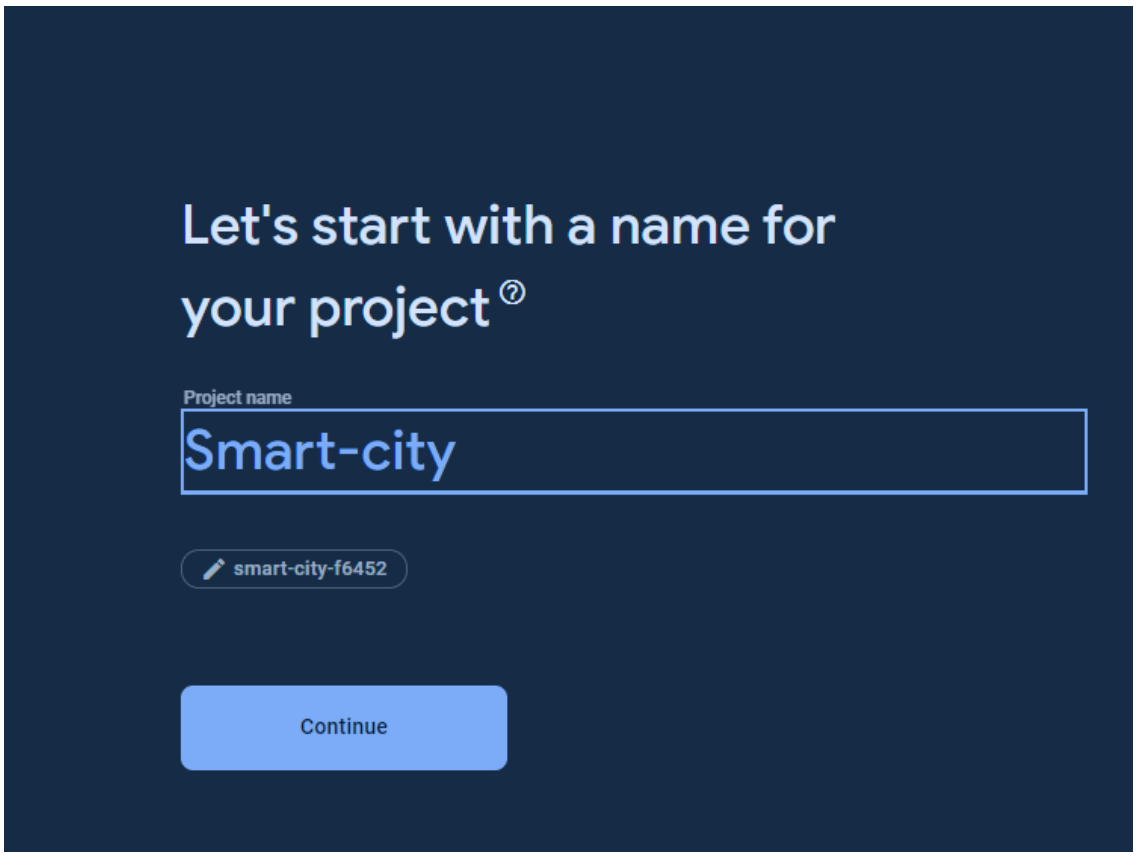


Рисунок 3.4 – Створення проєкту

Наступним етапом буде перевірка бази даних у реальному часі після того, як буде отримано доступ до консолі Firebase. У розділі «Розробники» -> «База даних» будуть доступні дві опції: «Хмарне сховище» та «Бази даних у реальному часі» (рис. 3.5).

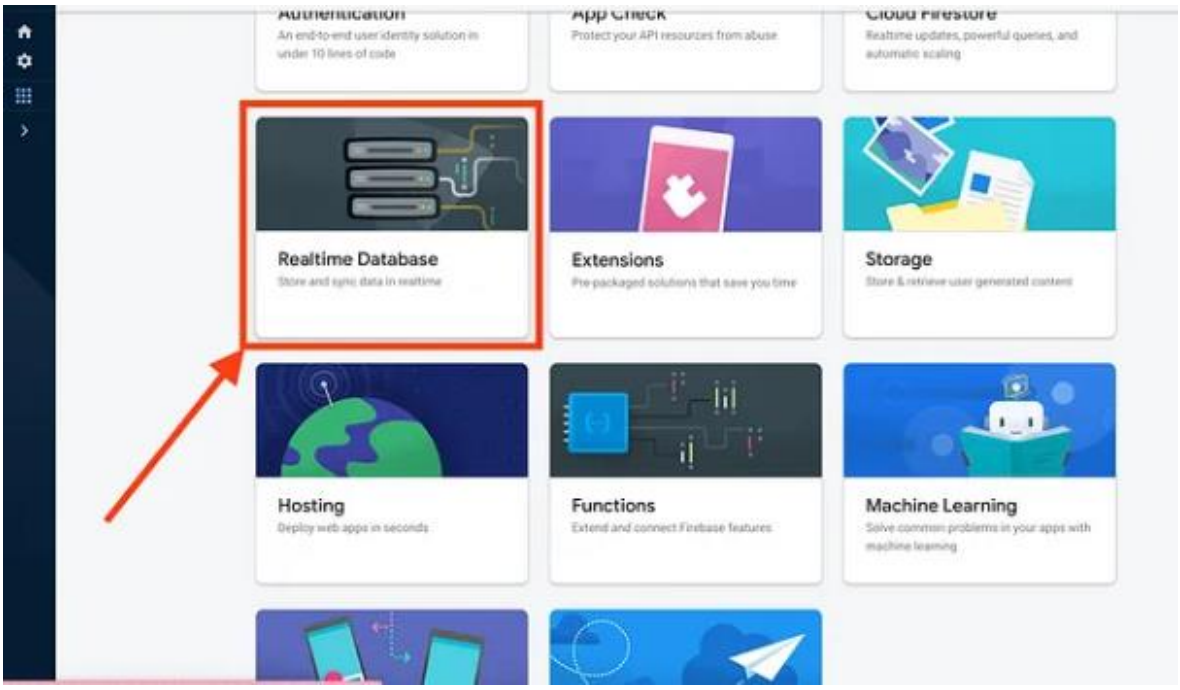


Рисунок 3.5 – Знаходження БД

Для створення бази даних спочатку потрібно перейти до консолі. У меню зліва потрібно обрати «База даних». Далі необхідно прокрутити до розділу «База даних у реальному часі» і натиснути кнопку «Створити базу даних» (рис. 3.6–3.7).

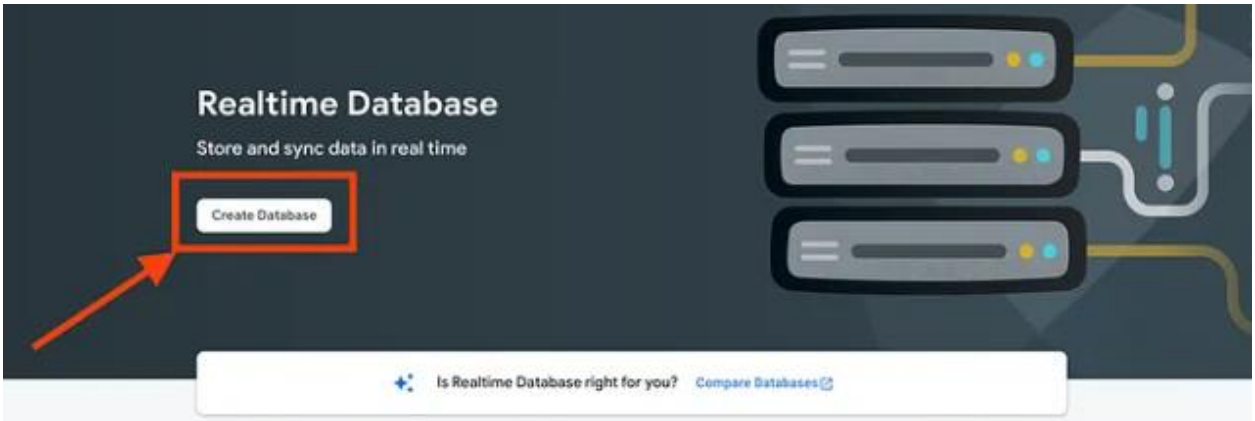


Рисунок 3.6 – Створення БД

Рисунок 3.7 – Структура БД

Опис БД:

– ID: Унікальний ідентифікатор для кожної камери;

– Координати: Широта та довгота, що вказують розташування камери;

– Статус: Вказує, чи камера активна (true) чи ні (false);

– Адреса: Фізична адреса, де розташована камера;

– Фотографії: Масив об'єктів, що містять інформацію про фотографії, пов'язані з камерою.

Кожен об'єкт фотографії включає:

– URL: Посилання на фотографію;

CreatedAt: Тимчасова мітка, яка вказує, коли було створено фотографію;

– ID: Унікальний ідентифікатор кожної фотографії;

– Count: Кількість переглядів фотографії;

– Модель: модель камери (в даному випадку "TP-LINK Tapo C310");

– Район: Ідентифікатор району, де розташовано камеру.

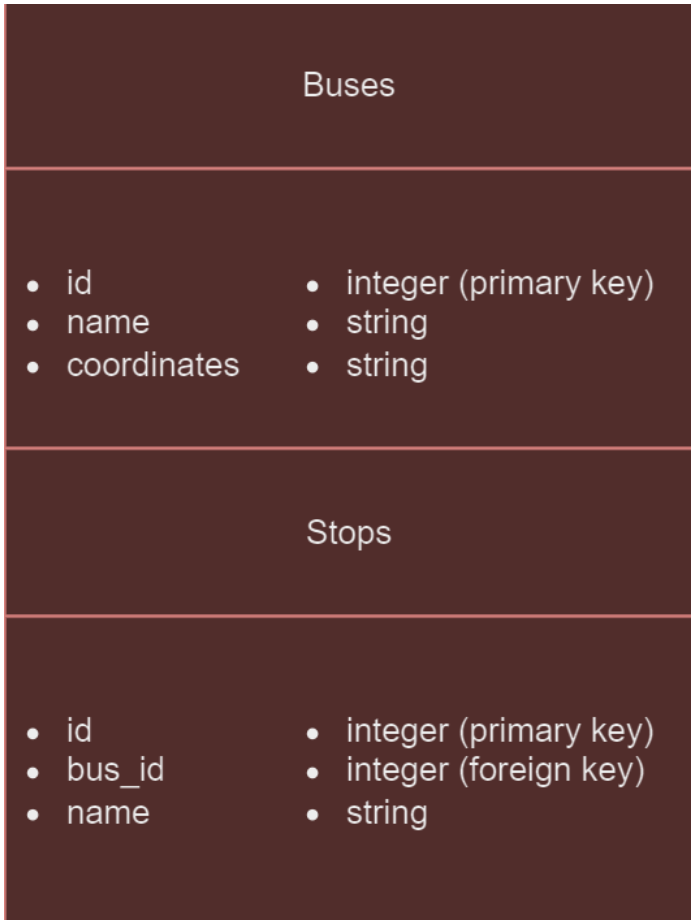


Рисунок 3.8 – Структура БД

Опис БД:

– ID: Унікальний ідентифікатор автобусного маршруту;

– name: Назва маршруту автобуса. Це рядкове значення, яке означає номер або назву конкретного маршруту;

– Coordinates: Рядок координат маршруту автобуса. Це рядкове значення, що містить координати маршруту у форматі «широта, довгота», розділені комами;

– Stops: містить зупинки, що належать кожному маршруту. Включає унікальний ідентифікатор зупинки (id), ідентифікатор маршруту, до якого вона відноситься (bus_id), індекс зупинки (index) і назва зупинки (name) ;

– id: Унікальний ідентифікатор зупинки. Це ціле значення, що використовується як первинний ключ для ідентифікації кожної зупинки в базі даних;

– bus_id: Ідентифікатор автобусного маршруту. Це ціле значення, що зв'язує зупинку з конкретним маршрутом автобуса, використовуючи зовнішній ключ;

– name: Назва зупинки. Це рядкове значення, що вказує на назву конкретної зупинки на маршруті.

3.3 Розробка алгоритму роботи програми

На початку скрипта відбувається імпорт необхідних модулів та бібліотек, які знадобляться для роботи скрипту. Ці імпортовані модулі забезпечують доступ до різних функцій та можливостей, таких як взаємодія з веббраузером, робота з часом, виявлення об'єктів на зображеннях, взаємодія з операційною системою та хмарними сервісами Google Cloud (рис. 3.9).

```
from selenium import webdriver
import time
from datetime import datetime
from imageai.Detection import ObjectDetection
import os
from google.cloud import firestore, storage
from datetime import timedelta
```

Рисунок 3.9 – Імпорт бібліотек

Модуль selenium надає можливості автоматизації дій веббраузера. Він використовується для відкриття вебсторінки, виконання дій на сторінці та отримання даних.

Модуль time надає функції для роботи з тимчасовими затримками та тимчасовими мітками.

Модуль datetime надає функції для роботи з поточним часом та датою.

Модуль ObjectDetection є частиною ImageAI бібліотеки і використовується для виявлення об'єктів на зображеннях. У цьому скрипті використовується виявлення людей на скріншотах.

Модуль os надає функції для роботи з операційною системою, такі як робота з шляхами файлів та каталогів, виконання команд та інші операції з файлами.

Модулі firestore та storage надають клієнтські інтерфейси для взаємодії з базою даних Firestore та сховищем файлів Cloud Storage відповідно. Вони дозволяють завантажувати дані в базу даних та зберігати файли у хмарному сховищі.

Імпортовані модулі та бібліотеки забезпечують основу для роботи скрипта, надаючи необхідні функції та можливості для взаємодії з веббраузером, обробки зображень та збереження даних у базі даних. Ці модулі забезпечують функціональність та розширюваність вашого скрипту, дозволяючи реалізувати різні завдання в автоматизації та обробці даних.

Після імпорту модулів та бібліотек встановлюються налаштування, такі як шлях до файлу ключа служби Google Cloud, ім'я колекції Firestore, клієнти для роботи з Firestore та Cloud Storage, а також поточний робочий каталог та шлях до файлу вебдрайвера Chrome (рис. 3.10).

```
os.environ["GOOGLE_APPLICATION_CREDENTIALS"] = "kay.json"

collection_name = 'cameras'
storage_client = storage.Client()
db = firestore.Client()

exec_path = os.getcwd()

driver_path = 'path/to/chromedriver.exe'
```

Рисунок 3.10 – Перші налаштування

Тепер створюється об'єкт ChromeOptions для конфігурації параметрів Chrome, таких як ігнорування помилок SSL та вимкнення автоматичного оновлення сторінки під час завантаження. Потім створюється об'єкт ChromeService із зазначенням шляху до файлу вебдрайвера, що виконується, а потім створюється екземпляр драйвера Chrome за допомогою створених параметрів і об'єкта сервісу (рис. 3.11).

```
chrome_options = webdriver.ChromeOptions()


chrome_options.add_argument('--ignore-certificate-errors')


chrome_options.add_argument('--disable-prompt-on-repost')

chrome_service = webdriver.chrome.service.Service(driver_path)

driver = webdriver.Chrome(service=chrome_service, options=chrome_options)
```

Рисунок 3.11 – Налаштування вебдрайвера Selenium

У скрипті визначено дві функції: upload_image_to_storage та update_document_with_image, які забезпечують взаємодію з сервісами Google Cloud для збереження інформації про завантажені зображення та їх подальше використання в базі даних Firestore.

Функція upload_image_to_storage відповідає за завантаження зображення у Firebase Storage та отримання його URL. У цій функції спочатку визначається ім'я бакета Cloud Storage, в якому зберігатиметься завантажене зображення. Потім із шляху до файлу вилучається ім'я файлу. Далі створюється об'єкт Blob, що представляє файл, що завантажується, і завантажується в бакет. Після цього генерується URL для доступу до завантаженого зображення за допомогою часу дії URL, встановленого на один рік. Нарешті URL зображення повертається з функції для подальшого використання.

Функція update_document_with_image відповідає за оновлення документа Firestore, додаючи інформацію про зображення. У цій функції спочатку виходить посилання на документ у колекції Firestore за вказаним ім'ям колекції та ідентифікатором документа. Потім створюється об'єкт даних зображення, який містить ідентифікатор, URL, кількість людей і час створення. Далі виходить поточний стан поля photos у документі. Новий об'єкт зображення додається до масиву зображень, а потім документ Firestore оновлюється з оновленим масивом зображень.

Код функцій можна переглянути на рис. 3.12–3.13.

```
def upload_image_to_storage(file_path):
    """Завантаження зображення в Firebase Storage та отримання його URL."""
    bucket_name = "smartcity-a7289.appspot.com"

    # Отримання імені файлу зі шляху
    file_name = os.path.basename(file_path)

    # Завантаження файлу в бакет
    blob = storage_client.bucket(bucket_name).blob(file_name)
    blob.upload_from_filename(file_path)

    # Отримання URL зображення
    expiration = datetime.now() + timedelta(days=365)
    image_url = blob.generate_signed_url(expiration)
    print(image_url)
    return image_url
```

Рисунок 3.12 – Функція «upload_image_to_storage»

```python
def update_document_with_image(collection_name, document_id, image_url, count):
    """Додати зображення до поля photos документа в Firestore."""
    try:
        # Посилання на документ
        doc_ref = db.collection(collection_name).document(document_id)

        # Створення об'єкта зображення
        image_data = {
            'id': datetime.now().strftime("%Y%m%d%H%M%S%f"),
            'url': image_url,
            'count': count,
            'createdAt': datetime.now()
        }

        # Отримання поточного стану поля photos
        doc = doc_ref.get()
        photos = doc.to_dict().get('photos', [])

        # Додавання нового об'єкта зображення до масиву
        photos.append(image_data)

        # Оновлення документа
        doc_ref.update({'photos': photos})

        print(f'Зображення успішно додано до документа {document_id} у колекції {collection_name}.')
    except Exception as e:
        print(f'Помилка при додаванні зображення до документа: {e}')
```

Рисунок 3.13 – Функція «update_document_with_image»

Основний цикл скрипту відкриває вебсторінку, робить скріншоти сторінки, обробляє їх для виявлення об'єктів (в даному випадку – людей), завантажує їх у сховище та оновлює документ у базі даних Firestore з даними про людей на зображенні. Після завершення роботи програми браузер закривається для звільнення ресурсів (рис. 3.14).

```python
try:
    # Відкриваємо сторінку один раз
    # Цикл скріншотів та обробки зображень
finally:
    driver.quit() # Закриття браузера після завершення програми
```

Рисунок 3.14 – Функція «Основний цикл скрипту»

Висновок до розділу 3

У третьому розділі було розроблено систему підрахунку кількості людей на зупинці міського автотранспорту на підставі аналізу зображень з IP-камер системи відеоспостереження «Розумне місто».

На початку розділу було описано Після встановлення та налаштування середовища розробки у WebStorm, включаючи вибір необхідних плагінів та створення нового проєкту з обраним фреймворком Vue.js, наступним кроком було завантаження необхідних модулів та бібліотек за допомогою пакетного менеджера npm. Це дозволило підготувати проєкт до подальшого написання коду.

Другим кроком було створення бази даних у Firebase. Після створення бази даних, було визначено та описано її структуру, включаючи поля для кожного об'єкта даних.

Останнім кроком було надано загальний огляд структури та функціоналу скрипта, який автоматизує процес виявлення об'єктів на зображеннях, їх завантаження у хмарне сховище та оновлення інформації у базі даних Firestore. Скрипт використовує різноманітні модулі та бібліотеки Python, такі як Selenium для взаємодії з веб-браузером, ImageAI для виявлення об'єктів на зображеннях, а також модулі Google Cloud для роботи з хмарними сервісами.

4 НАЛАШТУВАННЯ ТА ТЕСТУВАННЯ СИСТЕМИ ВИЗНАЧЕННЯ КІЛЬКОСТІ ПАСАЖИРІВ ДЛЯ ПОДАЧІ МІСЬКОГО ТРАНСПОРТА

4.1 Dashboard

Наступним кроком треба перейти до налаштувань і скопіювати конфігурацію для підключення бази даних к проєкту. У створеному js-файлі, вставимо конфігурацію та виведемо у лог змінну, щоб перевірити підключення бази даних(рис. 4.1–4.2).

```
import firebase from 'firebase/app';
import 'firebase/firestore';

const firebaseConfig = {
  apiKey: "AIzaSyAIkSQwZwOInDvZLHb-fVYRYViZw-wUhLs",
  authDomain: "smartcity-a7289.firebaseapp.com",
  databaseURL: "https://smartcity-a7289-default-rtdb.firebaseio.com",
  projectId: "smartcity-a7289",
  storageBucket: "smartcity-a7289.appspot.com",
  messagingSenderId: "390577141787",
  appId: "1:390577141787:web:148223d217fcfd3da8e0be"
};

if (!firebase.apps.length) {
  firebase.initializeApp(firebaseConfig);
}

export default firebase;
```

Рисунок 4.1 – Ініціалізація бази даних

Рисунок 4.2 – Перевірка підключення

Після створення та ретельного тестування компонента Vue.js для домашньої сторінки було виявлено такі результати:

– візуальна частина інтерфейсу успішно реалізована за допомогою бібліотеки Vuetify. Структура сторінки, побудована з використанням

компонентів v-row, v-col та v-card, надає користувачеві зручне відображення списку камер;

– функціональність пошуку, що втілена в компоненті v-text-field, демонструє коректну взаємодію з даними про камери, забезпечуючи ефективну фільтрацію;

– діаграми, представлені компонентами PieChart та DoughnutChart, візуалізують статистику за статусом камер та районів, забезпечуючи інтуїтивно зрозуміле представлення даних. Бейджі у діаграмах, що відповідають колірній схемі, відображаються чітко та ясно;

– модальне вікно, що активується компонентом CameraModel при натисканні на модель камери, надає користувачеві додаткову інформацію, що покращує досвід користувача;

– таблиця, реалізована за допомогою v-data-table, коректно відображає дані про камери та їх статус. Заголовки таблиці у headers відповідають структурі даних;

– методи openModal і closeModal взаємодіють із модальним вікном, забезпечуючи коректне відображення та приховування при необхідності;

– обробка даних у масиві cameras відповідає структурі таблиці, що підтверджує правильність подання інформації;

– структура коду та стилі у розділі style відповідають стандартам Vue.js та Vuetify, що забезпечує читання та підтримуваність коду;

– компонент виявляє стабільність у різних сценаріях взаємодії користувача та запобігає можливим помилкам;

Таким чином, тестування підтвердило, що компонент успішно виконує свої функції, надаючи користувачам зручний та інформативний інтерфейс для роботи з даними камери. Частину коду та саму готову сторінку можна побачити на рис. 4.3–4.4.

```
<v-row>
  <v-col cols="9">
    <v-card elevation="0">

      <v-data-table
          :loading="loadingTable"
          :headers="headers"
          :items="cameras"
          :search="search"
          :footer-props="{'items-per-page-options': [5, 10, 15]}"
          :items-per-page="15"
      >
        <template v-slot:top>
          <v-toolbar flat color="#ffdd69">
            <v-toolbar-title class="title">All cameras</v-toolbar-title>
            <v-spacer></v-spacer>
            <v-text-field
                v-model="search"
                append-icon="mdi-magnify"
                label="Search"
                single-line
                hide-details
                color="black"
            ></v-text-field>
          </v-toolbar>
        </template>

        <template v-slot:[`item.status`]="{ item }">
          <v-chip
              :color="item.status ? '#41b883' : '#ff5252'"
              dark
              small
              class="status"
          >
            {{ item.status ? 'online' : 'offline' }}
          </v-chip>
        </template>

        <template v-slot:[`item.model`]="{ item }">
          <span @click="openModal(item)">
```

Рисунок 4.3 – Частина коду першої сторінки

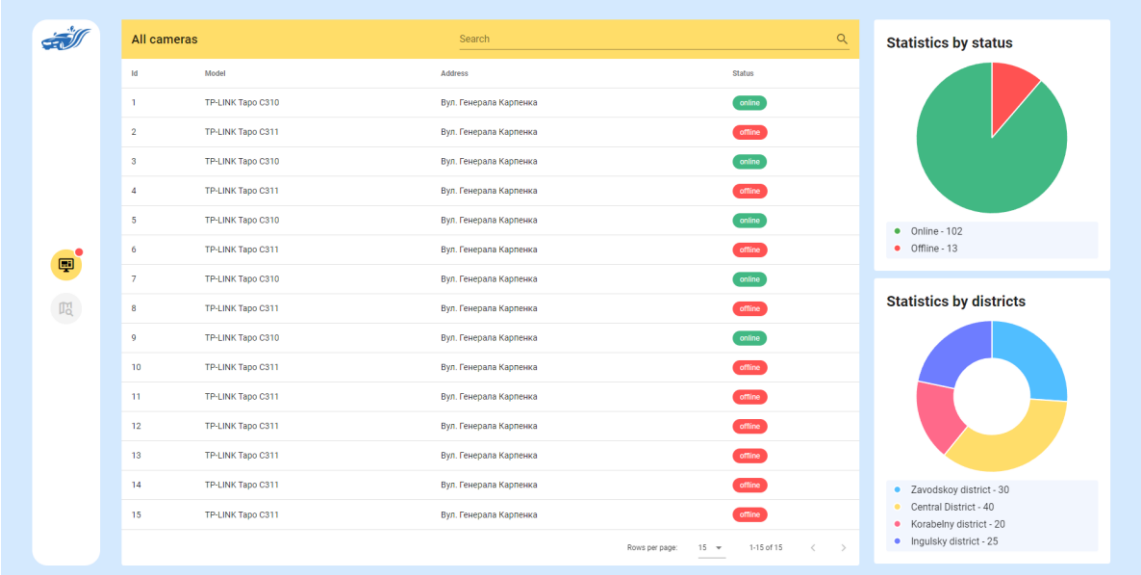


Рисунок 4.4 – Готова перша сторінка

Під час тестування компонента «MapPage» виявлено такі аспекти: ініціалізація картки Leaflet проходить успішно, гарантуючи коректний стартовий рівень масштабування та правильні центральні координати. Інтеграція шарів тайлів OpenStreetMap забезпечує надійне та якісне відображення картографічних даних. Полігони, що представляють різні райони, додаються на карту, реагуючи на взаємодію користувача під час кліку та відкриваючи спливаючі вікна з відповідною інформацією.

Маркери камер успішно відображаються на карті, і спливаючі вікна маркерів містять необхідну інформацію про модель камери, її адресу та кнопку «Info». При натисканні на цю кнопку відкривається модальне вікно CameraModel, де надається додаткова інформація про вибрану камеру.

Методи для керування станом модального вікна взаємодіють коректно, забезпечуючи відкриття та закриття вікна відповідно до очікувань користувача. Структура коду відповідає стандартам Vue.js, що забезпечує його читання та підтримуваність.

Стилі, які у компоненті, застосовуються ефективно, створюючи приємне візуальне сприйняття інтерфейсу. Висота карти адаптована для різних екранів, забезпечуючи чуйність інтерфейсу і приємний користувальницький досвід.

Загалом, компонент MapPage демонструє стабільну поведінку та надає користувачам інтуїтивно зрозумілий та функціональний інтерфейс для взаємодії з картографічними даними та інформацією про камери. Побачити сторінку можна на рис. 4.5–4.6.

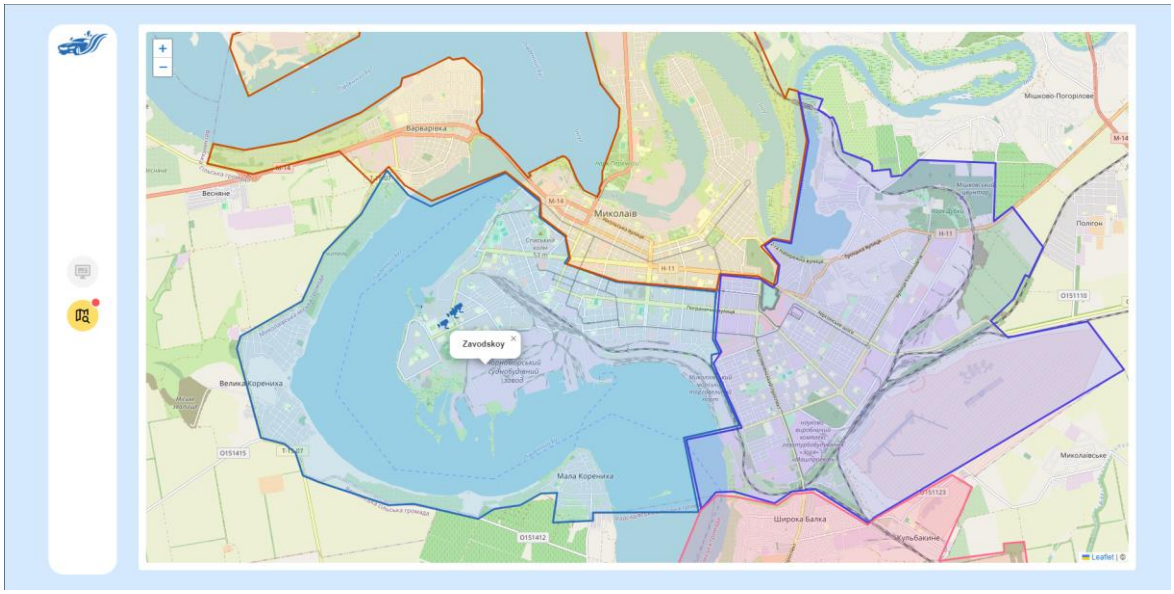


Рисунок 4.5 – Сторінка «MapPage»

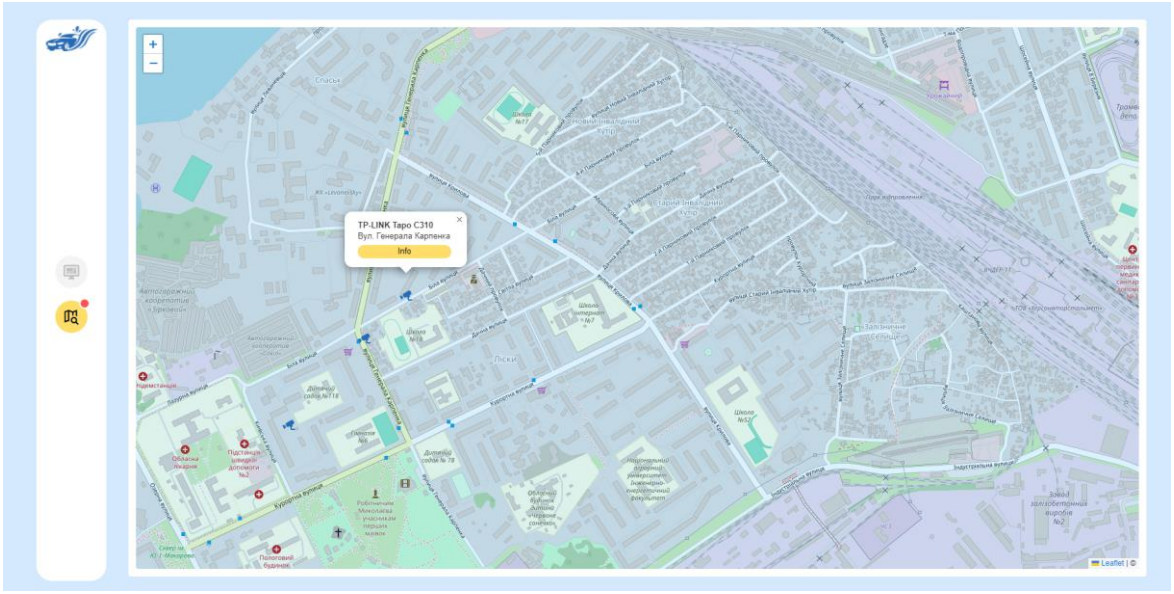


Рисунок 4.6 – Натиснення на камеру

Якщо натиснути ка кнопку «Info», то відкриється модальне вікно з додатковою інформацією, якщо навести на сам бар діаграми, то буде також видно точний час, коли було створено зображення. Якщо натиснути на бар, то це зображення відкриється у новій кладці. У модальному вікні відображається модель, адреса, та статистика людей на зупинці по годинам (рис. 4.7–4.8).

Рисунок 4.7 – Модальне вікно

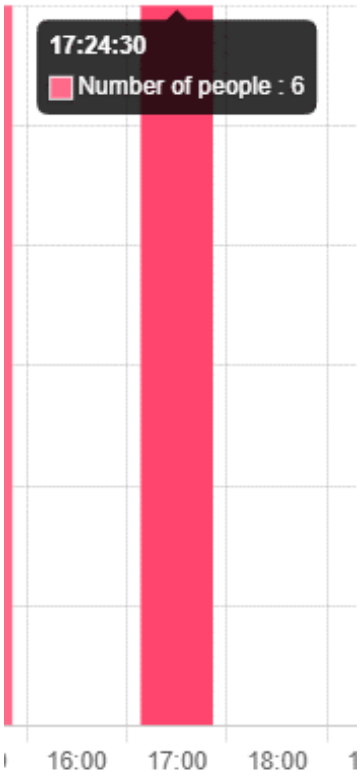


Рисунок 4.8 – Точний час

Рисунок 4.9 – Зображення у новій вкладці

4.2 Знаходження людей на фото

Для початку буде проведено тестування, щоб дізнатися, чи працює скрипт та чи знайде він людей на скриншоті (рис. 4.10).

Рисунок 4.10 – Перше тестування

Скрипт відав зображення, де люди обведенні прямокутником. Також у логах терміналу виведено, скільки саме було знайдено людей (рис. 4.11).

Рисунок 4.11 – Термінал

Тепер буде проведено тестування на зображенні у режимі нічного бачення. Результат видно на рис. 4.12.

Рисунок 4.12 – Знаходження людей у нічному режимі

Як можна побачити, все працює відмінно. Оскільки зараз триває війна, було прийнято рішення не фотографувати реальну зупинку в місті, а за допомогою Гугл-мапи знайти стару зупинку, та на її зображені провести тест.

Результат наведено на рис. 4.13.

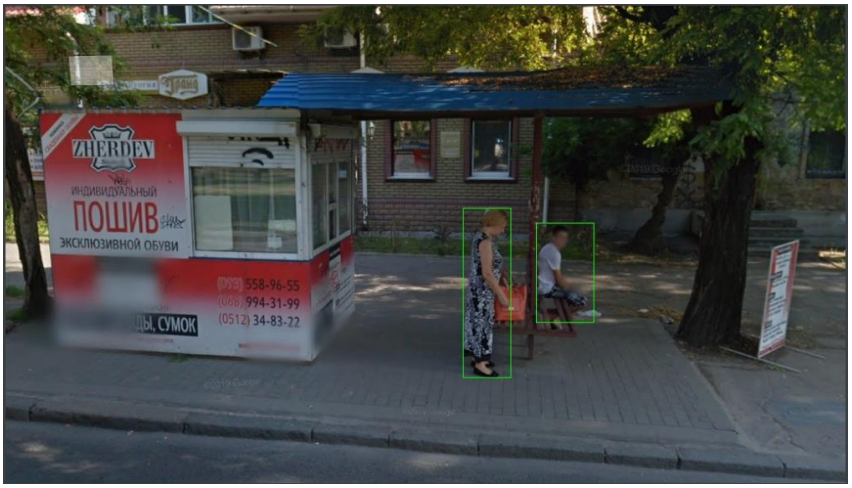


Рисунок 4.13 – Тестування скрипта на зображені зупинки

Після цього всі дані було збережено у базі даних, а саме кількість людей, дата, та посилання на зображення, щоб можна було його потім відкрити через dashboard (рис. 4.14).

Рисунок 4.14 – Дані в БД

4.3 Перегляд маршрутів на мапі

Для того щоб переглянути маршрути на мапі, перш за все буде добавлено дані до бази даних, а саме інформація про маршрути «16» та «56». Після переробки верстки, на сторінці справа від карти було розташовано список усіх маршрутів(рис. 4.15).

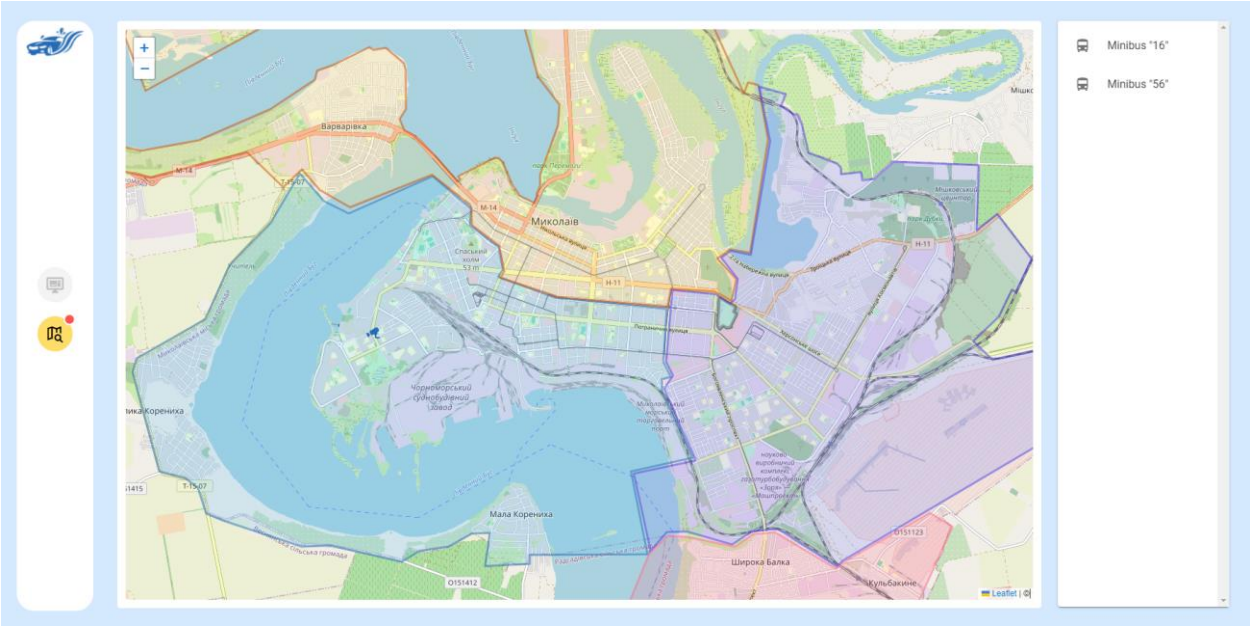


Рисунок 4.15 – Новий вид сторінки

Вибираємо один маршрут і натискаємо на нього. На карті буде створено зелена лінія, яка показує весь маршрут, а список заміниться на новий, де виводиться усі зупинки обраного маршруту (рис. 4.16).

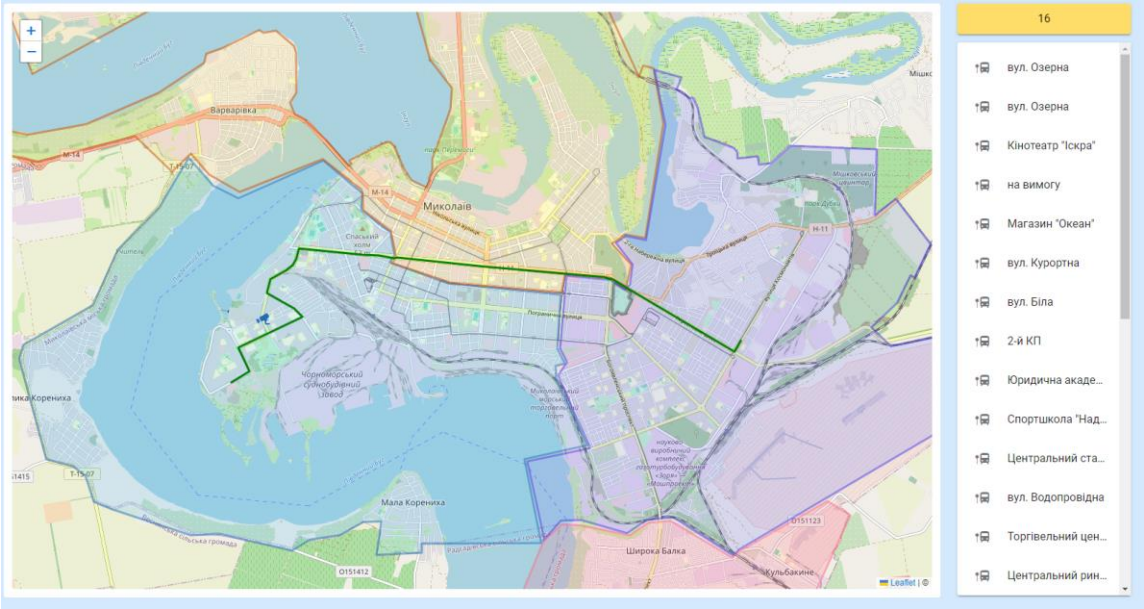


Рисунок 4.15 – Маршрут на карті

Щоб повернутися до списку усіх маршрутів, потрібно натиснуту на жовту кнопку з номером обраного маршруту.

4.5 Перспективи розвитку

4.5.1 Зробити відображення маршруток на карті у режимі онлайн

Для забезпечення режиму онлайн-відображення маршруток на мапі можна використовувати технології реального часу. Додавання GPS-трекера до кожного транспортного засобу дозволить отримувати актуальну інформацію про їх розташування. Цю інформацію можна передавати на сервер із картографічною системою, де за допомогою JavaScript або інших мов програмування вона буде відображатися на мапі у реальному часі.

4.5.2 Вивід списку водіїв та їх інформації

Для виведення списку водіїв та їх інформації можна використовувати базу даних, де зберігатиметься інформація про кожного водія, таку як ім'я, контактні дані, номер транспортного засобу тощо. Ця інформація може бути доступна через вебінтерфейс або мобільний застосунок для зручності користувачів.

4.5.3 Розробити мобільний застосунок для водіїв

Мобільний застосунок для водіїв може бути розроблений для платформ iOS та Android. Він буде зберігати персональну інформацію кожного водія, таку як ім'я, фотографія, контактні дані, інформація про транспортний засіб тощо. Застосунок також може мати функціонал для взаємодії з пасажирами, наприклад, можливість надсилати повідомлення про затримку або швидку відправку на зупинці, якщо багато людей чекають.

Висновок до розділу 4

У четвертому розділі кваліфікаційної роботи магістра було налаштовано та протестовано систему визначення кількості людей на зупинці транспорту шляхом обробки зображень з IP-камер.

У ході цього дослідження було проведено ретельне тестування та аналіз двох компонентів для веб-додатку. Перший компонент відповідає за відображення даних про камери спостереження, у той час як другий - за відображення картографічних даних та інформації про камери на мапі.

У результаті тестування скрипту було підтверджено його успішну роботу на різних типах зображень, включаючи звичайні та з нічним режимом. Отримані дані були збережені у базі даних для подальшого використання через інтерфейс Dashboard.

У кінці розділу було описано перспективи розвитку проєкту.

ВИСНОВКИ

У цій кваліфікаційній роботі була висвітлена актуальна проблема планування подачі міського транспорту, і вона була розв'язана через розробку системи, яка використовує високі технології та інфраструктуру «Розумного міста». Мета роботи полягала в автоматизації моніторингу та планування подачі транспорту на основі даних, отриманих із IP-камер.

Для досягнення цієї мети були вирішені конкретні завдання, такі як дослідження об'єктної та предметної областей, розробка бази даних для зберігання інформації, створення скрипту для аналізу зображень, розробка користувацького інтерфейсу та тестування застосунку.

Практична цінність роботи полягає в тому, що вона відкриває нові можливості для поліпшення системи планування подачі міського транспорту. Використання високих технологій та інфраструктури «Розумного міста» дозволяє забезпечити ефективний моніторинг та аналіз інформації на транспортних зупинках, що в свою чергу допомагає оптимізувати рух транспорту та адаптувати його до потреб пасажирів.

Застосування IP-камер для збору даних про кількість пасажирів на різних зупинках важливе для підвищення якості обслуговування громадян. Це дозволяє адаптувати розклади та ресурси транспорту в реальному часі, що робить подорожі більш комфортними та сприяє покращенню загального досвіду громадян.

Отже, розроблена система є важливим кроком у напрямку вдосконалення транспортної системи міста, а використання новітніх технологій відкриває шлях до ще більш ефективного та інноваційного міського планування та обслуговування громадян.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Пастушок І. А., Журавська І. М. Автоматизація планування подачі міського транспорту за даними IP відеоспостереження системи «розумне місто». *Інформаційні технології та інженерія* : тези доп. Всеукр. наук.-практ. конф. Миколаїв, 31 січ. – 02 лют. 2024 р. Миколаїв : Чорном. нац. ун-т ім. Петра Могили, 2024. С. 92–93.

2. Azure AI Vision : вебсайт. URL: https://azure.microsoft.com/en-us/products/ai-services/ai-vision (дата звернення: 12.12.2023).

3. EasyWay : вебсайт. URL: https://www.eway.in.ua/ua/cities/mykolaiv (дата звернення: 12.12.2023).

4. Amazon Rekognitio : вебсайт. URL: https://aws.amazon.com/rekognition/?nc1=h_ls (дата звернення: 12.12.2023).

5. Алгоритм Віоли-Джонса : вебсайт. URL: https://towardsdatascience.com/understanding-face-detection-with-the-viola-jones-object-detection-framework-c55cc2a9da14 (дата звернення 27.01.2024).

6. Model Builder : вебсайт. URL: http://surl.li/pvrnu (дата звернення: 27.01.2024).

7. WebStorm: вебсайт. URL: https://www.jetbrains.com/webstorm/ (дата звернення 27.01.2025).

8. VueJS : вебсайт. URL: https://stfalcon.com/ru/blog/post/vue-js-guide-to-tech (дата звернення 27.01.2024).

9. Vue-Router : вебсайт. URL: https://rukovodstvo.net/posts/id_1178/ (дата звернення 27.01.2024).

10. Vuetify : вебсайт. URL: https://vuetifyjs.com/en/ (дата звернення 27.01.2024).

11. Firebase : вебсайт. URL: https://firebase.google.com/ (дата звернення 27.01.2024).

12. PyCharm : вебсайт. URL: https://www.jetbrains.com/pycharm/ (дата звернення 27.01.2024).

13. Python : вебсайт. URL: https://www.jetbrains.com/pycharm/ (дата звернення 27.01.2024).

14. Чому розумні міста прискорять перехід систем відеоспостереження у хмару? *Worldvision* : інтернет-магазин систем безпеки. Опубл. 28.01.2022. URL: https://worldvision.com.ua/chomu-rozumni-mista-priskoryat-perekhid-sistem-videosposterezhennya-u-khmaru/ (дата звернення: 27.01.2024).

15. Мілякіна Т., Самойленко Г. «Безпечне місто». У Миколаєві впроваджують нові етапи проєкту. *Суспільне. Новини* : інтернет-видання. Опубл. 17.10.2023. URL: https://suspilne.media/596061-bezpecne-misto-u-mikolaevi-vprovadzuut-novi-etapi-proektu/ (дата звернення: 27.01.2024).

16. Standards by ISO/TC 268/SC 1. Smart community infrastructures. ISO: International Organization forStandardization. URL: https://www.iso.org/committee/656967/x/catalogue/ (Last accessed: 27.01.2024).

17. ITU-T Recommendations by series. Y series: Global information infrastructure, Internet protocol aspects,next-generation networks, Internet of Things and smart cities. ITU: Committed to connecting the world. URL: https://www.itu.int/itu-t/recommendations/index.aspx?ser=Y (Last accessed: 27.01.2024).

ДОДАТОК А
Довідка
про перевірку на унікальність пояснювальної записки

кваліфікаційної магістерської роботи

на тему: Автоматизація планування подачі міського транспорту за даними IP-відеоспостереження системи «Розумне місто»
студента спеціальності 123 «Комп'ютерна інженерія», групи 605м

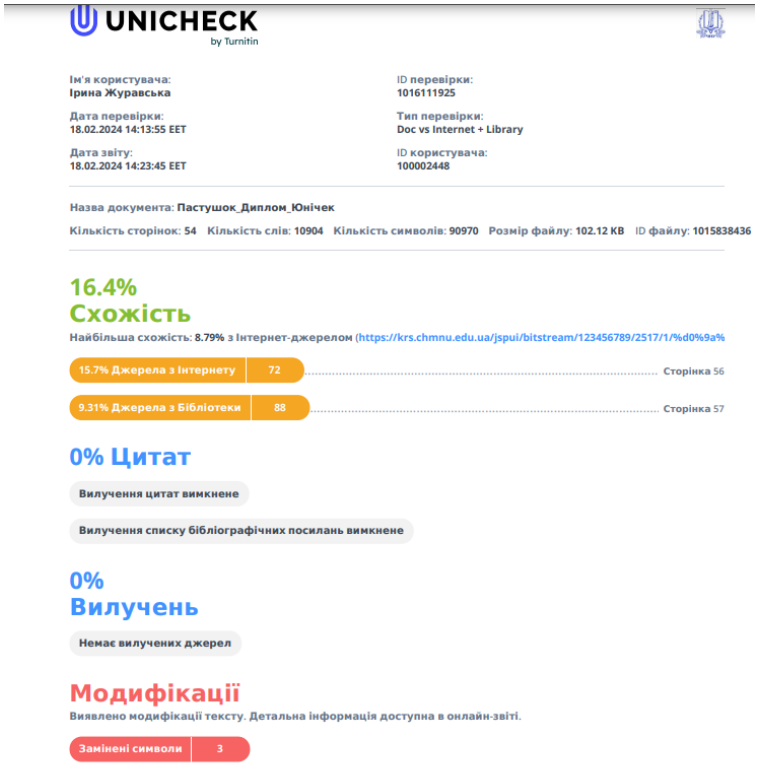
_____Пастушка Ігоря Анатолійовича_____

(прізвище, ім'я, по батькові)

прізвище, ім'я, по-батькові

Перевірку тексту здійснено сервісом: онлайн-сервіс Unicheck

Результат перевірки тексту кваліфікаційної роботи: схожість складає 16,4 %.



Студент:

_____ І. А. Пастушка
підпис ініціали, прізвище

Керівник:

Д-р техн. наук, професор

_____І. М. Журавська
підпис ініціали, прізвище

Дата: «___» _____ 2024 р.

ДОДАТОК Б
Код програми

Б.1 Python

```
# Вказівка шляху до файлу ключа служби
os.environ["GOOGLE_APPLICATION_CREDENTIALS"] = "kay.json"
# Ім'я колекції
collection_name = 'cameras'
storage_client = storage.Client()
db = firestore.Client()
exec_path = os.getcwd()
driver_path = path/to/chromedriver.exe'
# Створюємо об'єкт ChromeOptions
chrome_options = webdriver.ChromeOptions()
# Додаємо опцію для ігнорування помилок SSL
chrome_options.add_argument('--ignore-certificate-errors')
# Вимикаємо автоматичне оновлення сторінки під час завантаження
chrome_options.add_argument('--disable-prompt-on-repost')
# chrome_options.add_argument('--headless')
# Створюємо об'єкт ChromeService
chrome_service = webdriver.chrome.service.Service(driver_path)
# Створюємо екземпляр драйвера
driver = webdriver.Chrome(service=chrome_service,
options=chrome_options)
counter = 1
def upload_image_to_storage(file_path):
    """Завантаження зображення в Firebase Storage та отримання його
URL."""
    bucket_name = "smartcity-a7289.appspot.com"
    # Отримання імені файлу зі шляху
    file_name = os.path.basename(file_path)
    # Завантаження файлу в бакет
    blob = storage_client.bucket(bucket_name).blob(file_name)
    blob.upload_from_filename(file_path)
    # Отримання URL зображення
    expiration = datetime.now() + timedelta(days=365)
    image_url = blob.generate_signed_url(expiration)
    print(image_url)
    return image_url
def update_document_with_image(collection_name, document_id,
image_url, count):
    """Додати зображення до поля photos документа в Firestore."""
    try:
        # Посилання на документ
        doc_ref = db.collection(collection_name).document(document_id)
        # Створення об'єкта зображення
        image_data = {
```

```
            'id': datetime.now().strftime("%Y%m%d%H%M%S%f"),
            'url': image_url,
            'count': count,
            'createdAt': datetime.now()
        }
        # Отримання поточного стану поля photos
        doc = doc_ref.get()
        photos = doc.to_dict().get('photos', [])
        # Додавання нового об'єкта зображення до масиву
        photos.append(image_data)
        # Оновлення документа
        doc_ref.update({'photos': photos})
        print(f'Зображення успішно додано до документа {document_id} у
колекції {collection_name}.')
    except Exception as e:
        print(f'Помилка при додаванні зображення до документа: {e}')
try:
    url = ""
    driver.get(url)
    time.sleep(10)
    while True:
        current_time = datetime.now().strftime("%Y%m%d_%H%M%S")
        filename = f'screenshot_{current_time}_{counter}.png'
        driver.save_screenshot(filename)
        print(f'print #${counter}')
        detection = ObjectDetection()
        detection.setModelTypeAsRetinaNet()
        detection.setModelPath(os.path.join(exec_path,
"retinanet_resnet50_fpn_coco-eeacb38b.pth"))
        detection.loadModel()
        custom = detection.CustomObjects(person=True)
        list = detection.detectObjectsFromImage(
            custom_objects=custom,
            display_object_name=False,
            display_percentage_probability=False,
            input_image=(os.path.join(exec_path, filename)),
            output_image_path=(os.path.join(exec_path,
"output_ai.jpg")))
        people_count = len(list)
        # Шлях до файлу зображення
        image_path = "output_ai.jpg"
        image_url = upload_image_to_storage(image_path)
        update_document_with_image(collection_name, "1", image_url,
people_count)
        counter += 1
        time.sleep(15)
finally:
    driver.quit()
```

Б.2 JS

```
import Vue from 'vue'
import App from './App.vue'
import vuetify from './plugins/vuetify'
import router from '@/routes/router';
import VueRouter from 'vue-router';
Vue.config.productionTip = false
Vue.use(VueRouter);
new Vue({
  vuetify,
  router,
  render: h => h(App)
}).$mount('#app')
<script>
import PieChart from "@/components/Chart/PieChart.vue";
import DoughnutChart from "@/components/Chart/DoughnutChart.vue";
import CameraModel from "@/components/modal/CameraModel.vue";
import firebase from '../../fb.js';
export default {
  name: "HomePage",
  components: {CameraModel, DoughnutChart, PieChart},
  data () {
    return {
      loadingTable: true,
      showModal: false,
      districtsCamerasCount: {
        Zavodskoy: 0,
        Central: 0,
        Ingulsky: 0,
        Korabelny: 0
      },
      cameraId: null,
      search: '',
      headers: [
        {
          text: 'Id',
          align: 'start',
          sortable: false,
          value: 'id',
        },
        { text: 'Model', value: 'model', sortable: false },
        { text: 'Address', value: 'address', sortable: false },
        { text: 'Status', value: 'status', sortable: false },
      ],
      cameras: [
      ],
    }
```

```
  },
  created() {
    const db = firebase.firestore();
    const camerasCollection = db.collection('cameras');
    this.loadingTable = true
    camerasCollection.get().then((querySnapshot) => {
      querySnapshot.forEach((doc) => {
        this.cameras.push(doc.data());
        this.cameras.forEach(camera => {
          switch (camera.district) {
            case 0:
              this.districtsCamerasCount.Zavodskoy++;
              break;
            case 1:
              this.districtsCamerasCount.Central++;
              break;
            case 2:
              this.districtsCamerasCount.Ingulsky++;
              break;
            case 3:
              this.districtsCamerasCount.Korabelny++;
              break;
            default:
              break;
          }
        });
        this.loadingTable = false
        console.log('CAMERA', this.cameras)
      });
    }).catch((error) => {
      console.log("Ошибка при получении данных:", error);
    });
  },
  computed: {
    countCamerasIsOnline() {
      return this.cameras.filter(e => e.status).length
    },
    countCamerasIsOffline() {
      return this.cameras.filter(e => !e.status).length
    }
  },
  methods: {
    openModal(camera) {
      this.cameraId = camera.id
      this.showModal = true
    },
    closeModal() {
      this.showModal = false
    }
  }
}
</script>
```

ДОДАТОК В
Матеріали апробації роботи

В.1 Всеукраїнська науково-практична конференція «Інформаційні технології та інженерія»

Міністерство освіти і науки України
Чорноморський національний університет
імені Петра Могили

PETRO MOHYLA BLACK SEA NATIONAL UNIVERSITY
COMPUTER SCIENCE FACULTY
Est. 1996

«Інформаційні технології та інженерія»

Всеукраїнська науково-практична конференція молодих вчених, аспірантів і студентів

ТЕЗИ

31 січня – 2 лютого 2024 року

Миколаїв – 2024

ЗМІСТ

УДК 004.89:338.47

Пастушок І. А., Журавська І. М.
Чорноморський національний університет ім. Петра Могили, м. Миколаїв, Україна

АВТОМАТИЗАЦІЯ ПЛАНУВАННЯ ПОДАЧІ МІСЬКОГО ТРАНСПОРТУ ЗА ДАНИМИ IP ВІДЕОСПОСТЕРЕЖЕННЯ СИСТЕМИ «РОЗУМНЕ МІСТО»

Однією з ключових проблем, що виникають у контексті розвитку сучасних міст і підвищення якості життя громадян, є забезпечення оптимального та ефективного планування подачі міського транспорту.

Поєднання високих технологій та інфраструктури «Розумного міста» відкриває нові можливості для автоматизації цього процесу та покращення якості транспортного обслуговування громадян.

Система для розпізнання та підрахунку кількості людей на зображення може бути розроблена за допомогою Python та бібліотеки OpenCV. Після цього дані використовуються для аналізу та розроблення плану автоматизації подачі міського транспорту.

Підрахунок людей на зображеннях є корисним завданням, оскільки він широко використовується для громадської безпеки, планування людей у надзвичайних ситуаціях, інтелектуального потоку натовпу та з багатьох інших причин. Згідно з сучасними методами аналізу зображень із зазначеною метою є побудування гістограми зображення з наступним використанням алгоритму Кітлера для визначення оптимального порогу бінаризації зображення [1]. Однак, на переповнених зображеннях зі збільшеним щільності людей об'єкти частково оточують та перекривають один одного (рис. 1).

Рисунок 1 – Фото людей на зупинці транспорту з частковим перекриттям об'єктів одного іншим

В такому разі підрахунок об'єктів у зображеннях не має практичного сенсу, оскільки це займає багато часу та ніколи не дає точних результатів для щільних зображень. Ця проблема оклюзії об'єктів обмежує можливість підрахунку натовпу будь-якої традиційної моделі комп'ютерного зору.

Щоб подолати цю проблему, доцільно побудувати модель лінійної регресії згорткової нейронної мережі з динамічними ядрами (англ. Dynamic Kernel Convolution Neural Network-Linear Regression, DKCNN-LR) для підрахунку точної кількості людей у кадрах зображення, навіть якщо натовп дуже щільний і виникає проблема оклюзії [2]. Запропонована модель працює у два етапи: спочатку модель DKCNN-LR використовує шари згортки таким чином, що вага ядра кожного наступного шару становить половину ваги попереднього шару згортки. Перші три шари з важкими ядрами ідентифікують об'єкти далеких областей камери (низького рівня), а наступні шари з ядрами меншої ваги («легкими ядрами») допомагають ідентифікувати об'єкти поблизу камери (високого рівня). По-друге, модель лінійної регресії використовується для виконання параметричної регресії між фактичною кількістю людей (точні значення) та оціненою кількістю (прогнозовані значення).

Ефективність побудованої моделі перевірена на трьох наборах даних різної якості. Оцінка точності проводилась за показниками середня квадратична помилка (англ. Mean Squared Error, MSE), середня абсолютна помилка (англ. Mean Absolute Error, MAE), коефіцієнт детермінації R^2 [3].

Запропонований підхід дозволяє на підставі зображень, отриманих з комп'ютерної мережі IP-камер системи «Розумне місто», подавати міський транспорт відповідної місткості для своєчасного розвантаження зупинок та попередження скупченості людей у часи пік.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Буренко В. О. Аналіз наповненості зупинок пасажирського транспорту за допомогою алгоритмів обробки зображень з IP-камер «Розумного міста». *Вісник Хмельницького національного університету. Технічні науки.* 2023. Т.1, № 5 (325). С. 49-52. DOI: 10.31891/2307-5732-2023-325-5-49-52.
2. Tomar A., Kumar S., Pant B., Tiwari U. Dynamic Kernel CNN-LR model for people counting. *Applied Intelligence.* 2022. Vol. 52(3). P. 1-16. DOI: 10.1007/s10489-021-02375-6.
3. Pardoe I. Applied regression modeling. Wiley, 2020. 336 p.