КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

ФАКУЛЬТЕТ РАДІОФІЗИКИ, ЕЛЕКТРОНІКИ ТА КОМП'ЮТЕРНИХ СИСТЕМ

Кафедра комп'ютерної інженерії

До захисту допущено: «На правах рукопису»
Завідувач кафедри _____ Юрій Бойко
« _ » _____ 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему:

«КОМП'ЮТЕРНЕ РОЗПІЗНАВАННЯ МАРШРУТУ АВТОМОБІЛІВ»

Виконав:
студент 4-го курсу бакалаврату
денної форми навчання
спеціальності 123 Комп'ютерна інженерія
ОНП «_____»
Вадим Масляний _____

Науковий керівник:
кандидат технічних наук, доцент
Олександр Самощенко _____

Рецензент:
_____

Засвідчую, що у цій бакалаврській роботі
немає запозичень з праць інших авторів без
відповідних посилань
Студент _____

Робота допущена до захисту в ЕК рішенням кафедри _____
від «__»_____ 2023 р., протокол № __.

Завідувач кафедри _____,
кандидат фізико-математичних наук, доцент
Бойко Юрій Володимирович

(підпис)

Київ – 2023

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра містить 55 стор., 18 рис., 3 таблиці, 3 додатки, 13 використаних інформаційних джерел.

Об'єкт дослідження – способи побудови маршруту автомобіля за результатами фіксації камерами відеоспостереження.

Мета роботи - створення програми, для відтворення маршрут автомобіля по місту(при умові потрапляння автомобіля на камери відеофіксації у декількох різних місцях одного міста).

Робота не передбачає створення програми для фіксації номерних знаків, проте в ній описані методи розпізнавання символів та знаків.

В роботі проведений аналіз мов програмування та баз даних, їх порівняння та переваги. Реалізована програма, що дозволяє будувати маршрут автомобіля за його фіксацією на камерах відеоспостереження.

НЕЙРОННІ МЕРЕЖІ, PYTHON, БІБЛІОТЕКИ МОВИ ПРОГРАМУВАННЯ PYTHON, FOLIUM, MYSQL.

ЗМІСТ

ВСТУП

Інформаційні технології стають все більш важливими для розвитку та покращення різноманітних галузей людської діяльності, включаючи транспорт. Одним із напрямків, які знаходяться в активному розвитку, є системи відеофіксації руху транспорту, які дозволяють автоматично зафіксувати номерні знаки автомобілів на вулицях міст.

Метою дипломної роботи є створення програмного забезпечення для збору та відображення маршрутів автомобілів на основі номерних знаків, знятих з камер відеофіксації. Завдяки цій програмі, користувачі матимуть можливість переглядати маршрути транспортних засобів, які були зафіксовані на камерах в різних точках міста, що дозволить використовувати цю інформацію для аналізу транспортного потоку та вдосконалення дорожньої інфраструктури.

Для досягнення цієї мети розроблено базу даних для збереження зібраної інформації. Також створено програму, що дозволяє переглядати маршрути транспортних засобів на карті.

В результаті виконання дипломної роботи створено програму, яка дозволяє ефективно вести облік руху транспорту у місті за допомогою камер відеофіксації. Розроблена програма здатна заносити номерні знаки автомобілів, що зафіксовані камерами, в базу даних для подальшого аналізу. Також програма може надавати можливість відтворення маршруту руху автомобіля в межах міста, якщо його номерний знак вже був зафіксований на декількох камерах відеофіксації.

Робота є актуальною, оскільки в сучасних умовах велике значення набуває контроль за рухом транспорту в місті з метою покращення безпеки дорожнього руху та екологічної ситуації в місті. Розробка подібної програми може сприяти збільшенню ефективності діяльності відповідних служб, зокрема поліції, дорожньої служби, екологічної інспекції тощо. Також програма може

бути корисною для приватних охоронних служб, які займаються контролем руху на території підприємств чи інших закритих об'єктів.

У процесі розробки програми використані сучасні технології програмування та баз даних, зокрема мова Python та система управління базами даних MySQL. Також використані спеціальні бібліотеки для візуалізації геоданих.

Розробка програми для обліку руху транспорту у місті з використанням камер відеофіксації є важливим і перспективним напрямком в сучасній технологічній та соціальній сфері.

РОЗДІЛ 1
ОСНОВНІ ВІДОМОСТІ ПРО РОЗПІЗНАВАННЯ АВТОМОБІЛЬНИХ ЗНАКІВ

1.1 Основні методи розпізнавання номерних знаків

Розпізнавання автомобільних номерів є важливою задачею в галузі комп'ютерного зору. Її можна використовувати в різних областях, таких як автоматизоване відеоспостереження, автоматична оплата проїзду, електронний контроль доступу та інше. У цьому розділі розглянемо основні відомості про розпізнавання автомобільних номерів.

Завдання розпізнавання автомобільних номерів полягає у знаходженні номера на фотографії або відео та перетворенні його на текстовий формат. Зазвичай, номери містять буквено-цифрову комбінацію та можуть мати різну кількість символів залежно від країни та регіону.

Зазвичай, завдання розпізнавання автомобільних номерів складається з таких етапів:

- Пошук номера: на вхідному зображенні потрібно знайти номер та виділити характерну область. Для цього можуть використовуватися різні методи, такі як виявлення контуру за допомогою алгоритму Кенні, сегментація зображення за кольором або за допомогою нейронних мереж.
- Вирізання номера: після знаходження номера його потрібно вирізати з вхідного зображення та перетворити на чорно-біле зображення, що дозволяє спростити подальший аналіз.
- Сегментація символів: на чорно-білому зображенні номера потрібно виділити окремі символи. Це може бути складним завданням через різний розмір та форму символів, а також можливі шуми на зображенні. Для цього можуть використовуватися алгоритми обробки зображень, які допомагають виділити номерний знак з фону та підготувати його для подальшого розпізнавання[1].

Одним із основних методів розпізнавання номерних знаків є метод шаблонів. В цьому методі створюється набір шаблонів для кожного символу, який потім порівнюється з частинами зображення для визначення символу. Цей метод добре працює для номерних знаків з однаковим шрифтом та розміром символів.

Інший метод - це метод нейронних мереж. В цьому методі створюється модель нейронної мережі, яка навчається розпізнавати символи на зображеннях. Нейронна мережа здатна розпізнавати номерні знаки з різними шрифтами та розмірами символів.

Таблиця 1

Таблиця порівняння методів шаблонів та нейронних мереж:

Особливості	**Метод шаблонів**	**Нейронні мережі**
Розмір даних для навчання	Великий	Великий
Потрібна підготовка	Так	Так
Стійкість до шумів	Низько	Висока
Відповідність зображенню	Висока	Висока
Результат роботи	Залежить від якості шаблонів	Залежить від якості моделі
Навчання/перетворення моделі	Не потрібно	Потрібно
Швидкість роботи	Швидкий	Повільний

Також можуть використовуватися методи машинного навчання, зокрема метод опорних векторів та метод k-найближчих сусідів. В методах модель

навчається на великій кількості зображень з номерними знаками та символами, що дозволяє покращити точність розпізнавання.

Інші методи включають в себе геометричні методи розпізнавання, які використовують геометричні ознаки символів для розпізнавання, та статистичні методи, які використовують статистичні дані для розпізнавання символів.

Розпізнавання номерних знаків має велике застосування в автоматизованих системах контролю швидкості руху, паркування, охорони, дорожньої безпеки та інших галузях. Наприклад, системи автоматичної фіксації порушень дорожнього руху здатні розпізнавати номерні знаки та автоматично відправляти штрафи водіям, які порушують правила дорожнього руху.

Основні етапи процесу розпізнавання номерних знаків можна розділити на декілька кроків:

- Передпроцесінг - включає попередню обробку зображення, яка включає фільтрацію шуму, вирівнювання геометричних спотворень та видалення фону.
- Сегментація - процес виділення окремих символів на зображенні номерного знака. Він може бути виконаний за допомогою алгоритмів, що базуються на пороговій обробці, розмитті зображення, відніманні фону та інших методах.
- Витягнення ознак - етап включає в себе виділення ключових рис зображення символів, таких як форма, розмір, контрастність та інші параметри.
- Класифікація - останній етап процесу розпізнавання, де виконується ідентифікація символів та перетворення у текстовий формат. Для цього можуть використовуватися методи шаблонної відповідності, статистичні методи та методи машинного навчання.

Одним з головних викликів при розпізнаванні номерних знаків є їх варіативність - номерні знаки можуть відрізнятися за кольором, формою та

символами. Тому важливо використовувати різноманітні методи та алгоритми для досягнення високої точності розпізнавання.

Крім того, для роботи системи розпізнавання автомобільних номерних знаків необхідно враховувати такі чинники, як варіанти освітлення, швидкість руху автомобіля, наявність шумів та інші фактори, що можуть вплинути на якість отриманого зображення номерного знаку. Для забезпечення стабільної роботи системи розпізнавання можна використовувати різноманітні технології, які враховують ці чинники.

Один зі способів вирішення проблеми варіації освітлення полягає в застосуванні адаптивного порогового фільтрування. Метод дозволяє автоматично змінювати поріг, залежно від освітлення зображення, для підвищення якості розпізнавання. Крім того, для зменшення впливу шумів на зображення можуть використовуватися методи морфологічної обробки, які дозволяють видалити невиправдані деталі зображення.

Для підвищення швидкості розпізнавання номерних знаків можуть використовуватися методи зменшення кількості операцій. Наприклад, можна використовувати попередній аналіз зображення та відкинути всі регіони, що не містять номерні знаки, для скорочення часу обробки. Також можна застосовувати методи прискорення обробки зображень за допомогою графічних процесорів (GPU), які дозволяють збільшити швидкість розпізнавання в декілька разів порівняно зі звичайними центральними процесорами.

1.2 Розпізнавання за допомогою нейронних мереж

У сучасних системах розпізнавання номерних знаків з частотою використовують методи глибокого навчання та нейронні мережі. Ці методи дозволяють підвищити точність розпізнавання шляхом автоматичного виявлення важливих ознак на зображенні. Системи можуть бути навчені на великій кількості зображень, що дозволяє розпізнавати номерні знаки в різних умовах, включаючи зміну освітлення, зміну ракурсу та інші чинники.

Нейронні мережі використовуються в якості моделі для розпізнавання номерних знаків. Нейронна мережа - математична модель, що складається з нейронів, які здатні обробляти вхідні дані та виконувати складні обчислення. Для розпізнавання номерних знаків нейронні мережі використовуються для автоматичного визначення важливих ознак на зображенні, таких як контури та риси символів[2].

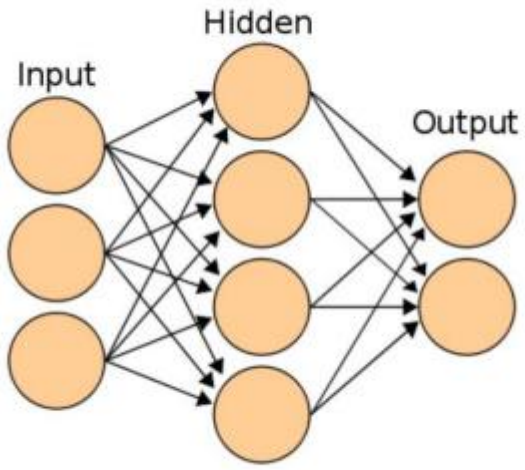

Рисунок 1.1 Модель нейронної мережі

Системи розпізнавання номерних знаків, які використовують нейронні мережі, зазвичай працюють у двох етапах: виявлення номерного знака та розпізнавання символів на номерному знаку. Перший етап полягає в виявленні номерного знака на зображенні та його вирізанні. Другий етап - розпізнавання символів на вирізаному номерному знаку.

Один з найбільш поширених методів розпізнавання номерних знаків на основі нейронних мереж - згорткова нейронна мережа (Convolutional Neural Network, CNN). Метод використовується для виявлення номерних знаків та розпізнавання символів на них. Згорткова нейронна мережа складається з кількох шарів, включаючи згорткові, підвибіркові та пов'язані шари, які допомагають виявляти та розпізнавати символи на номерних знаках.

Одним з викликів, які виникають при використанні нейронних мереж для розпізнавання номерних знаків, є нестабільність роботи в умовах зміни освітлення та інших факторів. Для вирішення питання доступними є методи адаптивного порогового значення та аугментування даних.

Метод адаптивного порогового значення полягає у визначенні порогового значення для кожного зображення окремо залежно від освітлення на зображенні. Це дає можливість підвищити стійкість розпізнавання до зміни освітлення та контрастності на зображенні.

Аугментування даних - це метод, який дозволяє збільшити обсяг тренувальної вибірки за допомогою штучного додавання до неї модифікованих зображень. Наприклад, можна застосовувати горизонтальне та вертикальне перевертання, зміну кольору та яскравості, відбиття та інші трансформації зображень. Цей метод дозволяє підвищити стійкість нейронної мережі до різноманітних ускладнень на зображеннях та покращити точність розпізнавання номерних знаків.

Одним з інших викликів є швидкість обробки в реальному часі, особливо при використанні в автоматизованих системах контролю дорожнього руху. Для розв'язання цієї проблеми використовують спеціалізовані апаратні рішення, такі як графічні процесори та тензорні процесори, які забезпечують високу швидкість обробки даних. Також можуть використовуватися методи оптимізації та прискорення нейронних мереж, такі як обчислення з використанням бінарних ваг, зменшення кількості шарів мережі та інші.

Узагалі, системи розпізнавання номерних знаків знаходять все більше застосувань у різних сферах, де вони можуть забезпечити автоматизацію та поліпшення різноманітних процесів. Наприклад, в галузі контролю швидкості руху системи автоматичної фіксації порушень дорожнього руху здатні розпізнавати номерні знаки та визначати швидкість руху автомобіля. У сфері охорони і безпеки такі системи можуть допомагати в ідентифікації транспортних засобів, які перебувають у режимі розшуку. В галузі паркування

системи розпізнавання номерних знаків можуть використовуватись для контролю доступу та визначення місць паркування.

Також системи розпізнавання номерних знаків знайшли своє застосування у транспортному бізнесі. Наприклад, такі системи можуть використовуватись для автоматизації процесу реєстрації автомобілів на дорозі, що дозволить зменшити час очікування та зробити процес більш ефективним. Також системи розпізнавання номерних знаків можуть бути використані для контролю вантажних перевезень та забезпечення безпеки на дорогах[3].

У банківському секторі системи розпізнавання номерних знаків можуть допомагати в автоматизації процесу оцінки автомобілів, що є важливим для прийняття рішень щодо надання кредитів та інших фінансових послуг.

Системи розпізнавання номерних знаків стають все більш популярними в різних галузях, де вони можуть забезпечити автоматизацію та покращення ефективності різних процесів. Що до майбутнього, можна очікувати подальшого розвитку цих систем та їх використання в нових сферах.

1.3 Локалізація, нормалізація, сегментація

Система автоматичного розпізнавання автомобільних номерів може бути реалізована як програмний або апаратно-програмний комплекс, який виконує алгоритм автоматичного розпізнавання номерних знаків з метою автоматизації введення даних та їх подальшої обробки.

Процес автоматичного розпізнавання автомобільних номерів включає такі етапи:

- Локалізація: ідентифікація та виокремлення області зображення, де знаходиться номерний знак.
- Нормалізація: приведення розмірів та орієнтації номерного знака до стандартизованого вигляду.
- Сегментація: розділення номерного знака на окремі символи.

- Розпізнавання: використання алгоритмів та моделей для ідентифікації розпізнаних символів.
- Синтаксичний аналіз: перевірка послідовності символів номерного знака відповідно до правил формату.

Кожен з цих етапів виконує певні операції для досягнення точного та надійного розпізнавання номерних знаків автомобілів.

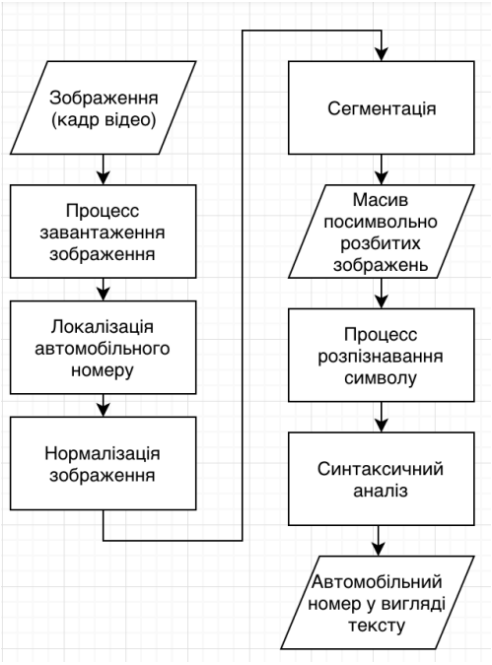

Рисунок 1.2 Алгоритм роботи системи розпізнавання автомобільного номеру

РОЗДІЛ 2
ВИБІР ІНСТРУМЕНТІВ ДЛЯ РЕАЛІЗАЦІЇ ПРОГРАМИ

2.1 Огляд мов програмування

JavaScript - мова програмування, що використовується для створення динамічних веб-сайтів та веб-додатків. JavaScript є однією з найпоширеніших мов програмування, має велику кількість бібліотек та фреймворків, які допомагають розробникам створювати високоякісні додатки з меншими затратами часу.

Переваги:

- Крос-браузерність: JavaScript можна використовувати на будь-якому браузері, що дозволяє розробникам створювати веб-сайти та веб-додатки, що працюють на будь-яких платформах.
- Висока швидкість виконання: JavaScript є відносно швидкою мовою програмування, що дозволяє створювати ефективні та швидкі додатки.
- Широке сприйняття: JavaScript є дуже популярною мовою програмування, що означає, що є багато ресурсів та підтримки для розробників.
- Є простим для вивчення: JavaScript має простий синтаксис, що дозволяє новачкам швидко освоювати мову та починати розробляти веб-додатки.

Недоліки:

- Вразливість: JavaScript може бути вразливим до атак, таких як XSS (Cross-Site Scripting) та CSRF (Cross-Site Request Forgery), що може призвести до викрадення конфіденційної інформації.
- Строгість типів: JavaScript є м'яко типізованою мовою програмування, що може призвести до помилок через неправильне використання типів даних.
- Обмеження безпеки: JavaScript не дозволяє робити деякі операції через обмеження безпеки, наприклад, взаємодія з файловою системою.

- Недостатня підтримка стандартів: JavaScript має багато різних реалізацій та версій, що може створювати проблеми зі сумісністю між різними браузерами та платформами[4].

Python - це високорівнева, інтерпретована мова програмування, яка була розроблена у кінці 1980-х років Гвідо ван Россумом. Python має простий та зрозумілий синтаксис, що дозволяє швидко розробляти якісний код з меншою кількістю рядків порівняно з іншими мовами програмування.

Основні переваги Python:

- Простота вивчення та використання: Python має простий та зрозумілий синтаксис, що робить його легким для вивчення та використання.
- Пристосованість: Python можна використовувати на різних платформах, таких як Windows, Linux та macOS.
- Велика кількість бібліотек та фреймворків: Python має велику кількість стандартних бібліотек та пакетів, а також десятки тисяч сторонніх бібліотек та фреймворків, що дозволяє розробляти швидко та ефективно.
- Підтримка ООП: Python підтримує об'єктно-орієнтоване програмування, що дозволяє створювати класи та об'єкти, що полегшує розробку та підтримку коду.

Основні недоліки Python:

- Повільна швидкість виконання: порівняно з деякими мовами програмування, такими як C або C++, Python може бути повільнішим у виконанні.

- Обмежені можливості для мобільної розробки: Python не має повної підтримки для розробки мобільних додатків, хоча є певні фреймворки, такі як Kivy, які дозволяють розробляти мобільні додатки на Python.
- Висока залежність від відступів: Python використовує відступи для визначення блоків коду, що може призводити до проблем з читабельністю та породжувати проблеми в програмах, які використовують різні редактори коду з різними налаштуваннями відступів.
- Обмежені можливості для малих проектів: Python може бути занадто потужним для дуже малих проектів, так як має велику кількість функціональності, яку можна не використати в малих проектах[5].

C# - це мова програмування, створена компанією Microsoft в 2000 році. Ця мова програмування є однією з найпопулярніших мов для розробки програмного забезпечення на платформі .NET. Ось декілька плюсів та мінусів використання мови програмування C#:

Переваги:

- Платформно-незалежна: C# є мовою, яка працює на платформі .NET, що дозволяє розробляти програмне забезпечення, яке працює на будь-якій операційній системі, що підтримує .NET Framework або .NET Core.
- Багато функцій та бібліотек: C# має велику кількість функцій та бібліотек, які дозволяють розробникам швидко та ефективно створювати програмне забезпечення.
- Безпека: C# має багато вбудованих механізмів безпеки, які допомагають запобігати атакам з зовнішнього середовища та забезпечують захист даних.

Недоліки:

- Висока вартість розробки: для розробки програмного забезпечення на C# потрібно використовувати середовище Visual Studio, що може бути високою вартістю для окремих розробників та компаній.
- Обмеження платформи: C# працює тільки на платформі .NET, що може бути недостатнім для деяких проектів, які мають більш широку аудиторію або використовують різні платформи[6].

C++ - це мова програмування, яка була розроблена в 1983 році як розширення мови C. C++ є однією з найпопулярніших мов програмування, особливо у веб-розробці та розробці ігор. Ось декілька плюсів та мінусів використання мови програмування C++:

Переваги:

- Висока продуктивність: C++ дозволяє розробляти дуже ефективний та швидкий код, що дозволяє використовувати його для створення високопродуктивного програмного забезпечення, такого як ігри або системи відображення.
- Широкі можливості: C++ має багато функцій та можливостей, які дозволяють розробникам створювати різноманітні типи програмного забезпечення, від малих додатків до великих систем.
- Низькорівневий доступ до апаратного забезпечення: C++ дозволяє розробникам отримати прямий доступ до апаратного забезпечення, що дозволяє більш точно контролювати процеси в системі та використовувати апаратне забезпечення більш ефективно.

Недоліки:

- Складність: C++ є однією з найскладніших мов програмування, що може призводити до складнощів у відладці та підтримці коду.

- Безпека: через прямий доступ до апаратного забезпечення та низькорівневі можливості, C++ може бути менш безпечною мовою програмування, ніж інші мови з вищим рівнем абстракції.
- Можливості на рівні апаратного забезпечення обмежені: хоча C++ надає можливості на рівні апаратного забезпечення, ці можливості обмежені архітектурою конкретного апаратного забезпечення та операційної системи, що може зробити код менш переносним між різними платформами.
- Можливості для використання пам'яті: у C++ є можливість прямого доступу до пам'яті, що може бути корисним для оптимізації програм, але водночас може призвести до серйозних проблем з пам'яттю, таких як витік пам'яті або переповнення буфера.
- Багатопоточність: C++ підтримує багатопоточність, але при цьому потрібно бути дуже обережним при роботі з пам'яттю, щоб уникнути ситуацій гонки за ресурсами та інших проблем, що пов'язані з багатопоточним програмуванням.
- Швидкодія: C++ є однією з найшвидших мов програмування, оскільки дозволяє прямий доступ до пам'яті та забезпечує більш точну контрольованість роботи з ресурсами. Проте, зі швидкодією пов'язані й деякі недоліки, такі як підвищені вимоги до ресурсів комп'ютера та складність розробки та тестування.
- Висока складність: C++ - це мова з високим рівнем складності, оскільки вона дозволяє прямий доступ до пам'яті та має багато можливостей на рівні апаратного забезпечення. Це може призвести до складнощів в розробці та тестуванні програм.
- Багатофункціональність: C++ має багато можливостей та функцій, що дозволяють розробникам створювати складні та високопродуктивні програми. Однак, ця багатофункціональність може також призвести до більшої складності коду та збільшення розміру програми[7].

Таблиця 2

Порівняння основних характеристик мов програмування

Мова програмування	Використання	Простота вивчення	Популярність	Екосистема
JavaScript	Веб-розробка	Середня	Висока	Широка
Python	Загальне	Висока	Висока	Розналужена
C#	Десктопні	Середня	Висока	Розналужена
C++	Системне	Висока	Висока	Розналужена

2.2 Огляд баз даних

SQL Server - це система управління реляційними базами даних від компанії Microsoft. Ось деякі плюси та мінуси SQL Server:

Переваги:

- Надійність: SQL Server відомий своєю надійністю та стабільністю. Він має механізми резервного копіювання та відновлення даних, що дозволяє забезпечити безпеку даних та уникнути втрати інформації.
- Потужність: SQL Server має багато потужних можливостей, таких як засоби забезпечення безпеки даних, масштабованість та підтримку різних типів даних.
- Швидкість: SQL Server забезпечує швидку роботу з даними та запитами завдяки ефективному використанню ресурсів комп'ютера та оптимізації запитів.
- Зручність: SQL Server має простий та зрозумілий інтерфейс користувача та широкий спектр інструментів для розробки та управління базами даних.

Недоліки:

- Вартість: SQL Server є комерційною продукцією, тому його вартість може бути значною, особливо для підприємств з обмеженим бюджетом.

- Системні вимоги: SQL Server потребує певної кількості ресурсів комп'ютера, що може призвести до додаткових витрат на обладнання.
- Складність: SQL Server може бути складним у налаштуванні та управлінні, що може вимагати наявності досвідченого персоналу або додаткової підготовки.
- Обмеження масштабування: SQL Server може бути обмеженим у масштабуванні, особливо для великих підприємств з високим обсягом даних та великою кількістю користувачів[8].

MySQL - це безкоштовна система управління базами даних з відкритим кодом, яка дозволяє зберігати та організовувати великі обсяги даних.

Основні особливості MySQL:

- Швидкість та продуктивність: MySQL може працювати з великим обсягом даних та забезпечувати швидкий доступ до них. Це досягається за рахунок оптимізації структури бази даних та оптимізації запитів.
- Масштабованість: MySQL може бути масштабовано для роботи з великими обсягами даних та великої кількості користувачів.
- Безпека: MySQL має різні рівні безпеки для захисту ваших даних, включаючи шифрування даних та захист від несанкціонованого доступу.
- Підтримка стандартів: MySQL дотримується стандартів SQL та є сумісним з багатьма іншими системами управління базами даних[9].
- Надійність: MySQL досить надійний та стабільний, що забезпечує стабільну роботу ваших даних та додатків, що працюють з базою даних.
- Підтримка: MySQL має широку підтримку спільноти, що забезпечує доступ до великої кількості ресурсів та допомогу в розв'язанні проблем.

Недоліки MySQL:

- Обмеження в розширенні: MySQL має деякі обмеження в розширенні, зокрема, відсутність деяких функцій, що можуть бути доступні в інших системах управління базами даних.
- Проблеми з відновленням даних: при неправильному використанні MySQL може виникнути проблема з відновленням даних, яка може призвести до втрати даних.
- Висока вимога до ресурсів: MySQL може вимагати значних ресурсів системи, особливо при обробці великих обсягів даних. Наприклад, збільшення обсягу бази даних може потребувати значних витрат на обладнання та пам'ять для зберігання даних. Також можуть виникати проблеми з продуктивністю при виконанні складних запитів, якщо не використовувати оптимізацію запитів та індексацію таблиць[10].

SQLite - це вбудована реляційна база даних, яка зберігає дані в локальному файлі, що розміщується на диску. Особливістю SQLite є те, що вона не потребує окремого сервера баз даних для роботи, тому можна використовувати безпосередньо в додатках.

Деякі з переваг SQLite:

- Просто використовувати: SQLite має простий синтаксис, що робить її досить простою у використанні.
- Портативність: SQLite може працювати на багатьох платформах, включаючи Windows, Mac та Linux.
- Невелика вимога до ресурсів: SQLite не потребує великих обсягів пам'яті та інших ресурсів, що робить її підходящою для використання на мобільних пристроях та вбудованих системах.
- Безкоштовна та відкрита: SQLite поширюється під ліцензією Public Domain, що означає, що вона безкоштовна для використання та модифікації.

Деякі з недоліків SQLite:

- Обмежена потужність: SQLite не рекомендується для великих баз даних з великою кількістю користувачів, оскільки вона може стати повільною при обробці великого обсягу даних.
- Обмежена функціональність: SQLite має обмежену підтримку деяких функцій, що доступні в інших СУБД.
- Обмежена безпека: SQLite не надає високого рівня захисту даних в порівнянні з іншими СУБД, такими як MySQL та PostgreSQL[11].

Таблиця 3

Порівняння основних характеристик баз даних

Характеристика	**SQL Server**	**MySQL**	**SQLite**
Тип ліцензії	Комерційна	Відкрита	Відкрита
Мова запитів	T-SQL	SQL	SQL
Підтримка реплікації	Так	Так	Ні
Підтримка транзакцій	Так	Так	Так
Підтримка схем	Так	Так	Ні
Обмеження цілісності	Так	Так	Частково
Підтримка індексів	Так	Так	Так
Підтримка збережених	процедур і функцій	процедур і функцій	процедур і функцій
Підтримка розподіленої	бази даних	бази даних	бази даних

2.3 Висновки до розділу

У дипломній роботі обрано мову програмування **Python** разом з **MySQL** базою даних для реалізації системи.

Python - високорівнева інтерпретована мова програмування з динамічним зв'язуванням типів, що має простий і зрозумілий синтаксис, дозволяє швидко писати програми. Python здатна вирішувати широкий спектр завдань, від обробки даних до наукових обчислень і веб-розробки.

Однією з основних переваг Python є те, що вона є дуже популярною в галузі машинного навчання та аналізу даних, завдяки чому має безліч сторонніх бібліотек, які спрощують і прискорюють процес розробки. Наприклад, для машинного навчання у Python є такі бібліотеки як TensorFlow, Keras, PyTorch, Scikit-Learn та багато інших.

Також, Python є переносним і доступним на багатьох платформах, таких як Windows, MacOS, Linux, а це означає, що нашу програму можна використовувати на різних пристроях та операційних системах.

Ще однією перевагою Python є швидкість розробки, тому що розробник може писати програми швидше, що дозволяє економити час і зусилля.

Робота відповідає характеристикам, що роблять Python однією з кращих мов програмування для її виконання. Python забезпечує зручний і швидкий розвиток, наявність безлічі бібліотек, що спрощують розробку, та можливість запускати програму на різних платформах.

Також, однією з головних причин вибору мови програмування Python є бібліотека folium.

Folium - це бібліотека для створення веб-карт та візуалізації геоданих в середовищі Python. Вона є розширенням для бібліотеки Leaflet.js, яка надає високоякісний та інтерактивний інтерфейс веб-карт. Завдяки цій бібліотеці можна створювати інтерактивні карти, на яких можна відображати

різноманітну інформацію, таку як місцезнаходження, маршрути, маркери та інші геодані.

Для використання бібліотеки folium потрібно спочатку встановити її за допомогою менеджера пакетів pip. Після цього можна імпортувати її у свій Python-код та використовувати для візуалізації геоданих. Однією з головних переваг folium є те, що вона дозволяє додавати на карту різноманітні шари та елементи, такі як маркери, лінії, круги, полігони та інші. Також, бібліотека має підтримку віджетів Jupyter Notebook, що дозволяє створювати інтерактивні карти та візуалізації прямо в ноутбуці[12].

Застосування folium може бути корисним в різних випадках, наприклад, при візуалізації результатів аналізу геоданих, відображенні розташування певних об'єктів, створенні інтерактивних карт для подорожей та інших додатків. Також, бібліотека може бути корисною для розробки геоінформаційних систем та додатків з аналітикою геоданих[13].

MySQL - це одне з найбільш популярних відкритих реляційних СУБД (систем управління базами даних), яке широко використовується в бізнесі, веб-розробці та інших галузях. MySQL дозволяє зберігати, організовувати та отримувати доступ до даних.

MySQL є ідеальним вибором для нашої роботи, оскільки ми збираємо та зберігаємо дані про маршрути руху автомобілів в реляційній базі даних. MySQL пропонує надійний та ефективний спосіб зберігання даних, які можуть бути відновлені та модифіковані. Крім того, MySQL має велику спільноту користувачів, тому для нього існує багато документації, підтримка та різноманітні інструменти для роботи з даними.

В дипломній роботі використовується Python для збору та обробки даних з камер відеофіксації, а потім ми зберігаємо ці дані в MySQL базі даних. Мову програмування Python можна використовувати для з'єднання з базою даних MySQL та взаємодіяти з даними, що зберігаються в ній. Для відображення

маршрутів на мапі ми використовуємо бібліотеку Folium, яка також дозволяє взаємодіяти з даними, що зберігаються в базі даних MySQL.

РОЗДІЛ 3
РОЗРОБЛЕННЯ ТА РЕАЛІЗАЦІЯ ПРОГРАМИ

3.1 Розробка програми

Для розробки програми необхідно імпортувати бібліотеки **folium**, **mysql.connector** та **MarkerCluster** з пакету **folium.plugins**.

folium - це бібліотека для візуалізації даних на інтерактивній картах, що базується на бібліотеці Leaflet.js. Ця бібліотека дозволяє відображати геодані, такі як точки, лінії, полігони, на інтерактивній карті.

mysql.connector - це бібліотека для з'єднання та взаємодії з базою даних MySQL. Вона дозволяє виконувати запити до бази даних, забирати та записувати дані.

MarkerCluster - це плагін для folium, що дозволяє групувати маркери на карті у кластери. Це корисно для відображення багатьох маркерів на карті, що допомагає зберігати чистоту та читабельність карти.

```
import folium
import mysql.connector
from folium.plugins import MarkerCluster
```

Рисунок 3.1 Імпорт бібліотек

Щоб працювати з мапою створена база даних cars до якої під’єднуємось вказавши значення host, user, password, database. в даному випадку ми встановлюємо з'єднання з базою даних MySQL, яка знаходиться на локальному сервері за адресою 127.0.0.1, з використанням імені користувача root та пароля Katavada324, і вибираємо базу даних з назвою cars.

```
# з'єднуємось з базою даних
mydb = mysql.connector.connect(
    host='127.0.0.1',
    user='root',
    password='Katavada324',
    database='cars'
)
```

Рисунок 3.2 З'єднання з базою даних

Наступним кроком прописується код, який відповідає за запит про номерний знак автомобіля, маршрут якого необхідно побудувати.

Створюється карта folium, яка буде використовуватися для відображення маршруту машини.

За допомогою класу MarkerCluster створюється кластер маркерів, який буде використовуватися для групування маркерів на карті.

Кластер маркерів додається до карти за допомогою методу add_to().

```
# запитуємо номерний знак машини
car_number = input("Введіть номерний знак машини: ")

# створюємо карту
m = folium.Map(location=[48.344746043429055, 33.50164576003915], zoom_start=16)

# створюємо кластер маркерів
marker_cluster = MarkerCluster().add_to(m)
```

Рисунок 3.3 Створення мапів та запиту до номерного знаку.

Необхідно створити курсор для виконання запитів до бази даних MySQL, яка була підключена раніше. Потім виконується запит до бази даних, в якому вибирається всі записи про маршрути транспортних засобів з таблиці "routes",

які належать автомобілю з заданим номерним знаком. Це здійснюється за допомогою підзапиту, який вибирає ідентифікатор автомобіля з таблиці "car_numbers" за номерним знаком, який ми ввели користувачем раніше.

Результат запиту зберігається у змінній "result". Це список з кортежів, де кожен кортеж містить координати та час місцезнаходження автомобіля. Кожний кортеж містить чотири елементи: широту (lat), довготу (lon), дату та час (date, time).

```
# витягуємо координати з бази даних
mycursor = mydb.cursor()
mycursor.execute(f"SELECT lat, lon, date, time FROM routes WHERE"
                 f" car_id=(SELECT car_id FROM car_numbers WHERE car_number='{car_number}')")
result = mycursor.fetchall()
```

Рисунок 3.4 Витягання даних з бази даних

Далі необхідно створити код, що відповідає за візуалізацію маршруту, пройденого автомобілем, на карті. Кожна точка маршруту зберігається в базі даних разом з датою та часом її проходження.

Код витягує всі точки маршруту для введеного користувачем номера автомобіля з бази даних MySQL та зберігає їх в змінну result.

Далі цикл for перебирає кожну точку та створює маркер на карті з відповідними координатами. Кожен маркер містить інформацію про дату та час проходження точки. Ці маркери додаються до кластера маркерів marker_cluster.

Наступний цикл for проходить по кожній парі сусідніх точок та додає лінію між ними до кластера маркерів. Ці лінії позначають шлях, пройдений автомобілем між кожною парою точок на маршруті. Кожна лінія має синій колір.

```
# проходимось по кожній точці та додаємо маркер на карту
for point in result:
    folium.Marker(location=[point[0], point[1]], popup=f"Дата: {point[2]}, Час: {point[3]}").add_to(marker_cluster)

# проходимось по кожній парі сусідніх точок та додаємо лінію між ними до кластера маркерів
for i in range(len(result) - 1):
    folium.PolyLine(locations=[result[i][:2], result[i + 1][:2]], color='blue').add_to(marker_cluster)
```

Рисунок 3.5 Проходження по кожній точці

В кінці необхідно саме відобразити мапу, на якій побудується маршрут. В останньому рядку коду використовується метод save(), який зберігає створену карту у вигляді HTML-файлу з ім'ям "map.html". Далі можна відкрити цей файл у будь-якому веб-браузері та переглянути результат роботи програми.

```
# відображаємо карту
m.save('map.html')
```

Рисунок 3.6 Відображення мапи

3.2 Реалізація програми.

Для прикладу обрано місто Жовті Води, автомобіль з номерним знаком AE1234BB.

Запустимо програму та введемо номерний знак:

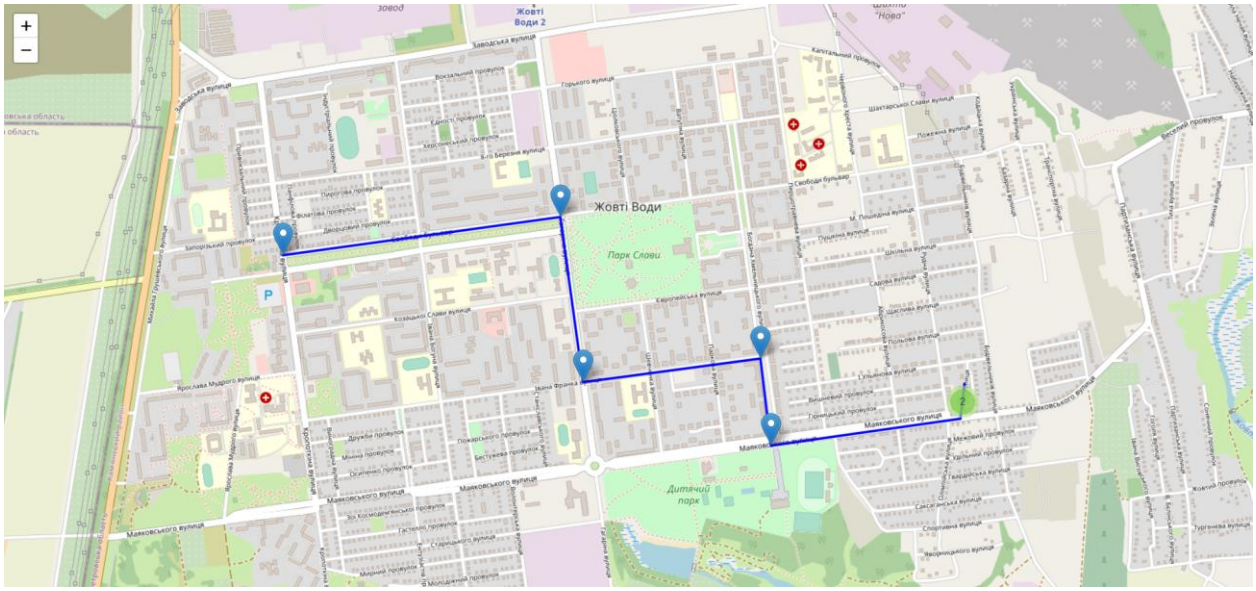

Рисунок 3.7 Повний маршрут, що пройшов автомобіль з номерним знаком AE1234BB

Як видно на малюнку, маршрут збудовано.

Для зручності можна приблизити карту колесом мишки або ж знаками + або - у лівому куті:

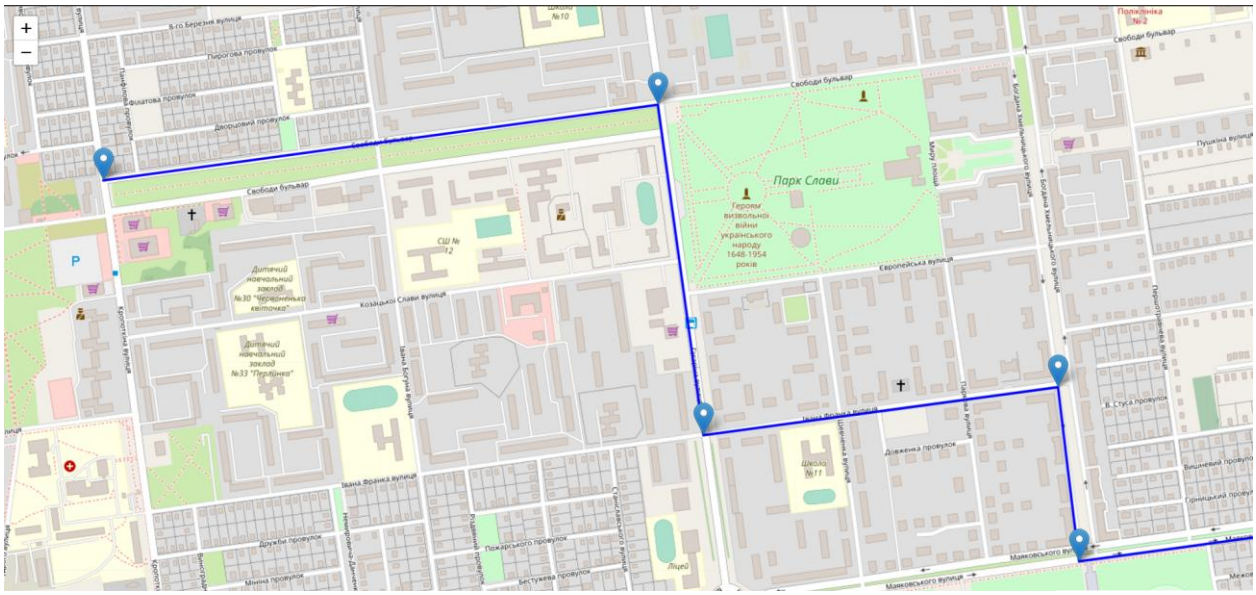

Рисунок 3.8 приближений маршрут автомобіля з номерним знаком AE1234BB

Аби дізнатись інформацію щодо того коли та о котрій кодині було зафіксовано автомобіль, необхідно натиснути на маркер камери:

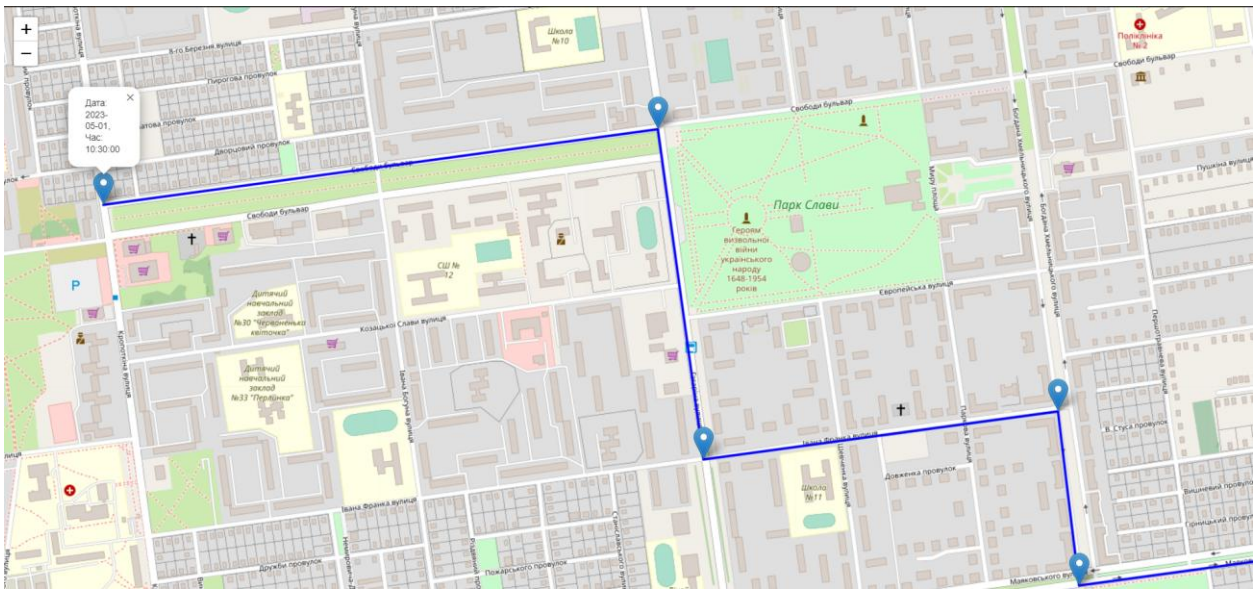

Рисунок 3.9 Демонстрація інформації про дату та час фіксації автомобіля з номерним знаком AE1234BB на першій камері

Натиснемо також на сусідній маркер, аби переконатись що інформацію внесено вірно:

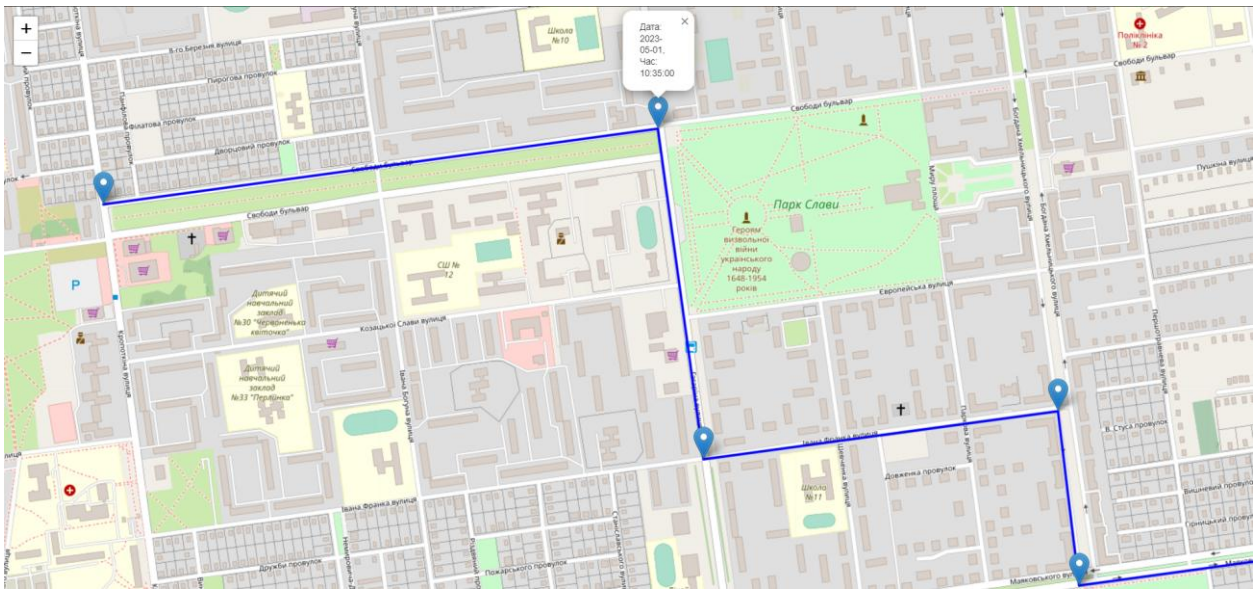

Рисунок 3.10 Демонстрація інформації про дату та час фіксації автомобіля з номерним знаком AE1234BB на другій камері

Для різноманіття результатів було проведено пошук за автомобільним номером BA2345KO:

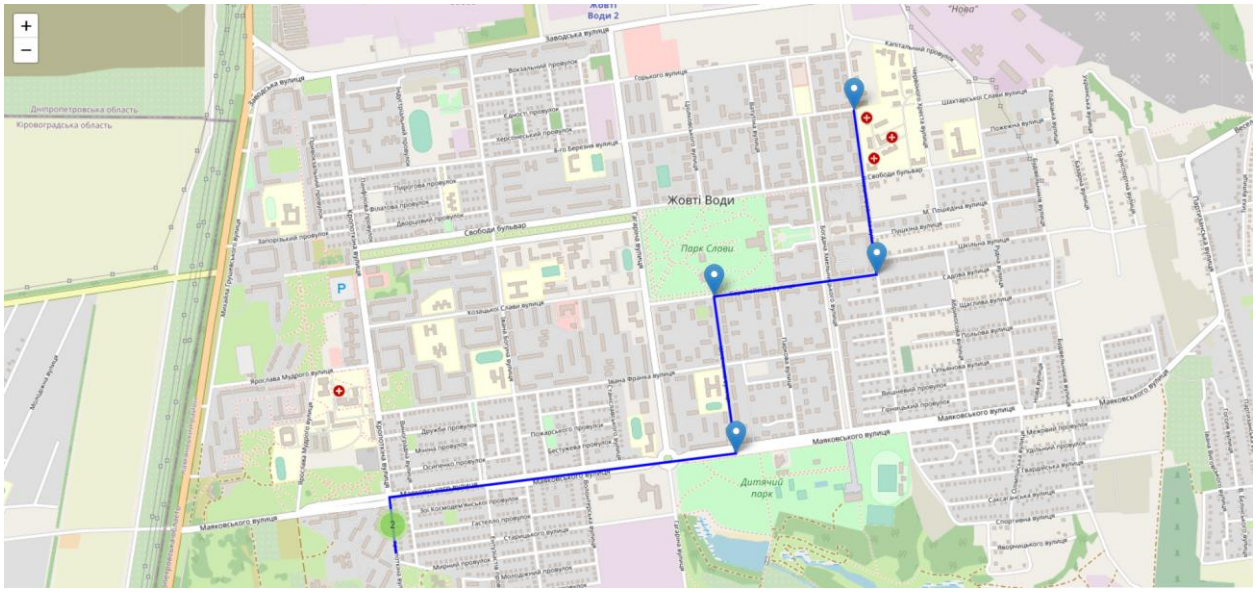

Рисунок 3.11 Повний маршрут, що пройшов автомобіль з номерним знаком BA2345KO

Для перевірки дати та часу фіксації тиснемо на сусідні маркери:

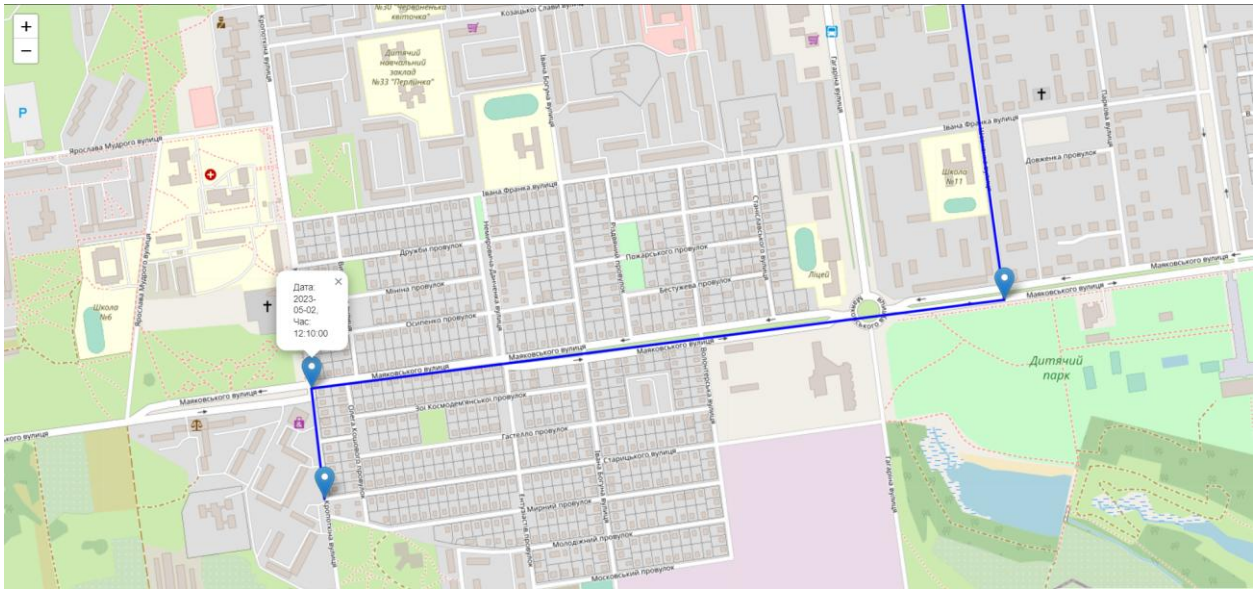

Рисунок 3.12 Демонстрація інформації про дату та час фіксації автомобіля з номерним знаком ВА2345КО на першій камері

Та відповідно на сусідній:

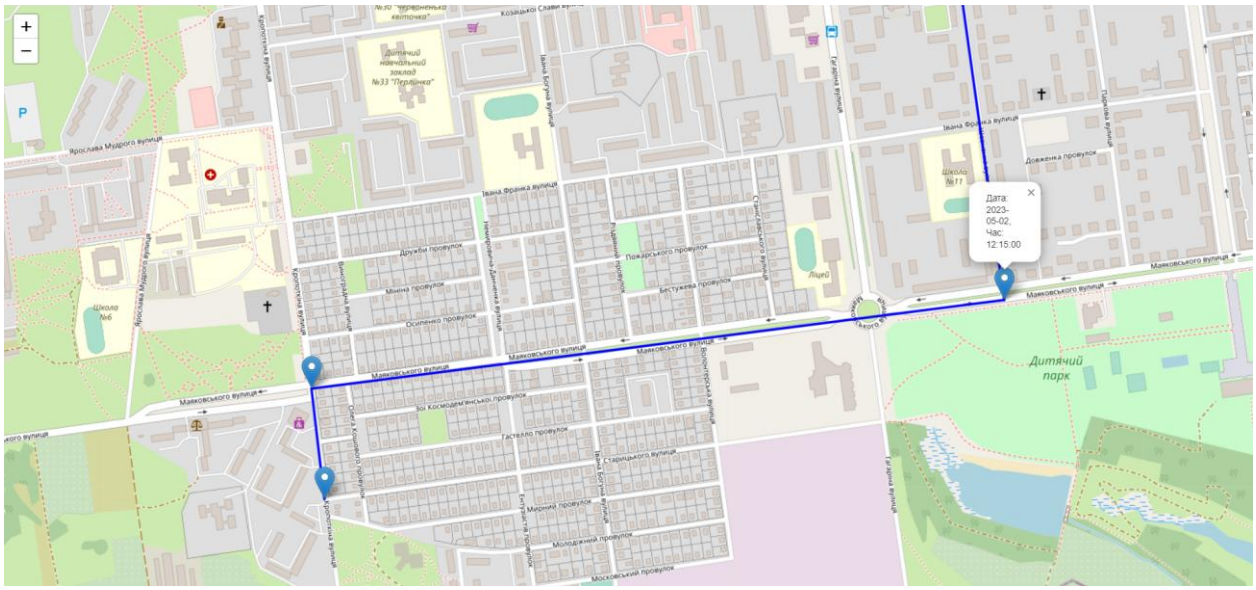

Рисунок 3.13 Демонстрація інформації про дату та час фіксації автомобіля з номерним знаком ВА2345КО на другій камері

Аналогічно побудовано маршрут для автомобіля з номерним знаком КА3456ОМ:

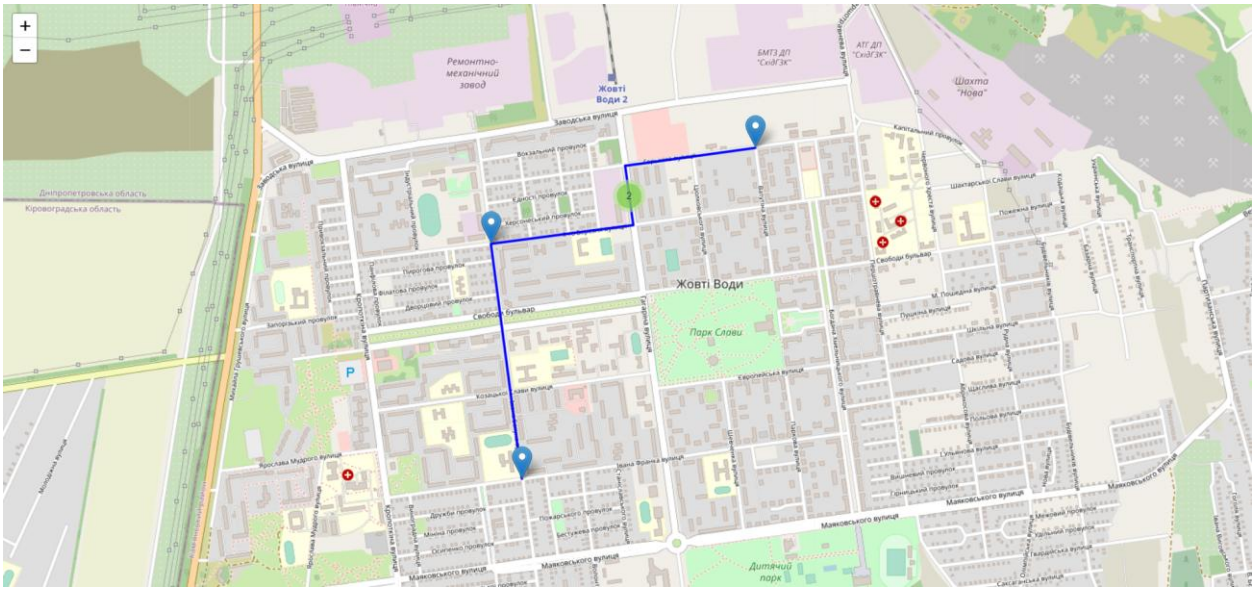

Рисунок 3.14 Повний маршрут, що пройшов автомобіль з номерним знаком КА3456ОМ

Для перевірки дати та часу фіксації тиснемо на сусідні маркери:

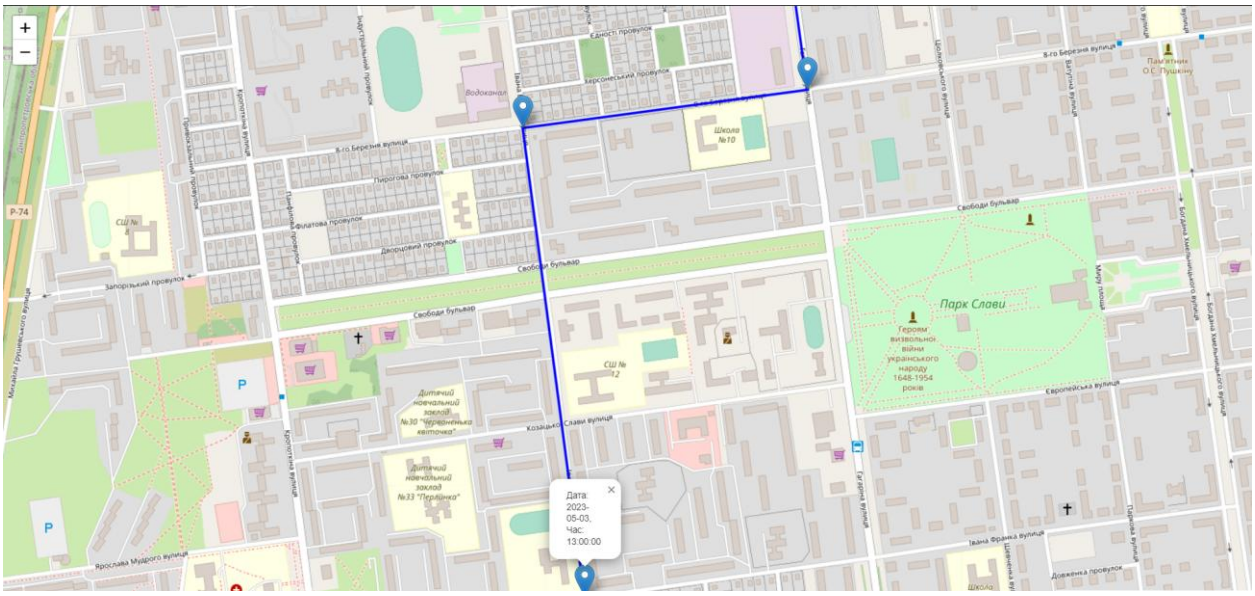

Рисунок 3.15 Демонстрація інформації про дату та час фіксації автомобіля з номерним знаком КА3456ОМ на першій камері

Та відповідно на сусідній:

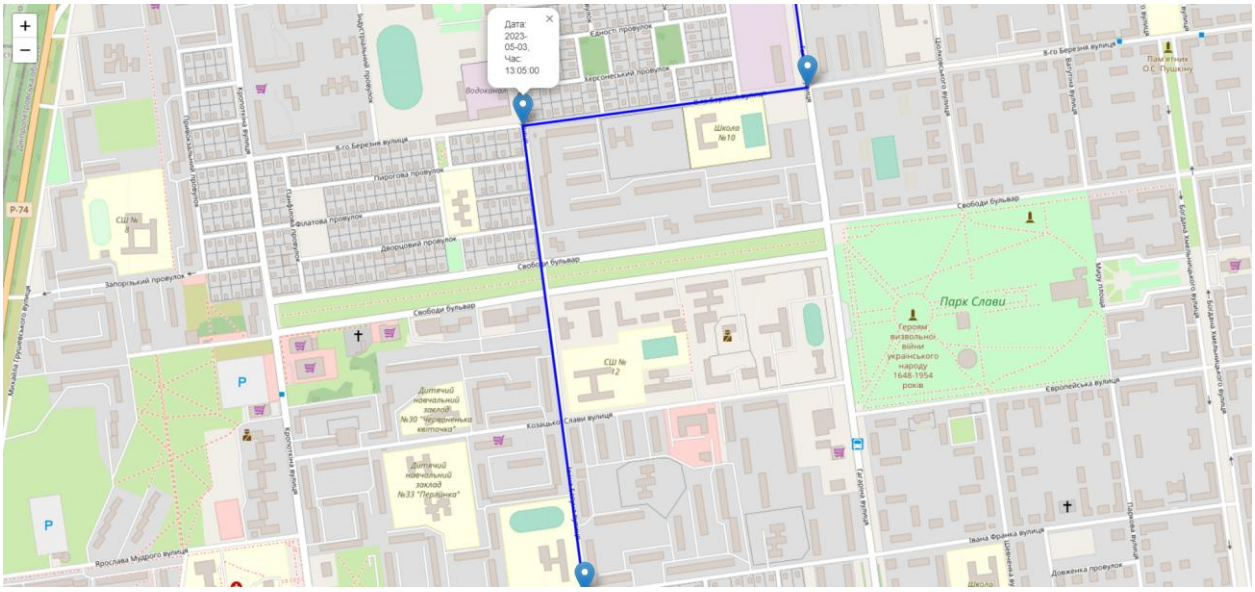

Рисунок 3.16 Демонстрація інформації про дату та час фіксації автомобіля з номерним знаком КА3456ОМ на другій камері

Програма вірно заносить данні з бази даних до мапи та видає маршрут та повну інформацію про дату та час фіксації.

ВИСНОВКИ

У дипломній роботі створено програму для відслідковування маршрутів руху автомобілів з використанням мови програмування Python та бібліотеки folium. Розробка подібної програми може сприяти збільшенню ефективності діяльності відповідних служб, зокрема поліції, дорожньої служби, екологічної інспекції тощо. Також програма може бути корисною для приватних охоронних служб, які займаються контролем руху на території підприємств чи інших закритих об'єктів.

В першому розділі детально розглянуто різні види розпізнавання символів та знаків за допомогою камер відеофіксації, більш детально розглянуто розпізнавання за допомогою нейронних мереж.

В другому розділі проведено аналіз різних мов програмування та обрано мову Python, оскільки ця мова є досить простою в освоєнні та має велику кількість корисних бібліотек, які значно спрощують процес розробки програмного забезпечення.

Також проаналізовано різні бази даних та було вибрано MySQL, оскільки ця система керування базами даних є досить простою у використанні, має велику кількість корисних функцій та масштабована.

Для відображення маршруту використана бібліотека folium, яка дозволяє створювати інтерактивні карти, на яких можна відображати маркери та лінії. Завдяки використанню цієї бібліотеки, користувач може бачити точні координати та час руху автомобіля.

Успішно розроблена програма, яка дозволяє відстежувати маршрути руху автомобілів з використанням мови програмування Python, бази даних MySQL та бібліотеки folium. Дана програма може бути використана в різних сферах,

таких як транспортні компанії, логістика, та будь-яких інших галузях, де важливо відслідковувати маршрути руху різних видів транспорту.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. НАУКОВА РОБОТА На тему: «Розробка програмного засобу розпізнавання автомобільних номерів» - с. 13 - [Електронний ресурс]. Режим доступу: https://nure.ua/wp-content/uploads/2020/Konkurs/69-avtomobilni-nomer.pdf
2. 402 КІНАХ О.В. ‘розпізнавання автомобільних номерів на основі нейронних мереж’ – с. 5 - [Електронний ресурс]. Режим доступу: http://surl.li/gwajy
3. Способи розпізнавання номерних знаків автомобілів у системі контрольно-пропускного пункту – с. 69 - [Електронний ресурс]. Режим доступу: https://ela.kpi.ua/bitstream/123456789/26680/1/Kampov_magistr.pdf
4. Сучасний підручник з JavaScript - дата доступу: [27.02.23] - [[Електронний ресурс]. Режим доступу: https://uk.javascript.info/
5. Python 3.11.3 documentation - дата доступу: [27.02.23] - [[Електронний ресурс]. Режим доступу: https://docs.python.org/uk/3/
6. C# documentation - дата доступу: [27.02.23] - [[Електронний ресурс]. Режим доступу: https://learn.microsoft.com/en-us/dotnet/csharp/
7. C++ Programming Language - дата доступу: [27.02.23] - [[Електронний ресурс]. Режим доступу: https://devdocs.io/cpp/
8. SQL Server technical documentation - дата доступу: [27.02.23] - [[Електронний ресурс]. Режим доступу: https://learn.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver16
9. MySQL Documentation - дата доступу: [13.04.23] - [Електронний ресурс]. Режим доступу: https://dev.mysql.com/doc/
10. Oracle MySQL - дата доступу: [13.04.23] - [Електронний ресурс]. Режим доступу: https://docs.oracle.com/cd/E17952_01/index.html
11. SQLite Documentation - дата доступу: [13.04.23] - [Електронний ресурс]. Режим доступу: https://www.sqlite.org/docs.html

12. Folium 0.14.0 documentation - дата доступу: [02.03.23] - [Електронний ресурс]. Режим доступу: https://python-visualization.github.io/folium/
13. streamlit-folium - дата доступу: [02.03.23] - [Електронний ресурс]. Режим доступу: https://folium.streamlit.app/

ДОДАТКИ

Додаток А

Лістинг програми

```
import folium
import mysql.connector
from folium.plugins import MarkerCluster

# з'єднуємось з базою даних
mydb = mysql.connector.connect(
    host='127.0.0.1',
    user='root',
    password='Katavada324',
    database='cars'
)

# запитуємо номерний знак машини
car_number = input("Введіть номерний знак машини: ")

# створюємо карту
m = folium.Map(location=[48.344746043429055, 33.50164576003915],
zoom_start=16)

# створюємо кластер маркерів
marker_cluster = MarkerCluster().add_to(m)

# витягуємо координати з бази даних
mycursor = mydb.cursor()
```

```
mycursor.execute(f"SELECT lat, lon, date, time FROM routes WHERE
car_id=(SELECT car_id FROM car_numbers WHERE
car_number='{car_number}')")

result = mycursor.fetchall()


# проходимось по кожній точці та додаємо маркер на карту

for point in result:

    folium.Marker(location=[point[0], point[1]], popup=f"Дата: {point[2]}, Час:
{point[3]}").add_to(marker_cluster)


# проходимось по кожній парі сусідніх точок та додаємо лінію між ними до
кластера маркерів

for i in range(len(result) - 1):

    folium.PolyLine(locations=[result[i][:2], result[i + 1][:2]],
color='blue').add_to(marker_cluster)


# відображаємо карту

m.save('map.html')
```

Додаток Б

Створення бази даних

```
create database cars character set utf8 collate utf8_general_ci;


use cars;


CREATE TABLE car_numbers (
    car_id INT PRIMARY KEY,
    car_number VARCHAR(10)
);


CREATE TABLE routes (
  id INT PRIMARY KEY AUTO_INCREMENT,
  car_id INT NOT NULL,
  lat DECIMAL(10, 8) NOT NULL,
  lon DECIMAL(11, 8) NOT NULL,
  date DATE NOT NULL,
  time TIME NOT NULL
);
```

INSERT INTO car_numbers (car_id, car_number) VALUES (1, 'AE1234BB'), (2, 'BA2345KO'), (3, 'KA3456OM');

INSERT INTO routes (car_id, lat, lon, date, time) VALUES

(1, 48.34482578517953, 33.48455751507143, '2023-05-01', '10:30:00'),

(1, 48.34615211121674, 33.49919164744028, '2023-05-01', '10:35:00'),

(1, 48.34036166000522, 33.500393277077336, '2023-05-01', '10:40:00'),

(1, 48.34120317036216, 33.50974882211024, '2023-05-01', '10:45:00'),

(1, 48.338147962232924, 33.51029869419981, '2023-05-01', '10:50:00'),

(1, 48.339070977859784, 33.52032973689893, '2023-05-01', '10:55:00'),

(1, 48.34030719606737, 33.520548534242145, '2023-05-01', '11:00:00'),

(2, 48.334131002495354, 33.48666631693455, '2023-05-02', '12:05:00'),

(2, 48.336070971353, 33.48632299418172, '2023-05-02', '12:10:00'),

(2, 48.33762531373246, 33.504604930769936, '2023-05-02', '12:15:00'),

(2, 48.343120338330074, 33.50345192792752, '2023-05-02', '12:20:00'),

(2, 48.34390127095952, 33.51202810715836, '2023-05-02', '12:25:00'),

(2, 48.34965661386406, 33.51083511465834, '2023-05-02', '12:30:00'),

(3, 48.33969131788929, 33.492861494747885, '2023-05-03', '13:00:00'),

(3, 48.34791166282302, 33.491220739200074, '2023-05-03', '13:05:00'),

(3, 48.34857697907477, 33.49872217132257, '2023-05-03', '13:10:00'),

(3, 48.35064886840955, 33.49829710667186, '2023-05-03', '13:15:00'),

(3, 48.35129012715253, 33.505157383496595, '2023-05-03', '13:20:00');

Додаток В

Лістинг map.html

```
<!DOCTYPE html>
<html>
<head>

    <meta http-equiv="content-type" content="text/html; charset=UTF-8" />

        <script>
            L_NO_TOUCH = false;
            L_DISABLE_3D = false;
        </script>

    <style>html, body {width: 100%;height: 100%;margin: 0;padding: 0;}</style>
    <style>#map {position:absolute;top:0;bottom:0;right:0;left:0;}</style>
    <script src="https://cdn.jsdelivr.net/npm/leaflet@1.9.3/dist/leaflet.js"></script>
    <script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"></
script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/Leaflet.awesome-
markers/2.0.2/leaflet.awesome-markers.js"></script>
```

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/leaflet@1.9.3/dist/leaflet.css"/>

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"/>

<link rel="stylesheet" href="https://netdna.bootstrapcdn.com/bootstrap/3.0.0/css/bootstrap.min.css"/>

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/@fortawesome/fontawesome-free@6.2.0/css/all.min.css"/>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/Leaflet.awesome-markers/2.0.2/leaflet.awesome-markers.css"/>

<link rel="stylesheet" href="https://cdn.jsdelivr.net/gh/python-visualization/folium/folium/templates/leaflet.awesome.rotate.min.css"/>

<meta name="viewport" content="width=device-width,

initial-scale=1.0, maximum-scale=1.0, user-scalable=no" />

<style>

#map_7cee125645bac8f43d40f9b034dcdee4 {

position: relative;

width: 100.0%;

height: 100.0%;

left: 0.0%;

top: 0.0%;

```
            }

            .leaflet-container { font-size: 1rem; }

        </style>


    <script
src="https://cdnjs.cloudflare.com/ajax/libs/leaflet.markercluster/1.1.0/leaflet.markercl
uster.js"></script>

    <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/leaflet.markercluster/1.1.0/MarkerCluster
.css"/>

    <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/leaflet.markercluster/1.1.0/MarkerCluster
.Default.css"/>

</head>

<body>



        <div class="folium-map" id="map_7cee125645bac8f43d40f9b034dcdee4"
></div>


</body>

<script>
```

```
        var map_7cee125645bac8f43d40f9b034dcdee4 = L.map(
            "map_7cee125645bac8f43d40f9b034dcdee4",
            {
                center: [48.344746043429055, 33.50164576003915],
                crs: L.CRS.EPSG3857,
                zoom: 16,
                zoomControl: true,
                preferCanvas: false,
            }
        );

        var tile_layer_b99c2862d86f4022bdbff6a1695859b7 = L.tileLayer(
            "https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png",
            {"attribution": "Data by \u0026copy; \u003ca target=\"_blank\"
href=\"http://openstreetmap.org\"\u003eOpenStreetMap\u003c/a\u003e, under
\u003ca target=\"_blank\"
href=\"http://www.openstreetmap.org/copyright\"\u003eODbL\u003c/a\u003e.",
"detectRetina": false, "maxNativeZoom": 18, "maxZoom": 18, "minZoom": 0,
"noWrap": false, "opacity": 1, "subdomains": "abc", "tms": false}
```

```
        ).addTo(map_7cee125645bac8f43d40f9b034dcdee4);


        var marker_cluster_cfe3c5020b721895bf67ef40c2f9bda2 =
L.markerClusterGroup(
            {}
        );

map_7cee125645bac8f43d40f9b034dcdee4.addLayer(marker_cluster_cfe3c5020b72
1895bf67ef40c2f9bda2);


        var marker_2461ee174671200d0ca88ebde349e305 = L.marker(
            [48.33969132, 33.49286149],
            {}
        ).addTo(marker_cluster_cfe3c5020b721895bf67ef40c2f9bda2);


    var popup_9239dc549264327748281b6750ba76c2 = L.popup({"maxWidth":
"100%"});
```

var html_2bae5e75b69841803d09118817abde2e = $(`<div id="html_2bae5e75b69841803d09118817abde2e" style="width: 100.0%; height: 100.0%;">Дата: 2023-05-03, Час: 13:00:00</div>`)[0];

popup_9239dc549264327748281b6750ba76c2.setContent(html_2bae5e75b69841803d09118817abde2e);

marker_2461ee174671200d0ca88ebde349e305.bindPopup(popup_9239dc549264327748281b6750ba76c2)

;

var marker_1dab8b8bc5dba0519b6c193e79be7f56 = L.marker(

[48.34791166, 33.49122074],

{}

).addTo(marker_cluster_cfe3c5020b721895bf67ef40c2f9bda2);

var popup_cb6b85a90d186faa827e722334343598 = L.popup({"maxWidth": "100%"});

var html_8900dca746829f90e8330a8c2abe6af5 = $(`<div id="html_8900dca746829f90e8330a8c2abe6af5" style="width: 100.0%; height: 100.0%;">Дата: 2023-05-03, Час: 13:05:00</div>`)[0];

popup_cb6b85a90d186faa827e722334343598.setContent(html_8900dca746829f90e8330a8c2abe6af5);

marker_1dab8b8bc5dba0519b6c193e79be7f56.bindPopup(popup_cb6b85a90d186faa827e722334343598)

;

var marker_a6df4335d91817a30912a0fabd494985 = L.marker(

[48.34857698, 33.49872217],

{}

).addTo(marker_cluster_cfe3c5020b721895bf67ef40c2f9bda2);

var popup_c15fec6c6b114e74a67952e297bc8f93 = L.popup({"maxWidth": "100%"});

var html_048986d5c24e862576584cdfeb114651 = $(`<div id="html_048986d5c24e862576584cdfeb114651" style="width: 100.0%; height: 100.0%;">Дата: 2023-05-03, Час: 13:10:00</div>`)[0];

popup_c15fec6c6b114e74a67952e297bc8f93.setContent(html_048986d5c24e862576584cdfeb114651);

marker_a6df4335d91817a30912a0fabd494985.bindPopup(popup_c15fec6c6b114e74a67952e297bc8f93)

;

```
        var marker_e1ada166b931cd16fba7799bfaa797dd = L.marker(
            [48.35064887, 33.49829711],
            {}
        ).addTo(marker_cluster_cfe3c5020b721895bf67ef40c2f9bda2);


        var popup_cee3bb272ba168184946f4447970af98 = L.popup({"maxWidth": "100%"});


            var html_2caa7726d64c8da227147cc286ab87c9 = $(`<div id="html_2caa7726d64c8da227147cc286ab87c9" style="width: 100.0%; height: 100.0%;">Дата: 2023-05-03, Час: 13:15:00</div>`)[0];

        popup_cee3bb272ba168184946f4447970af98.setContent(html_2caa7726d64c8da227147cc286ab87c9);
```

marker_e1ada166b931cd16fba7799bfaa797dd.bindPopup(popup_cee3bb272ba16818
4946f4447970af98)

;

var marker_c76bd2d6c983d3bd80bc0669d118a583 = L.marker(

[48.35129013, 33.50515738],

{}

).addTo(marker_cluster_cfe3c5020b721895bf67ef40c2f9bda2);

var popup_6fed6c0d9c95ff3ffb85bad9b2772547 = L.popup({"maxWidth":
"100%"});

var html_c04d807f8ec7ac3880b99fc0b80c2736 = $(`<div
id="html_c04d807f8ec7ac3880b99fc0b80c2736" style="width: 100.0%; height:
100.0%;">Дата: 2023-05-03, Час: 13:20:00 `)[0];

popup_6fed6c0d9c95ff3ffb85bad9b2772547.setContent(html_c04d807f8ec7ac3880b99fc0b80c2736);

marker_c76bd2d6c983d3bd80bc0669d118a583.bindPopup(popup_6fed6c0d9c95ff3ffb85bad9b2772547)

;

var poly_line_1e4d5cec43236e6d2d080b43f4e7aaf6 = L.polyline(

[[48.33969132, 33.49286149], [48.34791166, 33.49122074]],

{"bubblingMouseEvents": true, "color": "blue", "dashArray": null, "dashOffset": null, "fill": false, "fillColor": "blue", "fillOpacity": 0.2, "fillRule": "evenodd", "lineCap": "round", "lineJoin": "round", "noClip": false, "opacity": 1.0, "smoothFactor": 1.0, "stroke": true, "weight": 3}

).addTo(marker_cluster_cfe3c5020b721895bf67ef40c2f9bda2);

var poly_line_466e52ded45bf933c79c92ea6cb3d0c7 = L.polyline(

[[48.34791166, 33.49122074], [48.34857698, 33.49872217]],

{"bubblingMouseEvents": true, "color": "blue", "dashArray": null, "dashOffset": null, "fill": false, "fillColor": "blue", "fillOpacity": 0.2, "fillRule": "evenodd", "lineCap": "round", "lineJoin": "round", "noClip": false, "opacity": 1.0, "smoothFactor": 1.0, "stroke": true, "weight": 3}

).addTo(marker_cluster_cfe3c5020b721895bf67ef40c2f9bda2);

var poly_line_43e4c81ba3c3c664001781d950eb24ad = L.polyline(

[[48.34857698, 33.49872217], [48.35064887, 33.49829711]],

{"bubblingMouseEvents": true, "color": "blue", "dashArray": null, "dashOffset": null, "fill": false, "fillColor": "blue", "fillOpacity": 0.2, "fillRule": "evenodd", "lineCap": "round", "lineJoin": "round", "noClip": false, "opacity": 1.0, "smoothFactor": 1.0, "stroke": true, "weight": 3}

).addTo(marker_cluster_cfe3c5020b721895bf67ef40c2f9bda2);

var poly_line_2d70b5359ea245dc3dc96504565723dc = L.polyline(

[[48.35064887, 33.49829711], [48.35129013, 33.50515738]],

{"bubblingMouseEvents": true, "color": "blue", "dashArray": null, "dashOffset": null, "fill": false, "fillColor": "blue", "fillOpacity": 0.2, "fillRule": "evenodd", "lineCap": "round", "lineJoin": "round", "noClip": false, "opacity": 1.0, "smoothFactor": 1.0, "stroke": true, "weight": 3}

).addTo(marker_cluster_cfe3c5020b721895bf67ef40c2f9bda2);

</script>

```
</html>
```