МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА ТЕХНОЛОГІЙ

Кафедра Комп'ютерних інформаційних технологій

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
Аліна САВЧЕНКО

«_____»_____2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

(ДИПЛОМНА РОБОТА, ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИЦІ ОСВІТНЬОГО СТУПЕНЯ

"МАГІСТРА"

ЗА ОСВІТНЬО-ПРОФЕСІЙНОЮ ПРОГРАМОЮ "ІНФОРМАЦІЙНІ УПРАВЛЯЮЧІ СИСТЕМИ ТА ТЕХНОЛОГІЇ"

Тема: «Інформаційна система моніторингу руху міського транспорту»

Виконала: Оласюк Наталія Миколаївна

Керівник: к.т.н., доцент кафедри КІТ Райчев Ігор Едуардович

Нормоконтролер Ігор РАЙЧЕВ

Київ – 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет *Комп'ютерних наук та технологій*.

Кафедра *Комп'ютерних інформаційних* технологій.

Галузь знань, спеціальність, освітньо-професійна програма: 12 "Інформаційні технології", 122 "Комп'ютерні науки", "Інформаційні управляючі системи та технології".

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____Аліна САВЧЕНКО

"___"_____ 2023 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи студентки

Оласюк Наталії Миколаївни

(прізвище, ім'я, по батькові)

1.Тема роботи: «Інформаційна система моніторингу руху міського транспорту», затверджена наказом ректора від "29"вересня 2023 р. за № 1976/ст..

2. Термін виконання роботи: : з 02 жовтня 2023 р. по 31 грудня 2023 р. .

3. Вихідні дані до роботи: методи та засоби розробки для реалізації покращеного функціоналу інформаційної системи для відстежування маршрутних таксі міста.

4. Зміст пояснювальної записки (перелік питань, що підлягають розробці): існуючі рішення для моніторингу для планування поїздок на маршрутних таксі міста, способи покращення існуючого функціоналу, інструменти розробки мобільних додатків, додаткових технологій та бібліотек для розробки мобільних додатків, вимоги до інформаційної системи для аналізу існуючих рішень моніторингу для планування поїздок на маршрутних таксі міста..

5. Перелік обов'язкового графічного матеріалу: інформативні рисунки, графічні скріншоти роботи системи, слайди презентації в MS PowerPoint.

6. Календарний план-графік

№ з/п	Завдання	Термін виконання	Підпис керівни
1.	Пошук і дослідження наукових джерел.	06.09.2023 – 08.09.2023	
2.	Розроблення та затвердження календарного плану виконання дипломної роботи.	09.09.2023 – 10.09.2023	
3.	Проведення консультацій з науковим керівником.	11.09.2023 – 12.09.2023	
4.	Написання Розділу 1. Аналіз існуючих рішень по плануванню поїздок на маршрутних таксі міста.	13.09.2023 –27.09.2023	
5.	Написання Розділу 2. Огляд методів, інструментів та підходів для реалізації нової функціональності в інформаційній системі.	28.09.2023 – 16.10.2023	
6.	Написання Розділу 3.	17.10.2023 – 31.10.2023	
7.	Оформлення пояснювальної записки дипломної роботи.	01.11.2023 – 07.11.2023	
8.	Написання, друк та підписання Рецензії у рецензента та Відгуку керівника у встановленому порядку.	08.11.2023 – 11.11.2023	
9.	Створення Презентації та доповіді.	12.11.2023 – 14.11.2023	
10.	Підготовка до захисту та попередній захист дипломної роботи на випусковій кафедрі.	15.11.2023 – 05.12.2023	

7. Дата видачі завдання: «02» жовтня 2023 р.

Керівник дипломної роботи _____ Ігор РАЙЧЕВ
(підпис керівника) (П.І.Б.)

Завдання прийняла до виконання _____ Наталія ОЛАСЮК
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Пояснювальна записка до дипломної роботи «Інформаційна система моніторингу руху міського транспорту» викладена на 84 сторінках і містить 15 рисунків. Список бібліографічних посилань складається з 16 найменувань.

МАШИННЕ НАВЧАННЯ, ГЕОЛОКАЦІЙНІ ДАНІ, АЛГОРИТМИ ТА ФУНКЦІЇ, АНАЛІЗ ТЕКСУ, МЕТОД ВИЯВЛЕННЯ КЛЮЧОВИХ СЛІВ, КРИТЕРІЇ РОЗРОБКИ, ВІЗУАЛІЗАЦІЯ КОРИСТУВАЦЬКОГО ІНТЕРФЕЙСУ.

Об’єкт дослідження – процес оптимізації системи моніторингу міського транспорту.

Предмет дослідження – інформаційна система моніторингу руху міського транспорту.

Мета роботи. Оптимізація інформаційної системимоніторингу руху міського транспорту, що враховують недоліки створеною раніше системи та пропонує наповнення системи новим функціоналом.

Актуальність дипломної роботи полягає в необхідності покращення інформаційної моніторингу руху міського транспорту для більш зручного та надійного використання, порівняно з існуючою інформаційними системою, що була розроблена раніше.

У роботі висвітлено:

- Існуючі рішенння для моніторингу руху міського транспорту та їх недоліки;
- Особливості створення інформаційної системи із залученням сучасних технологій та мов програмування;
- Функціонування інформаційної системи з доданим функціоналом та оптимізація користувацького інтерфейсу.

Подальший розвиток розробленої інформаційної системи можливий у напрямку покращення інтеграції зі штучним інтелектом для розрахування оптимального способу пересування містом міським транспортом.

ЗМІСТ

ВСТУП

Сучасні міста є складними екосистемами, де міський транспорт відіграє ключову роль у забезпеченні мобільності громадян та розвитку інфраструктури. Підвищення рівня комфорту та ефективності міського транспорту стає надзвичайно важливим завданням для влади та міських підприємців. Водночас, науковці та інженери активно працюють над розробкою нових технологій та інформаційних систем для підтримки цього процесу.

Інформаційна система моніторингу руху міського транспорту є однією з інноваційних розробок, спрямованих на оптимізацію та управління транспортним потоком в місті. Ця система використовує сучасні технології збору та аналізу даних, щоб забезпечити точний та надійний моніторинг руху міських транспортних засобів. Вона дозволяє владі та операторам транспорту отримувати в режимі реального часу важливу інформацію про рух транспорту, робити прогнози, приймати управлінські рішення та надавати громадянам можливість зручно користуватися міськими перевезеннями.

Мета даної дипломної роботи полягає в розробці та впровадженні нових рішень та функціоналу до інформаційної системи моніторингу руху міського транспорту, яка сприятиме підвищенню якості послуг громадянам та оптимізації роботи міських транспортних служб. У цьому проекті буде проведений аналіз сучасних підходів до моніторингу міського транспорту, розроблена архітектура системи, розглянуті питання зі збору та обробки даних, а також надані рекомендації з її впровадження та подальшого розвитку.

Дослідження та розробка інформаційної системи моніторингу руху міського транспорту має великий потенціал для поліпшення якості життя мешканців міст та сприяє сталому розвитку урбаністичних областей. Результати цього проекту можуть бути використані владою міст, транспортними компаніями та дослідницькими установами для забезпечення

більш зручної та ефективної роботи міського транспорту, що відповідає сучасним вимогам суспільства.

РОЗДІЛ 1
АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ПО ПЛАНУВАННЮ ПОЇЗДОК НА МАРШРУТНИХ ТАКСІ МІСТА

1.1. Easyway

Easyway - це додаток для моніторингу руху міського транспорту, який надає користувачам доступ до різноманітної інформації та функціоналу, спрямованого на полегшення використання громадського транспорту та підвищення комфорту подорожей в містах.

Як тільки користувач заходить в додаток, він одразу бачить найближчі до нього зупинки. Потім він обирає потрібну зупинку та маршрут, і дивиться приблизний час прибуття транспортного засобу.

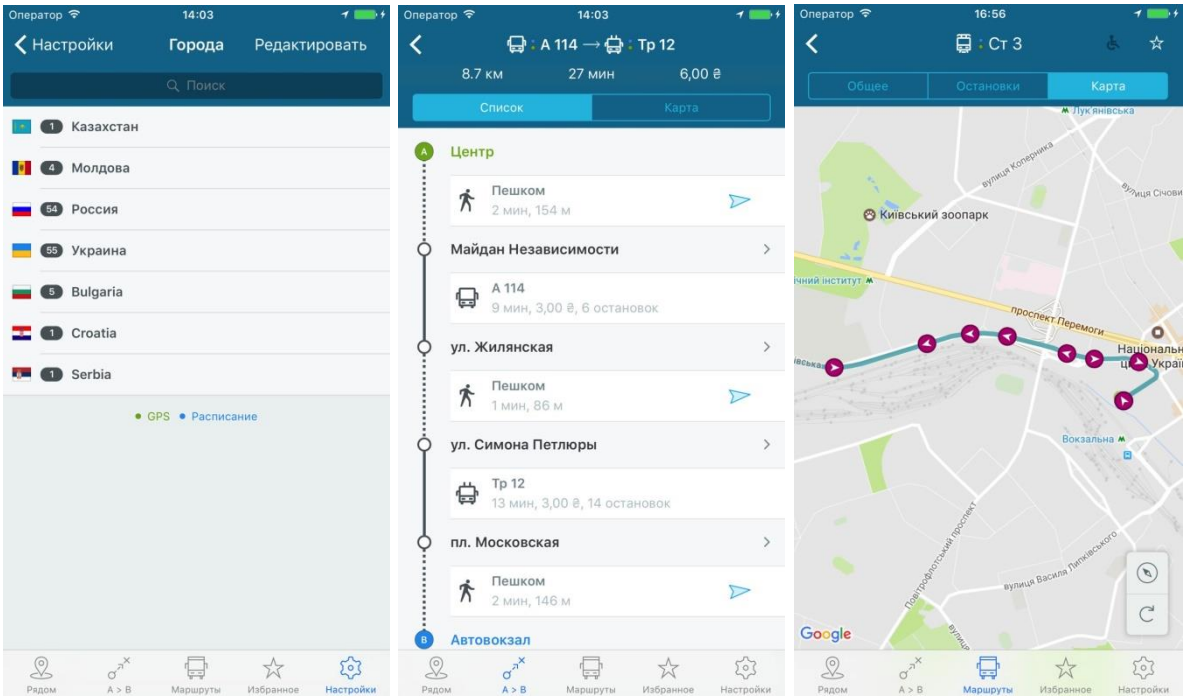

Рис. 1.1. Мобільний додаток EasyWay

Кафедра				**НАУ 23.15.48.000 ПЗ**		
Виконала	*Оласюк Н.М.*			*Інформаційна система моніторингу руху міського транспорту*	**Літ.**	**Арк.** 9 / **Аркушів** 14
Керівник	*Райчев І.Е.*					
Консульт.					**УС-211М**	**122**
Н. Контр.	*Райчев І.Е.*					

Основними недоліками додатку є:

- Залежність від інтернет-з'єднання: Для коректної роботи додатку потрібне постійне підключення до Інтернету. Це може створювати несприятливі умови для користувачів у випадку відсутності мережі або поганих умов зв'язку.
- Не завжди точна інформація: Інформація про рух транспорту в реальному часі може бути не завжди абсолютно точною через різні фактори, такі як погодні умови, технічні проблеми транспортних засобів тощо.
- Залежність від даних операторів транспорту: Додаток зазвичай залежить від даних, які надають оператори міського транспорту. Якщо оператор надає обмежену або неякісну інформацію, це може вплинути на якість роботи додатку.
- Специфіка регіону: "Easyway" може бути більш ефективним у певних регіонах або містах, де існують добре розвинені системи моніторингу та передачі даних про рух транспорту. У менш розвинених регіонах додаток може бути менш корисним.

1.2. Moovit

Moovit - це додаток для смартфонів та веб-платформа, які призначені для надання інформації про громадський транспорт та допомоги користувачам у плануванні та здійсненні подорожей в міському середовищі. Основні функції та можливості додатка Moovit включають:

- Розклади та маршрути: Moovit надає інформацію про розклади громадського транспорту;
- Моніторинг руху в реальному часі: Можливість відстежувати рух громадського транспорту в режимі реального часу;
- Планування маршруту: Користувачі можуть вказати початкову та кінцеву точки своєї подорожі;

- Інформація про доступність для осіб з обмеженими можливостями.

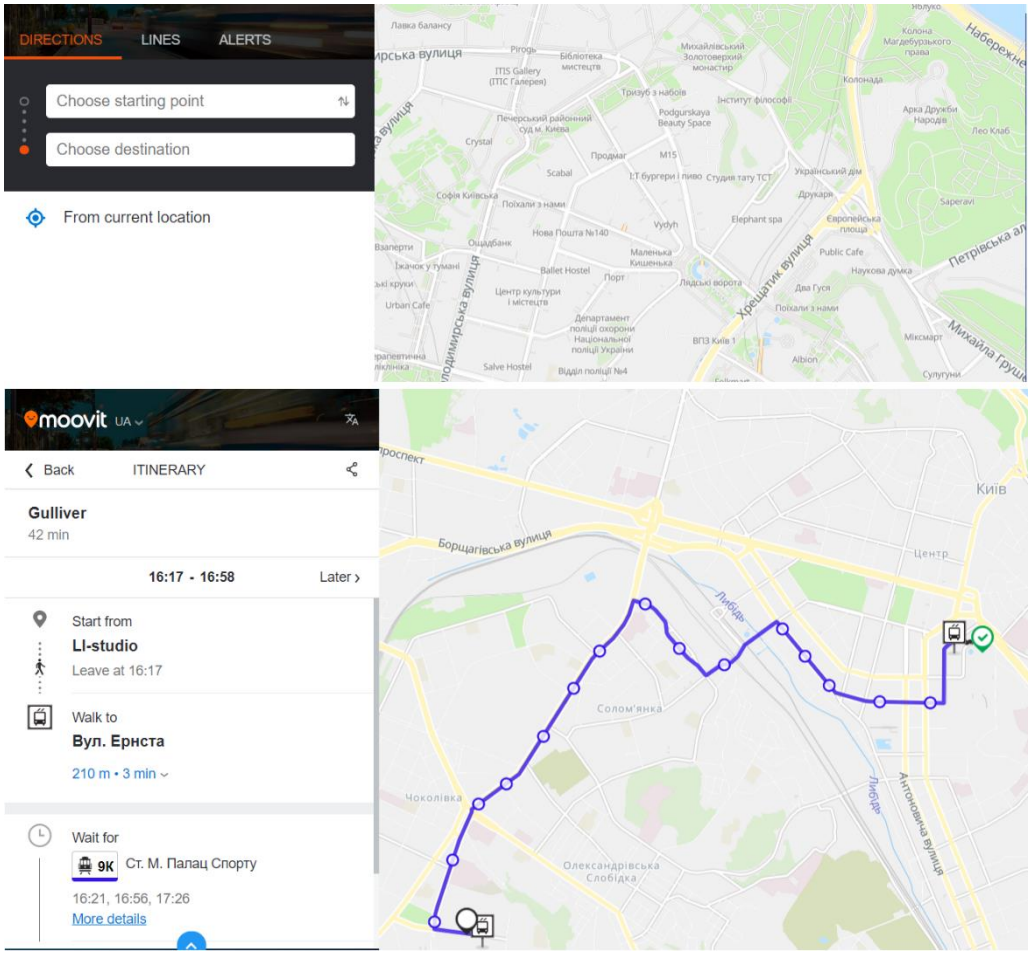

Рис. 1.2. Мобільний додаток Moovit

Основними недоліками додатку є:

- Додаток має даних про поточне положення транспорту, але покриває дуже багато міст, та населених пунктів. Загалом додаток більше придатний для туристів, які подорожують по містам. Раціонально спланувати час за допомогою цього додатку ви не зможете.

- Неточність інформації: У деяких випадках, особливо в містах з складною інфраструктурою та непостійним графіком, інформація про рух транспорту може бути неточною або застарілою. Це може призводити до неприємних ситуацій для користувачів, які розраховуються за цю інформацію.

- Залежність від користувачів: Moovit використовує принцип спільної інформації, іншими словами, інформація про рух транспорту надходить від

користувачів, які діляться своєю поточною локацією та досвідом подорожей. Це може призводити до низької достовірності даних, особливо в малозаселених або малорухомих районах.

- Не всюди доступний: Moovit може бути доступний не в усіх містах і регіонах світу. Деякі користувачі можуть виявити, що додаток не надає повну підтримку для їхнього регіону.
- Не завжди інтегрований з міськими системами транспорту: Деякі міські системи транспорту можуть не бути повністю інтегрованими з Moovit, що може призводити до несправностей в інформації про рух транспорту та можливих розбіжностей з офіційними джерелами інформації.
- Залежність від Інтернету: Для коректної роботи Moovit потрібне постійне підключення до Інтернету. Це може створювати проблеми в умовах обмеженого доступу до мобільного зв'язку або в місцях з поганим покриттям.
- Реклама та споживчі дані: Додаток може містити рекламу та збирати споживчі дані користувачів для вивчення їхніх вподобань і звичок. Деякі користувачі можуть вважати це надокучливим або порушенням приватності.

1.3. Google Maps

Google Maps – широко відомий додаток, який призначений для перегляду вулиць, визначних місць, доріг та багато іншого. До того ж Google Maps може будувати маршрути від точки А, до точки В, показуючи весь доступний для цього наземний транспорт. Google Maps містить величезну кількість можливих маршрутів, та велику базу транспортних засобів, але так як і попереденій додаток Moveit, Google Maps не відображає поточного розташування транспорту.

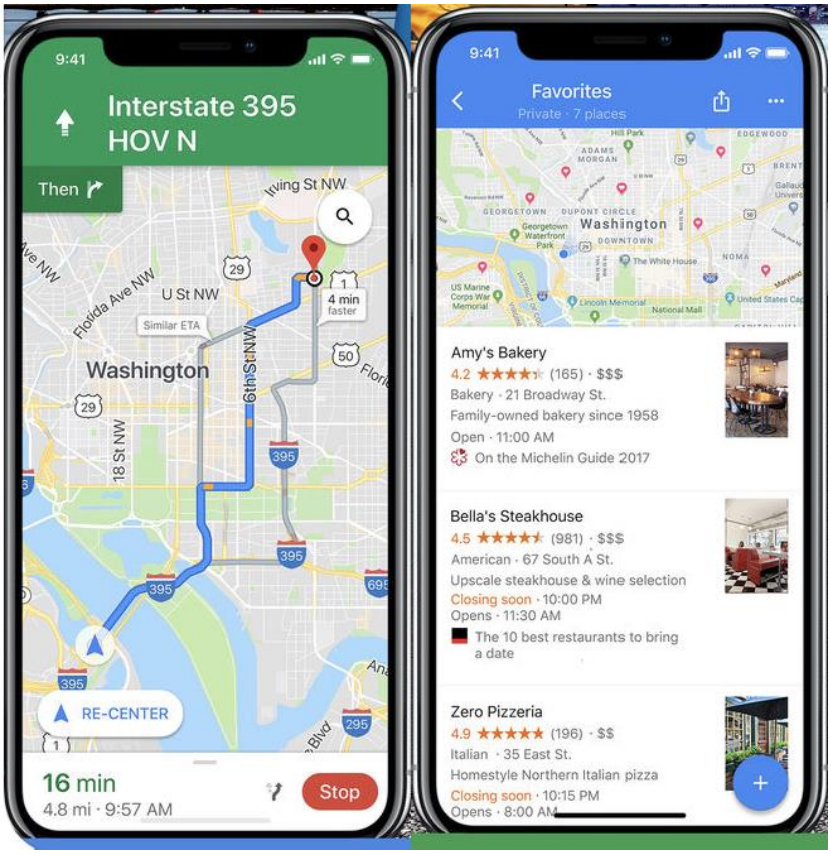

Рис. 1.3. Мобільний додаток Google Maps

Основними недоліками додатку є:

- Не завжди актуальна інформація про громадський транспорт: У деяких регіонах та містах Google Maps може не мати актуальної інформації про розклади громадського транспорту або не підтримувати всі види транспорту. Це може бути не зручно для користувачів, які полагаються на громадський транспорт.

- Неоптимальні маршрути: Інколи Google Maps може пропонувати неоптимальні маршрути або рекомендувати пересадки, які можуть займати більше часу, ніж інші варіанти. Це може призвести до надмірної тривалості подорожі.

1.4. Транспорт онлайн

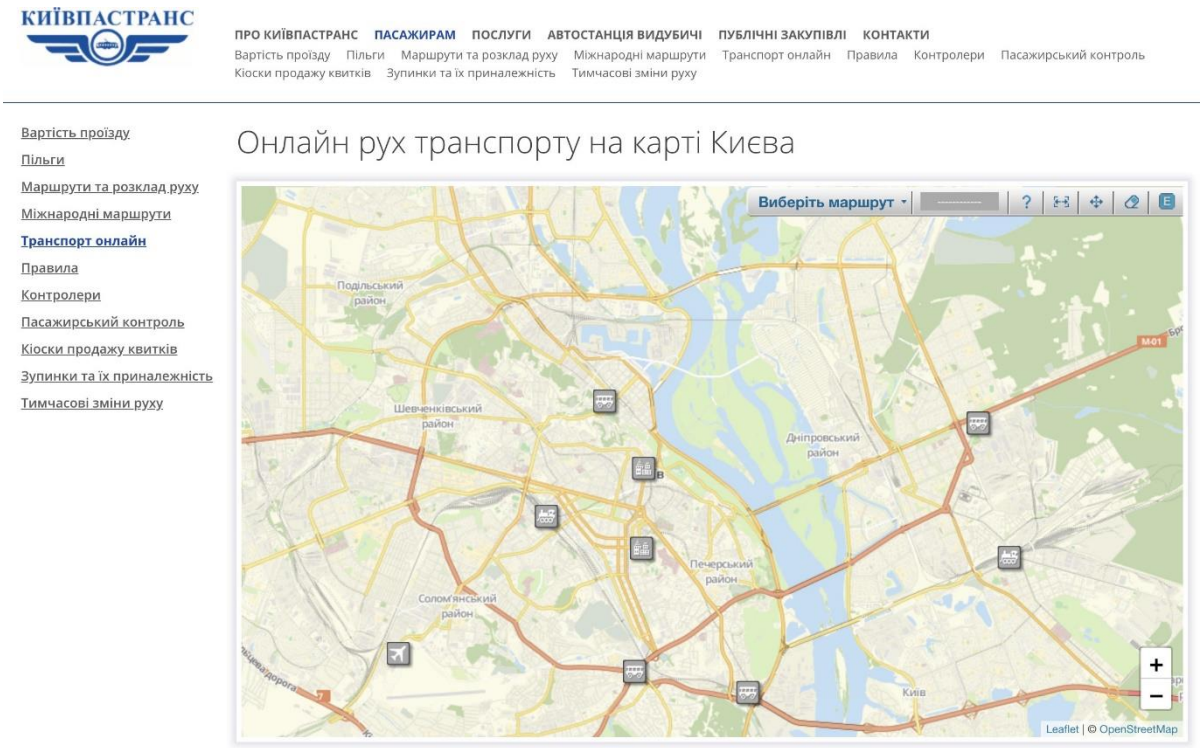

Рис. 1.4. Головна сторінка сайту «Транспорт онлайн»

Транспорт онлайн – це сайт де ви можете переглядати положення громадського транспорту в реальному часі. Сайт використовує ті ж самі пристрої GPS, що і додаток Easyway, тому не має переваг, а лише наслідує ті ж недоліки, що і вищезгаданий додаток.

Основними недоліками додатку є:

– не всі транспортні засоби міста оснащені GPS трекерами;

– частина засобів, що оснащена GPS не показує своє розташування у реальному часі;

– не зручний інтерфейс додатку;

– застарілий дизайн;

– не відображає часу прибуття, лише положення на карті.

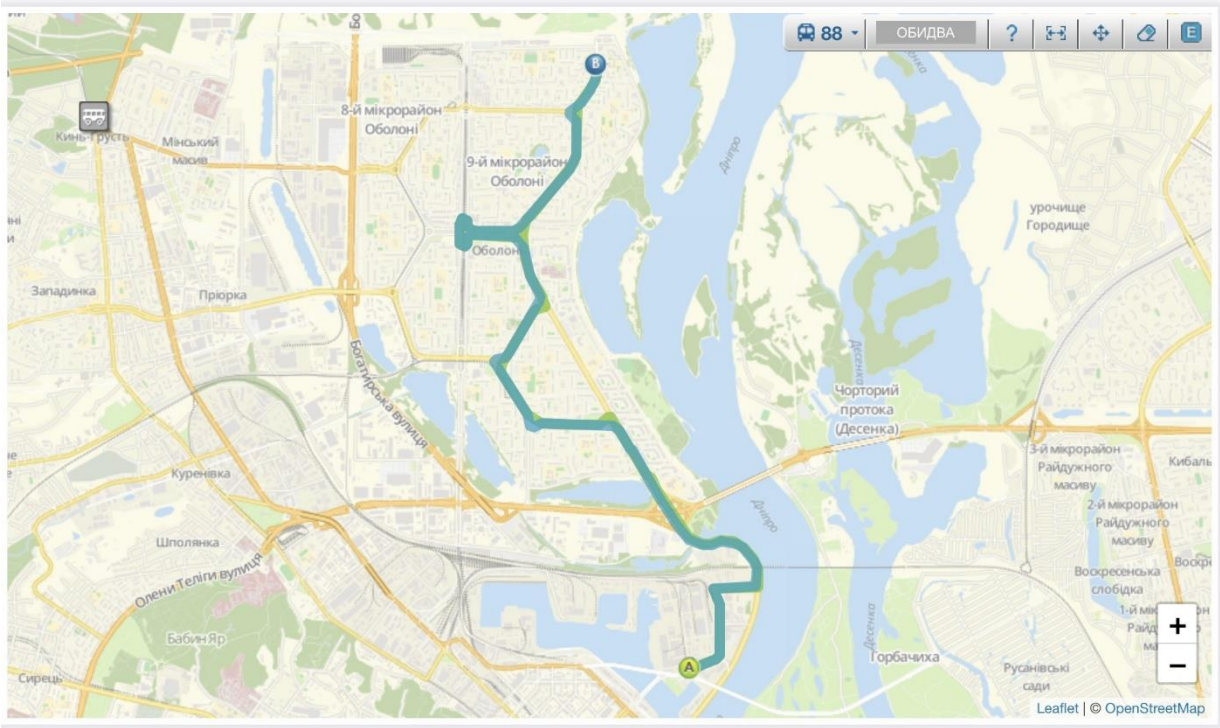

Рис. 1.5. Карта з маршрутом транспортного засобу на сайті «Транспорт онлайн»

1.5. Актуальність інформаціних систем для відстежування маршрутних таксі міста

Створення мобільних додатків для відстежування маршрутних таксі міста є дуже актуальним завданням в сучасному світі. Ось кілька основних причин, чому це актуально:

- Зручність для користувачів: Мобільні додатки дозволяють пасажирам легко та швидко викликати маршрутне таксі, відстежувати приблизний час прибуття та маршрут руху таксі на мапі. Це забезпечує зручність та ефективність подорожі.
- Покращення безпеки: Додатки для відстежування маршрутного таксі можуть сприяти покращенню безпеки пасажирів. Пасажири можуть слідкувати за маршрутом та ділитися інформацією про свою подорож з друзями або сім'єю, щоб забезпечити свою безпеку.

- Ефективне управління автопарком: Для операторів маршрутного таксі мобільні додатки надають можливість відстежувати всі свої транспортні засоби в режимі реального часу, оптимізувати маршрути та знижувати час очікування пасажирів, що дозволяє підвищити якість обслуговування та залучити нових клієнтів.

- Споживчий попит: Попит на маршрутне таксі продовжує зростати, особливо в мегаполісах та містах з високою щільністю населення. Мобільні додатки роблять взаємодію з таксі більш зручною для користувачів, що впливає на зростання популярності цих послуг.

- Конкурентна перевага для бізнесу: Для компаній, які надають послуги маршрутного таксі, наявність власного мобільного додатку може бути конкурентною перевагою. Це дозволяє залучати більше клієнтів та покращувати обслуговування.

- Можливість збільшити ефективність міського транспорту: У ряді міст додатки для маршрутного таксі допомагають оптимізувати роботу громадського транспорту, розподіляти пасажирів більш ефективно та зменшувати затори.

- Екологічні переваги: Виклик маршрутного таксі через мобільний додаток може сприяти зменшенню кількості особистих автомобілів на дорогах, що допомагає знижувати забруднення повітря та транспортні затори.

Узагальнюючи, створення мобільних додатків для відстежування маршрутного таксі міста є актуальним та важливим напрямком розвитку транспортної та технологічної індустрії. Вони покращують зручність та безпеку подорожей для користувачів і допомагають оптимізувати роботу операторів таксі.

1.6. Способи покращення існуючих додатків для відстеження маршрутних таксі міста

Існуючі додатки для відстежування маршрутного таксі міста вже мають багато корисних функцій, але є кілька аспектів, які можуть бути покращені або вдосконалені:

- Інтеграція з громадським транспортом: Деякі додатки для відстежування маршрутного таксі не мають інтеграції з іншими видами громадського транспорту, такими як автобуси, метро, трамваї тощо. Удосконалення додатків для включення всіх варіантів транспорту може полегшити пасажирам планування мультимодальних маршрутів.
- Оптимізація маршруту: Більшість додатків дозволяють користувачам вказати початкову і кінцеву точки, але їхні можливості оптимізації маршруту можуть бути вдосконалені. Оптимальний маршрут може включати в себе врахування трафіку, пересадок і найкоротших шляхів.
- Опції платежу: Розширення варіантів платежу, включаючи можливість оплати за допомогою електронних грошей, мобільних платіжних систем або криптовалют, може зробити послугу більш зручною для користувачів.
- Зменшення часу очікування: Оптимізація маршрутів та робота з водіями для зменшення часу очікування таксі може підвищити задоволення клієнтів.
- Додаткові функції для водіїв: Розробники можуть розглянути можливість додавання додаткових функцій для водіїв, таких як інформація про трафік, навігація та засоби для підтримки водіїв під час роботи.
- Покращення досвіду: Додатки можуть вдосконалити систему оцінок та відгуків користувачів та водіїв для забезпечення більшої надійності та довіри до системи.

- Більше даних для користувачів: Додатки можуть надавати користувачам більше інформації про водіїв, таку як їхню рейтингову оцінку, відгуки попередніх пасажирів і фотографії автомобілів.
- Збільшення безпеки: Вдосконалення систем безпеки для пасажирів і водіїв, такі як можливість обмежити виклики лише з певних діапазонів часу або регіонів, може зменшити ризик неприємних ситуацій.

Загалом, розвиток та вдосконалення додатків для відстежування маршрутного таксі міста важливі для забезпечення зручності та ефективності транспортних послуг для користувачів та підвищення конкурентоспроможності операторів маршрутних таксі.

1.7. Впровадження технологій штучного інтелекту (машинного навчання) до мобільних додатків для відстежування маршрутних таксі міста

Впровадження штучного інтелекту (машинного навчання) до мобільних додатків для відстежування маршрутних таксі міста може відкрити нові можливості та поліпшити функціональність таких додатків. Наведу приклади способів, які можна використовувати машинне навчання для покращення додатків для відстежування маршрутного таксі.

Прогнозування часу прибуття: Машинне навчання може аналізувати дані про рух таксі в реальному часі та враховувати фактори, такі як трафік, погода, дорожні умови і навіть стиль водіння, для точного прогнозування часу прибуття маршрутного таксі до місця призначення.

Оптимізація маршруту: Машинне навчання може автоматично аналізувати поточні умови на дорозі і вибирати найкращий маршрут для подорожі, що допомагає зменшити час подорожі та споживання пального.

Персоналізація: Машинне навчання може враховувати індивідуальні вподобання та звички користувачів. Наприклад, воно може надавати рекомендації щодо популярних місць або підходящих часів для подорожі.

Підвищення безпеки: Машинне навчання може допомагати виявляти аномалії та небезпеку під час подорожі. Наприклад, воно може виявляти небезпечну їзду водіїв та надсилати попередження.

Автоматичний розрахунок ціни: Машинне навчання може автоматично розраховувати вартість поїздки на основі відстані, часу та інших факторів. Воно також може надавати рекомендації щодо оптимальних варіантів для зниження вартості поїздки.

Оцінка якості обслуговування: Машинне навчання може аналізувати відгуки та оцінки користувачів для оцінки якості обслуговування водіїв та додатку. Воно може виявляти проблемні ситуації та реагувати на них.

Прогнозування попиту: Машинне навчання може аналізувати дані про попит на маршрутне таксі в різні часи доби та на різних маршрутах, щоб встановлювати оптимальні ціни та розподіляти ресурси ефективно.

Впровадження машинного навчання в мобільні додатки для відстежування маршрутного таксі може сприяти покращенню ефективності та якості послуг для користувачів, підвищити безпеку та оптимізувати операційні процеси для операторів таксі. До того ж, це може зробити такі додатки більш конкурентоспроможними та привабливими для користувачів.

Для впровадження штучного інтелекту (машинного навчання) до мобільних додатків для відстежування маршрутних таксі міста можна використовувати різні технології та інструменти, такі як:

- Моделі машинного навчання: використання моделей машинного навчання, таких як нейронні мережі, дерева рішень, методи класифікації та регресії для аналізу даних та роботи з прогнозуванням, оптимізацією маршрутів та іншими завданнями.

- Аналітика в реальному часі: використання потужних систем аналітики в реальному часі, щоб відстежувати рух транспорту, аналізувати трафік і робити прогнози для кращого управління транспортними потоками.

- Геолокація і GPS: використання GPS та геолокаційних даних для точного визначення місцезнаходження та руху транспортних засобів, що допомагає в реалізації послуг відстеження.

- Обробка природних мов: використання обробки природних мов для аналізу відгуків та коментарів користувачів, що допомагає зрозуміти їхні вподобання та скарги.

- Системи рекомендацій: використання алгоритмів рекомендацій для надання користувачам персоналізованих порад та пропозицій щодо подорожей та водіїв.

- Аналітика даних: збір та аналіз великих обсягів даних, для виявлення тенденції та пошуку можливостей для оптимізації та покращення сервісу.

- Аналіз зображень та відео: використання аналізу зображень та відео для розпізнавання номерів автомобілів, аналізу водійської поведінки та іншого моніторингу.

Ці технології можуть бути використані як окремо, так і в комбінації для створення розумного та продуктивного мобільного додатку для відстежування маршрутних таксі міста. Вони допоможуть покращити якість обслуговування та забезпечити більшу зручність та безпеку для користувачів і водіїв.

Висновки до Розділу 1

У цьому розділі дипломної роботи було проведено аналіз та дослідження сучасних мобільних додатків для відстежування маршрутних таксі міста та технологій, які можна використовувати для їхнього впровадження та вдосконалення. На основі цього дослідження я сформулювала висновки про: Актуальність розвитку мобільних додатків для відстежування маршрутного таксі так як сучасний світ стикається з ростом популярності маршрутного таксі, і створення мобільних додатків для його відстежування є надзвичайно актуальним завданням.

Плюси та мінуси існуючих додатків провівши аналіз додатків, таких як Easyway, Moovit та Google Maps, показала, що вони мають свої переваги та недоліки. Потенційні користувачі таких додатків повинні враховувати ці аспекти при виборі сервісу.

Можливості застосування штучного інтелекту проаналізувавши плюси впровадження штучного інтелекту, зокрема машинного навчання, у мобільні додатки для відстежування маршрутного таксі та описала, що саме такий спосіб розробки відкриває нові можливості. Це включає в себе прогнозування часу прибуття, оптимізацію маршрутів, персоналізацію обслуговування та багато інших функцій.

Технології для впровадження штучного інтелекту з впровадженням машинного навчання в додатки для відстежування маршрутного таксі можна використовувати різні технології, такі як геолокація, аналітика даних, аналіз зображень та інші. Комбінація цих технологій дозволяє створити більш розумний та продуктивний додаток.

Усі ці висновки підкреслюють важливість розробки та впровадження мобільних додатків для відстежування маршрутного таксі міста, а також потенціал, який надає використання штучного інтелекту для покращення їхньої функціональності та забезпечення кращого обслуговування користувачів. Перш

за все, потрібно розуміти, наскільки масовими стали мобільні телефони в наш час. Вже на даний момент кількість активованих SIM карт перевищує населення землі. До того ж майже у кожного на вулиці можна побачити смартфон з операційною системою Android чи iOS. Саме тому замість GPS приладів, які будуть напряму встановлюватись на транспортних засобах та вимагатимуть фізичної та технічної підтримки в роботі, я пропоную використовувати мобільні телефони водіїв в якості GPS трекерів. Ця ідея чимось схожа на відомий додаток для водіїв таксі UBER, де користувач може спостерігати за переміщенням водія по карті в реальному часі, і додаток використовує GPS координати з телефону водія для того, щоб відображати розташування водія на карті. До того ж, якщо в дію вступає водій, як один з користувачів системи, він може отримувати оцінки, мати відгуки і т.д. Для покращення вже існуючого фунціоналу мого додатку було прийнято рішення інтегрувати технологію машинного навчання, для аналітики швидкості переміщення користувачів додатку та водіїв в реальному часі; технологію геолокації і GPS: використання GPS та геолокаційних даних для точного визначення місцезнаходження та руху транспортних засобів, що допомагає в реалізації послуг відстеження користувачів та водіїв та технологію обробки природних мов для аналізу відгуків та коментарів користувачів, щоб розуміти їхні вподобання та скарги, і автомачно створювати список покращень та проблем у додатку. Також планується розробити особистий кабінет для двох типів користувачів – пасажирів та водіїв, де вони зможуть відслідковувати свої минулі поїздки; дивитись середній час, витрачений на подолання певного маршруту; встановлювати час отримання сповіщень для завчасного планування часу початку маршруту.

РОЗДІЛ 2
ОГЛЯД МЕТОДІВ, ІНСТРУМЕНТІВ ТА ПІДХОДІВ ДЛЯ РЕАЛІЗАЦІЇ НОВОЇ ФУНКЦІОНАЛЬНОСТІ В ІНФОРМАЦІЙНІЙ СИСТЕМІ.

У цьому розділі я детальніше розгляну інновації та функціонал, які заплановані для впровадження в інформаційну систему моніторингу руху міського транспорту. Моя ціль полягає в створенні більш сучасного та корисного інструменту для пасажирів та водіїв, щоб поліпшити їх досвід використання громадського транспорту та забезпечити ефективнішу організацію міського руху.

Однією з головних інновацій є використання технології машинного навчання для аналізу швидкості переміщення користувачів додатку та водіїв у реальному часі. Ця функція дозволить системі не лише надавати актуальну інформацію про рух транспорту, але й прогнозувати оптимальний час прибуття та маршрути з урахуванням попереднього руху.

Другою ключовою інновацією є використання GPS та геолокаційних даних для точного визначення місцезнаходження та руху транспортних засобів. Це сприятиме вдосконаленню послуг відстеження користувачів та водіїв, а також забезпечить більш точну інформацію щодо розташування та доступності транспорту в режимі реального часу.

Третьою інновацією, яку я розгляну, є використання технології обробки природних мов для аналізу відгуків та коментарів користувачів. Ця функція допоможе зрозуміти вподобання та скарги наших користувачів та автоматично виявляти питання чи пропозиції щодо поліпшень у додатку.

Крім цього, я також розроблю особистий кабінет для двох типів

користувачів – пасажирів та водіїв, де вони зможуть відслідковувати свої минулі поїздки, дивитись середній час, витрачений на подолання певного маршруту та встановлювати час отримання сповіщень для завчасного планування поїздок.

У цьому розділі ми розглянемо методи, інструменти та підходи, які використовуватимуться для успішної реалізації цих інновацій та забезпечення їх ефективної роботи інформаційній системі моніторингу руху міського транспорту.

2.1. Використання технології машинного навчання

2.1.1 Аналіз використання технології машинного навчання для аналізу швидкості переміщення користувачів та водіїв в реальному часі.

Технологія машинного навчання може бути використана для аналізу швидкості переміщення користувачів та водіїв у реальному часі завдяки своїй здатності автоматично виявляти та прогнозувати зміни в швидкості руху на основі інформації, яка надходить з додатку, з датчиків мобільних пристроїв та транспортних засобів.

Ось кроки та способи, якими машинне навчання може бути використано для цієї задачі:

Збір даних: Першим кроком є збір даних про рух користувачів та водіїв. Ці дані можуть бути зібрані з мобільних додатків, смартфонів, GPS-пристроїв, а також з інших джерел, які можуть надавати інформацію про місцезнаходження та швидкість руху. Збір даних - це один із ключових етапів для впровадження технології машинного навчання для аналізу швидкості переміщення користувачів та водіїв в реальному часі у системі моніторингу руху міського транспорту.

Перше, що потрібно визначити, це джерела даних. У багатьох випадках, головним джерелом інформації про рух транспорту та користувачів є мобільні

додатки, які встановлені на смартфонах пасажирів та водіїв. Ці додатки можуть збирати геолокаційні дані, які включають координати місцезнаходження та інформацію про швидкість руху. Інші джерела можуть включати GPS-пристрої в транспортних засобах або системи моніторингу руху.

Після визначення джерел даних, потрібно розглянути формат даних. Дані можуть надходити у різних форматах, таких як текстові файли, JSON-структури, бази даних або навіть потокові дані у режимі реального часу. Важливо визначити, яким чином ці дані будуть зберігатися та передаватися до системи аналізу.

Збір геолокаційних даних повинен бути здійснений з дотриманням приватності користувачів і відповідно до законодавства про захист даних. Дані повинні бути анонімізовані та захищені від несанкціонованого доступу. Це особливо важливо, оскільки ці дані можуть містити особисту інформацію.

Для збору геолокаційних даних з мобільних додатків я повинна інтегрувати функціональність отримання доступу до GPS та інших сенсорів в додаток. Дані можуть надсилатися до сервера системи моніторингу через безпечне з'єднання, наприклад, за допомогою HTTPS-протоколу.

Після отримання даних на сервері, вони можуть бути оброблені та агреговані. Це включає в себе видалення непотрібної інформації, фільтрацію аномальних даних і обчислення різних статистичних показників, таких як середня швидкість руху на певному маршруті.

Зберігання даних відіграє важливу роль, оскільки історичні дані можуть бути корисними для аналізу та навчання моделей машинного навчання. Дані можуть бути зберігатися в базі даних або в хмарному сховищі, в залежності від обсягу та вимог до масштабованості.

Дані мають бути постійно оновлювані, оскільки рух на дорогах є динамічним і змінюється з часом. Система повинна бути здатною синхронізувати дані між різними джерелами та пристроями для забезпечення актуальності інформації.

Збір даних - це важливий етап, який вимагає уважного планування та налагодження для забезпечення точності та надійності інформації, яка буде використовуватися для аналізу швидкості переміщення користувачів та водіїв у реальному часі.

Побудова навчального набору даних: Для того щоб навчити модель машинного навчання передбачати швидкість руху, потрібно побудувати навчальний набір даних, який містить приклади швидкості руху та відповідні характеристики, такі як час доби, день тижня, тип транспорту, маршрут і т. д.

Побудова навчального набору даних - це важливий етап у використанні технології машинного навчання для аналізу швидкості переміщення користувачів та водіїв в реальному часі. Він передбачає створення даних, які будуть використовуватися для тренування моделі машинного навчання. Я розглянула цей процес більш детально:

- Визначення параметрів даних:

Спочатку важливо визначити, які саме параметри даних будуть включені в навчальний набір. Для аналізу швидкості переміщення можуть бути важливими параметри, такі як:

- Час та дата: Для врахування сезонних та часових змін у швидкості руху.
- Місцезнаходження: Геолокаційні координати, які визначають місце руху.
- Швидкість руху: Фактична швидкість переміщення користувача або водія.
- Тип транспорту: Інформація про вид транспорту, на якому знаходиться користувач або водій.
- Маршрут: Інформація про конкретний маршрут або ділянку дороги.

- Збір даних:

Після визначення параметрів даних, потрібно зібрати відповідні дані. Це може включати в себе встановлення мобільних додатків на смартфони користувачів та водіїв, які будуть автоматично записувати дані про рух. Збір

даних також може включати в себе використання GPS-пристроїв у транспортних засобах або сенсорів на дорогах.

- Формат даних:

Дані повинні бути зібрані та збережені в структурованому форматі, який відповідає параметрам, визначеним на попередньому етапі. Зазвичай це може бути таблиця або база даних, де кожен рядок представляє запис про рух, а стовпці представляють параметри.

- Анотація даних:

Важливо анотувати дані, щоб визначити правильні вихідні значення швидкості руху. Це може вимагати вручного або автоматизованого аналізу. Наприклад, швидкість може бути виміряна у кілометрах на годину (км/год) або метрах на секунду (м/с).

- Очистка даних:

Для покращення якості даних може бути потрібна очистка. Це включає в себе видалення аномальних даних, коригування помилкових значень та заповнення відсутніх даних.

- Створення навчального та тестового набору:

Дані можуть бути поділені на два набори: навчальний та тестовий. Навчальний набір використовується для тренування моделі машинного навчання, тоді як тестовий набір використовується для оцінки точності моделі.

- Зберігання та резервне копіювання:

Дані повинні бути збережені в безпечному місці з можливістю резервного копіювання. Це допоможе запобігти втраті даних у разі непередбачуваних ситуаціях, таких як відмова обладнання або видалення даних.

Побудова навчального набору даних є важливим кроком у створенні моделі машинного навчання, яка може передбачати швидкість переміщення. Якість та точність моделі значною мірою залежить від якості цього навчального набору.

Вибір моделі машинного навчання: Наступним кроком є вибір моделі машинного навчання, яка найкраще підходить для завдання передбачення швидкості руху. Моделі, такі як регресія, нейронні мережі або методи часових рядів, можуть бути використані для цієї задачі.

Вибір моделі машинного навчання - це важливий етап в розробці системи для аналізу швидкості переміщення користувачів та водіїв в реальному часі. Вибір моделі визначає, як саме моя система буде навчатися та робити передбачення на основі навчальних даних. Тому я розглянула цей процес докладніше:

- Визначення задачі:

Спочатку необхідно чітко визначити, яку саме задачу ви хочете вирішити за допомогою машинного навчання. У вашому випадку, це передбачення швидкості переміщення на основі доступних даних.

- Розгляд різних типів моделей:

Існує велика кількість різних типів моделей машинного навчання, і вибір залежить від природи вашої задачі. Основні категорії моделей включають:

- Лінійна регресія: Використовується для прогнозування числових значень на основі лінійних залежностей між вхідними та вихідними змінними.
- Нейронні мережі: Можуть використовуватися для складних задач, включаючи роботу з послідовними даними, такими як часові ряди.
- Методи часових рядів: Ідеально підходять для аналізу швидкості переміщення в часовому контексті.
- Дерева рішень і випадковий ліс: Використовуються для задач класифікації та регресії, і можуть розглядатися для задачі передбачення швидкості руху на основі факторів, які впливають на неї.

- Вибір моделі:

Після розгляду різних типів моделей, потрібно обрати той, який найкраще відповідає моїй задачі та характеру використовуваних даних. Важливо також

враховувати обчислювальну складність моделі та ресурси, доступні для її навчання та використання.

- Настроювання гіперпараметрів:

Вибір моделі також включає в себе налаштування гіперпараметрів. Гіперпараметри - це параметри моделі, які не навчаються під час тренування, але впливають на її ефективність. Наприклад, для нейронних мереж це може бути кількість шарів та нейронів у кожному шарі.

- Навчання та оцінка моделі:

Після вибору моделі і налаштування параметрів, модель повинна бути навчена на вашому навчальному наборі даних. Після тренування моделі, важливо оцінити її точність та здатність до передбачення швидкості руху на вашому тестовому наборі даних.

- Оптимізація та налаштування:

Може знадобитися оптимізувати модель та налаштувати її параметри, щоб досягнути кращої точності. Цей процес може включати в себе зміну гіперпараметрів, додавання додаткових функцій або використання альтернативних алгоритмів.

- Перевірка на відмовостійкість та масштабованість:

Важливо перевірити, наскільки моя модель відмовостійка до помилок та чи вона може масштабуватися для обробки великого обсягу даних в реальному часі.

Вибір моделі машинного навчання - це важливий етап, який може впливати на якість та ефективність системи аналізу швидкості переміщення. Важливо враховувати характер вашої задачі та доступні ресурси при виборі моделі.

Навчання моделі: Використовуючи навчальний набір даних, модель машинного навчання навчається передбачати швидкість руху на основі доступної інформації. Під час навчання модель аналізує залежності між різними параметрами та швидкістю руху.

Навчання моделі машинного навчання є одним із ключових етапів в розробці системи для аналізу швидкості переміщення користувачів та водіїв в реальному часі. Цей етап передбачає навчання моделі на навчальному наборі даних, так щоб вона могла робити передбачення на основі нових вхідних даних. Долі розглянула цей процес більш детально:

- Підготовка даних:

Спочатку необхідно підготувати дані для навчання моделі. Це включає в себе:

- Нормалізацію даних: Масштабування вхідних функцій для забезпечення стандартизації та обробки невеликих значень.
- Розділення даних: Розбиття навчального набору на тренувальну та валідаційну підвибірки для оцінки точності моделі.
- Обробка відсутніх даних: Заповнення або видалення відсутніх значень в даних.

- Вибір алгоритму навчання:

Вибір самого алгоритму навчання або методу оптимізації визначає, як саме модель буде навчатися на даних. Це може бути градієнтний спуск, метод найменших квадратів, метод опорних векторів, нейронні мережі, тощо.

- Ініціалізація параметрів моделі:

Модель потребує початкової ініціалізації параметрів перед початком навчання. Ці початкові значення можуть бути вибрані випадково або з урахуванням деяких експертних знань.

- Процес навчання:

Під час навчання модель аналізує навчальний набір даних і намагається встановити залежність між вхідними та вихідними змінними. Цей процес може бути ітеративним, і модель оновлює свої параметри так, щоб мінімізувати помилку передбачення.

- Функція втрат та оптимізація:

Для визначення того, наскільки точно модель передбачає результати, використовується функція втрат (або функція помилки). Мета полягає в тому, щоб ця функція була мінімізована під час навчання. Алгоритм оптимізації (наприклад, градієнтний спуск) використовується для налаштування параметрів моделі так, щоб мінімізувати цю функцію втрат.

- Контроль над перенавчанням:

Перенавчання - це ситуація, коли модель навчена настільки добре на навчальних даних, що вона не може ефективно узагальнювати на нові дані. Для запобігання перенавчанню використовуються техніки регуляризації, розділення навчальних та валідаційних даних, та інші методи.

- Моніторинг та оцінка навчання:

Під час навчання моделі важливо моніторити її ефективність на валідаційних даних. Це допомагає визначити, чи досягла модель заданого рівня точності. Якщо навчання не вдається, можливо, потрібно змінити архітектуру моделі, гіперпараметри або дані.

- Завершення навчання та зберігання моделі:

Після досягнення бажаної точності навчання модель завершується, і її параметри можуть бути збережені для подальшого використання.

Навчання моделі машинного навчання - це процес, який може вимагати багато експериментів та налаштувань.

Тестування та оцінка моделі: Після навчання модель перевіряється на тестовому наборі даних, щоб оцінити її точність та здатність передбачати швидкість руху. Цей процес допомагає визначити, наскільки ефективно модель працює у реальних умовах.

Тестування та оцінка моделі є важливим етапом в розробці системи для аналізу швидкості переміщення користувачів та водіїв в реальному часі. Після навчання моделі на навчальних даних важливо перевірити її ефективність та здатність до передбачення на нових даних. Ось докладний опис цього процесу:

- Вибір тестового набору даних:

Спочатку необхідно вибрати тестовий набір даних, який буде використовуватися для оцінки моделі. Це повинні бути дані, які модель раніше не бачила і які відображають реальні умови її застосування.

- Застосування моделі до тестових даних:

Потім модель застосовується до тестового набору даних для створення передбачень швидкості переміщення на основі вхідних даних.

- Розрахунок метрик ефективності:

Після отримання передбачень важливо розрахувати різні метрики ефективності, які допоможуть оцінити якість моделі. Деякі з популярних метрик включають:

− Середньоквадратична помилка (MSE): Вимірює середньоквадратичну різницю між передбаченими значеннями та фактичними значеннями швидкості переміщення. Менший MSE вказує на кращу точність.

− Коефіцієнт детермінації (R-squared): Визначає, наскільки змінні в моделі відповідають дійсним даним. Зазвичай вимірюється від 0 до 1, де 1 вказує на ідеальну відповідність.

− Середня абсолютна похибка (MAE): Вимірює середню абсолютну різницю між передбаченими та фактичними значеннями. MAE дозволяє оцінити розмір помилок без залежності від їхнього знаку.

- Аналіз результатів:

Після розрахунку метрик ефективності важливо аналізувати результати. Якщо модель має низьку точність або не відповідає вимогам, може бути необхідно внести зміни в архітектуру моделі, гіперпараметри або дані.

- Перевірка на перенавчання та недонавчання:

Важливо також перевірити, чи модель перенавчена (вона добре пасує до навчальних даних, але погано узагальнює на нових) або недонавчена (вона не виявляє закономірностей у даних). Це допоможе визначити, чи потрібно вносити корективи в навчання.

- Перевірка на стабільність та відмовостійкість:

Модель також повинна бути перевірена на стабільність та відмовостійкість. Це означає, що вона має працювати ефективно під час непередбачуваних ситуацій та в умовах, відмінних від тих, на яких вона була навчена.

- Повторення процесу за необхідності:

Якщо результати тестування не задовольняють очікування, можливо, доведеться повторити процес навчання з новими даними, параметрами або моделями.

Тестування та оцінка моделі - це важливий крок у розробці системи для аналізу швидкості переміщення та допомагають визначити, наскільки ефективна та точна ваша модель у вирішенні завдань аналізу швидкості руху.

Інтеграція з додатком: Остаточна модель може бути інтегрована з інформаційною системою моніторингу транспорту таким чином, що вона надає актуальну інформацію про швидкість руху користувачам та водіям у реальному часі.

Постійне оновлення та налаштування: Модель машинного навчання може потребувати постійного оновлення та налаштування для врахування змінних умов на дорогах та поведінки користувачів.

Загалом, машинне навчання дозволяє системі моніторингу руху міського транспорту не лише збирати дані про швидкість руху, але й надавати користувачам та водіям корисну та актуальну інформацію для більш комфортної та ефективної подорожі.

2.1.2 Вибір алгоритмів машинного навчання та підходів до навчання моделей.

Вибір алгоритмів машинного навчання та підходів до навчання моделей є важливими завданнями в розробці системи для аналізу швидкості переміщення

користувачів та водіїв в реальному часі. Цей вибір визначає ефективність та точність вашої моделі. Ось деякі широкі та відкриті обговорення щодо вибору алгоритмів та підходів:

Тип задачі та дані: Перший крок - визначення типу задачі, яку я намагаюсь вирішити. У моєму випадку це задача регресії (прогнозування числового значення - швидкості переміщення). Також важливо врахувати природу даних, наприклад, часові ряди, геолокаційні дані, текстові коментарі тощо.

Вибір алгоритмів: Вибір алгоритмів залежить від задачі. Лінійна регресія може бути добрим вибором для базової моделі, якщо є числові функції, які впливають на швидкість переміщення. Якщо я матиму часові ряди, то методи часових рядів або нейронні мережі можуть бути корисними. Важливо розглядати не тільки класичні алгоритми, але й сучасні підходи, такі як глибоке навчання.

Регуляризація та оптимізація гіперпараметрів: Для кращого навчання моделі рекомендується використовувати методи регуляризації, такі як L1 та L2 регуляризація. Також важливо налаштувати гіперпараметри моделі, такі як швидкість навчання, кількість шарів та нейронів у нейронних мережах.

Валідація та перевірка на перенавчання: Після навчання моделі важливо використовувати валідаційний набір даних для перевірки її точності та визначення можливості перенавчання. Важливо пам'ятати, що модель повинна бути здатною узагальнювати на нові дані, а не просто пам'ятати навчальний набір.

Використання ансамблів: У деяких випадках ефективність може бути покращена за допомогою ансамблів моделей. Наприклад, використання випадкового лісу або градієнтного бустінгу може допомогти підвищити точність передбачень.

Врахування обчислювальних ресурсів: Врахування обчислювальних ресурсів є важливим аспектом в розробці системи для аналізу швидкості

переміщення користувачів та водіїв в реальному часі. Обчислювальні ресурси можуть включати в себе обчислювальну потужність (процесори), обсяг доступної оперативної пам'яті, а також можливості графічного процесора (GPU) у випадку глибокого навчання. Важливо враховувати ці ресурси при виборі алгоритмів та підходів до навчання моделей.

Постійне навчання та оновлення моделі: При розробці системи реального часу важливо розглядати можливість постійного навчання та оновлення моделі на нових даних. Це допоможе забезпечити актуальність та точність передбачень з плином часу.

Аналіз результатів та оптимізація: Після впровадження системи важливо аналізувати її результати та виконувати оптимізацію, якщо це необхідно. Результати можуть вказувати на можливість покращення або корекції в роботі системи.

Вибір алгоритмів та підходів до навчання моделей є процесом, який може вимагати експериментів та налаштувань. Важливо розглядати конкретні особливості завдання та обмеження, щоб вибрати найбільш підходящі методи для досягнення бажаних результатів у системі аналізу швидкості переміщення.

2.2. Використання GPS та геолокаційних даних

GPS (Global Positioning System) та геолокаційні дані є надзвичайно важливими складовими системи моніторингу транспорту. Вони надають можливість визначити точне місцезнаходження транспортних засобів у реальному часі та забезпечують безліч переваг для якості та ефективності моніторингової системи.

GPS надає дуже високу точність у визначенні місцезнаходження, що дозволяє точно відстежувати рух транспортних засобів. Ця точність особливо важлива в міському середовищі, де транспорт може рухатися на коротких відстанях.

GPS дозволяє отримувати дані про місцезнаходження транспортних засобів у реальному часі. Це робить систему моніторингу динамічною і здатною реагувати на зміни у русі та обставинах. Геолокаційні дані дозволяють системі моніторингу оптимізувати маршрути та планувати шляхи для транспортних засобів. Це може покращити ефективність та скоротити час в дорозі.

GPS може бути використаний для покращення безпеки транспорту та пасажирів. Наприклад, в разі аварій або надзвичайних ситуацій можна негайно локалізувати транспортний засіб та надати допомогу. Також може бути використаний для відстеження робочих годин водіїв, що важливо з точки зору дотримання правил щодо обмеження часу за кермом та запобігання перевтомленості. Геолокаційні використовуються для моніторингу викидів, споживання пального та інших екологічних параметрів транспортних засобів, сприяючи екологічній стійкості транспорту. Дозволяє відстежувати рух вантажу та пасажирів, що може бути важливим для логістики, безпеки та управління ресурсами.

Геолокаційні дані можуть бути використані для створення звітів, аналітики та візуалізації руху транспортних засобів. Це допомагає приймати обґрунтовані рішення та покращувати продуктивність системи.

Усі ці переваги GPS та геолокаційних даних дозволяють створити потужну систему моніторингу транспорту, яка сприяє покращенню роботи транспортних компаній, пасажирів та влади, а також сприяє загальній безпеці, ефективності та збереженню ресурсів.

2.2.1 Інструменти та технології для збору та обробки геолокаційних даних

Використання GPS для точного визначення місцезнаходження та руху транспортних засобів є ключовим елементом систем моніторингу транспорту.

GPS (Global Positioning System) - це система супутникового навігації, яка дозволяє визначити точне географічне положення об'єкта в будь-якому місці на Землі. Ось як GPS використовується для цієї цілі, разом із засобами та технологіями для збору та обробки геолокаційних даних:

- Супутникові сигнали GPS:

GPS отримує сигнали від супутників, які обертаються навколо Землі. Ці супутники надсилають сигнали із точно відомими часами та положеннями. GPS-приймачі на транспортних засобах приймають ці сигнали та вимірюють час, який потрібний для того, щоб сигнал досягнув точки прийому.

- Триангуляція:

GPS-приймачі використовують принцип триангуляції для визначення свого місцезнаходження. Вони приймають сигнали від мінімум трьох супутників і використовують часові затримки для визначення відстаней до кожного супутника. Після цього вони обчислюють своє точне місцезнаходження на основі перетину сфер, центри яких знаходяться в точках супутників.

- DGPS (Диференційна GPS):

Для ще більшої точності може використовуватися технологія диференційного GPS. Вона включає в себе станції, які знаходяться на відомих точках і збирають GPS-сигнали. Ці станції можуть коригувати дані, надіслані GPS-приймачем, для покращення точності місцезнаходження.

- GPS-приймачі та технології збору даних:

GPS-приймачі можуть бути встановлені на транспортних засобах та приймати сигнали від супутників. Існують різні типи GPS-приймачів, включаючи вбудовані в сучасні смартфони та спеціалізовані пристрої для великих флотів транспорту.

- Обробка та передача даних:

Геолокаційні дані, отримані від GPS-приймачів, можуть бути оброблені та передані на сервери для подальшого аналізу. Це може включати в себе збереження координат, швидкості, часу, напрямку руху та іншої інформації.

- Технології передачі даних:

Для передачі геолокаційних даних можуть використовуватися різні технології, такі як мобільний зв'язок (3G, 4G, 5G), Wi-Fi, Bluetooth, LoRa та інші.

- Системи моніторингу та аналітики:

Отримані геолокаційні дані можуть бути використані в системах моніторингу та аналітики для відстеження руху транспортних засобів, планування маршрутів, аналізу продуктивності та багатьох інших завдань.

Загалом, використання GPS та геолокаційних даних дозволяє точно відстежувати рух транспортних засобів, що є критичним для оптимізації транспортних процесів, забезпечення безпеки та покращення якості обслуговування.

2.3. Технологія обробки природних мов для аналізу відгуків та коментарів користувачів

Обробка природних мов (Natural Language Processing, NLP) є потужним інструментом для розуміння вподобань та скарг користувачів в інформаційних системах. Ця технологія дозволяє комп'ютерам аналізувати та інтерпретувати людську мову, що включає в себе текстові повідомлення, коментарі, відгуки та інші форми текстуального виразу. Ось докладніше про використання обробки природних мов для розуміння вподобань та скарг користувачів:

Аналіз відгуків та коментарів: За допомогою NLP можна аналізувати великі обсяги текстових відгуків та коментарів користувачів. Алгоритми NLP можуть визначити настрій (позитивний, негативний, нейтральний) та ідентифікувати ключові слова та фрази, які вказують на вподобання або скарги.

Створення категорій та тематичний аналіз: NLP може автоматично створювати категорії та класифікувати відгуки за темами. Наприклад, визначення, чи стосуються відгуки продукту його функціональності, якості, ціни тощо. Це дозволяє ідентифікувати конкретні аспекти, які цікавлять користувачів.

Виявлення ключових слів та термінів: NLP може виділяти ключові слова та терміни, які найчастіше зустрічаються у відгуках та коментарях. Це допомагає ідентифікувати основні проблеми або популярні аспекти, на які звертають увагу користувачі.

Сентиментний аналіз: Застосовуючи сентиментний аналіз, NLP може визначити, як користувачі ставляться до певного продукту або послуги. Це допомагає виявити негативні скарги та пропозиції для покращення, а також позитивні відгуки для підвищення реклами.

Автоматична обробка скарг та запитів: NLP може бути використаний для автоматичної обробки скарг та запитів користувачів. Наприклад, автоматизовані системи можуть розподіляти скарги до відповідних відділів або створювати автоматичні відповіді на типові запити.

Персоналізовані рекомендації: На основі аналізу відгуків та коментарів NLP може генерувати персоналізовані рекомендації для користувачів. Це може покращити їхній досвід та задоволення від використання продукту або послуги.

Аналіз трендів та попиту: NLP допомагає виявляти тренди та попит серед користувачів, що дозволяє підприємствам адаптувати свої пропозиції та стратегії.

Загалом, обробка природних мов створює можливість зрозуміти вподобання та скарги користувачів на більш глибокому рівні, що дає змогу покращити продукти та послуги, підвищити задоволення клієнтів та підтримати впровадження змін в відповідь на їхні потреби.

2.3.1. Інструменти та бібліотеки для аналізу тексту, опис методів виявлення ключових слів та тематичного аналізу коментарів.

Для реалізації ідей мого дипломного проекту, потрібні інструменти та бібліотеки для аналізу тексту, які здатні працювати з великими обсягами текстових даних, виконувати сентиментний аналіз, класифікацію тексту, виділення ключових слів та інші завдання NLP. Ось декілька інструментів та бібліотек, що можуть бути використані:

NLTK (Natural Language Toolkit): NLTK - це популярна бібліотека для обробки природних мов в Python. Вона має велику кількість інструментів для токенізації, стеммінгу, лематизації, сентиментного аналізу, класифікації тексту та інших NLP-завдань.

spaCy: spaCy - це швидка та ефективна бібліотека для обробки тексту, яка також має моделі для визначення іменованих сутностей та інших завдань NLP. Вона може бути корисною для аналізу великих обсягів текстових даних у реальному часі.

TextBlob: TextBlob - це простий у використанні інструмент для аналізу тексту, який має вбудовані функції для сентиментного аналізу, перекладу тексту, витягнення ключових слів та інших завдань.

scikit-learn: Scikit-learn - це бібліотека для машинного навчання в Python, яка також має функціонал для класифікації тексту та роботи з векторами слів.

Gensim: Gensim - це бібліотека для роботи з тематичним моделюванням та векторною обробкою тексту. Вона корисна для витягнення тем з тексту та створення векторних представлень слів.

Transformers (Hugging Face): Бібліотека Transformers від Hugging Face містить передові моделі для різних завдань NLP, включаючи сентиментний аналіз, класифікацію тексту та переклад. Ви можете використовувати ці моделі для досягнення високих результатів у вашому проекті.

Stanford NLP: Якщо ви шукаєте надійні інструменти для парсингу та аналізу тексту, Stanford NLP може бути вибором. Вона включає в себе моделі для багатьох завдань NLP та має широкий функціонал.

FastText: FastText - це бібліотека для роботи з векторною обробкою тексту, яка може бути корисною для класифікації тексту та векторизації слів.

Вибір конкретних інструментів та бібліотек буде залежати від конкретних завдань під час написання інформаційної системи. Деякі з цих інструментів можна комбінувати для досягнення найкращих результатів.

Методи виявлення ключових слів та тематичного аналізу коментарів є важливими для витягнення цінної інформації з великих обсягів текстових даних. Ось деякі методи для цих завдань:

Аналіз частоти слів (TF-IDF): Метод TF-IDF (Term Frequency-Inverse Document Frequency) визначає важливість кожного слова в тексті в порівнянні з іншими документами. Слова, які високочастотні в даному документі, але низько високочастотні в інших, вважаються ключовими. TF-IDF дозволяє виокремити ключові слова для кожного коментаря та визначити їхню важливість.

Виокремлення іменованих сутностей: Визначення та виділення іменованих сутностей (наприклад, імена, місця, організації) може допомогти виявити ключові теми в коментарях. Багато NLP-бібліотек мають моделі для розпізнавання іменованих сутностей.

LDA (Latent Dirichlet Allocation): Метод LDA - це популярний алгоритм тематичного аналізу. Він допомагає виявити теми, які присутні у текстах, і визначити, які слова пов'язані з кожною темою. LDA може допомогти виокремити ключові теми в коментарях та призначити їх кожному коментарю.

NMF (Non-Negative Matrix Factorization): NMF - це інший метод тематичного аналізу, який розкладає матрицю текстових даних на дві матриці - тематичну та словникову. Він також допомагає виокремити ключові теми та ваги слів для кожної теми.

Word Embeddings: Методи векторної обробки тексту (наприклад, Word2Vec, FastText) можуть створити векторні представлення слів, які можна використовувати для визначення семантичних відношень між словами та визначення схожих слів. Це може допомогти виявити ключові слова та теми.

Сентиментний аналіз: Аналіз сентименту допомагає визначити настрій коментаря (позитивний, негативний, нейтральний). Коментарі з високим сентиментом можуть містити ключові слова, які вказують на проблеми або позитивні аспекти.

Асоціативні правила: Методи асоціативних правил, такі як Apriori або FP-Growth, дозволяють виявити зв'язки між словами та фразами у коментарях. Це може виявити ключові слова, які часто зустрічаються разом.

Моделі глибокого навчання: Деякі глибокі нейронні мережі, такі як Recurrent Neural Networks (RNN) та Transformer, можуть бути використані для виконання тематичного аналізу та виявлення ключових слів.

Комбінування цих методів та інструментів може допомогти мені досягти кращих результатів у виявленні ключових слів та тематичному аналізі коментарів у інформаційній системі.

2.4. Розробка особистого кабінету для користувачів

Організація особистого кабінету для пасажирів та водіїв є важливою складовою вашої інформаційної системи моніторингу руху міського транспорту. Ось деякі основні можливості кабінету користувача:

- Особистий кабінет пасажира

Реєстрація та вхід: Пасажир повинен мати можливість зареєструватися в системі, вказавши особисті дані, такі як ім'я, номер телефону та електронну адресу. Після реєстрації він може увійти в особистий кабінет за допомогою свого логіна та паролю.

Профіль користувача: Пасажир може переглядати та редагувати свій профіль, додавати фотографію, змінювати контактні дані та інші особисті відомості.

Історія поїздок: Пасажир може переглядати історію своїх минулих поїздок, включаючи дату, час, маршрут, витрати часу та інші деталі.

Пошук та бронювання маршрутів: Пасажир може шукати доступні маршрути міського транспорту, переглядати розклад руху, вартість та іншу інформацію. Він може також бронювати квитки на певний маршрут та зручний час.

Сповіщення та повідомлення: Пасажир може підписатися на сповіщення щодо змін у розкладі, транспортних пригодах, акціях та інші важливі повідомлення.

Зворотний зв'язок та скарги: Пасажир може залишати відгуки, скарги та пропозиції через систему зворотного зв'язку та спілкуватися з службою підтримки.

- Особистий кабінет водія:

Реєстрація та вхід: Водій також має можливість реєстрації та входу в систему, але його профіль відрізняється від пасажира.

Профіль водія: Водій може створити та редагувати свій профіль, включаючи особисті дані, фотографію, номер транспортного засобу та інші деталі. Важливо також додати інформацію про маршрути, які він обслуговує.

Моніторинг руху транспорту: Водій може переглядати мапу з розташуванням свого транспортного засобу в реальному часі, а також інформацію про швидкість руху, розклад зупинок та інше.

Історія поїздок: Водій може переглядати історію своїх минулих поїздок, включаючи дату, час, маршрут та інші деталі.

Фінансова інформація: Важливо мати доступ до фінансової інформації, такої витрати на пальне та інші фінансові аспекти.

Зворотний зв'язок та повідомлення: Водій може спілкуватися з пасажирами та службою підтримки через систему повідомлень.

Організація особистого кабінету для пасажирів та водіїв допомагає забезпечити зручний та інтерактивний досвід користувачів у вашій інформаційній системі моніторингу руху міського транспорту, а також дозволяє їм ефективно керувати своїми поїздками та отримувати необхідну інформацію.

Висновки до Розділу 2

У розділі 2 мого дипломного проекту я розглянула різні методи, інструменти та підходи, які будуть використані для реалізації інновацій у системі моніторингу руху міського транспорту. Підсумовуючи цей розділ, можна виділити декілька ключових висновків:

Використання машинного навчання: Впровадження технології машинного навчання дозволить аналізувати швидкість переміщення користувачів та водіїв в реальному часі. Це допоможе оптимізувати роботу системи, покращувати точність та надійність інформації та надавати користувачам актуальні дані.

Геолокація та GPS: Використання GPS та геолокаційних даних дозволить точно визначати місцезнаходження та рух транспортних засобів. Це необхідно для відстеження руху автобусів, тролейбусів і інших транспортних засобів у реальному часі.

Обробка природних мов: Використання обробки природних мов дозволить розуміти вподобання та скарги користувачів, а також автоматично створювати список покращень та проблем у додатку. Це допомагає забезпечити задоволення користувачів та покращити якість обслуговування.

Особистий кабінет для користувачів: Розробка особистого кабінету для пасажирів та водіїв дозволяє забезпечити зручний та інтерактивний досвід користувачів, де вони можуть керувати своїми поїздками, отримувати необхідну інформацію та взаємодіяти з системою.

Загальний результат розділу 2 полягає в тому, що за допомогою інноваційних методів, інструментів та підходів, я створила більш зручну, ефективну та інтерактивну інформаційну систему моніторингу руху міського транспорту, яка сприяє покращенню якості обслуговування користувачів та забезпечує актуальну та достовірну інформацію.

РОЗДІЛ 3
ОПИС АРХІТЕКТУРИ ТА РОЗРОБЛЕНИХ АЛГОРИТМІВ

3.1. Загальний опис архітектури

Основна структура цього додатку розділена на дві важливі частини: клієнтську та серверну. Розділ клієнта, створений за допомогою Javascript і фреймворку React Native, відповідає за демонстрацію поточного розташування маршрутних таксі на міській карті, показуючи маршрути та місця зупинок. Крім того, клієнтська частина передає поточні координати телефону, коли водій увійшов у додаток.

Серверна частина, розроблена на Node.JS, служить для забезпечення взаємодії між клієнтом і базою даних. У моєму випадку, клієнт відправляє запити на сервер для отримання даних про наявні маршрутні таксі, їхні поточні місця розташування, маршрути та зупинки. Коли водій використовує додаток, сервер отримує запити від клієнта для оновлення даних про розташування телефону водія, яке відображає фактичне розташування транспортного засобу.

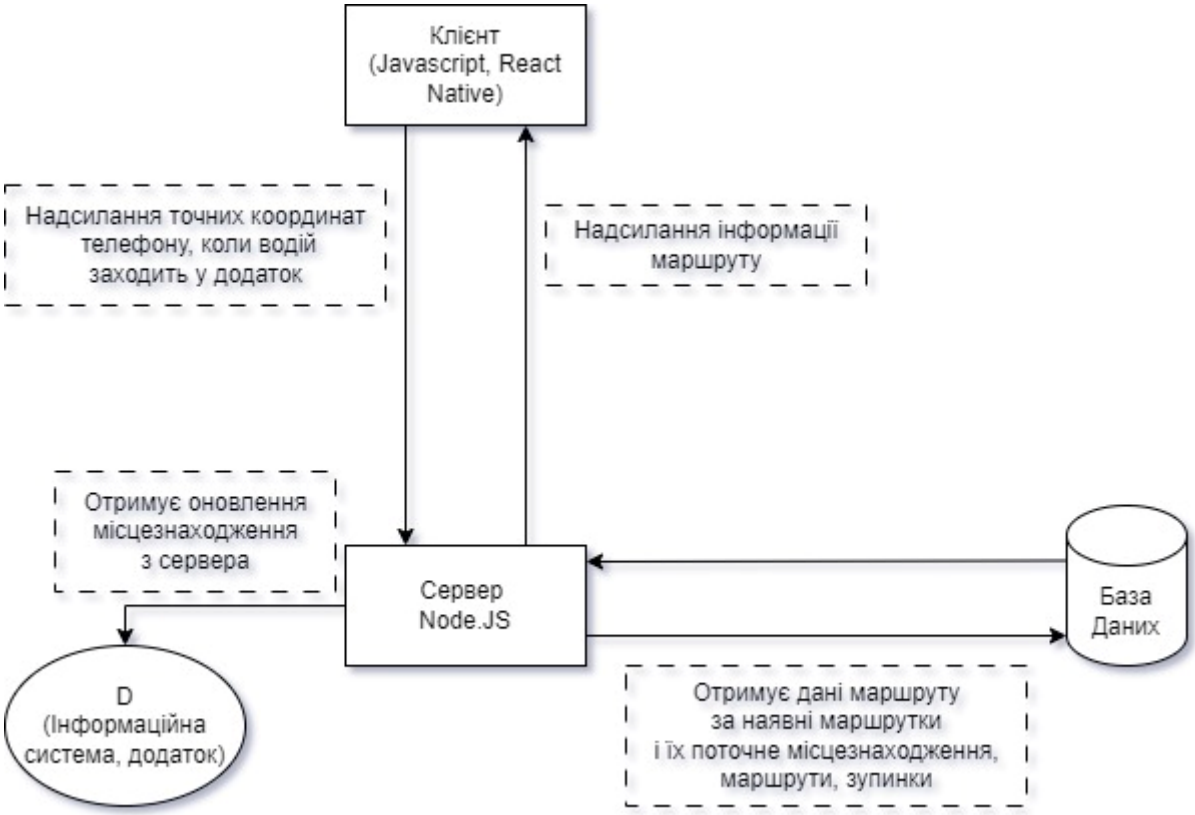

Рис.3.1. Діаграма зв’язку сервера та бази даних

3.1.1. Основні вимоги до функціоналу для аналітики переміщення користувачів та водіїв в реальному часі.

- Аналітика переміщення користувачів

Збір даних: Функціонал повинен мати доступ до історичних даних про рух маршрутних таксі на конкретному маршруті та актуальні дані про рух транспорту в реальному часі. Це включає дані про розташування транспорту (за допомогою GPS), швидкість руху, зупинки, час, який витрачається на кожній зупинці тощо.

Критерії прийняття:

1.1. Система повинна мати можливість збирати історичні дані про рух маршрутних таксі на конкретному маршруті.

1.2. Збір історичних даних повинен включати в себе інформацію про час відправлення та прибуття маршрутного таксі на кожній зупинці, маршрути та

їхні відстані, інформацію про транспортні пригоди та затримки, які сталися під час поїздок.

1.3. Система повинна здійснювати збір актуальних даних про рух маршрутного таксі в реальному часі. Ці дані повинні бути зібрані з GPS-датчиків, які встановили водії та які використовують додаток.

1.4. Для забезпечення точності та надійності збору даних, система повинна автоматично фільтрувати та видаляти помилкові або некоректні дані.

1.5. Зібрані дані повинні зберігатися в безпечному та захищеному від несанкціонованого доступу сховищі.

1.6. Система повинна мати механізм регулярного оновлення історичних даних для врахування нової інформації та змін у русі маршрутних таксі.

1.7. Для забезпечення якості даних, система повинна мати можливість проводити аналіз та перевірку історичних даних на наявність аномалій та відхилень.

1.8. Система повинна забезпечити можливість резервного копіювання та відновлення даних в разі втрати або пошкодження.

Аналіз та обробка даних: Система повинна обробляти дані, враховуючи фактори, такі як трафік, погодні умови, події на дорозі та інші фактори, що можуть впливати на час прибуття транспорту на зупинку.

Критерії прийняття:

1.1. Система повинна забезпечувати неперервний та автоматичний збір актуальних даних про рух маршрутних таксі та події на дорозі у реальному часі.

1.2. Для аналізу даних про трафік, система повинна інтегруватися з відомими джерелами трафікової інформації, такими як GPS-навігатори, карти з реальним часом та інші джерела, і використовувати ці дані для розрахунку поточних умов руху.

1.3. Аналіз даних повинен бути проведений в реальному часі та оновлюватися на підставі нових інформаційних внесків, щоб надавати користувачам точну та актуальну інформацію.

1.4. Система повинна мати механізм корегування прогнозів прибуття у випадку, якщо з'являються нові дані, які вказують на зміну умов руху або інших факторів.

1.5. Забезпечити вивід результатів аналізу та прогнозування часу прибуття маршрутного таксі на зручному і зрозумілому для користувача інтерфейсі, який відображається на мапі або списку зупинок.

1.6. Забезпечити можливість відображення прогнозів прибуття в різних форматах, таких як години і хвилини або залишений час до прибуття (наприклад, "через 5 хвилин").

Розрахунок прогнозу: На основі оброблених даних система повинна розраховувати прогнозований час прибуття маршрутного таксі на конкретну зупинку. Цей прогноз може бути оновлюваний в реальному часі, з урахуванням змін в умовах руху.

Критерії прийняття:

1.1. Система повинна використовувати оброблені дані з пункту "Аналіз та обробка даних" для розрахунку прогнозованого часу прибуття маршрутного таксі на конкретну зупинку.

1.2. Розрахунок прогнозу має враховувати величини, такі як поточна відстань між маршрутним таксі та зупинкою, середню швидкість руху на даному маршруті, погодні умови та інші фактори, які впливають на час подорожі.

1.3. При оновленні прогнозу система повинна виводити користувачам повідомлення або сповіщення зі зміненим часом прибуття та поясненнями про причини змін.

Доступність інтерфейсу: Інтерфейс користувача повинен бути інтуїтивно зрозумілим і доступним для всіх категорій користувачів.

Аналітика переміщення водіїв:

Збір та надання даних: Система повинна забезпечувати водіїв актуальними даними про рух транспорту на їхньому маршруті та наступній

зупинці. Інформація про маршрути, зупинки, поточне місцезнаходження водія та інші важливі дані повинні бути доступними для водіїв у реальному часі через їхні смартфони або планшети.

Автоматичне оновлення даних: Система повинна автоматично оновлювати дані про рух транспорту та події на дорозі в реальному часі без потреби в ручному оновленні водієм. Дані повинні оновлюватися достатньо часто, щоб водії мали можливість отримувати актуальну інформацію про маршрут.

Прогноз прибуття: Система повинна розраховувати прогнозований час прибуття водія до наступної зупинки на основі актуальних даних про рух.

Оповіщення та сповіщення: Система повинна надавати можливість водіям отримувати повідомлення та сповіщення про зміни у прогнозах прибуття, затримки або інші важливі події на маршруті. Оповіщення повинні бути візуальними та аудіальними, а також можливість налаштування пріоритетності та звуку оповіщень.

Відображення маршруту: Система повинна надавати водіям можливість переглядати маршрут на карті, включаючи маркери зупинок та поточне місцезнаходження. Водії повинні мати доступ до докладної інформації про маршрут, включаючи назви зупинок та відстань між ними.

3.1.2. Розробка функціоналу для аналітики переміщення користувачів та водіїв в реальному часі.

На основі аналізу вимог до функціоналу для аналітики переміщення користувачів та водіїв в реальному часі, можна написати програмний модуль на мові JavaScript, який використовує алгоритми машинного навчання для прогнозування часу прибуття. Для цього можна використати бібліотеки, такі як TensorFlow.js. , що є потужною бібліотекою для машинного навчання, яка працює на JavaScript і надає можливості впровадження ML-моделей

безпосередньо в браузер або на Node.js. TensorFlow.js дозволяє створювати, тренувати та використовувати моделі машинного навчання прямо в JavaScript. Це означає, що ви можете інтегрувати ML-функціонал безпосередньо у мою інформаційну систему. Також надає функціонал для ефективної обробки даних, включаючи операції з тензорами (багатовимірні масиви даних), які є основою машинного навчання та автоматично використовує WebGL для прискорення обчислень на GPU, що значно підвищує швидкість обробки та тренування моделей.

Лістинг 1:Функція прогнозування часу прибуття.

```
import * as tf from '@tensorflow/tfjs';
class TrafficAnalytics {
 constructor() {
  this.model = null;
  this.historicalTrafficData = []; // Для зберігання історичних даних про рух
  this.realTimeTrafficData = []; // Для зберігання даних про рух у реальному часі
  this.weatherData = []; // Для зберігання даних про погодні умови
 }
 async loadModel(modelPath) {
  // Завантаження попередньо навченої моделі
  this.model = await tf.loadLayersModel(modelPath);
 }
 async updateTrafficData(newTrafficData) {
  // Оновлення даних про рух
  this.realTimeTrafficData = newTrafficData;
 }
 async updateWeatherData(newWeatherData) {
  // Оновлення погодних даних
  this.weatherData = newWeatherData;
 }
 processData(rawData) {
  // Обробка вхідних даних
  return processedData;
 }
```

```
  predictArrivalTime(processedData) {
    // Прогнозування часу прибуття
    const prediction = this.model.predict(tf.tensor2d([processedData]));
    return prediction.dataSync()[0];
  }
  updatePredictions() {
    // Оновлення прогнозів на основі останніх даних
    const processedData = this.processData(this.realTimeTrafficData);
    return this.predictArrivalTime(processedData);
  }
}
// Використання модуля
const analytics = new TrafficAnalytics();
analytics.loadModel('/path/to/model.json').then(() => {
  // Припустимо, ми отримали нові дані про рух
  const newTrafficData = [/* ... */];
  analytics.updateTrafficData(newTrafficData);
  // Оновлення та отримання прогнозів
  const updatedPrediction = analytics.updatePredictions();
  console.log(`Оновлений прогноз часу прибуття: ${updatedPrediction}`);
});
```

Цей модуль включає в себе:

Клас TrafficAnalytics для роботи з моделлю машинного навчання. Клас TrafficAnalytics є програмною конструкцією, розробленою для аналізу даних трафіку, зокрема, в контексті аналітики руху міського транспорту. Клас TrafficAnalytics є самостійним компонентом, який може бути інтегрований в більші системи для обробки та аналізу даних руху. Всі дані та методи, що стосуються аналітики руху, зберігаються в межах класу, забезпечуючи чітку структуру та легкість управління. Клас може збирати дані з різних джерел, таких як GPS-трекери, сенсори руху, або інші системи моніторингу. Він може обробляти ці дані, наприклад, конвертувати координати GPS у відстані або

швидкості, виявляти шаблони руху тощо. Використання алгоритмів машинного навчання для прогнозування таких параметрів, як час прибуття, рівні заторів тощо. Можлива інтеграція з інтерактивними картами для відображення інформації про рух в реальному часі. Цей клас є прикладом об'єктно-орієнтованого підходу до розробки програмного забезпечення, який дозволяє ефективно керувати складними даними та функціями, пов'язаними з аналітикою руху в міському середовищі.

Функцію loadModel для завантаження навченої моделі. У контексті класу TrafficAnalytics, який використовується для аналітики руху, відіграє важливу роль у завантаженні та ініціалізації моделі машинного навчання. Вона відповідає за завантаження попередньо навченої моделі машинного навчання. Це може бути модель, навчена з використанням TensorFlow. Оскільки завантаження моделі може бути часозатратним процесом, функція loadModel зазвичай виконується асинхронно. Це означає, що вона повертає Promise, який розв'язується після завершення завантаження. Функція приймає як аргумент шлях до файлу моделі (часто це URL або локальний шлях), який використовується для визначення місця розташування моделі. Після завантаження моделі, loadModel ініціалізує її та готує до подальшого використання. Це може включати налаштування параметрів моделі, визначення вхідних та вихідних шарів тощо. Коли модель успішно завантажена та ініціалізована, функція повертає цю модель, що дозволяє іншим частинам програми використовувати її для прогнозування, аналізу даних тощо.

Функцію predictArrivalTime для прогнозування часу прибуття на основі оброблених даних. У контексті класу TrafficAnalytics, який використовується для аналітики руху, призначена для прогнозування часу прибуття транспортного засобу. Функція predictArrivalTime приймає вхідні дані, які можуть включати різні параметри, такі як поточне місцезнаходження, швидкість, напрямок руху, час доби, погодні умови тощо. Ці дані використовуються для роботи прогнозу. Функція перетворює ці вхідні дані у

формат, який може бути використаний моделлю машинного навчання. Це часто включає перетворення даних у тензори або нормалізацію даних. predictArrivalTime використовує попередньо завантажену модель машинного навчання для виконання прогнозування. Модель обробляє вхідні дані та генерує вихідні дані, які відображають прогнозований час прибуття. Функція повертає результат прогнозування, який може бути відображений користувачу або використаний для подальших розрахунків у системі. Результатом може бути конкретний час прибуття або оцінка затримки. У разі виникнення помилок під час прогнозування або обробки даних, predictArrivalTime може включати механізми для їх обробки або повідомлення про помилки.

Функцію processData для підготовки даних для моделі. У контексті класу TrafficAnalytics, який використовується для аналітики руху, відіграє ключову роль у підготовці та обробці вхідних даних, необхідних для аналізу та прогнозування. Функція processData приймає сирий набір даних, який може включати GPS-координати, інформацію про швидкість, час, погодні умови, та інші параметри, що впливають на рух транспорту. Вона обробляє та перетворює ці дані у формат, придатний для аналізу та прогнозування. Це може включати нормалізацію даних (наприклад, масштабування значень), обробку пропущених значень, перетворення географічних даних тощо. Функція може виділяти важливі особливості з даних, які є ключовими для аналізу. Наприклад, вона може визначати маршрути, розраховувати відстані між точками, аналізувати швидкісні профілі тощо. Після обробки та перетворення даних, processData підготовлює їх до подальшого використання в моделях машинного навчання. Це може включати перетворення даних у тензори або інші формати, які використовуються моделями. По завершенню обробки, функція повертає оброблені дані, які можуть бути використані для прогнозування часу прибуття, аналізу трафіку, або інших завдань аналітики руху.

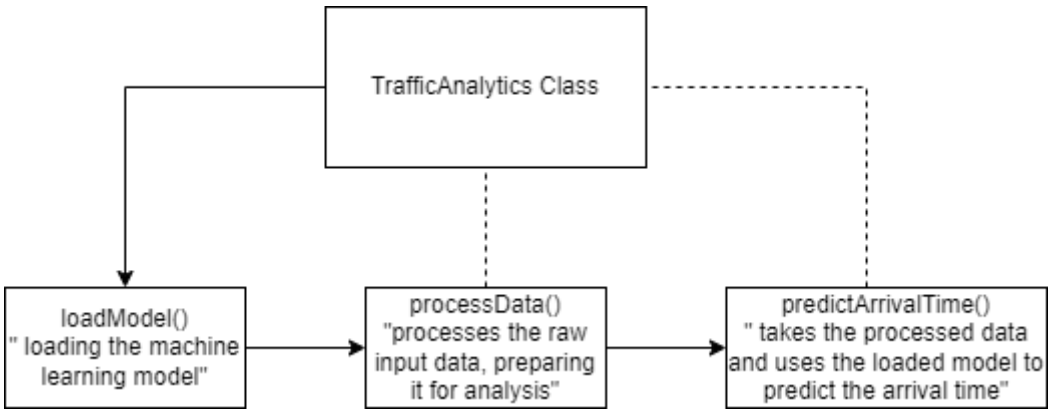

Рис. 3.2. Діаграма послідновності виконання модулю.

3.2. Впровадження технології обробки природних мов для аналізу відгуків та коментарів користувачів

Основні завдання та вимоги до NLP (Natural Language Processing):

- Сентимент-аналіз (аналіз настрою)

Вимога: Система повинна автоматично аналізувати емоційне забарвлення відгуків та коментарів користувачів, класифікуючи їх як позитивні, негативні, або нейтральні.

- Коментарі користувачів класифікуються з точністю не менше 80%.
- Система виявляє ключові емоційні слова або фрази, що впливають на оцінку настрою.

- Класифікація коментарів

Вимога: Система повинна автоматично класифікувати коментарі на основі їх змісту в категорії, такі як скарги, пропозиції, або запитання.

- Класифікація коментарів має точність класифікації не менше 75%.
- Система ідентифікує ключові слова або фрази, які вказують на конкретну категорію.

- Видобування інформації

Вимога: Система повинна виявляти та виділяти ключові слова та фрази в коментарях, які можуть вказувати на конкретні проблеми або аспекти.

- Система здатна ідентифікувати та виділяти ключові слова або фрази з високою точністю.

- Аналіз трендів

Вимога: Система повинна виявляти зміни в настроях та відгуках користувачів протягом часу, щоб відстежувати ефективність змін у послугах.

- Система відстежує та аналізує зміни в настроях користувачів протягом визначеного періоду часу.

- Аналіз трендів включає в себе зіставлення змін у настроях із внесеними змінами у послуги.

Генерація звітного файлу:

Після аналізу та класифікації коментарів за емоційним забарвленням, система генерує файл з відсортованими коментарями. Кожен коментар у цьому файлі містить ідентифікатор користувача та анотацію емоційного забарвлення (позитивний, негативний, нейтральний).

Цей файл надається розробникам, щоб вони могли ознайомитися з відгуками та врахувати важливі аспекти для подальшого покращення послуг.

Нарахування бонусних балів:

Якщо користувач залишає позитивний відгук, йому нараховується 2 бонусних бали.

Якщо відгук негативний, користувач отримує 1 бонусний бал за те, що поділився своєю думкою.

Бонусні бали відразу після відправлення оцінки та/або коментаря автоматично нараховуються на рахунок користувача та відображаються у його профілі.

Отримання промокоду:

За кожні накопичені 100 бонусних балів, користувач автоматично отримує промокод, який можна використати у партнерському книжковому магазині. Промокод надсилається на електронну пошту користувача та відображається у його профілі в системі.

Оновлення статусу профілю:

Статус балів та наявність промокоду оновлюються в реальному часі в профілі користувача. Користувачі можуть переглядати свої нараховані бали та доступні промокоди через інтерфейс користувацького профілю.

Кнопка "Почати/Закінчити Подорож":

На інтерфейсі користувача додано кнопку, яка дозволяє позначити початок та закінчення подорожі.

При натисканні на кнопку на початку подорожі, система фіксує час та геолокаційні дані.

При натисканні кнопки для закінчення подорожі, система фіксує кінцеву точку та час подорожі.

Модальне вікно для оцінювання та коментування:

Після закінчення подорожі автоматично з'являється модальне вікно, яке пропонує користувачу поставити оцінку від 1 до 5 зірок. В модальному вікні також присутнє поле для того щоб залишити коментар (необов'язково).

Повідомлення про набрані бали:

Після нарахування балів користувачу надсилається повідомлення з інформацією про кількість зароблених балів.

Оцінка Маршрутів Для Інших Користувачів

Оцінки та коментарі від користувачів будуть використані для формування загальної оцінки маршрутів. Інші користувачі можуть переглядати ці оцінки під час планування своїх подорожей, що допоможе їм обирати оптимальні маршрути.

Використання цього функціоналу буде стимулювати активність користувачів та збільшить кількість корисних відгуків та оцінок. Користувачі, які активно залишають відгуки та оцінки, швидше наберуть потрібну кількість балів для отримання промокодів.

Цей процес не тільки забезпечує цінний зворотний зв'язок для команди розробників, але й стимулює користувачів до активної участі в процесі вдосконалення послуг, надаючи їм відчуття внеску та винагороди.

Лістинг 2. Функція технології обробки природних мов:

```
// Припускаємо, що TensorFlow.js вже підключений
// Завантаження попередньо навченої моделі NLP для сентимент-аналізу
async function loadSentimentModel() {
   const model = await tf.loadLayersModel('path/to/sentiment/model.json');
   return model;
}
// Функція для аналізу відгуку
async function analyzeSentiment(text, model) {
   // Попередня обробка тексту (токенізація, нормалізація тощо)
   // Перетворення тексту в тензор
   const inputTensor = tf.tensor(/* ... */);
   // Використання моделі для аналізу
   const prediction = model.predict(inputTensor);
   return prediction.dataSync()[0];
}
// Використання
const model = await loadSentimentModel();
const sentiment = await analyzeSentiment("Дуже задоволений послугами!", model);
console.log(`Sentiment score: ${sentiment}`);
```

Функція loadSentimentModel: Завантажує попередньо навчену модель NLP з зазначеного шляху (або URL). Використовує tf.loadLayersModel для асинхронного завантаження моделі машинного навчання. Повертає Promise, який при успішному виконанні повертає інстанс моделі.

Функція analyzeSentiment: Аналізує текстовий відгук на емоційне забарвлення, використовуючи завантажену модель.

- text: Вхідний текст відгуку для аналізу.
- model: Модель NLP, завантажена функцією loadSentimentModel.

Код містить коментар, що передбачає попередню обробку тексту, таку як токенізацію, нормалізацію тощо, яка необхідна перед подачею тексту в модель. Використовую метод predict моделі для отримання прогнозу емоційного забарвлення відгуку. Код повертає результат аналізу (наприклад, оцінку сентименту відгуку).

Використання функцій демонструє, як використовувати вищеописані функції для аналізу конкретного відгуку. Результат виводиться у консоль сентимент-аналізу для прикладу тексту.

Лістинг3.Функція відстеження початку та закінчення маршруту і відображення модального вікна:

```
import React, { useState } from 'react';
function TravelComponent() {
  const [travelStarted, setTravelStarted] = useState(false);
  const [modalVisible, setModalVisible] = useState(false);
  const startTravel = () => {
    // Позначити початок подорожі
    setTravelStarted(true);
    // Логіка відстеження подорожі
  };
  const endTravel = () => {
    // Позначити закінчення подорожі
    setTravelStarted(false);
    // Показати модальне вікно для відгуку
    setModalVisible(true);
  };
  const submitFeedback = (stars, comment) => {
    // Обробка відгуку користувача
    setModalVisible(false);
  };
  return (
    <div>
      {!travelStarted ? (
```

```
            <button onClick={startTravel}>Почати Подорож</button>
          ) : (
            <button onClick={endTravel}>Закінчити Подорож</button>
          )}
          {modalVisible && (
            <FeedbackModal onSubmit={submitFeedback} />
          )}
        </div> );
}
function FeedbackModal({ onSubmit }) {
  const [stars, setStars] = useState(0);
  const [comment, setComment] = useState('');
  return (
    <div className="modal">
      {/* Логіка відображення та збору даних модального вікна */}
      <button onClick={() => onSubmit(stars, comment)}>Надіслати Відгук</button>
    </div>
  ); } export default TravelComponent;
```

У цьому коді TravelComponent використовується для управління станом подорожі та відображення модального вікна з відгуками. Компонент FeedbackModal відповідає за збір оцінок та коментарів користувачів.

3.3. Серверна частина

Бекенд системи складається з REST API, яке забезпечує взаємодію між клієнтською частиною, базою даних і Redis.

Сервер здатний обслуговувати наступні типи запитів:

GET:

buses – відсилає дані про усі доступні автобуси;

routes – відсилає дані про всі наявні маршрути;

stops – відсилає дані про всі наявні зупинки;

buses/:id/location – відсилає інформацію про розташування конкретного автобуса за його ідентифікатором;

feedbacks/:id – відсилає відгуки за ідентифікатором водія.

POST:

registration – обробляє запити на реєстрацію;

authorization – обробляє запити на авторизацію;

feedbacks – дозволяє додавати відгуки про водіїв.

PUT:

buses/:id/location – оновлює дані про розташування автобуса в Redis.

Лістинг 4. Список функцій обробників запитів на сервер

```
api.use('/buses', buses({ config, db }));
api.use('/routes', routes({ config, db }));
api.use('/stops', stops({ config, db }));
api.get('/buses/:id/location', (res, req) => getLocation(res,
req, db.redis));
api.get('/feedbacks/:id', (res, req) => getFeedbacksByUser(res,
req, db.mysql));
api.post('/registration', (res, req) => register(res, req,
db.mysql));
api.post('/authorization', (res, req) => authorization(res,
req, db.mysql));
api.post('/feedbacks', (res, req) => postFeedback(res, req,
db.mysql));
api.put('/buses/:id/location', (res, req) => putLocation(res,
req, db.mysql, db.redis, io));
```

Розглянемо детальніше кожен з запитів.

GET /buses, /routes, /stops: дані запити досить примітивні у своїй логіці. Сервер робить запит до бази даних та просто надсилає отриманий результат у відповідь.

Лістинг 5. Функція обробник запиту GET /buses

```
db.mysql.query("SELECT * FROM bus", function (err, result, fields) {
```

```
 if (err) throw err;
 res.json(result);
});
```

Лістинг 6. Функція обробник запиту GET /routes до API

```
db.mysql.query("SELECT * FROM route", function (err, result,
fields) {
 if (err) throw err;
 res.json(result.map(res => ({
 ...res,
 stops_ids: res.stops_ids.split(',').map(id => Number(id))
 })));
});
```

Лістинг 7. Функція обробник запиту GET /stops до API

```
db.mysql.query("SELECT * FROM stop", function (err, result,
fields) {
 if (err) throw err;
 res.json(result);
});
```

GET /buses/:id/location: при даному запиті сервер дивитья чи в Redis є поле з таким ключем. Якщо поле є, у відповідь надсилається об’єкт з даними про довготу, широту, швидкість та останній час оновлення даних. Якщо такого поля нема, приходить повідомлення що автобус не в мережі.

Лістинг 8. Функція обробник запиту GET /buses/:id/location

```
export const getLocation = (req, res, redis) => {
 if (!req.params.id) res.json({error: 'Bus id required'});
 redis.get(req.params.id, (err, value) => {
 if (err) {
 res.json({
 error: 'Bus is offline'
 });
 return;
 }
```

```
 if (value) {
 const payload = value.split(',');
 res.json({
 lon: Number(payload[0]),
lat: Number(payload[1]),
 timestamp: Number(payload[2]),
 speed: payload[3] ? Number(payload[3]) : 0
 })
 }
```

GET /feedbacks/:id: при даному запиті сервер робить запит до бази даних, який одразу повертає відгуки на конкретного водія, та надсилає їх у відповідь на запит.

Лістинг 6. Функція обробник запиту GET /feedbacks/:id

```
export const getFeedbacksByUser = (req, res, mysql) => {
 const user_id = Number(req.params.user_id);
 if (!user_id) {
 res.json({
 error: 'User ID required.'
 })
 }
 mysql.query(`SELECT * FROM feedback WHERE user_id =
${user_id}`, (err, result) => {
 if (err) {
 return;
 }
 res.json(result)
 })
}
```

POST /registration: запит на реєстрацію, на відміну від попередніх, потребує передачу параметрів в тілі (body) запиту. Запит повинен містити такі параметри:

− ім’я користувача

− логін користувача

− пароль

− роль в додатку (водій або звичайний користувач)

Спочатку сервер перевіряє, чи прийшли в тілі запиту ім’я та логін. Якщо вони прийшли, сервер робить запит до бази даних, та перевіряє чи даний логін не зайнятий іншим користувачем. Якщо такий користувач вже існує, сервер надсилає у відповдь на запит помилку. Якщо такого користувача не існує, сервер перевіряє наявність паролю та ролі користувача у додатку. Якщо ці дані є, сервер хешує пароль та додає нового користувача до бази даних. В цей же момент сервер генерує токен. Далі сервер додає до бази даних нову сесію, яка зберігає токен, дату коли токен стане не дійсним на основі поточного часу (до поточного часу додається один місяць, тобто токени в додатку дійсні один місяць), роль та id користувача. У відповідь на даний запит надсилається об’єкт який містить всю інформацію що прийшла в запиті окрім пароля, а разом з цим. надсилається токен, та id щойно створеного користувача.

Лістинг 7. Функція обробник запиту POST /registration

```
export const register = (req, res, mysql) => {
 const { name, username, password, role } = req.body;
 if (!name) { res.json({ error: 'Name required.'}); return; }
 if (!username) { res.json({ error: 'Username required.'});
return; }
 mysql.query('SELECT * FROM user WHERE username = ?',
username, (err, results) => {
 if (err) {
 res.json({
 error: err
 })
 return;
 }
 if (results.length !== 0) { res.json({ error: 'User with
this username exist' }); return;}
 if (!password) { res.json({ error: 'Password required.'});
```

```
return;}
 if (!role) { res.json({ error: 'Role required.'}); return;}
 const passwordHash =
crypto.createHash('md5').update(password).digest("hex");
 const user = {
name,
 username,
 passwordHash,
 role
 };
 mysql.query('INSERT INTO user SET ?', user, (err, result) => {
 if (err) {
 console.log(err);
 res.json({
 error: err
 })
 }
 const token = randToken.generate(32);
 res.json({
 success: true,
 user: {
 name,
 username,
 role,
 token,
 id: result.insertId,
 }
 })
 const date = new Date();
 date.setMonth(date.getMonth() + 1);
 const session = {
 token,
 user_id: result.insertId,
 user_role: role,
```

```
expire_date: date.toISOString()
}
mysql.query('INSERT INTO session SET ?', session)
})
});
}
```

POST /authorization: запит на авторизацію потребує два параметри: логін та хеш паролю. Спочатку сервер перевіряє наявність цих двох параметрів у запиті. Якщо якийсь з них пропущений, сервер надішле у відповідь повідомлення з відповідною помилкою. Далі сервер робить запит до бази даних та шукає користувача за вказаним логіном. Якщо користувач не знайдений, сервер надсилає помилку, що користувача з даним логіном не існує. Якщо користувач існує сервер перевіряє чи збігається хеш з тим хешом, що зберігається у базі даних. Якщо хеші не збігаються, сервер надсилає відповідну помилку. Якщо хеші збігаються, відбувається те саме, що і на останньому етапі реєстрації. У відповідь сервер надсилає дані про користувача разом з токеном, і створює новий запис у таблиці сесій в базі даних.

Лістинг 8. Функція обробник запиту POST /authorization

```
export const authorization = (req, res, mysql) => {
 const { username, passwordHash } = req.body;
 if (!username) { res.json({ error: 'Username required.',
success: false}); return;}
 if (!passwordHash) { res.json({ error: 'Password required.',
success: false}); return;}
 mysql.query('SELECT * FROM user WHERE username = ?',
username, (err, results) => {
 if (err) {
 res.json({
 error: err
 })
 return;
 }
```

```
 if (results.length === 0) { res.json({ error: `There is no
user with username '${username}'`, success: false }); return;}
 if (results[0].passwordHash !== passwordHash) { res.json({
error: 'Password is incorrect', success: false }); return;}
 const token = randToken.generate(256);
 res.json({
 success: true,
 user: {
 name: results[0].name,
 role: results[0].role,
 id: results[0].id,
 token
 }
 })
 const date = new Date();
 date.setMonth(date.getMonth() + 1);
const session = {
 token,
 user_role: results[0].role,
 user_id: results[0].id,
 expire_date: date.toISOString() }
 mysql.query('INSERT INTO session SET ?', session)
 });
}
```

POST /feedbacks: запит на відгуки повинен містити id водія, оцінку поставлену водію та відгук (необов’язково). Спочатку сервер перевіряє чи наявний id водія та оцінка в тілі запиту. Якщо ці параметри відсутні у відповідь сервер надсилає відповідні помилки. Якщо всі дані, що потрібні, наявні у запиті, сервер записує даний відгук до бази даних, а у відповідь надсилає об’єкт з одним полем, яке містить булеву змінну, яка вказує на те що запит пройшов успішно.

Лістинг 9. Функція обробник запиту POST /feedbacks

```
export const postFeedback = (req, res, mysql) => {
```

```
const {rate, user_id} = req.body;
if (!user_id) {
res.json({
error: 'User ID required.'
})
}
if (!rate) {
res.json({
error: 'Rating is required.'
})
}
let text = req.body.text;
if (!text) text = '';
const feedback = {
text,
user_id,
rate,
date_created: new Date().toISOString()
};
mysql.query('INSERT INTO feedback SET ?', feedback, (err) =>
{
if (err) {
return;
}
res.json({
success: true
})
})
}
```

PUT /buses/:id/location: даний запит являється основним в даному додатку. Перш за все сервер перевіряє чи запит містить id атобуса. Якщо id відсутній, у відповідь прийде помилка. Якщо id наявний, сервер робить запит до бази даних та шукає автобус по id. Якщо таких автобусів немає у

відповідь прийде помилка. Якщо id водія який зробив запит не відповідає id водія автобуса з бази даних, то сервер надішле помилку, що даний водій не може оновлювати дані про даний транспортний засіб. Потім сервер перевіряє наявність координат широти та довготи в запиті. При їх відсутності виникає помилка. Якщо всі перевірки пройшли успішно, сервер робить запит до Redis, та дивиться чи є поле з ключем який дорівнює id маршрутки, що прийшов в запиті. Якщо такого поля не існує, сервер створює нове поле в таблиці Redis, куди записуються дані довготи, широти та поточного часу через кому. Якщо ж запис по такому id вже є, сервер проводить розрахунок середньої швидкості на пройденому відрізку шляху, та також робить запис до таблиці Redis. В будь–якому випадку, у відповідь надсилається відповідь, про успішне виконання запиту.

Лістинг 10. Функція обробник запиту PUT /buses/:id/location

```
export const putLocation = (req, res, mysql, redis, io) => {
/buses/:id/location
 const id = Number(req.params.id);
 if (!id) {
 res.json({error: 'Bus id required.'});
 return;
 }
 mysql.query('SELECT * FROM bus WHERE id = ?', id, (err,
results) => {
 if (err) {
 res.json({
 error: err
 });
 return;
 }
 if (results.length === 0) {
 res.json({
 error: `There is no bus with id '${id}'`
```

```
 });
 return;
 }
 const bus = results[0];
 if (req.userId !== bus.user_id) {
 res.json({
 error: 'Driver with this token cannot update this bus
location'
 });
 return;
 }
 const { lon, lat } = req.body;
 if (!lon || !lat) {
 res.json({error: 'Request should contain longitude and
latitude.'});
 return;
 }
 redis.get(id, (err, value) => {
 if (err) throw (err);
 const currMilliseconds = Date.now();
 if (value) {
 const payload = value.split(',');
 const lonPrev = Number(payload[0]);
 const latPrev = Number(payload[1]);
 const distance = getDistanceBetwenCoords({lat: latPrev,
lon: lonPrev}, {lat, lon});
 const prevMilliseconds = Number(payload[2]);
/buses/:id/location
 const deltaMilliseconds = currMilliseconds -
prevMilliseconds;
 const speed = distance / deltaMilliseconds * 1000;
 const newPayload =
`${lon},${lat},${currMilliseconds},${speed}`;
 io.emit('bus_location', { id, lon, lat, timestamp:
```

```
currMilliseconds, speed});
 redis.set(id, newPayload)
 res.json({
 updated: true
 });
 return;
 }
 io.emit('bus_location', { id, lon, lat, timestamp:
currMilliseconds, speed: 0})
 const newPayload = `${lon},${lat},${currMilliseconds}`;
 redis.set(id, newPayload)
 res.json({
 updated: true
 });
 }
```

GET /feedbacks/:id: при даному запиті сервер робить запит до бази даних, який одразу повертає відгуки на конкретного водія, та надсилає їх у відповідь на запит.

Лістинг 11. Функція обробник запиту GET /feedbacks/:id

```
export const getFeedbacksByUser = (req, res, mysql) => {
  const user_id = Number(req.params.user_id);
  // Перевірка наявності ідентифікатора користувача
  if (!user_id) {
    return res.json({
      error: 'User ID is required.'
    });
  }
  // Запит до бази даних для отримання відгуків користувача
  mysql.query(`SELECT * FROM feedback WHERE user_id = ${user_id}`, (err, result) =>
{
    if (err) {
      // Обробка помилки сервера
      return res.status(500).json({
```

```
      error: 'Server error occurred.'
    });
  }
  // Повернення результатів у форматі JSON
  res.json(result);
 });
};
```

POST /feedbacks: запит для додавання відгуків має включати ідентифікатор водія, присвоєну йому оцінку та, за бажанням, текст відгуку. Спершу сервер перевіряє наявність ідентифікатора водія та оцінки у контенті запиту. У разі відсутності цих обов'язкових параметрів сервер повідомляє про помилки відповідними повідомленнями. У випадку, коли запит включає всі необхідні дані, сервер заносить відгук до бази даних і відсилає у відповідь об'єкт з булевим атрибутом, що підтверджує успішне виконання операції.

Лістинг 12. Функція обробник запиту POST /feedbacks

```
export const postFeedback = (req, res, mysql) => {
 const { rate, user_id } = req.body;
 // Перевірка наявності ідентифікатора користувача
 if (!user_id) {
  return res.json({
   error: 'User ID is required.'
  });
 }
 // Перевірка наявності оцінки
 if (!rate) {
  return res.json({
   error: 'Rating is required.'
  });
 }
 // Якщо текст відгуку не надано, встановлюється пустий рядок
 let text = req.body.text || '';
 // Створення об'єкту відгуку
```

```
  const feedback = {
    text: text,
    user_id: user_id,
    rate: rate,
    date_created: new Date().toISOString()
  };
  // Вставлення відгуку до бази даних
  mysql.query('INSERT INTO feedback SET ?', feedback, (err) => {
    if (err) {
      // Обробка помилки бази даних
      return res.status(500).json({
        error: 'Database error occurred.'
      });
    }
    // Успішне додавання відгуку
    res.json({
      success: true
    });
  });
};
```

3.4. Екрани користувача: карта та особистий кабінет

Основні вимоги, що визначають компоненти та функціональність інтерфейсу користувача для системи моніторингу руху міського транспорту, забезпечуючи зручність та ефективність у використанні додатку.

Загальний Вид Екрану:

- Екран користувача містить карту міста.
- На карті в реальному часі відображається маячок, що позначає місцезнаходження користувача.

- Поля введення "Звідки" і "Куди": мають чітко відображати місце, звідки користувач починає та закінує свою подорож. Повинні мати функцію автозаповнення для полегшення введення адрес.
- Кнопка обміну місцями "Звідки" і "Куди": має забезпечувати можливість швидко обміняти введені точки між собою.
- Іконки Типів Транспорту (Автобус, Метро, Трамвай, Поїзд, Таксі): кожна іконка має бути виразною та легко впізнаваною. При натисканні на іконку відбувається фільтрація доступних маршрутів за вибраним типом транспорту.
- Розділ "Мої минулі поїздки": перелік збережених маршрутів має бути легко доступним і читабельним. Кожен збережений маршрут має містити назву місця та приблизний час до нього. При натисканні на збережений маршрут користувач має переходити до деталей маршруту.
- Навігаційне Меню (У дорогу, Карта, Профіль): Іконки мають бути інтуїтивно зрозумілими та легко впізнаваними. Навігаційне меню має забезпечувати швидке перемикання між різними розділами додатку. Активний розділ має бути візуально виділений.

Деталі Маршруту:

- Після вибору маршруту користувачу відображаються деталі:
- Середній час подорожі до зупинки відправлення.
- Середній час у дорозі на транспорті.
- Орієнтовний час прибуття до кінцевої точки від останньої зупинки.

Відстеження Руху Транспорту:

- Користувач має змогу в реальному часі стежити за рухом обраного транспорту на карті.

Кнопка "Почати Подорож":

- Наявність кнопки для позначення початку подорожі.

Кнопка "Закінчити Подорож":

- Наявність кнопки для позначення кінця подорожі.

Модальне Вікно для Відгуків:

- Після закінчення подорожі користувачу відображається модальне вікно, де він може:
- Поставити оцінку маршруту (від 1 до 5 зірок).
- Залишити текстовий коментар (необов'язково).

Загальні Вимоги до Дизайну:

- Кольорова палітра та шрифти мають бути універсальними та доступними для користувачів з різними візуальними потребами.
- Інтерфейс має бути адаптованим до різних розмірів екранів.
- Елементи управління мають бути розміщені таким чином, щоб їх було зручно використовувати однією рукою.

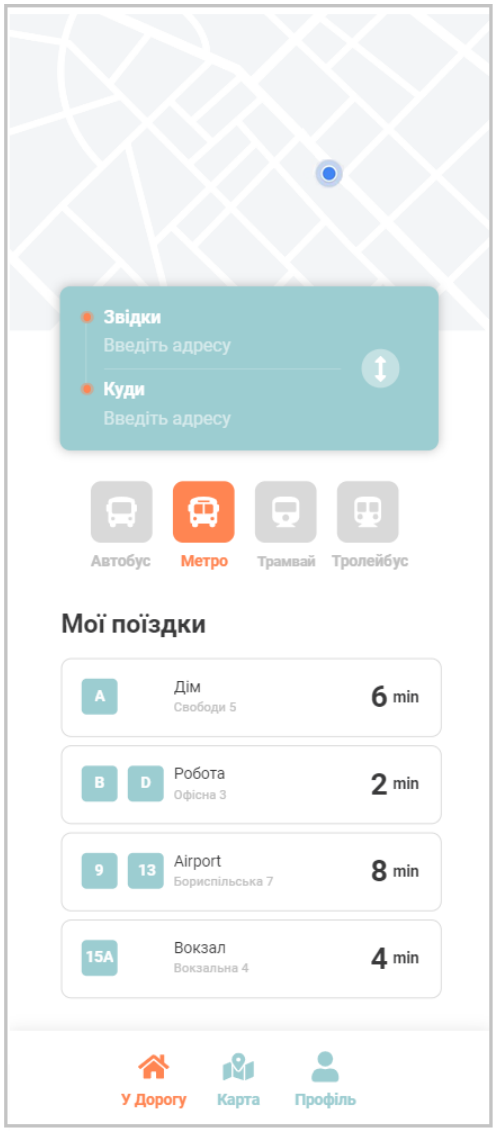

Рис. 3.3. Головний екран користувача

Вид екрану з результатами запиту:

Екран з результатами запиту "Знайдені маршрути ":

- Має бути чітко видимий на верхній частині екрану.
- Повинен вказувати, що наступний вміст екрану є результатом пошуку маршрутів.

Навігаційні Кнопки:

- Ліва стрілка вгорі екрана має повертати користувача на попередній екран.
- Кнопка меню (три смужки) вгорі екрана має відкривати додаткові опції або налаштування.

Поля Введення "Звідки" та "Куди":

- Поля повинні відображати обрані користувачем пункти відправлення та призначення.
- Кнопка обміну місцями має дозволяти швидко поміняти місцями введені адреси.

Список "Кращі маршрути":

- Кожен маршрут у списку повинен містити інформацію про очікуваний час подорожі та тип транспорту.
- Іконки поруч з текстом мають відображати тип транспорту (таксі, метро, пішки, велосипед).
- Кожен елемент списку має бути вибірковим та вести до детальнішої інформації про маршрут.

Дизайн Елементів Списку:

- Елементи списку мають бути достатньо великими для зручного натискання.

- Візуальна відмінність між різними видами транспорту (колір, іконка) для кращого визначення.

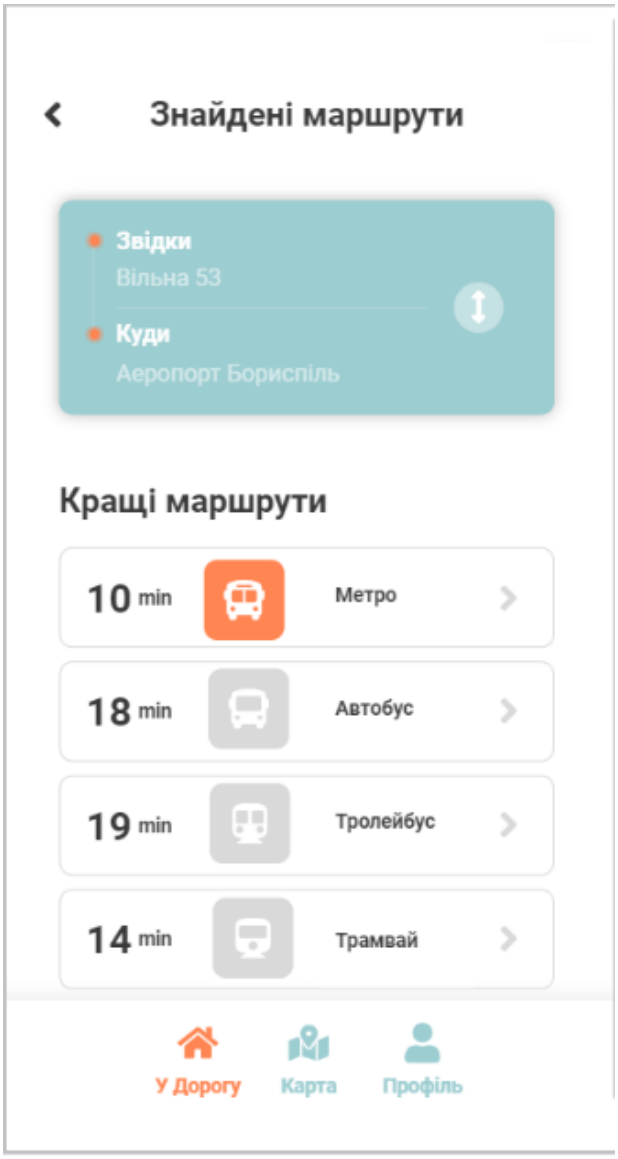

Рис. 3.4. Екран результатів пошуку маршрутів

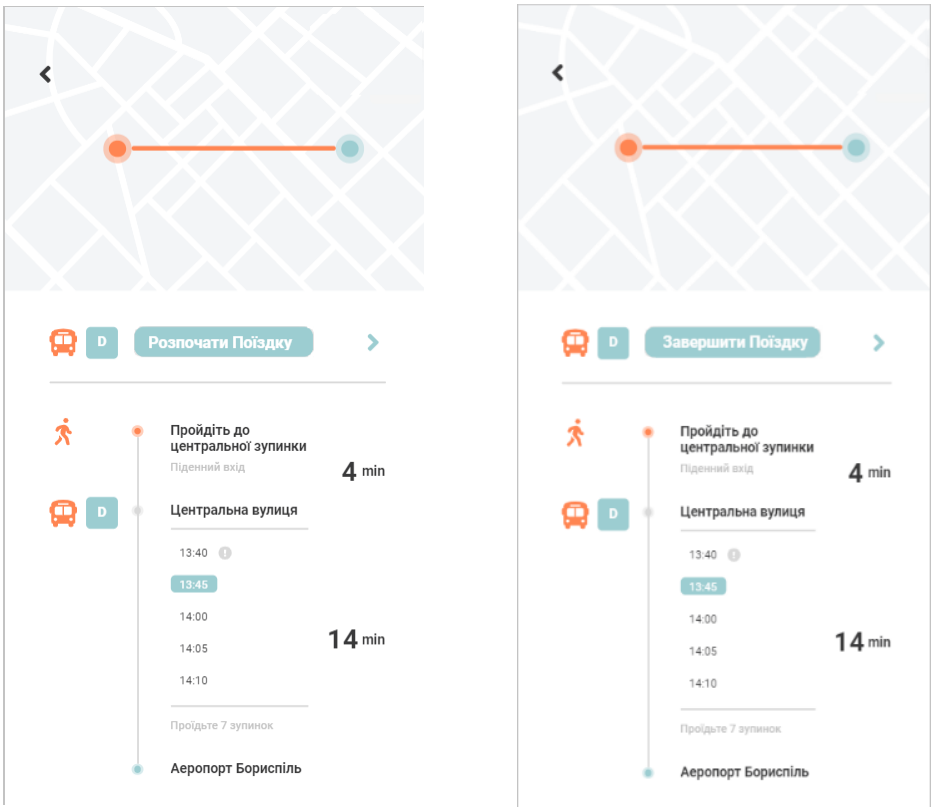

Рис. 3.5. , 3.6. Екрани з приблизно розрахованим часом поїздки.

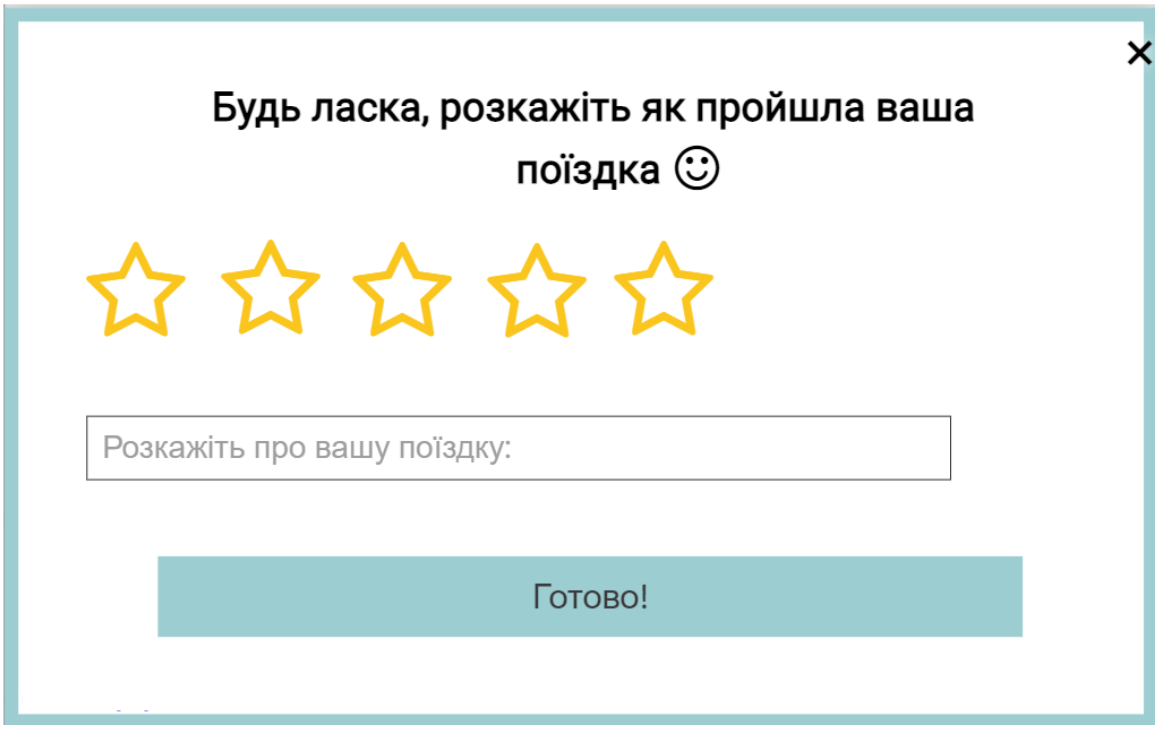

Рис. 3.7. Вигляд модального вікна для оцінки та відгуку про поїздку

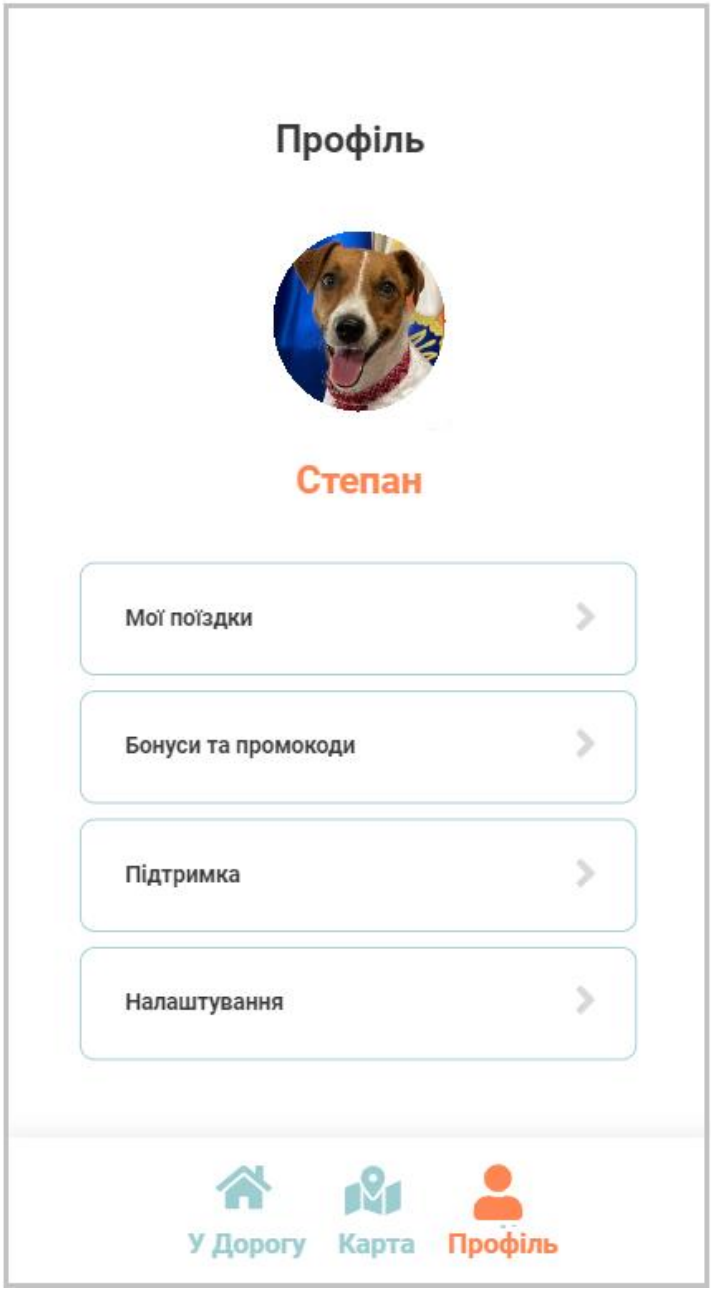

Рис. 3.8. Екран особистого кабінету користувач (Профіль).

Рис. 3.9. Екран бонусного рахунку та наявних промокодів.

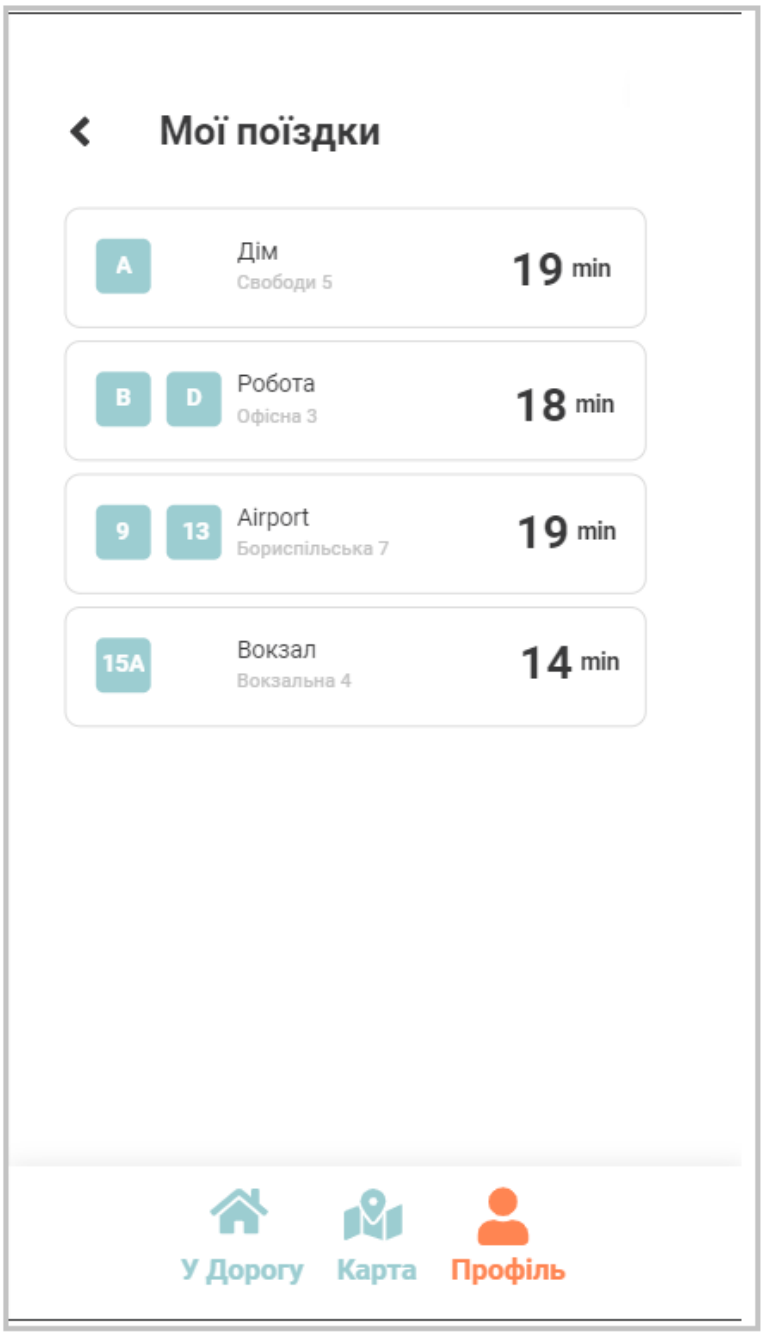

Рис. 3.10. Екран минулих поїздок із відображенням середнього часу витраченого на подорож.

3.5. Тестування системи

Мобільні додатки відрізняються від десктопних рішень через використання різних операційних систем, обмежений розмір екрану, присутність багатьох сенсорів, які працюють у режимі реального часу, та інші

особливості. Тому тестування мобільних додатків вимагає багатофакторного підходу та оцінки на багато критеріїв.

Навіть при появі новіших стандартів мобільного зв'язку, не всі користувачі можуть користуватися постійним та стабільним з'єднанням. Зв'язок часто може ставати менш стійким або навіть втрачатися зовсім. Як я вже пояснювала у третьому розділі, наша клієнтська частина програми створена так, що хоча б раз отримавши координати, користувач отримує інформацію щодо швидкості. Після цього транспортний засіб починає рухатися по карті з тією самою швидкістю, яка була отримана останнім запитом. Саме такий підхід дозволяє користувачам відстежувати рух транспортного засобу навіть при поганому з'єднанні. Зрозуміло, що прогнозовані координати можуть бути недосконалими та не завжди точно відображати реальну ситуацію. Однак це надає користувачам хоча б загальне уявлення про місцезнаходження транспортного засобу.

Також часто виникають проблеми у користувачів, коли вони вимикають або ставлять додаток на паузу. В цих випадках дані можуть бути втрачені, а додаток може видачувати фатальні помилки. Для перевірки цього сценарію, я вимикала та ставила додаток на паузу, виконуючи при цьому багато додаткових операцій у додатку. На щастя, під час тестування не було виявлено жодних помилок або проблем.

Як я вже згадувала у вимогах до програмного забезпечення, розробка під мобільні пристрої має свої особливості, і однією з них є обмежена оперативна пам'ять пристрою. Під час тестування було помічено, що при наявності великої кількості маркерів зупинок на карті, швидкість роботи пристрою помітно погіршувалася. Тому було прийнято рішення додати параметр, який дозволяє користувачу вказати радіус навколо свого поточного місцезнаходження, в межах якого він бажає бачити зупинки. Це допомагає зменшити навантаження на пам'ять та покращити продуктивність додатку на пристроях з обмеженими ресурсами.

ВИСНОВКИ

Основною метою цього дипломного проекту було розробити зручний додаток, який дозволить користувачам відстежувати рух маршрутних таксі на міській мапі та планувати свої поїздки на основі наданих даних.

Система розділена на клієнтську і серверну частини, кожна з яких виконує специфічні функції. Клієнтська частина, розроблена на JavaScript та React Native, візуалізує місцезнаходження транспорту на міській карті, в той час як серверна частина, реалізована на Node.JS, забезпечує обмін даними між клієнтом і базою даних. Розроблений функціонал забезпечує збір даних про переміщення користувачів і водіїв, включаючи історичні дані та інформацію в реальному часі. Особлива увага приділяється точності, надійності та безпеці збору та зберігання даних. Аналітика та обробка даних включає в себе врахування різноманітних факторів, які можуть впливати на рух транспорту.

Реалізовано програмний модуль на JavaScript для прогнозування часу прибуття, використовуючи алгоритми машинного навчання та бібліотеку TensorFlow.js. Модуль використовує історичні та актуальні дані для точного прогнозування, демонструючи гнучкість та інноваційність підходу.

Система включає NLP для аналізу відгуків та коментарів користувачів, що дозволяє збирати цінний зворотний зв'язок. Виконується сентимент-аналіз, класифікація коментарів, видобування ключових слова та фраз, а також аналіз трендів. Це не лише забезпечує важливу інформацію для покращення послуг, але й стимулює активну участь користувачів через систему нагород та бонусів.

Дипломний проект має детальний опис розробки інформаційної системи для моніторингу руху міського транспорту, вказуючи на важливість інтеграції передових технологій та інноваційних підходів.

СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ

1. Google map help. View traffic, satellite, terrain, biking, and transit [Електронний ресурс]. – Режим доступу: https://goo.gl/9zxHhZ . Дата доступу: вересень 2023. Назва з екрану.
2. EasyWay [Електронний ресурс]. – Режим доступу: https://www.eway.in.ua/ua/cities/kyiv. Дата доступу: вересень 2023. Назва з екрану.
3. Mooveit [Електронний ресурс]. – Режим доступу: https://moovit.com/about-us/?_ga=2.4733809.37944187.1695127192-968595489.1695127191. Дата доступу: вересень 2023. Назва з екрану.
4. Транспорт онлайн Київ [Електронний ресурс]. – Режим доступу: https://techukraine.net/7. Дата доступу: вересень 2023. Назва з екрану.
5. Machine Learning [Електронний ресурс]. – Режим доступу: https://www.it.ua/knowledge-base/technology-innovation/machine-learning. Дата доступу: травень 2023. Назва з екрану.
6. Громадський транспорт у Києві: маршрути, програми відстеження та актуальні питання [Електронний ресурс]. – Режим доступу: https://znaj.ua/society/279839-gromadskiy-transport-u-kiyevi-marshruti-programi-vidstezhennya-ta-aktualni-pitannya. Дата доступу: вересень 2023. Назва з екрану.
7. Огляд додатків міського транспорту[Електроний ресурс] Режим доступу: https://hi-tech.ua/article/poehali-obzor-prilozheniy-dlya-obshhestvennogo-transport/ . Дата доступу: листопад 2023. Назва з екрану.
8. Swift [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Swift_(%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F). Дата доступу: листопад 2023. Назва з екрану.

9. Java overview [Електронний ресурс]. – Режим доступу: https://goo.gl/T8tZr0 . Дата доступу: листопад 2023. Назва з екрану.
10. Переваги та недоліки Java-технологій [Електронний ресурс]. – Режим доступу: https://goo.gl/lPsTgx . Дата доступу: листопад 2023. Назва з екрану.
11. Javascript [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/JavaScript . Дата доступу: листопад 2023. Назва з екрану.
12. MySQL [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/MySQL . Дата доступу: листопад 2023. Назва з екрану.
13. Node.JS [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Node.js . Дата доступу: листопад 2023. Назва з екрану.
14. Робота з Google Maps API [Електронний ресурс]. – Режим доступу: https://goo.gl/2vYMhN . Дата доступу: листопад 2023. Назва з екрану.
15. 5 принципів тестування мобільних додатків [Електронний ресурс]. – Режим доступу: https://habr.com/post/244345/ . Дата доступу: листопад 2023. Назва з екрану.
16. Smoke testing (software) [Електронний ресурс]. – Режим доступу: https://goo.gl/qkJrxI . Дата доступу: листопад 2023. Назва з екрану.