

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

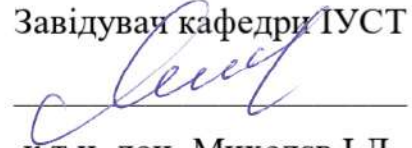
Національний університет кораблебудування імені адмірала Макарова

Навчально-науковий інститут комп'ютерних наук та управління проектами

Кафедра інформаційних управляючих систем та технологій

"Допущений до захисту"

Завідувач кафедри ІУСТ



к.т.н, доц. Михелєв І.Л.

" ____ " _____ 2026 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти "бакалавр"

на тему:

Розроблення підсистеми збору даних про пасажиропотоки в інтелектуальній
системі управління громадським транспортом

Виконав: студент групи 4144



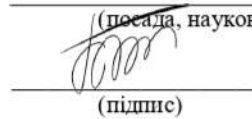
(підпис)

Покровський М.В.

Керівник роботи:

викладач

(посада, науковий ступінь вчене звання)



(підпис)

Бобровський О.С.

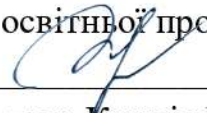
м. Миколаїв – 2026 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет кораблебудування імені адмірала Макарова
Навчально-науковий інститут комп'ютерних наук та управління проектами
Кафедра інформаційних управляючих систем та технологій
Спеціальність 126 "Інформаційні системи та технології"
Освітня програма "Інформаційні системи та технології"

"ЗАТВЕРДЖУЮ"

Гарант освітньої програми


к.т.н, доц. Кнirik Н.Р.

" ____ " ____ 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти "бакалавр"

Студенту Покровському Михайлу Володимировичу

(прізвище, ім'я, по батькові)

1. Тема роботи: Розроблення підсистеми збору даних про пасажиропотоки в інтелектуальній системі управління громадським транспортом

Керівник роботи
Бобровський О.С.

Затверджені наказом ректора № 270-уч від "31" березня 2026 року.

2. Термін подання роботи: 22 червня 2026 р.

3. Вихідні дані до проекту (роботи) ДСТУ щодо обробки інформації, літературні джерела, технічна документація на існуючі аналоги систем,

4. Перелік питань, що належать до розробки (найменування розділів)
1. ФОРМУВАННЯ ВИМОГ ДО ІНФОРМАЦІЙНОЇ СИСТЕМИ ОБЛІКУ ПАСАЖИРОПОТОКУ, РОЗРОБКА ЇЇ КОНЦЕПЦІЇ І ПОСТАНОВКА ЗАДАЧІ
2. СИСТЕМА ЗБОРУ ТА ПЕРЕДАЧІ ДАНИХ
3. ОБРОБКА ТА ЗБЕРІГАННЯ ДАНИХ
4. ОХОРОНА ПРАЦІ

5. Перелік презентаційних матеріалів 1. Загальний вигляд інтелектуальної транспортної системи; 2. Архітектура інтелектуальної системи управління громадським транспортом; 3. Модуль збору даних; 4. Загальна структура системи обліку пасажиропотоку; 5. Алгоритм програми збору даних; 6. ER-діаграма таблиць бази даних; 7. UML-діаграма класів; 8. UML-діаграма послідовності.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Бобровський О.С.	02.02.2026	23.02.2026
2	Бобровський О.С.	23.02.2026	20.03.2026
3	Бобровський О.С.	20.03.2026	24.04.2026
4	Бобровський О.С.	24.04.2026	01.06.2026

7. Дата видачі завдання 02 лютого 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів роботи	Примітка
1	Аналіз предметної галузі	02.02.2026	виконано
2	Постановка задачі	16.02.2026	виконано
3	Розробка концепції	23.02.2026	виконано
4	Розробка проекту ІС	23.03.2026	виконано
5	Реалізація проекту	27.04.2026	виконано
6	Розробка питань з охорони праці	01.06.2026	виконано
7	Розробка графічних матеріалів	05.06.2026	виконано
8	Подання роботи на перевірку на плагіат	08.06.2026	виконано
9	Подання роботи рецензенту	15.06.2026	виконано
10	Подання роботи на захист	22.06.2026	виконано

Студент


(підпис)

Керівник роботи


(підпис)

Покровський М.В.

(ПБ)

Бобровський О.С.

(ПБ)

АНОТАЦІЯ

Кваліфікаційна робота складається із 4 розділів та нараховує 59 сторінок, 3 додатка, 6 таблиць, 13 рисунків та 20 джерел.

У кваліфікаційній роботі розглянуто проблему автоматичного обліку пасажиропотоку у громадському транспорті, яка б збирала потрібні статистичні дані для компанії перевізника та міських служб. Метою роботи являється побудова концепту інтелектуальної транспортної системи, яка дозволить покращити точність обліку пасажирів громадського транспорту.

Для реалізації поставленої мети сформульовані наступні завдання:

- Аналіз предметної області;
- Вибір технології обліку;
- Створення архітектури системи;
- Розробка програмного забезпечення та бази даних.

Об'єктом дослідження є процеси збору та інтеграції даних у реальному часі в межах інтелектуальних транспортних систем.

Предметом дослідження є методи, алгоритми та програмно-апаратні засоби автоматизованого обліку пасажиропотоку.

Ключові слова: інтелектуальна транспортна система, інформаційні технології, моніторинг транспортних засобів, пасажиропотік.

ABSTRACT

The qualification work consists of 4 sections and has 59 pages, 3 appendices, 6 tables, 13 figures and 20 sources.

The qualification work considers the problem of automatic passenger flow accounting in public transport, which would collect the necessary statistical data for the carrier company and city services. The purpose of the work is to build a concept of an intelligent transport system, which will improve the accuracy of accounting for public transport passengers.

To achieve this goal, the following tasks have been formulated:

- Analysis of the subject area;
- Selection of accounting technology;
- Creation of system architecture;
- Development of software and database.

The object of the study is the processes of collecting and integrating data in real time within intelligent transport systems.

The subject of the study is methods, algorithms and software and hardware for automated passenger flow accounting.

Keywords: Intellectual Transport System, Information Technology, Monitoring of Vehicles, Passenger Flow.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	7
ВСТУП.....	9
РОЗДІЛ 1. ФОРМУВАННЯ ВИМОГ ДО ІНФОРМАЦІЙНОЇ СИСТЕМИ ОБЛІКУ ПАСАЖИРОПОТОКУ, РОЗРОБКА ЇЇ КОНЦЕПЦІЇ І ПОСТАНОВКА ЗАДАЧІ ...	11
1.1. Загальна концепція інтелектуальної транспортної системи	11
1.2. Організаційна структура інтелектуальної системи управління громадським транспортом	15
1.3. Концепція інформаційної системи обліку пасажиропотоку та постановка задачі.....	18
РОЗДІЛ 2. СИСТЕМА ЗБОРУ ТА ПЕРЕДАЧІ ДАНИХ.....	21
2.1. Огляд датчиків для вимірювання пасажиропотоку.....	21
2.2. Огляд технологій та протоколів передачі даних	23
2.3. Вибір елементів системи збору та передачі даних.....	27
РОЗДІЛ 3. ОБРОБКА ТА ЗБЕРІГАННЯ ДАНИХ	34
3.1. Загальна структура системи обліку пасажиропотоку	34
3.2. Розробка програми модуля збору даних	34
3.3. Вибір бази даних.....	37
3.4. Розробка бази даних для системи обліку пасажиропотоку	40
3.5. Розробка програми обробки даних на сервері.....	43
РОЗДІЛ 4. ОХОРОНА ПРАЦІ	47
4.1. Правове регулювання охорони праці в Україні	47
4.2. Значення охорони праці у роботі фахівців з інформаційних технологій	48
4.3. Інформаційна культура як важлива складова в управлінні охорони праці ..	50
4.4. Вдосконалення охорони праці фахівців з інформаційних технологій.....	53
4.5. Розрахунок освітленості робочих місць.....	55
ВИСНОВКИ.....	57
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	58
Додаток А. Програма модуля збору даних.....	60
Додаток Б. База даних.....	65
Додаток В. Програма обробки даних на сервері.....	68

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

BLE – Bluetooth Low Energy
CoAP – Constrained Application Protocol
ER-діаграма – діаграма «сутність - зв'язок» (Entity-Relationship Diagram)
GNSS – Global Navigation Satellite System
GPS – Global Positioning System
HDTV – High-Definition Television
HTML – HyperText Markup Language
HTTP – HyperText Transfer Protocol
HTTPS – HyperText Transfer Protocol Secure
I2N – Infrastructure-to-Network
IDE – Integrated Development Environment
IoT – Internet of Things
JSON – JavaScript Object Notation
LoRa – Long Range
LTE – Long-Term Evolution
MQTT – Message Queuing Telemetry Transport
NMEA – National Marine Electronics Association
OBU – On-Board Unit
PoE – Power over Ethernet
QoS – Quality of Service
REST API – Representational State Transfer Application Programming Interface
RSU – Roadside Unit
SIM – Subscriber Identification Module
SQL – Structured Query Language
SSL – Secure Sockets Layer
TCP – Transmission Control Protocol
TLS – Transport Layer Security
UART – Universal Asynchronous Serial Transmitter

UDP – User Datagram Protocol

UML – Unified Modeling Language

URL – Uniform Resource Locator

USB – Universal Serial Bus

V2I – Vehicle-to-Infrastructure

V2N – Vehicle-to-Network

V2P – Vehicle-to-Pedestrian

V2V – Vehicle-to-Vehicle

V2X – Vehicle-to-Everything

VRU – Vulnerable Road User

XML – Extensible Markup Language

БД – база даних

ІСУГТ – інтелектуальна система управління громадським транспортом

ІТ – інформаційні технології

ІТС – інтелектуальна транспортна система

ОЗП – оперативний запам'ятовувальний пристрій

СУБД – система управління базами даних

ВСТУП

Інтелектуальна транспортна система (ІТС) – це система, яка використовує інформаційні та комунікаційні технології для експлуатації та управління інфраструктурою дорожньої мережі, яка включає міжміські та міські дороги. Коли наприкінці 1960-х років були досліджені основи ІТС, у Японії з'явилися комплексні системи управління автомобільним рухом, а також системи електронного орієнтування на маршруті у США та Німеччині. Після цих років додатки ІТС почали набувати широкого поширення в середині 1980-х років. Вони вперше були впроваджені в 1990-х роках з такими додатками, як електронні системи збору плати, інтелектуальні системи контролю перехресть, інформаційні системи для пасажирів і водіїв і центри керування дорожнім рухом. Після того, як дослідження в міжнародному масштабі почали набувати значення, в 1994 році відбувся перший конгрес ІТС і з того часу продовжував проводитися регулярно щороку. В результаті цієї діяльності та академічних досліджень ІТС країни почали створювати власні організації ІТС відповідно до отриманих знань, нових технологічних розробок і потреб. Окрім організацій, створених на національному рівні, є регіональні організації, такі як European Road Transport Telematics Implementation Coordination, ITS America та ITS Asia Pacific [1].

Покращення роботи системи збору даних про пасажиропотоки в ІТС є досить актуальним тому, що отримання належних даних дозволяє оптимізувати кількість одиниць транспорту для перевезення пасажирів на кожному маршруті та покращити економічні показники ІТС в цілому. Метою роботи є розробка підсистеми збору даних про пасажиропотоки в інтелектуальній системі управління громадським транспортом, яка забезпечить більш зручне керування заповненням транспорту в різний час доби та підвищить продуктивність диспетчерських служб. Об'єкт дослідження – процеси збору та інтеграції даних у реальному часі в межах системі управління громадським транспортом. Предметом дослідження є методи, алгоритми та програмно-апаратні засоби автоматизованого обліку пасажиропотоку.

У наслідок проведеного аналізу поточної роботи співробітника диспетчерської служби, було сформовано вимоги до проекту інформаційної

системи. Використовуючи сформовані вимоги розроблено концепцію, що дозволяє вирішити так задачі, як: реєстрація транспортних засобів, ведення їх обліку, розподілення пасажирів між маршрутами, отримання вибірок для створення звітів. На основі концепції розробляється технічний проект, в якому приймаються та розробляються загальносистемні рішення, рішення з інформаційного, організаційного, технічного та програмного забезпечення. Використовуючи розроблену концепцію, для реалізації інформаційної системи обрано сучасні програмні та технічні засоби. Для системи розробляється інструкція користувача, інструкція по формуванню бази даних, інструкція з початку роботи з інформаційною системою. Проведено аналіз небезпечних та шкідливих факторів, розроблено техніку безпеки, заходи по зменшенню впливу небезпечних та шкідливих факторів на робочому місці диспетчера.

РОЗДІЛ 1. ФОРМУВАННЯ ВИМОГ ДО ІНФОРМАЦІЙНОЇ СИСТЕМИ ОБЛІКУ ПАСАЖИРОПОТОКУ, РОЗРОБКА ЇЇ КОНЦЕПЦІЇ І ПОСТАНОВКА ЗАДАЧІ

1.1. Загальна концепція інтелектуальної транспортної системи

ІТС було розроблено, щоб скоротити час у дорозі, підвищити безпеку дорожнього руху, ефективно використовувати існуючу пропускну здатність доріг, підвищити мобільність, ефективно використовувати енергію та зменшити шкоду навколишньому середовищу за допомогою різноманітного обміну даними між користувачами, транспортними засобами, інфраструктурою та навколишнім середовищем. Крім того, системи, засновані на інформаційному зв'язку, включають в себе центральні механізми вимірювання параметрів транспортних засобів, аналізу та контролю дорожньої ситуації. ІТС спрямована на мінімізацію помилок, спричинених людиною. Крім того, можна запобігти втраті часу, пов'язаній із дорожнім рухом, аваріям зі смертельними наслідками та травмами, фінансовим збиткам, забрудненню повітря та багатьом подібним негативним ситуаціям. Загальний вигляд ІТС та її застосування в містах показано на рис. 1.1. [2]

ІТС – це інтегрована транспортна система, що підтримується інформаційно-комунікаційними технологіями. ІТС охоплює стійкі, безпечні та взаємопов'язані транспортні системи, включаючи трамваї, автобуси, метро, автомобілі, морський та повітряний транспорт, пішоходів, велосипедистів, мотоциклістів та інших учасників дорожнього руху. В ІТС існує два основних види вузлів: бортові пристрої транспортних засобів (On-Board Unit – OBU) і пристрої дорожньої інфраструктури (Roadside Unit – RSU). Ці вузли утворюють спеціальну мережу та взаємодіють один з одним за допомогою або безпосереднього зв'язку, або зв'язку з проміжними вузлами. OBU – це електронний пристрій, встановлений у транспортному засобі, який записує дані про дорожній рух і процес водіння та може підключатися до дорожньої інфраструктури та супутникової навігаційної системи. OBU зазвичай

використовуються для автоматизованого виставлення рахунків і реєстрації плати за проїзд, але також можуть використовуватися для додаткових послуг.

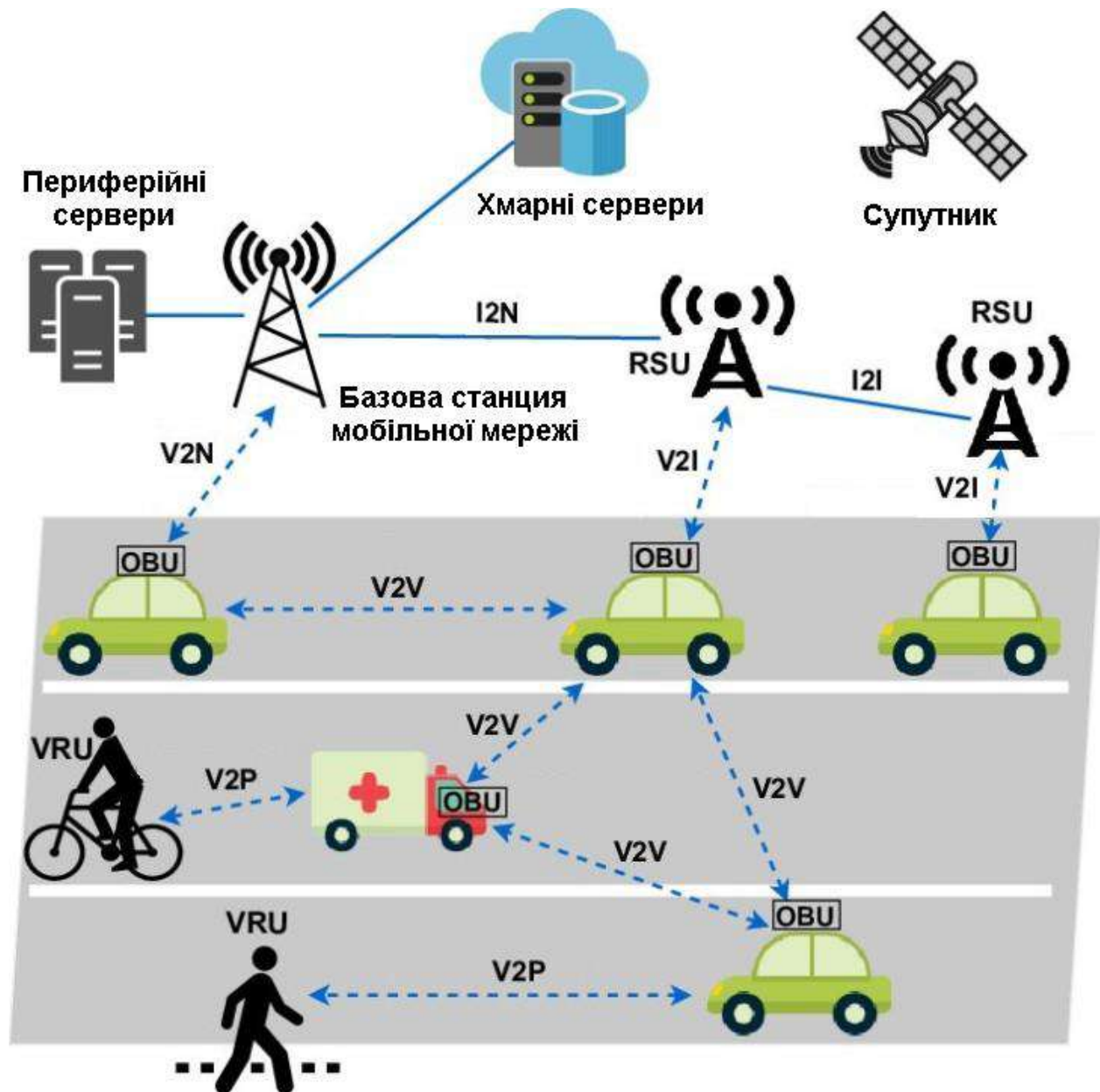


Рис. 1.1. Загальний вигляд ІТС

Наприклад, ОБУ можна використовувати для діагностики та зберігання даних для надзвичайних ситуацій, планування маршруту та навігації. RSU збирають дані про дорожній рух із статичної зони зондування вздовж дороги та передають ці дані на пристрої керування дорожнім рухом, а також у центральний центр керування дорожнім рухом. Ці пристрої також служать джерелом інформації для

інтелектуальних транспортних засобів для збирання майбутньої інформації про дорожній рух. Наведемо декілька прикладів інформації, яку можуть забезпечувати RSU:

- Поточний стан фазування сигналу світлофорів на перехрестях і час, що залишився в цій фазі;
- Місцезнаходження та доступність стоянок, обмеження за часом обслуговування, умови руху та графік навантаження/розвантаження;
- Попередження про негабаритні розміри, які повідомляють водії, якщо їхній транспортний засіб не може проїхати через міст або тунель;
- Попередження про обмеження смуги руху для автомобілів із високою завантаженістю, громадського чи спеціального транспорту;
- Інші варіанти попереджень: про швидкість на повороті, про знижену швидкість, про пішоходів на переході для громадського транспорту, про перетин залізниці, про погодні умови тощо.

Також розглядають так званих вразливих учасників дорожнього руху (Vulnerable Road User – VRU) – це узагальнюючий термін, що використовується для позначення групи учасників дорожнього руху, які більш схильні до аварій і травм на дорозі, ніж середньостатистичний учасник дорожнього руху. До цієї категорії входять пішоходи, велосипедисти, мотоциклісти та інші. Взаємодія VRU з іншими вузлами ІТС дозволяє уникнути небезпечних ситуацій на дорозі.

Окрім вищезгаданого, до складу ІТС входять базові станції мобільної мережі, які забезпечують зв'язок учасників дорожнього руху та інфраструктурних пристроїв з центрами керування дорожнім рухом у вигляді периферійних та хмарних серверів. Також слід відмітити використання супутникових систем навігації (Global Navigation Satellite System – GNSS) для отримання інформації про положення учасників руху.

Для взаємодії між елементами ІТС використовуються наступні зв'язки:

- Зв'язок між транспортними засобами (Vehicle-to-Vehicle – V2V);
- Зв'язок між транспортним засобом та елементом дорожньої інфраструктури (Vehicle-to-Infrastructure – V2I);

- Зв'язок між транспортними засобами та VRU (Vehicle-to-Pedestrian – V2P);
- Зв'язок між транспортним засобом та мобільною мережею (Vehicle-to-Network – V2N);
- Зв'язок між елементом дорожньої інфраструктури та мобільною мережею (Infrastructure-to-Network – I2N);
- У деяких випадках замість V2V, V2I, V2P, V2N використовується узагальнений термін «зв'язок між транспортним засобом та його оточенням» (Vehicle-to-Everything – V2X).

Додатки ІТС охоплюють різноманітні технології та рішення для підвищення ефективності, безпеки та стійкості транспортних мереж. Ось деякі ключові застосування ІТС [3, 4]:

Управління дорожнім рухом: ІТС використовує збір даних у реальному часі, датчики моніторингу руху та прогнозу аналітику для оптимізації транспортного потоку та зменшення заторів на дорозі. Адаптивні системи управління сигналами дорожнього руху регулюють час сигналів на основі поточних умов руху, щоб підвищити ефективність перехрестя.

Управління громадським транспортом: ІТС покращує роботу та управління системами громадського транспорту, включаючи автобуси, поїзди та метро. Системи інформації про пасажирів у режимі реального часу надають пасажирам актуальну інформацію про розклади, затримки та збої в обслуговуванні, покращуючи загальний досвід пасажирів.

Інформаційні системи для подорожуючих: додатки ІТС надають мандрівникам інформацію в режимі реального часу про стан дорожнього руху, закриті дороги, погоду та альтернативні маршрути. Це дає змогу користувачам приймати обґрунтовані рішення та ефективніше планувати свої подорожі.

Ці додатки демонструють різні способи, як ІТС революціонізують наш спосіб переміщення пасажирів і товарів, створюючи безпечніші, ефективніші та стійкіші транспортні мережі.

1.2. Організаційна структура інтелектуальної системи управління громадським транспортом

Організаційна структура інтелектуальної системи управління громадським транспортом (ІСУГТ) інтегрує апаратно-програмні комплекси, транспортних операторів та міську інфраструктуру задля оптимізації пасажирських перевезень. Ця структура об'єднує управлінські, інформаційні та технічні ланки в єдину мережу [5]. Основні елементи організаційної структури ІСУГТ:

1. Адміністрація;
2. Оперативно-диспетчерський контроль;
3. Інформаційно-комунікаційна система;
4. Виконавці.

Адміністрація відповідає за стратегічне планування, фінансування та формування транспортної політики на рівні міста чи регіону. Органи місцевого самоврядування (міська рада, департаменти транспорту) затверджують маршрутну мережу, тарифи, інвестиційні програми. Оператори-координатори (наприклад, єдиний оператор муніципального транспорту) укладають договори з перевізниками (принцип «керівник-виконавець»), розподіляють транспортну роботу та координують розклади.

Оперативно-диспетчерський контроль – це основне ядро, де приймаються поточні управлінські рішення на основі телематичних даних. Центр управління (ситуаційний центр) збирає, аналізує інформацію та здійснює пряме втручання в роботу мережі. Диспетчери (маршрутні та мережеві) контролюють дотримання графіка руху, реагують на форс-мажорні ситуації (аварії, затори), коригують інтервали руху в режимі реального часу.

Інформаційно-комунікаційна система забезпечує технічний збір даних та обмін інформацією між рухомим складом і центром. Бортові комп'ютери транспортних засобів фіксують координати, швидкість, технічний стан та передають їх через бездротові мережі до диспетчерського центру. Автоматизована система контролю та оплати проїзду включає до себе валідатори, термінали,

системи обліку пасажиропотоку. Підсистеми зв'язку забезпечують безперебійний зв'язок транспортних засобів з інфраструктурою.

Виконавці безпосереднє забезпечують роботу транспорту та взаємодію з пасажиром. Транспортні підприємства здійснюють випуск рухомого складу на лінію, проводять технічне обслуговування та ремонт, забезпечують роботу водіїв. Розумні зупинки показують точний час прибуття транспорту. Мобільні застосунки та портали використовуються для планування маршрутів з урахуванням пересадок, відстеження руху онлайн та купівлі єдиного електронного квитка. Пасажири – це кінцеві користувачі, які впливають на систему через генерацію попиту, використання електронних квитків та отримання інформації про прибуття транспорту.

Завдяки функціонуванню цієї структури місто отримує можливість автоматизовано оптимізувати маршрути, управляти світлофорами для надання пріоритету громадському транспорту, а також оперативно інформувати пасажирів. На рис. 1.2 наведено архітектуру інтелектуальної системи управління громадським транспортом з виділенням трьох рівнів:

1. Користувачі;
2. Сервіси;
3. Інфраструктура.

Система повинна вирішувати наступні завдання:

1. Ведення реєстрів маршрутів, зупиночних пунктів, об'єктів транспортної інфраструктури, організацій та транспортних засобів, що обслуговують маршрути регулярних перевезень пасажирів;
2. Облік даних витрати палива, транспортної роботи за маршрутами регулярних перевезень;
3. Прогнозування прибуття транспортних засобів на зупиночні пункти та фактичних інтервалів, потреби у рухомому складі в залежності від змін пасажиропотоку;

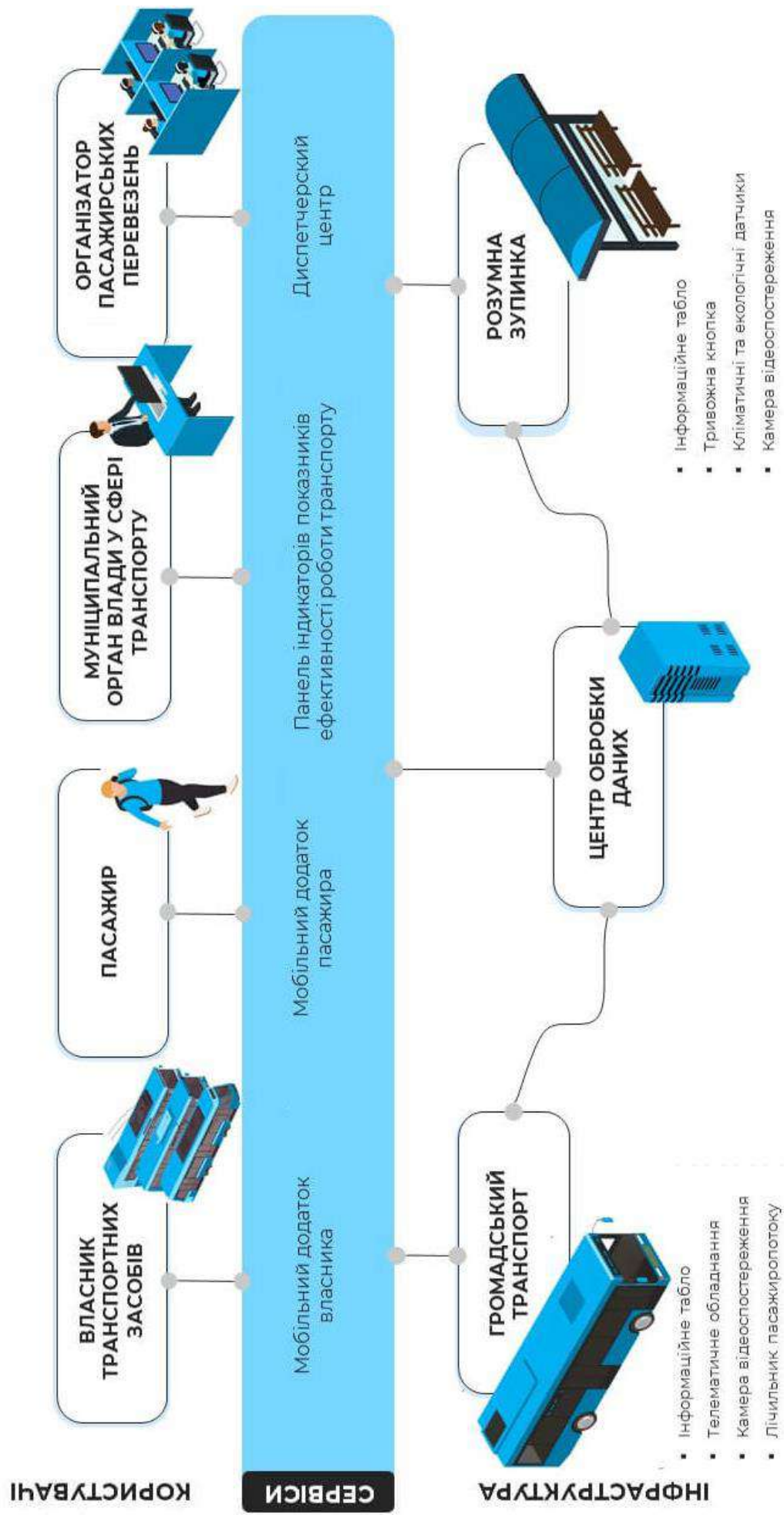


Рис. 1.2. Архітектура інтелектуальної системи управління громадським транспортом

4. Надання даних про місце розташування об'єктів моніторингу, обсяг пасажиропотоку, дотримання траси маршруту та відхилення від графіка руху;
5. Оперативне повідомлення населення про роботу громадського транспорту за допомогою мобільних додатків, Інтернет-порталу, сторонніх сервісів, систем інформування пасажирів на пунктах зупинки;
6. Контроль виконання транспортної роботи та облік претензій до перевізників;
7. Планування роботи по оптимізації розкладу руху, графіків та маршрутів громадського транспорту;
8. Підготовка звітів про маршрути руху громадського транспорту, розклади руху, розміщення зупинок, пасажиропотік та інше.

Впровадження такої системи дозволить покращати безпеку руху пасажирського автотранспорту, підвищити якість обслуговування пасажирів, підвищити ефективність використання рухомого складу пасажирського автотранспорту, а також продуктивність праці працівників, зайнятих пасажирськими перевезеннями.

1.3. Концепція інформаційної системи обліку пасажиропотоку та постановка задачі

Інформаційна система обліку пасажиропотоку – це комплекс апаратно-програмних засобів для автоматизованого вимірювання, аналізу та прогнозування переміщень людей у мережі громадського транспорту. Вона є цифровою основою для управління міською мобільністю та оптимізації транспортної інфраструктури. Концепція системи базується на інтегрованій цифровій платформі, яка збирає інформацію в режимі реального часу. Основа концепції включає чотири рівні:

1. Рівень збору забезпечує автоматичні підрахунки на масиву датчиків, відеокамер (комп'ютерний зір) та/або валідаторів електронних квитків;

2. Рівень передачі реалізує бездротові канали, які передають потрібну інформацію між бортовими комп'ютерами транспортних засобів та серверами;
3. Рівень обробки – це хмарні або локальні сервери, які зберігають та оброблюють дані, а також забезпечують інформаційні портали для користувачів системи;
4. Рівень аналітики забезпечує формування звітів, матриць кореспонденції та динамічних моделей для прийняття рішень.

Система вирішує наступні технічні та управлінські завдання:

1. Аналітичні:

- Оцінка обсягів – визначення точної кількості перевезених пасажирів за певний проміжок часу;
- Аналіз навантаження – розрахунок коефіцієнта наповнюваності транспорту на різних ділянках маршруту;
- Матриця кореспонденції – визначення найпопулярніших напрямків поїздок для проектування нових маршрутів;

2. Оперативні:

- Корекція графіків – збільшення чи зменшення кількості рухомого складу на лінії залежно від фактичного попиту;
- Контроль за дотриманням розкладу – синхронізація зі службою диспетчеризації для уникнення інтервалів та скупчень транспорту;

3. Економічні:

- Оптимізація витрат – уникнення холостих пробігів автобусів, що зменшує витрати палива та викиди;
- Фінансовий контроль – зіставлення даних фізичного пасажиропотоку з транзакціями по оплаті для мінімізації тіньових перевезень.

Висновки за розділом 1

Проаналізовано основні компоненти ІТС та взаємодію між ним. Розглянуто організаційну структуру ІСУГТ. Сформовано концепцію інформаційної системи обліку пасажиропотоку.

РОЗДІЛ 2. СИСТЕМА ЗБОРУ ТА ПЕРЕДАЧІ ДАНИХ

2.1. Огляд датчиків для вимірювання пасажиропотоку

Існує велика кількість методів підрахунку, що використовуються автоматичними системами обліку пасажиропотоку [6-10]. Найчастіше використовуються наступні датчики:

- Натискні пластини;
- Інфрачервоні датчики;
- Системи комп'ютерного зору.

Датчики відрізняються між собою методами встановлення, технологіями підрахунку пасажирів, ціною та ефективністю тощо. Для вибору датчика автоматичної системи обліку потрібно виділити їхні переваги та недоліки.

Натискні пластини реєструють пасажирів, коли вони наступають на пластину. Датчики розташовуються на сходах транспортних засобів, або за їх відсутності поблизу дверей на підлозі. Металева конструкція датчику може бути покрита шаром гуми і прикріплена до ступенів за допомогою спеціально виділених кріпильних конструкцій або просто склеєна за допомогою клею.

Переваги натискних пластин:

- Простий монтаж;
- Не потрібні великі обчислювальні ресурси для обліку;
- Анонімні дані.

Недоліки натискних пластин:

- Є рухомі частини;
- Для визначення напрямку руху пасажирів треба встановлювати подвійну кількість датчиків;
- Менш точний, оскільки важко підрахувати великі групи людей.

Інфрачервоні датчики реагують на перетин пасажиром інфрачервоного променя. Інфрачервоні датчики встановлюють біля дверних прорізів.

Їх можна розділити на дві основні категорії, а саме:

- Датчики активного типу – складаються з передавача і приймача, вони створюють промінь і працюють як детектор увімкнення/вимкнення;
- Датчики пасивного типу – ділять інфрачервоний промінь на пучки, створюючи таким чином зону об'ємного виявлення.

Переваги інфрачервоних датчиків:

- Є одними з найдоступніших варіантів для підрахунку людей;
- Не потрібні великі обчислювальні ресурси для обліку;
- Анонімні дані;
- Немає рухомих частин.

Недоліки інфрачервоних датчиків:

- Більш складний монтаж;
- Для визначення напрямку руху пасажирів треба встановлювати подвійну кількість датчиків.

Системи комп'ютерного зору переважно засновані на двох стереоскопічних камерах. Дані системи встановлюють над дверима, вони фіксують зображення в області під пристроями. Камери оснащуються світлодіодами, що дозволяє системі працювати при будь-яких умовах освітлення.

Переваги та систем комп'ютерного зору:

- Висока точність підрахунку;
- Може визначати напрямок руху пасажирів, їх стать та вік, а також відрізняти пасажирів від інших об'єктів;
- Дозволяє перевіряти пасажирів з метою безпеки.

Недоліки оптичних датчиків та систем комп'ютерного зору:

- Потрібні обчислювальні ресурси для обробки зображень;
- Можливо порушення анонімності пасажирів;
- Велика ціна обладнання;
- Дорогий і складний монтаж.

Також можливий облік за допомогою аналізу Wi-Fi/Bluetooth запитів (Probe Requests) смартфонів пасажирів. Аналіз Probe Requests – це метод пасивного перехоплення сигналів, які смартфони пасажирів безперервно випромінюють для

пошуку мереж Wi-Fi або Bluetooth. Це дозволяє виявляти присутність пристроїв та, відповідно, їх власників. Хоча подібні системи є досить простими для реалізації, їх використання обмежено відносно низькою точністю тому, що наявність запитів залежить від налаштувань смартфонів пасажирів. Також пасажир може мати більше одного смартфона.

2.2. Огляд технологій та протоколів передачі даних

Вибір технології передачі даних відіграє важливу роль у проектуванні системи обліку пасажиропотоку [11-16]. Нижче наведено критичні параметри для вибору технології, щоб зробити канал передачі доступним, масштабованим та ефективним відповідно до потреб системи:

- Ефективна радіовідстань/швидкість передачі даних;
- Робочий діапазон частот;
- Моделі розгортання мережі;
- Підтримувані функції безпеки.

Найчастіше для взаємодії з транспортними засобами використовуються наступні комунікаційні технології:

- Технології низькоенергетичних широкосмугових мереж, такі як LoRa;
- Стільникові технології, зараз в основному використовується LTE;
- Бездротові технології малого радіусу дії, такі як Wi-Fi, Bluetooth, Zigbee.

Специфікації мережевих технологій представлені в таблиці 2.1.

Всі розглянуті технології мають достатні можливості з безпеки даних, що передаються, тому основними критеріями залишаються перші три. Оскільки дані потрібно передавати в режимі реального часу на великі відстані, то найбільш прийнятними технологіями є LTE і LoRa.

Таблиця 2.1. Специфікації мережевих технологій

Технологія	Відстань	Частота	Швидкість	Споживання
LTE	~10 км	900, 1800, 2100, 2300, 2600 МГц	> 10 Мбіт/с	Високе
BLE	< 100 м	2,4 ГГц	< 3 Мбіт/с	Низьке
WI-FI	< 300 м	2,4 ГГц, 5 ГГц	> 50 Мбіт/с	Середнє
ZigBee	< 100 м	2,4 ГГц	250 кбіт/с	Низьке
LoRa	1-15 км	< 1 ГГц	0,3-50 кбіт/с	Низьке

Мережі LTE та LoRa в цілому мають схожі концепції побудови – пристрої зв'язуються один з одним не безпосередньо, а за допомогою базових станцій, які утворюють інфраструктуру мережі. LTE використовує ліцензований спектр частот, а LoRa – безліцензійні радіочастотні діапазони нижче 1 ГГц. Мережі LTE вже розгорнуті та обслуговуються постачальниками цих мереж. На відміну від цього, мережу LoRa, як правило, потрібно розгорнути та обслуговувати самостійно.

Для взаємодії між пристроями в автоматизованих системах запропоновано декілька протоколів спілкування. Найбільш часто використовуються протокол MQTT і протоколи на основі HTTP – CoAP або REST API.

MQTT – це легкий мережевий протокол для обміну повідомленнями. Він є стандартом для Інтернету речей (IoT) та міжмашинної взаємодії завдяки низькому споживанню трафіку та енергії. Працює за моделлю «видавець – підписник». Пристрої не спілкуються напряму, а використовують брокер – центральний сервер. Повідомлення сортуються за «темами». Пристрій-видавець публікує дані (наприклад, показник температури) у відповідну «тему». Брокер пересилає це повідомлення всім пристроям, які підписані на дану «тему».

Переваги:

- Низькі витрати – заголовки пакетів дуже малі, тому протокол ідеально підходить для слабких мереж та пристроїв з обмеженим живленням;

- Невимогливість до каналу – легко відновлює з'єднання при розривах зв'язку;
- Рівні надійності (QoS) – має налаштування, що гарантують доставку повідомлення, навіть якщо зв'язок тимчасово зник.

Недоліки:

- Залежність від єдиної точки відмови (брокера) – якщо брокер вийде з ладу або втратить зв'язок, усі пристрої втратять здатність комунікувати;
- Нестандартність для потокового відео чи файлів – MQTT ідеально підходить для невеликих текстових команд та сенсорів, але абсолютно не оптимізований для передачі великих обсягів даних;
- Складнощі маршрутизації – оскільки пристрої обмінюються повідомленнями через «теми», а не прямими URL-адресами, то динамічне створення та управління складними ієрархіями «тем» вимагає багато ресурсів і ретельного планування;
- Базова безпека – протокол часто працює без шифрування, тому для захисту даних від перехоплення та забезпечення автентифікації необхідно обов'язково розгорнути шифрування TLS/SSL.

REST API та CoAP – це програмні інтерфейси для обміну даними, які базуються на архітектурному стилі REST, але оптимізовані для абсолютно різних умов. REST API – це фундаментальний підхід до взаємодії веб-додатків. Використовується для веб-сайтів, мобільних додатків, хмарних сервісів. Працює поверх протоколу HTTP/HTTPS. Використовує з'єднання TCP, яке гарантує доставку всіх пакетів, але потребує багато ресурсів та потужності. Найчастіше обмінюється інформацією у вигляді текстових файлів JSON, XML та HTML. Використовує стандартні команди запитів для створення, читання, зміни або видалення даних: GET, POST, PUT, DELETE.

Переваги:

- Висока швидкість і простота – працює на базі звичайних HTTP/HTTPS запитів, що полегшує розробку та налагодження;

- Гнучкість форматів – підтримує передачу даних у форматах JSON, XML та HTML;
- Потужне кешування – має вбудовану підтримку механізмів HTTP-кешування для зниження навантаження на сервер;
- Екосистема – величезна кількість готових бібліотек і інструментів для будь-якої мови програмування.

Недоліки:

- Надмірність даних – вимагає передачі великих заголовків, що збільшує трафік;
- Нестача оптимізації для IoT – занадто важкий, вимагає значної енергії та пам'яті для обробки пристроями з обмеженими ресурсами.

CoAP – це спеціалізований "полегшений" протокол, створений для пристроїв з обмеженими ресурсами, яким не вистачає потужності для роботи з класичним HTTP/REST. Використовується для IoT, бездротових датчиків, в розумних будинках (сенсори, лічильники). Працює поверх UDP. Завдяки цьому він використовує мінімальний розмір заголовків, заощаджує заряд батареї та витрачає менше трафіку. Зазвичай використовує стислий бінарний формат, що дозволяє передавати дані дуже швидко. Так само використовує методи GET, POST, PUT, DELETE. Крім того, він має функцію Observe – коли сервер може сповіщати клієнтів про зміну даних автоматично, без необхідності періодично опитувати їх.

Переваги:

- Енергоефективність і легкість – використовує протокол UDP замість TCP, а також розмір заголовка CoAP становить від 4 байт;
- Модель "клієнт-сервер" – подібно до REST, використовує базові методи запиту (GET, POST, PUT, DELETE), що дозволяє легко конвертувати його у звичайний REST API через спеціальні шлюзи;
- Вбудована підтримка підписок – завдяки механізму спостереження (Observe), клієнт може підписатися на зміни даних датчика, не роблячи постійні запити;

- Робота в автономному режимі – краще справляється з поганим зв'язком і втратами пакетів завдяки вбудованому управлінню надійністю передачі (можливість підтвердження доставки).

Недоліки:

- Необхідність шлюзів – веб-браузери не розуміють CoAP безпосередньо, для взаємодії з додатком, потрібен CoAP-HTTP проксі-сервер.
- Обмеження безпеки – замість важкого SSL/TLS використовує DTLS, що іноді створює додаткові складнощі при налаштуванні захисту.

В цілому слід відмітити, що використання певного протоколу залежить від можливостей пристроїв, які будуть взаємодіяти, а також від можливостей мережі, за допомогою якої будуть передаватися дані. REST API є найбільш універсальним та масштабованим, але важким. MQTT є найбільш легким та пристосованим до низького споживання. CoAP є проміжною версією.

2.3. Вибір елементів системи збору та передачі даних

Базовим модулем системи збору та передачі даних є модуль, який буде встановлюватись на транспортний засіб. Типовий транспортний засіб для великого міста зазвичай має 2 або 3 двері, потік пасажирів через які необхідно контролювати. Контроль всередині транспортного засобу відразу дає можливість прив'язки до певного маршруту та практично не має залежності точності обліку від пори року та погоди. При контролі на зупинці додатково потрібно розпізнавати номер маршруту, одночасно контролювати декілька транспортних засобів, а також мати більш потужний пристрій обробки зображень для точного розпізнавання пасажирів в умовах поганої погоди. Тому в цієї роботі контроль пасажирів на зупинці не розглядається.

Незважаючи на високу ціну, для побудови системи обліку пасажиропотоку в якості датчику, що буде використовуватись для обліку, обираємо системи комп'ютерного зору. Це обумовлено високою точністю обліку (приблизно 98%) та додатковою функціональністю, яку забезпечують такі системи. Можливість

відрізняти пасажирів від інших об'єктів дозволяє уникнути помилкових спрацьовувань. А можливість визначати вік пасажирів дозволить ефективно контролювати процес оплати проїзду.

На рис. 2.1 показана типова конфігурація системи комп'ютерного зору для транспортного засобу з 2 дверима. Також показано зовнішній вигляд 3D-камери.

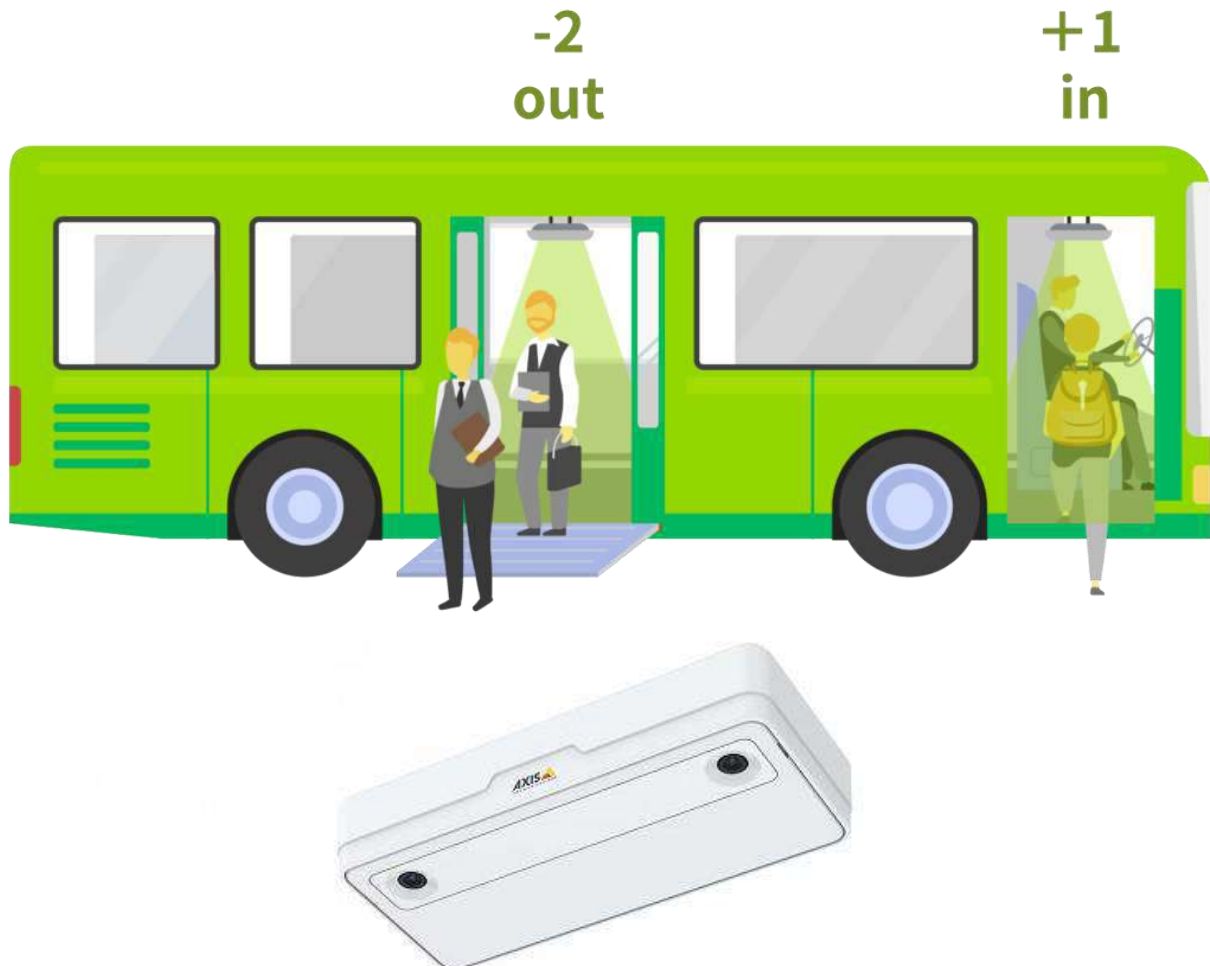


Рис. 2.1. Типова конфігурація системи комп'ютерного зору для транспортного засобу з 2 дверима

Кількість необхідних пристроїв дорівнює кількості дверей транспортного засобу, через які проходять пасажирів. Тобто в нашому випадку для базового модуля знадобиться дві або три 3D-камери.

Обрано пристрій AXIS P8815-2 3D People Counter [17]. Він має наступні параметри:

1. Функціональність:

- Двонаправлений підрахунок;

- Оцінка заповненості;
 - Гнучка зона підрахунку;
 - Підтримка широких входів завдяки використанню двох лічильників;
 - Анонімізація відеопотоку;
 - Виключає об'єкти розміром нижче 110 см;
2. Максимальний розмір зони підрахунку при монтажі на висоті 250 см – 199 x 46 см;
 3. Мінімальне освітлення – 5 люкс;
 4. Контролер:
 - Модель ARTPEC-6;
 - Пам'ять 1024 МБ ОЗП, 512 МБ флеш-пам'яті;
 5. Відео:
 - Стиснення – H.264 та Motion JPEG;
 - Роздільна здатність – 1920x1080 HDTV 1080p;
 - Частота кадрів 30/25 кадрів/с (60/50 Гц);
 - Потоків відео – кілька індивідуально налаштовуваних потоків у H.264 та Motion JPEG;
 6. Живлення:
 - Живлення через Ethernet (PoE) IEEE 802.3af/802.3at Type 1 Class 3\$
 - Споживання – типове 4,4 Вт, максимальне 6 Вт.

Також для контролю стану дверей транспортного засобу необхідні датчики відкриття/закриття. Обираємо герконовий сенсор 59140-1-V-02-F від Littelfuse Inc. Додатково потрібні магніти 57140-000 від Littelfuse Inc. Вони показані на рис. 2.2.



Рис. 2.2. Датчик відкриття/закриття

Для визначення координат транспортного засобу обираємо GPS-модуль GY-NEO6MV2 [18, 19]. Його технічні характеристики:

- Чіпсет: NEO-6M;
- Інтерфейс: UART;
- Швидкість передачі даних: 9600 (за замовчуванням);
- Протокол даних: NMEA 0183;
- Точність визначення позиції: до 2.5 м;
- Холодний старт: приблизно 27 с;
- Гарячий старт: приблизно 1 с;
- Антена: зовнішня керамічна патч-антена з коаксіальним роз'ємом;
- Напруга живлення: 3.3-5В;
- Резервне живлення: вбудована батарейка для збереження даних альманаху та ефемерид супутників.

Зовнішній вигляд GPS-модуля показано на рис. 2.3.

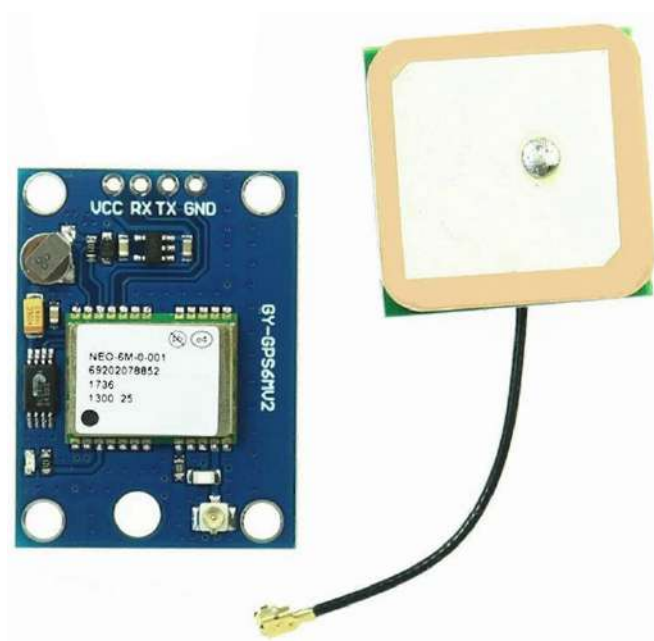


Рис. 2.3. GPS-модуль

Для обробки даних від всіх сенсорів та формування пакету даних для серверу обираємо Raspberry Pi 5 як найбільш актуальний мікрокомп'ютер з великою

кількістю бібліотек. Окрім обліку пасажиропотоку він буде займатися опитуванням датчиків стану транспортного засобу, реалізацією системи візуального та звукового оповіщення пасажирів про зупинки, зміни маршруту та інше. На рис.2.4 наведено зовнішній вигляд Raspberry Pi 5.

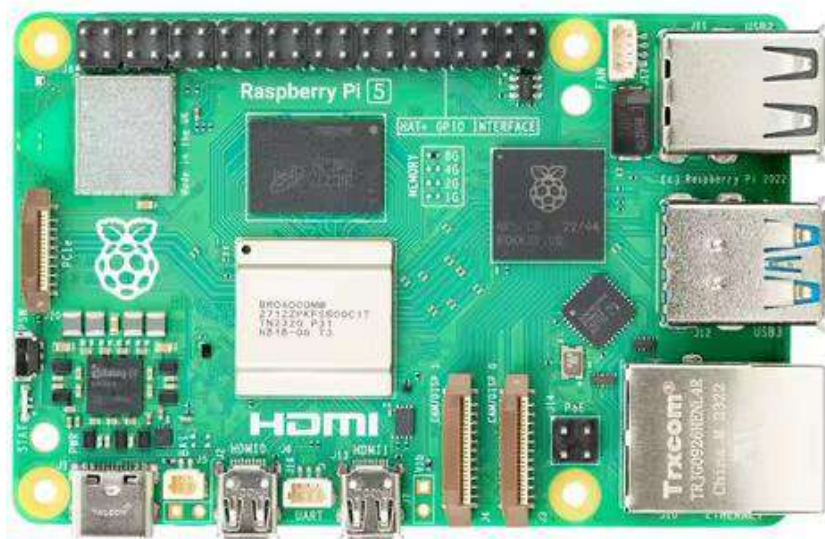


Рис. 2.4. Raspberry Pi 5

Для зв'язку з сервером обираємо технологію LTE тому, що необхідний лише пристрій для передачі даних, а мережа вже розгорнута. В якості пристрою зв'язку обираємо USB-модем Huawei E3372-325. Він має наступні параметри:

- Частотні діапазони:
 - 3G: 850/900/2100 МГц;
 - LTE FDD: Band 1/3/5/7/8/20/28 (2100/1800/850/2600/900/800/700 МГц);
 - LTE TDD: Band 38/39/40/41 (2600/1900/2300/2500 МГц);
- Швидкість прийому: до 150 Мбіт/с;
- Формат SIM: Mini-SIM;
- Чіпсет: ASR1803E;
- Роз'єми: 1 x Mini-SIM та 2 x CRC9 (для підключення 3G/4G-антен);
- Максимальне енергоспоживання: 3,5 Вт.

Зовнішній вигляд модема показано на рис. 2.5.



Рис. 2.5. USB-модем

Також для побудови системи необхідний комутатор. Обираємо TP-LINK LS1005G з наступними параметрами:

- Стандарти та протоколи: IEEE 802.3i/802.3u/ 802.3ab/802.3x;
- Інтерфейс: 5 портів 10/100/1000 Мбіт/с;
- Напруга живлення: 5 В;
- Максимальне споживання: 3,7 Вт.

Зовнішній вигляд комутатора наведено на рис.2.6.



Рис. 2.6. Комутатор TP-LINK LS1005G

Обрано всі елементи системи збору та передачі даних, яка буде встановлюватись на транспортний засіб.

Висновки за розділом 2

Розглянуто використання датчиків та їх функції в системі розумного громадського транспорту. Обрано технологію зв'язку в системі розумного громадського транспорту. Обрано елементи системи розумного громадського транспорту.

РОЗДІЛ 3. ОБРОБКА ТА ЗБЕРІГАННЯ ДАНИХ

3.1. Загальна структура системи обліку пасажиропотоку

Загальна структура системи обліку пасажиропотоку зображено на рис. 3.1.



Рис. 3.1 Загальна структура системи обліку пасажиропотоку

Апаратна частина системи розташована на транспортному засобі (модуль транспортного засобу). Цей модуль надсилає дані до серверу, де серверний додаток оброблює дані та записує результат у базу даних. Також серверний додаток може обмінюватись інформацією з клієнтським додатком.

3.2. Розробка програми модуля збору даних

Базовий модуль, встановлений на транспортному засобі містить Raspberry Pi, який має велику кількість різноманітних бібліотек. Розробимо програму збору даних на мові Python. Алгоритм програми наведено на рис. 3.2, а програму – в додатку А. Програма складається з блоку ініціалізації та основного робочого циклу. В робочому циклі перевіряється стан датчиків відкриття/закриття дверей транспортного засобу (з додатковою фільтрацією брязкоту контактів).

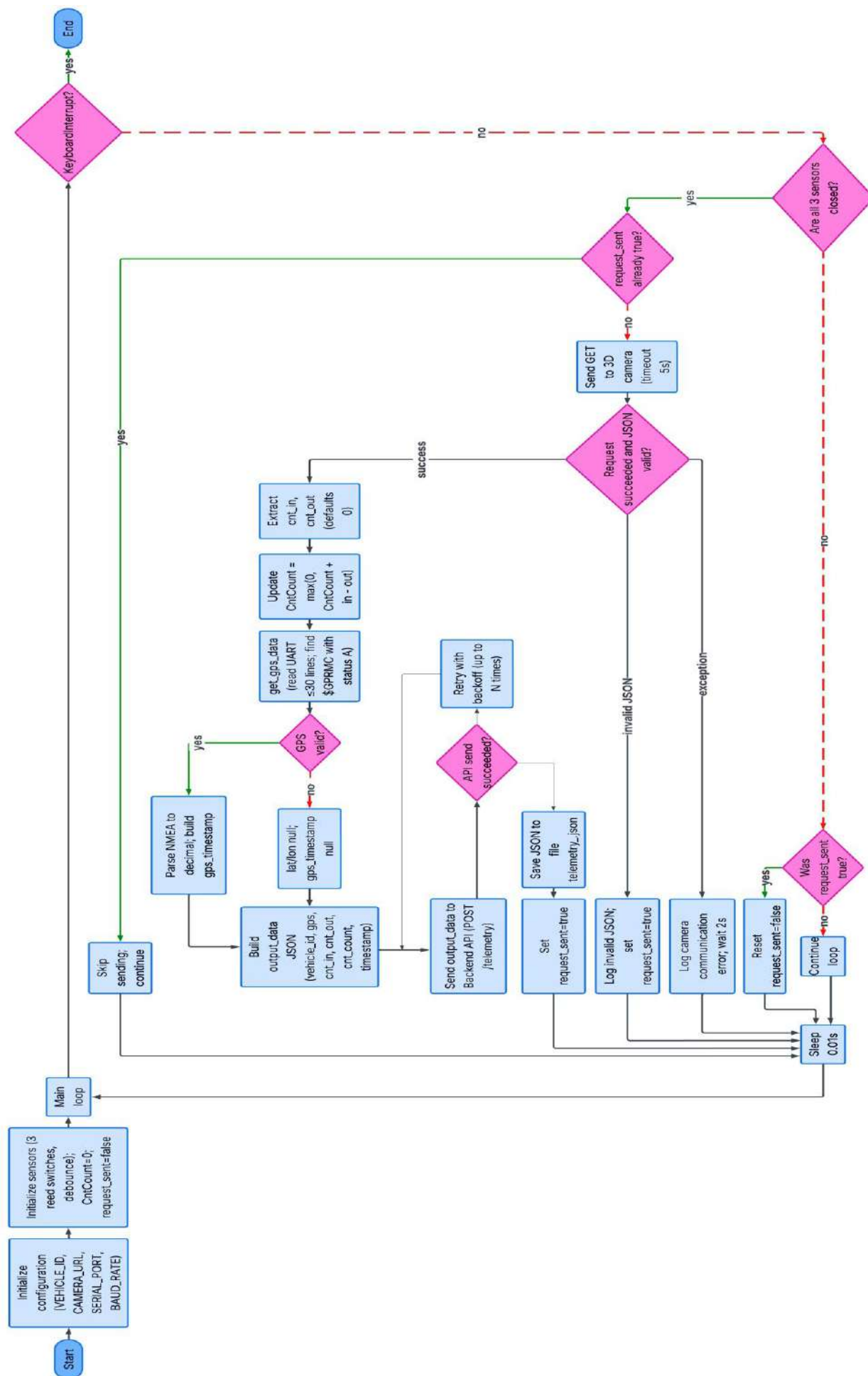


Рис. 3.2. Алгоритм програми збору даних

Якщо всі двері зачинені, то перевіряється змінна `request_sent`, яка показує успішність попереднього запиту до датчиків. Якщо попередній запит відбувся успішно, то очікується наступний цикл відкриття/закриття дверей.

Якщо з моменту останнього закриття дверей запит ще не відбувся, то починається спроба отримати інформацію від 3D-камер (в коді та алгоритмі наведено запит тільки до однієї камери для зменшення об'єму коду та алгоритму). Для отримання даних від 3D-камери необхідно відправити запит:

`http://<DeviceAddr>/stereo/people-counter/counts.json`

де `DeviceAddr` – адреса пристрою в локальній мережі.

У відповідь пристрій надсилає JSON-файл:

```
{
  "in": CntIN,
  "name": "DeviceName",
  "out": CntOUT,
  "serial": "MACAddr",
  "timestamp": "YYYYMMDDhhmmss"
}
```

де `CntIN` – кількість пасажирів, що увійшли на даний момент,

`CntOUT` – кількість пасажирів, що вийшли на даний момент,

`DeviceName` – ім'я пристрою,

`MACAddr` – MAC адреса пристрою,

`YYYYMMDDhhmmss` – поточний час.

Якщо отримані дані є коректними, то них виділяється інформація про кількість пасажирів, які увійшли або вийшли, потім обчислюється кількість пасажирів в салоні. Далі відбувається запит до GPS-модуля для отримання координат. Якщо дані від GPS-модуля також є коректними, то формується JSON-файл, який відправляється на сервер та зберігається на пристрої. Після чого змінна `request_sent` переключається в стан, який відповідає успішному запиту. Робочий цикл продовжується після паузи 10 мс.

3.3. Вибір бази даних

База даних це сховище в якому зберігається дані в залежності від критеріїв які їм підходять. Баз даних є велика кількість як реляційних так і нереляційних. Для огляду обираємо наступні системи управління базами даних (СУБД):

- MySQL;
- SQLite;
- PostgreSQL;
- MongoDB.

Кожна з цих СУБД використовує структуровану мову запитів SQL. Для вибору СУБД розберемо та проаналізуємо кожную з вибраних СУБД.

MySQL – це одна з найпоширеніших та популярних систем управління базами даних. MySQL являється реляційною СУДБ. MySQL безкоштовне та багато платформне програмне забезпечення. Головною особливістю є безпека даних. У всьому світі дана СУБД вважається найбільш безпечною та надійною. Всі дані захищенні паролем які зберігаються зашифрованому вигляді. У СУБД має велику кількість розробників яка займається випуском оновлень та виправлень.

До особливостей даної СУБД можна віднести:

- Безпека даних;
- Масштабування;
- Гнучкість;
- Швидкість;
- Портативність.

SQLite – безплатна реляційна система управління базами даних яка може працювати незалежно від обраної платформи. Особливістю даної СУБД є легке вбудовування в додатки, так як БД складається з одного файлу, що дозволяє без зайвих проблем переносити БД з однієї машини на іншу. Системи управління базами даних SQLite не підходить для використання якщо записуються великі обсяги даних у БД.

Особливості SQLite:

- Продуктивність;
- Надійність;
- Портативність;
- Не потрібне встановлення.

Об'єктно-реляційна СУБД PostgreSQL яка базується на мові SQL та має відкритий вихідний код. Вона підтримує БД необмеженого розміру, має надійні механізми транзакцій. Її можуть модифікувати одночасно кілька користувачів. Добре отримує дані із зовнішнього джерела даних. В порівнянні з іншими БД відносно не дорога. БД регулярно оновлюється, а також завжди можна отримати підтримку від спільноти. Але налаштування можуть бути складними для початківців.

Переваги СУБД PostgreSQL:

- Ефективне отримання даних із зовнішніх джерел;
- Недорога в порівнянні з іншими лідерами ринку;
- Висока продуктивністю;
- Стабільність;
- Швидкість реагування;
- Управління ресурсами;
- Адаптивність у різних середовищах.

Недоліки СУБД PostgreSQL:

- Налаштування за замовчуванням не оптимізовано для сучасного обладнання;
- Відсутня вбудована підтримка налаштувань із кількох майстрами

MongoDB – це документно-орієнтована база даних NoSQL, яка використовується для зберігання великих обсягів даних. MongoDB працює як своєрідна база даних, тільки вона має інший формат, ніж більшість, оскільки вона повністю заснована на документації. Ця БД з'явилася приблизно в середині 2000-х років. Цей інструмент використовувався для проектів, у яких дані в реальному часі

використовуються для аналізу. В MongoDB з'єднання з інструментами досить просте, також обробка даних відбувається дуже швидко.

Переваги СУБД MongoDB:

- Зчитує дані в режимі реального часу дуже точно і швидко;
- Легкий та зрозумілий синтаксис;
- Зберігання будь-яких типів даних шляхом створення типів персональних даних;
- Потужні запити та аналітика;
- База даних без схем, що означає легкість масштабування.

Недоліки СУБД MongoDB:

- MongoDB не має з'єднань, що робить її менш оптимальною;
- Велике споживання пам'яті;
- Функція транзакції недоступна.

Для обліку пасажиропотоку на транспорті найефективніше використовувати реляційну базу даних з таблицею, що фіксує кожну подію (вхід/вихід пасажирів), прив'язану до часу, маршруту та геолокації. Для побудови нашої системи обираємо PostgreSQL, тому що вона має наступні переваги для роботи з транспортними системами [20]:

- Структурованість – ідеально підходить для фіксації маршрутів, зупинок, графіків руху та транзакцій оплати;
- Складна аналітика – завдяки потужним JOIN і віконним функціям в SQL, зручно будувати аналітичні звіти про завантаженість напрямків за годинами;
- Робота з геоданими – має офіційне розширення PostGIS, яке дозволяє обчислювати відстані від транспортного засобу до зупинок, робити просторові запити;
- Надійність – забезпечує безкомпромісну цілісність даних, що критично важливо для фінансових транзакцій при оплаті проїзду.

3.4. Розробка бази даних для системи обліку пасажиропотоку

Оскільки присутній досить великий об'єм інформації, яку потрібно зберігати, було вирішено створити базу даних, та розбити дані по різних таблицях. База даних включає в себе п'ять таблиць:

- Таблиця маршрутів;
- Таблиця транспортних засобів;
- Таблиця зупинок;
- Таблиця розкладу рейсів;
- Таблиця обліку пасажиропотоку.

На рис.3.3 показано ER-діаграму таблиць бази. Код наведено в Додатку Б.

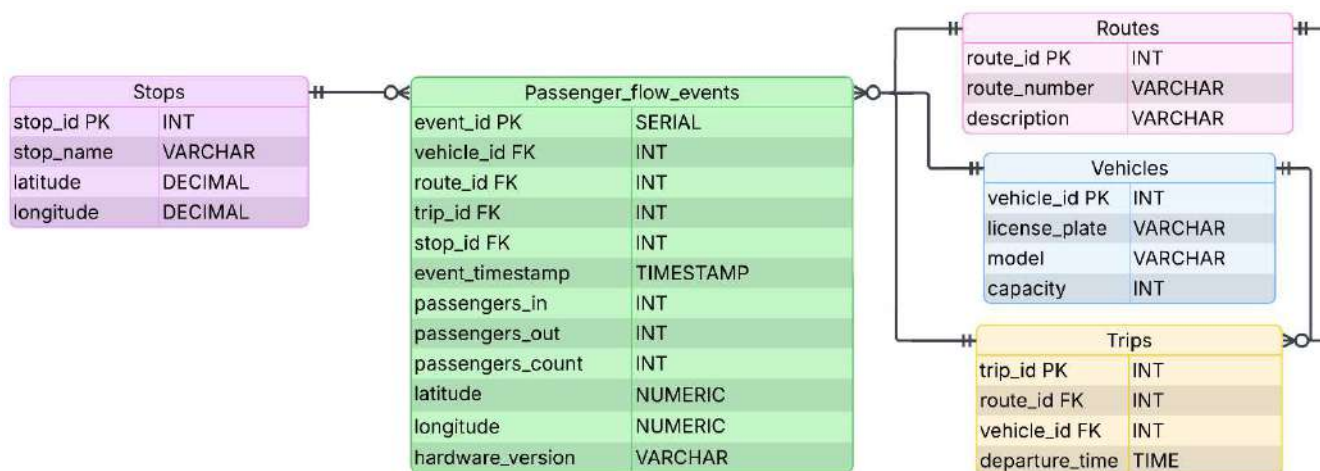


Рис.3.3. ER-діаграма таблиць бази даних

Таблиця «Routes» зберігає дані про маршрути: номер маршруту та його опис. Її структура представлена у таблиці 3.1.

Таблиця 3.1. Структура таблиця «Routes»

Стовпець	Тип даних	Опис	Ключ
route_id	INT	—	Первинний
route_number	VARCHAR(10)	Номер маршруту	
description	VARCHAR(255)	Опис маршруту	

Таблиця «Vehicles» зберігає дані про транспортні засоби: реєстраційний номер, модель та пасажиромісткість. Її структура представлена у таблиці 3.2.

Таблиця 3.2. Структура таблиця «Vehicles»

Стовпець	Тип даних	Опис	Ключ
vehicle_id	INT	—	Первинний
license_plate	VARCHAR(20)	Реєстраційний номер	
model	VARCHAR(50)	Модель	
capacity	INT	Пасажиромісткість	

Таблиця «Stops» зберігає дані про зупинки: назва, широту та довготу. Її структура представлена у таблиці 3.3.

Таблиця 3.3. Структура таблиця «Stops»

Стовпець	Тип даних	Опис	Ключ
stop_id	INT	—	Первинний
stop_name	VARCHAR(100)	Назва	
latitude	DECIMAL(9,6)	Широта	
longitude	DECIMAL(9,6)	Довгота	

Таблиця «Trips» зберігає дані про рейси. Вона пов'язана з таблицями «Routes»: та «Vehicles», а також містить час відправлення. Її структура представлена у таблиці 3.4.

Таблиця 3.4. Структура таблиця «Trips»

Стовпець	Тип даних	Опис	Ключ
trip_id	INT	—	Первинний
route_id	INT	—	Зовнішній
vehicle_id	INT	—	Зовнішній
departure_time	TIME	Час відправлення	

Таблиця «Passenger_flow_events» зберігає дані про події відносно пасажиропотоку. Вона пов'язана зі всіма іншими таблицями, а також містить час фіксації події, кількість пасажирів, які увійшли та вийшли, розрахункову кількість пасажирів у салоні, широту та довготу міста події, версію датчика для моніторингу працездатності. Її структура представлена у таблиці 3.5.

Таблиця 3.5. Структура таблиця «Passenger_flow_events»

Стовпець	Тип даних	Опис	Ключ
event_id	SERIAL	—	Первинний
vehicle_id	INT	—	Зовнішній
route_id	INT	—	Зовнішній
trip_id	INT	—	Зовнішній
stop_id	INT	—	Зовнішній
event_timestamp	TIMESTAMP	Час фіксації події	
passengers_in	INT	Кількість пасажирів, які увійшли	
passengers_out	INT	Кількість пасажирів, які вийшли	
passengers_count	INT	Розрахункова кількість пасажирів у салоні	
latitude	NUMERIC(9, 6)	Широта міста події	
longitude	NUMERIC(9, 6)	Довгота міста події	
hardware_version	VARCHAR(20)	Версія датчика для моніторингу працездатності	

3.5. Розробка програми обробки даних на сервері

Для розробки програми серверної частини обираємо FastAPI, який має наступні переваги:

- Надзвичайна швидкість – це один із найшвидших фреймворків на Python, продуктивність якого порівнюють з NodeJS та Go;
- Вбудована асинхронність – підтримка `async` та `await` дозволяє ефективно обробляти тисячі одночасних запитів;
- Автоматична документація – фреймворк генерує інтерактивну документацію (Swagger UI та ReDoc) прямо на льоту на основі стандартів OpenAPI;
- Автоматична валідація даних – використання бібліотеки Pydantic гарантує, що вхідні дані завжди матимуть правильний тип, зменшуючи кількість помилок;
- Зручність розробника – написання коду стає швидшим завдяки підказкам у сучасних IDE;
- Гнучкість – на відміну від монолітних фреймворків (наприклад, Django), FastAPI не нав'язує жорстку структуру проекту, даючи повну свободу архітектури.

Код серверної частини наведено в Додатку В. На рис. 3.4 наведено UML-діаграму класів, а на рис. 3.5 – діаграму послідовності.

Коли трекер надсилає пакет, сервер обчислює нове поточне значення кількості пасажирів в салоні. Також сервер запобігає ситуаціям, коли через збій лічильників у салоні виявляється негативна кількість пасажирів. Завдяки окремій таблиці `vehicle_occupancy`, для отримання поточної завантаженості автобуса диспетчерській системі чи мобільному додатку пасажирів не потрібно сканувати мільйони рядків історії – достатньо миттєво зчитати один рядок за `vehicle_id`.

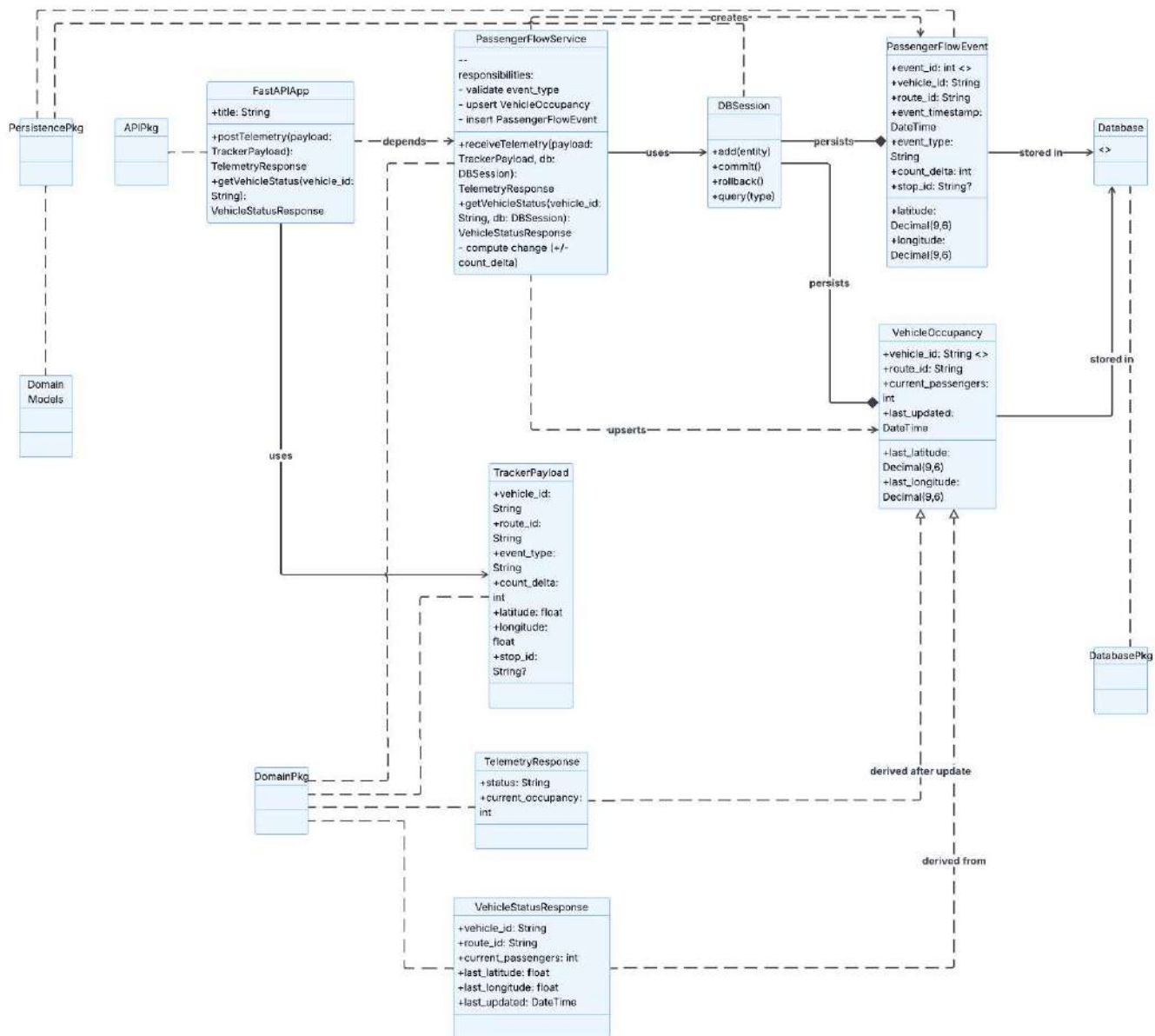


Рис. 3.4. UML-діаграма класів

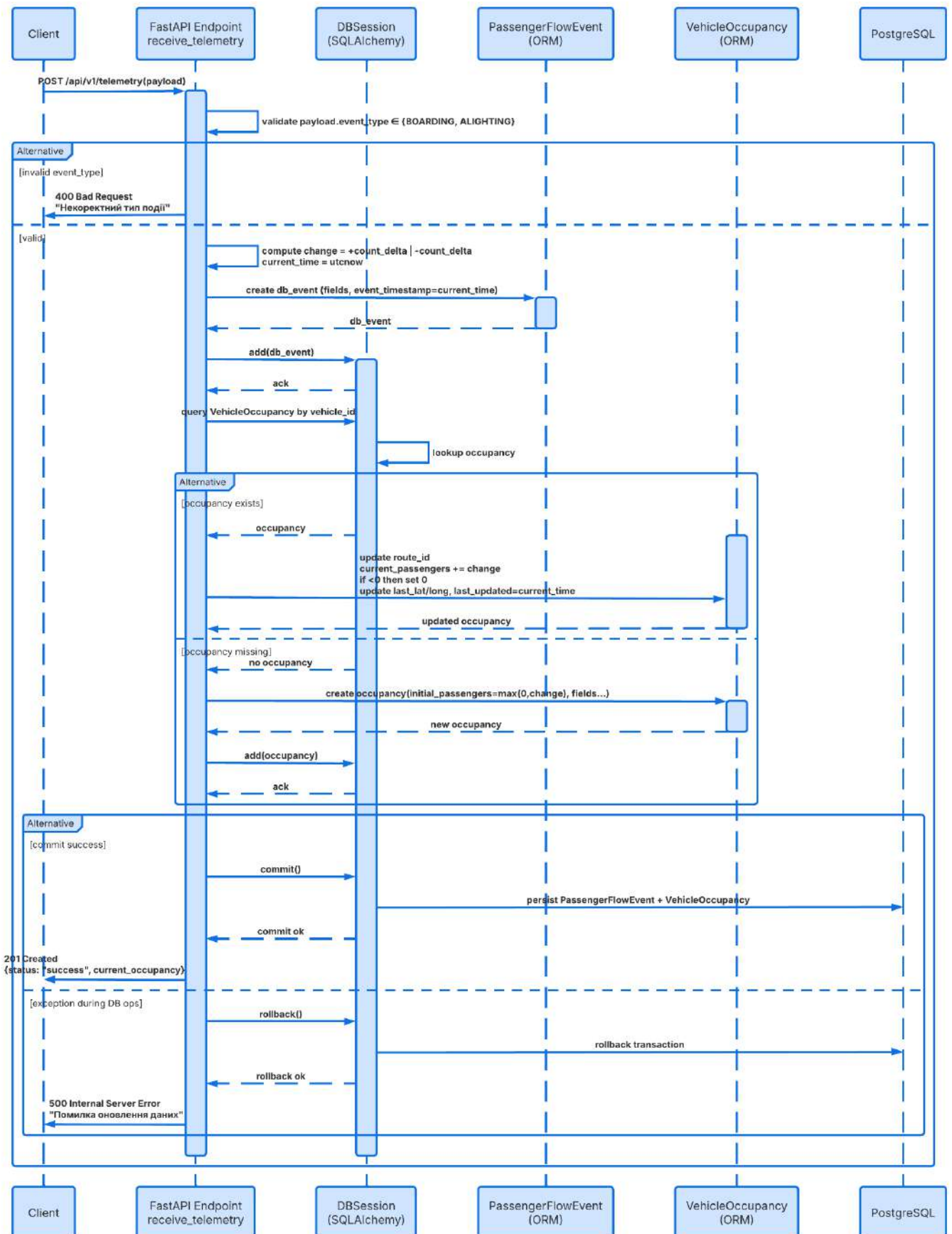


Рис. 3.5. UML-діаграма послідовності

Висновки за розділом 3

Розглянуто загальну структуру системи обліку пасажиропотоку. Розроблено алгоритм та програму для модуля збору даних. Проведено огляд баз даних та обрано базу даних для реалізації системи. Розроблено базу даних для системи обліку пасажиропотоку. Розроблено програму для серверної частини.

РОЗДІЛ 4. ОХОРОНА ПРАЦІ

4.1. Правове регулювання охорони праці в Україні

Поняття охорони праці визначається в ст. 1 Закону України від 14 жовтня 1992 «Про охорону праці». Охорона праці – система забезпечення здоров'я й життя особи під час трудової діяльності. До поняття включаються правові, соціально-економічні, організаційні, технічні, санітарно-гігієнічні, лікувально-профілактичні, реабілітаційні та інші подібні заходи. До трактовки поняття охорони праці також включаються всі інші заходи, що спрямовані для полегшення праці людей вагітних, неповнолітніх або з вадами здоров'я.

Нагляд за додержанням норм праці лежить на плечах двох органів: прокуратури та соціальних спілок. Також контроль за безпекою праці проводиться державних та відомчих спеціалізованих інспекцій.

В соціальній політиці охорона праці та здоров'я є однією з основних напрямків. Так, Конституція України одним з основних соціальних прав громадян визначає право кожного на належні, безпечні й здорові умови праці, встановлює, що використання праці жінок і неповнолітніх на небезпечних для їхнього здоров'я роботах забороняється.

Конституція України, Закони України "Про охорону праці", "Про охорону здоров'я", "Про пожежну безпеку", "Про використання ядерної енергії та радіаційний захист", "Про забезпечення санітарного та епідемічного благополуччя населення", "Про загальнообов'язкове державне соціальне страхування від нещасного випадку на виробництві та професійного захворювання, які спричинили втрату працездатності", кодекс законів про працю України.

4.2. Значення охорони праці у роботі фахівців з інформаційних технологій

Комп'ютери вже давно увійшли в повсякденне життя, а норми роботи з ними вже давно вкоренилися в зарубіжних ІТ-компаніях, проте на території України цей процес все ще триває, хоч і підходить до завершальної стадії. Така складна апаратура, як комп'ютери вже стала частиною повсякденного життя, й зарубіжні компанії з легкістю адаптувались до нових умов роботи з ними. Україна все ще в процесі адаптації до цієї зміни, хоча й близька до фінішу. По цій причині на території будь-якого міста України це питання залишається актуальним, хоча в обласних центрах це питання має значно більш приємний відтінок.

З першого погляду – робота перед комп'ютером безпечна, розробник не підпадає ні під які критерії що загрожують його життю або здоров'я, але це не так, адже фахівці згаданої сфери мають бути нерухомими досить довгий час, далеко не завжди знаходячись в комфортному положенні, також це сфера пов'язана з значними розумовими напруженнями.

Розробники настільки залучені, що навіть відволікаючись від роботи над проектом, продовжують думати про рішення, що їм потрібно реалізувати на роботі. Часто відпочинком для такого типу людей є паралельна зміна діяльності, вивчення мов програмування, читання технічної літератури й інше. Нажаль, мозок не може безкінечно приймати корисну інформацію, що розробник прагне направляти в русло особистісного та професійного зростання. Адже мозок людини не машина: він не може нескінченно зберігати і переробляти дані практично не втрачаючи продуктивності. Людина програє комп'ютеру за багатьма показниками, тому для підтримки працездатності вимагає певних умов.

По цій причині ІТ-компанії обладнують свої офіси різноманітними кімнатами й послугами, що, на здивування, відволікають робітників від роботи, адже це знижує психічне навантаження і сприяє росту продуктивності. Окремим робочим столом з ноутбуком вже давно нікого не здивуєш. Тому, в бажанні підвищити продуктивність працівників, компанії змагаються, перетворюючи нудні одноманітні офіси в

креативні простори, де нові ідеї народжуються без титанічних зусиль. Наприклад, робочі місця співробітників компанії Google в Цюриху нагадують гігантські вулики, а офіс шведського інтернет-провайдера Bahnhof розташувався в бомбосховищі часів холодної війни і походить на підземний притулок землян після глобальної катастрофи. А щоб співробітників не тягнуло додому, роботодавці створюють і можливість відпочинку поряд з робочими місцями, обладнавши басейни, ігрові кімнати та спортзали. Адже можна цілий день просидіти біля монітора, а у вечорі відчувати страшно втому, як від важкої фізичної праці. Однак це почуття помилкове і боротися з ним допоможе спорт. Активний відпочинок відмінно бадьорить, розганяє кров, додає сил. Чимало програмістів відчувають потребу в активному відпочинку на підсвідомому рівні, вибираючи спорт як хобі для проведення вільного часу. При цьому не варто забувати, що умови праці програмістів також характеризуються можливістю впливу на них наступних небезпечних і шкідливих виробничих факторів: шуму; тепловиділень, причому шкоди організму можуть завдати не тільки високі, але і низькі температури; іонізуючих і неіонізуючих випромінювань: рентгенівське, інфрачервоне, високочастотне електромагнітне випромінювання; статичної електрики; недостатнє штучне та природне освітлення; візуальні фактори: яскравість, контрастність, мерехтіння зображення, відблиски тощо. За таких умов зростає роль та значення охорони праці, як системи правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження здоров'я і працездатності людини в процесі праці. Адже в кінцевому рахунку плоди науково-технічного прогресу можуть бути ефективними лише в тій мірі, в якій вони забезпечують людині безпеку, комфортність і зручність трудової діяльності.

Критична ситуація в Україні у сфері безпеки праці проявляється високим рівнем виробничого травматизму і професійної захворюваності, незадовільними умовами праці та санітарним станом підприємств, внаслідок чого держава втрачає кваліфікованих працівників, а натомість отримує десятки тисяч осіб, які потребують повноцінного соціального захисту.

4.3. Інформаційна культура як важлива складова в управлінні охорони праці

Для підвищення ефективності системи управління охорони праці дуже важлива роль належить формуванню і розвитку інформаційної культури фахівців ІТ-технологій, яка впливає на удосконалення інформаційного контуру сучасних підприємств, дозволяє створювати надійні прогнози щодо стану умов праці, показників здоров'я та працездатності, виробничого травматизму і професійної захворюваності, визначати політику розвитку підприємств, установ та організацій на основі різноманітних стратегій охорони праці. Поряд з інформаційною культурою важливо використовувати в рамках системи управління охорони праці «трикутник» її складових: правову, організаційну, управлінську. В управлінні охороною праці потрібно реалізувати основні положення, окремі теоретико-методологічні підходи інформаційного менеджменту. Головну роль та відповідальність за стан системи управління охорони праці мають нести фахівці служби охорони праці сучасного підприємства. Сучасне суспільство називають постіндустріальним, постекономічним, інформаційним, оскільки йдеться про багатосторонні і кардинальні зміни у розвитку цивілізації. Інформаційне суспільство передбачає докорінну зміну, яка полягає у перетворенні інформації і знань у головний професійно-виробничий потенціал особистості, соціуму і держави. На постіндустріальному етапі розвитку суспільства вирішальним фактором стає інформація. Її домінування ініціювала науково-технічна революція, яку ще іменують інформаційною, оскільки нею охоплена будь-яка інтелектуальна діяльність, починаючи з інформаційних образів штучного інтелекту у нових технологіях, економіки, і продовжуючи інформатизацією суспільства в умовах світової глобалізації науки й освіти тощо. Інформаційні технології розглядаються як потужний важіль економічного зростання України. Для цього необхідні значні стратегічні інвестиції у комп'ютерну та комунікаційну інфраструктуру, програми досліджень і розробок, освітню галузь. Під інформаційною культурою розуміють сукупність, складову новітні інформаційні технології, технологічну, правову,

психологічну, соціологічну та ергономічну підсистеми, що сприяють спрямованому впливу на протікання соціальних процесів у суспільстві, колективі і вихованню свідомого відношення людини до праці, виконання прав та обов'язків. Поняття інформаційної культури виникло в процесі активізації дослідницької уваги до механізмів інформаційного обміну у зв'язку зі значним підвищення ролі інформації в соціокультурних процесах суспільства, яке розглядають як інформаційне суспільство знань, де в центрі знаходяться інформаційні технології. Робота з інформацією та інформаційна культура в цілому є одним з найважливіших компонентів спроб компанії управляти змінами. Є три принципові причини, в силу яких сьогодні необхідно дбати про інформаційну культуру компанії. По-перше, вона все більше і більше стає найважливішою частиною загальної організаційної (корпоративної) культури компанії. Все більше компаній розуміють необхідність перетворень, орієнтованих на задоволення очікувань споживача. Щоб сьогодні впливати на майбутнє, потрібно уявляти собі на що вона буде схожа. А для цього потрібно працювати з різноманітною діловою, професійною, технологічною, соціальною, ринковою та політичною інформацією. По-друге, інформаційні технології роблять можливим створення в компаніях комп'ютерних мереж, за допомогою яких йде спілкування між менеджерами, але важливо знати, як люди використовують цю інформацію. Саме по собі створення такої мережі з усіма її робочими станціями і мультимедійними можливостями не гарантує того, що інформація буде використовуватися більш розумно і більш ефективно. По-третє, для різних функціональних служб, підрозділів та робочих груп сучасних підприємств в сфері охорони праці інформаційна культура різна, а це означає відмінність методологічних підходів до процесів усвідомлення, збору, організації, обробки, поширення і використання інформації. Тому багато менеджерів погодяться з тим, що корпоративна інформаційна культура важлива для вироблення різних стратегій охорони праці та запровадження відповідних заходів з її вдосконалення. Для деяких галузей, таких як розробка програмного забезпечення, інформаційна культура є необхідною умовою виживання, тому що зміна технологій в розробці програмного забезпечення відбувається кожні 6-8 місяців, а інвестиції на підготовку персоналу і

освоєння нової технології величезні і у великих компаніях варіюються від 1,5 до 2 млрд. доларів на рік. Аналіз свідчить, що інформатизація та інтеграція комунікаційного простору України сприяє різкому підвищенню інформаційної та професійної компетентності, ділової активності, стимулюванню конкуренції, створенню інноваційних підприємств та організацій, нових робочих місць, зниженню витрат на утримання управлінського апарату.

Слід розуміти, що комп'ютерні технології, а особливо їх мережі істотно впливають на життєдіяльність людини, припускаючи глобалізацію і технократизацію суспільства. Але в ще більшій мірі цей вплив поширюється безпосередньо на центральну нервову систему, яка звикає працювати в дуже інтенсивному режимі багатозадачності, де вже переважають не тривалі логічні роздуми, а інтуїтивно-реактивні ланцюжки розумових формулювань у зв'язку з величезним обсягом оброблюваної щодня інформації, кількість якої зростає за експоненціальною швидкістю. Виникає припущення, що саме збільшення обсягу інформації та прискорення її обробки людиною може згубно вплинути на розвиток розумових здібностей людини. Аналіз продуктивності розумової праці в найбільших за чисельністю фахівців ІТ-фірм показав, що велике значення з точки зору впливу на її результати має організаційна (корпоративна) культура. В цьому напрямі влаштовуються різні тимблдинги, заходи, тренінги для розвитку персоналу. Також кожен керівник повинен добре розуміти свого співробітника, що саме для нього важливо, що його мотивує. Важливо відвести потрібну роль відповідному співробітнику, щоб він виконував ті завдання, які йому цікаві. На подібних тренінгах в тому числі повинна розглядатися інформаційна культура працівника, в освоєнні, володінні, мотивуванні, застосуванні, перетворенні інформації із застосуванням сучасних інформаційних технологій і використанням цих умінь в навчанні з охорони праці і в подальшій професійній діяльності. Особливо вони будуть корисні, як доповнення до існуючих інструктажів з охорони праці на підприємстві, або як контроль психологічного стану та взаємовідносин у колективі. Інформаційна культура як інтегративне утворення абсолютно не зводиться до розрізнених знань, вмінь та навичок роботи за комп'ютером. Вона передбачає

інформативну спрямованість цілісної особистості, яка володіє мотивацією до застосування і засвоєння нових даних. Інформаційну культуру можна розглядати, як одну з граней особистісного розвитку промислових робітників. Це шлях універсалізації якостей людини. Оволодіння інформаційною культурою сприяє реальному розумінню особистістю свого місця, себе і своєї ролі у виробничому колективі. Вона має сприяти формуванню нового покоління фахівців інформаційного суспільства, який повинен володіти наступними навичками: виділення релевантної, значущої інформації, диференціації вихідних даних, розробки інформативних критеріїв її оцінки інформації.

4.4. Вдосконалення охорони праці фахівців з інформаційних технологій

Поява та впровадження нових інформаційно-комунікаційних технологій зумовлює необхідність подальшого вдосконалення охорони праці фахівців ІТ-індустрії. З метою належного правового забезпечення необхідно розширити ти доповнити перелік основних професій комп'ютерної галузі у національному класифікаторі ДК-003-2010, а також підготувати відповідний випуск у кваліфікаційному довіднику посад фахівців ІТ-індустрії, що сприятиме вирішенню питань їх соціального захисту, пенсійного забезпечення, атестації робочих місць основних професій за умовами праці на предмет подальших певних видів пільг та компенсацій за важкі шкідливі і небезпечні умови праці. Важливим напрямом стосовно визначення професійної придатності фахівців з інформаційних технологій є проведення психофізіологічної експертизи відповідно до 5 статті Закону України «Про охорону праці». Робота з комп'ютерами нового покоління характеризується певним психофізіологічними перенавантаженнями, втомую зорового аналізатора, гіпокінезією, відсутність диференційованих норм праці при роботі з новою комп'ютерною технікою в залежності від віку, статі, категорії зорової роботи, режимів праці і відпочинку (протягом робочого дня, тижня, щорічного режиму відпусток). Все це потребує розробки нових нормативно-правових актів з регламентації праці та відпочинку фахівців ІТ-індустрії і стандартів підприємств,

центрів комп'ютерної техніки, центрів інформаційних технологій, сучасних комп'ютерних класів. Особлива роль з точки зору збереження та відновлення здоров'я працюючих в комп'ютерній галузі належить попереднім та періодичним наглядам з подальшої психофізіологічної експертизи і встановленням професійної придатності при роботі з комп'ютерами нового покоління, який супроводжується виникненням певних факторів професійного ризику електротравматизму при їх ремонті та обслуговуванні. За результатами експертизи повинні прийматися рішення про відповідність комп'ютерів нормам безпеки, терміни чергової експертизи, оформлюються протоколи вимірювань і випробувань, проведені у разі потреби розрахунки та експертний висновок. Для підвищення розумової працездатності то зорової роботи повинна здійснюватися ергономічна оптимізація в рамках системи «оператор-термінал», яка сприятиме результативній фізичній та інтелектуальній працездатності і відновленню психосоматичного здоров'я фахівців ІТ-індустрії. Заслуговує на увагу зарубіжний досвід створення у приміщеннях та в зоні їх розміщення на територіях підприємств спеціальних візуальних комфортних умов та забезпечення вимог виробничої естетики, дотримання норм рівнів виробничого шуму та акустичної тиші за межами офісу. Також дуже важливим є використання в офісних приміщеннях та кабінетах психофізіологічного розвантаження функціональної музики, яка сприяє попередженню перевтоми і підтриманню необхідного рівня розумової працездатності фахівців комп'ютерної галузі. В цьому напрямі заслуговує на увагу створення при великих центрах інформаційних технологій кімнат (кабінетів) психофізіологічного розвантаження працівників галузі (на 5 місць). Такий підхід дозволяє зняти надлишкове психофізіологічне перевантаження, підвищити працездатність центральної нервової системи, попередити перевтому зорового аналізатора. Всі наведені заходи щодо вдосконалення охорони праці фахівців ІТ індустрії повинні контролюватися службою охорони праці та комісією з охорони праці підприємства. Особливе значення у соціальному захисті цієї категорії працівників належить прийняття комплексного договору, який може забезпечити фахівців додатковими пільгами та компенсаціями

4.5. Розрахунок освітленості робочих місць

Для забезпечення безпечних умов праці необхідно розрахувати освітлення робочого приміщення. Згідно з нормами ДБН В.2.5-28:2018, мінімальна освітленість робочої зони повинна становити 300 лк.

Розрахунок ведеться за формулою:

$$E = \frac{F \cdot n \cdot \eta}{S \cdot k} \quad (4.1)$$

де E - освітленість, лк;

F - світловий потік однієї лампи, лм;

n - кількість ламп;

η - коефіцієнт використання світлового потоку;

S - площа приміщення, м²;

k - коефіцієнт запасу (1,4–1,6).

Розрахуємо:

Площа приміщення – 55 м², використовується 8 люмінесцентних ламп по 3200 лм кожна, коефіцієнт використання – 0,5, коефіцієнт запасу – 1,5.

$$E = \frac{3200 \cdot 8 \cdot 0,5}{55 \cdot 1,5} = 155,2 \text{ лк.}$$

Отже, необхідно збільшити кількість ламп в 2 рази – до 16, щоб досягти мінімальної освітленості на рівні 300 лк.

Охорона праці на автомобільному пасажирському транспорті є багатофакторною системою, яка включає технічні, організаційні, санітарно-гігієнічні та соціальні заходи. Ефективне функціонування цієї системи забезпечує не лише безпеку працівників, а й підвищення якості транспортних послуг.

Підвищення рівня безпеки праці досягається завдяки комплексному підходу - модернізації транспортних засобів, автоматизації контролю, удосконаленню системи управління охороною праці, впровадженню психологічної підтримки водіїв, дотриманню режимів праці та відпочинку.

Впровадження сучасних інформаційних технологій у систему управління охороною праці дозволяє оперативно виявляти порушення, прогнозувати ризики та

формувати безпечну виробничу культуру, що є необхідною умовою стабільного функціонування автомобільного пасажирського транспорту.

Висновки за розділом 4

Розглянуто різноманітні позиції по відповідності норм охорони праці в ІТ сфері. Розглянуто ризики, що можуть мати ІТ спеціалісти в відношенні свого здоров'я. Проведено огляд впливу ІТ сфери на загальну картину охорони праці в світі. Розглянуто різноманітні рішення, що сприяють продуктивності роботи ІТ спеціалістів.

ВИСНОВКИ

Проаналізовано основні компоненти ІТС та взаємодію між ним. Розглянуто організаційну структуру ІСУГТ. Сформовано концепцію інформаційної системи обліку пасажиропотоку.

Розглянуто використання датчиків та їх функції в системі розумного громадського транспорту. Обрано технологію зв'язку в системі розумного громадського транспорту. Обрано елементи системи розумного громадського транспорту.

Розглянуто загальну структуру системи обліку пасажиропотоку. Розроблено алгоритм та програму для модуля збору даних. Проведено огляд баз даних та обрано базу даних для реалізації системи. Розроблено базу даних для системи обліку пасажиропотоку. Розроблено програму для серверної частини.

Також розглянуто питання охорони праці.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Katanalp, B.Y.; Yildirim, Z.B.; Eren, E.; Uz, V.E. An Evaluation on Intelligent Transportation Systems. *ISAS 2018 : Proceedings of the Second International Symposium on Innovative Approaches in Scientific Studies*, Samsun, Turkey, 30 November – 2 December 2018. SETSCI, 2018. P.743-749.
2. Javed, M.A.; Ben Hamida, E.; Al-Fuqaha, A.; Bhargava, B. Adaptive Security for Intelligent Transport System Applications. *IEEE Intelligent Transportation Systems Magazine*. 2018. No. 10. P. 110-120.
3. Hamida, E.B.; Noura, H.; Znaidi, W. Security of Cooperative Intelligent Transport Systems: Standards, Threats Analysis and Cryptographic Countermeasures. *Electronics*. 2015. No. 4, P.380-423.
4. Cyber Security for Consumer Internet of Things. *ETSI*: official web-site. URL: https://www.etsi.org/deliver/etsi_en/303600_303699/303645
5. Сталий транспорт: збірка матеріалів для відповідальних за вироблення політики у містах, що розвиваються URL: https://transformative-mobility.org/wp-content/uploads/2024/01/3f_Transit_Alliances_SUTP_TUMI_GIZ_UKRAINIAN.pdf
6. Система підрахунку пасажирів GPSM Pass-track – URL: <https://gpsm.ua/ua/sistema-podschjota-passazhirov-gpsm-pass-track/>
7. Система автоматичного підрахунку пасажирів GPSM Clever Scan – URL: <https://gpsm.ua/ua/sistema-avtomaticheskogo-podscheta-passazhirov-gpsm-clever-scan/>
8. Automatic Passenger Counting Solutions - INFODEV EDI inc – URL: <https://www.infodev.ca/products/gateway-series/>
9. Датчик підрахунку пасажиропотоку IRMA Matrix – URL: <https://transcontrol.com.ua/oborudovanie-gps/gps-glonass-sat-aksessuary/prochiedadchiki-identifikatory-i-aksessuary/irma-matrix.html>

10. Automatic passenger counting systems for public transport – URL: <https://www.intelligenttransport.com/transport-articles/3116/automatic-passengercounting-systems-for-public-transport/>
11. An, S.; Lee, B.-H.; Shin, D.-R. A Survey of Intelligent Transportation Systems. *Communication Systems and Networks* : Proceedings of the 2011 Third International Conference on Computational Intelligence, Bali, Indonesia, 26-28 July 2011. IEEE Computer Society, 2011. P.332-337.
12. Cheek, E.; Alghodhaifi, H.; Adam, C.; Andres, R.; Lakshmanan, S. Dedicated short range communications used as fail-safe in autonomous navigation. *Unmanned System Technology*. 2020. No. 11. P. 159-177.
13. Gu, F.; Niu, J.; Jiang, L.; Liu, X.; Atiquzzaman, M. Survey of the low power wide area network technologies. *Network Computer App*. 2020. No. 14. P.102-107.
14. Cooperative Intelligent Transport Systems and Services. *CAR 2 CAR Communication Consortium*: official web-site. URL: <https://www.car-2-car.org/about-c-its>
15. Zhang, H.; Lu, X. Vehicle communication network in intelligent transportation system based on Internet of Things. *Computer Communication*. 2020. No. 160. P.799-806.
16. Liébana-Cabanillas, F.; Molinillo, S.; Ruiz-Montañez, M. To use or not to use, that is the question: Analysis of the determining factors for using NFC mobile payment systems in public transportation. *Technological Forecasting and Social Change*. 2019. No. 139. P.266-276.
17. AXIS P8815-2 3D People Counter – Product support URL: <https://www.axis.com/products/axis-p8815-2-3d-people-counter/support>
18. Data sheet NEO-6 u-blox 6 GPS Modules. Version– URL: https://www.u-blox.com/sites/default/files/products/documents/NEO6_DataSheet.pdf
19. Модуль GPS NEO-6M Mini SMA – URL: <https://arduino.ua/prod4091-modul-gps-neo-6m-mini-sma>
20. PostgreSQL: The World's Most Advanced Open Source Relational Database – URL: <https://www.postgresql.org/>

Додаток А. Програма модуля збору даних

```
import time
import json
import requests
import serial
from gpiozero import Button
# НАЛАШТУВАННЯ
VEHICLE_ID = "BUS-1234" # ID транспортного засобу
CAMERA_URL = "http://192.168.1"
SERIAL_PORT = "/dev/serial0" # Стандартний порт UART на Raspberry Pi
BAUD_RATE = 9600 # Стандартна швидкість для GY-NEO6MV2
# Ініціалізація датчиків із захистом від брязкіту контактів
sensor_1 = Button(17, pull_up=True, bounce_time=0.05)
sensor_2 = Button(27, pull_up=True, bounce_time=0.05)
sensor_3 = Button(22, pull_up=True, bounce_time=0.05)
# Глобальний лічильник
CntCount = 0
request_sent = False

def parse_nmea_degrees(raw_value, direction):
    """Конвертує координати з формату NMEA (DDMM.MMMM) у десяткові
    градуси (DD.DDDD)"""
    if not raw_value:
        return None
    try:
        # Для широти перші 2 знаки - градуси, для довготи - 3.
        dot_idx = raw_value.find('.')
        deg_len = dot_idx - 2
        degrees = float(raw_value[:deg_len])
        minutes = float(raw_value[deg_len:])
```

```

    decimal = degrees + (minutes / 60.0)
    if direction in ['S', 'W']:
        decimal = -decimal
    return round(decimal, 6)
except ValueError:
    return None

def get_gps_data():
    """Зчитує порт UART та намагається отримати поточні координати і час з
    GPS-модуля"""
    gps_data = {"latitude": None, "longitude": None, "gps_timestamp": None}

    try:
        # Відкриваємо порт із таймаутом, щоб скрипт не завис, якщо супутників
        немає
        with serial.Serial(SERIAL_PORT, baudrate=BAUD_RATE, timeout=1) as
        ser:
            # Читаємо порцію рядків (модуль шле дані беззупинно)
            for _ in range(30):
                line = ser.readline().decode('ascii', errors='replace').strip()

                # Шукаємо рядок $GPRMC (містить час, статус, координати)
                if line.startswith('$GPRMC'):
                    parts = line.split(',')
                    if len(parts) > 9 and parts[2] == 'A': # 'A' означає, що дані
активні/валідні
                        raw_time = parts[1] # hhmmss.ss
                        raw_lat = parts[3]
                        lat_dir = parts[4]
                        raw_lon = parts[5]

```

```

lon_dir = parts[6]
raw_date = parts[9] # ddmmyy

gps_data["latitude"] = parse_nmea_degrees(raw_lat, lat_dir)
gps_data["longitude"] = parse_nmea_degrees(raw_lon, lon_dir)

if raw_time and raw_date:
    gps_data["gps_timestamp"] = f"20{raw_date[4:6]}-{raw_date[2:4]}-{raw_date[0:2]}{raw_time[0:2]}:{raw_time[2:4]}:{raw_time[4:6]} UTC"
    return gps_data
except Exception as e:
    print(f"Помилка читання GPS модуля: {e}")

return gps_data # Поверне None значення, якщо супутники не знайдені або
модуль не підключено

print(f"Система запущена. Початковий CntCount = {CntCount}. Очікування
герконів...")

try:
    while True:
        all_closed = sensor_1.is_pressed and sensor_2.is_pressed and
sensor_3.is_pressed

        if all_closed:
            if not request_sent:
                print("Всі 3 геркони замкнені! Запит до 3D-камери...")

            try:

```

```

response = requests.get(CAMERA_URL, timeout=5)
response.raise_for_status()
camera_json = response.json()

# Витягуємо дані з камери
cnt_in = camera_json.get("in", 0)
cnt_out = camera_json.get("out", 0)

# Розрахунок лічильника пасажирів
CntCount = CntCount + cnt_in - cnt_out
if CntCount < 0:
    CntCount = 0

print("JSON від камери оброблено успішно. Опитування GPS...")
# Опитуємо GPS модуль
gps_info = get_gps_data()

# Формуємо новий підсумковий словник (JSON-структуру)
output_data = {
    "vehicle_id": VEHICLE_ID,
    "gps": {
        "latitude": gps_info["latitude"],
        "longitude": gps_info["longitude"]
    },
    "cnt_in": cnt_in,
    "cnt_out": cnt_out,
    "cnt_count": CntCount,
    "timestamp": gps_info["gps_timestamp"] if
gps_info["gps_timestamp"] else time.strftime("%Y-%m-%d %H:%M:%S")
}

```

```

# Конвертуємо словник у JSON-рядок (або файл)
output_json_string = json.dumps(output_data, indent=4,
ensure_ascii=False)

print("\nСФОРМОВАНО ПІДСУМКОВИЙ JSON:")
print(output_json_string)

# Приклад збереження у локальний файл на диск Raspberry Pi:
filename = f"telemetry_{int(time.time())}.json"
with open(filename, "w", encoding="utf-8") as f:
    f.write(output_json_string)
print(f"Дані збережено у файл {filename}\n" + "-"*40)

request_sent = True

except requests.exceptions.RequestException as e:
    print(f"Помилка зв'язку з камерою: {e}")
    time.sleep(2)
except ValueError:
    print("Помилка: відповідь від камери не є валідним JSON")
    request_sent = True
else:
    if request_sent:
        print("Геркони розімкнено. Скидання прапорця.")
        request_sent = False

time.sleep(0.01)

except KeyboardInterrupt:
    print("\nПрограму зупинено.")

```

Додаток Б. База даних

-- Таблиця маршрутів

```
CREATE TABLE Routes (  
    route_id INT PRIMARY KEY AUTO_INCREMENT,  
    route_number VARCHAR(10) NOT NULL,  
    description VARCHAR(255)  
);
```

-- Таблиця транспортних засобів (автобуси, тролейбуси тощо)

```
CREATE TABLE Vehicles (  
    vehicle_id INT PRIMARY KEY AUTO_INCREMENT,  
    license_plate VARCHAR(20) NOT NULL,  
    model VARCHAR(50) NOT NULL,  
    capacity INT NOT NULL  
);
```

-- Таблиця зупинок

```
CREATE TABLE Stops (  
    stop_id INT PRIMARY KEY AUTO_INCREMENT,  
    stop_name VARCHAR(100) NOT NULL,  
    latitude DECIMAL(9,6) NOT NULL,  
    longitude DECIMAL(9,6) NOT NULL);
```

-- Таблиця розкладу рейсів (пов'язує транспорт, маршрут та час)

```
CREATE TABLE Trips (  
    trip_id INT PRIMARY KEY AUTO_INCREMENT,  
    route_id INT NOT NULL,  
    vehicle_id INT NOT NULL,  
    departure_time TIME,  
    FOREIGN KEY (route_id) REFERENCES Routes(route_id),
```



```
FOREIGN KEY (vehicle_id) REFERENCES Vehicles(vehicle_id)
);
```

-- Таблиця обліку пасажиропотоку

-- Фіксує скільки пасажирів увійшло/вийшло на конкретній зупинці під час конкретного рейсу

```
CREATE TABLE Passenger_flow_events (
    event_id SERIAL PRIMARY KEY,
    vehicle_id INT NOT NULL,          -- Ідентифікатор транспортного засобу
    (автобус, трамвай)
    route_id INT NOT NULL,           -- Номер або ідентифікатор маршруту
    trip_id INT NOT NULL,
    stop_id INT NOT NULL,            -- Ідентифікатор зупинки (якщо є
    прив'язка до інфраструктури)
    event_timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP, -- Час
    фіксації події
    passengers_in INT DEFAULT 0,
    passengers_out INT DEFAULT 0,
    passengers_count INT DEFAULT 0, -- Розрахункова кількість пасажирів у
    салоні після зупинки
    latitude NUMERIC(9, 6) NOT NULL, -- Широта розташування
    longitude NUMERIC(9, 6) NOT NULL, -- Довгота розташування
    hardware_version VARCHAR(20)      -- Версія датчика для моніторингу
    працездатності
    FOREIGN KEY (vehicle_id) REFERENCES Vehicles(vehicle_id),
    FOREIGN KEY (route_id) REFERENCES Routes(route_id)
    FOREIGN KEY (trip_id) REFERENCES Trips(trip_id),
    FOREIGN KEY (stop_id) REFERENCES Stops(stop_id)
);
```

```
-- Індекси для швидкого пошуку та аналітики  
CREATE INDEX idx_vehicle_time ON passenger_flow_events(vehicle_id,  
event_timestamp);  
CREATE INDEX idx_route_time ON passenger_flow_events(route_id,  
event_timestamp);  
CREATE INDEX idx_geolocation ON passenger_flow_events(latitude, longitude);
```

Додаток В. Програма обробки даних на сервері

```
from fastapi import FastAPI, Depends, HTTPException, status
from pydantic import BaseModel, Field
from datetime import datetime
from typing import Optional
from sqlalchemy import create_engine, Column, Integer, String, DateTime,
Numeric
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker, Session

DATABASE_URL = "postgresql://user:password@localhost/dbname"

engine = create_engine(DATABASE_URL)
SessionLocal = sessionmaker(autocommit=False, autoflush=False, bind=engine)
Base = declarative_base()

# 1. ТАБЛИЦЯ 1: Лог усіх подій
class PassengerFlowEvent(Base):
    __tablename__ = "passenger_flow_events"

    event_id = Column(Integer, primary_key=True, index=True)
    vehicle_id = Column(String(50), nullable=False)
    route_id = Column(String(50), nullable=False)
    event_timestamp = Column(DateTime, default=datetime.utcnow)
    event_type = Column(String(10), nullable=False)
    count_delta = Column(Integer, nullable=False)
    latitude = Column(Numeric(9, 6), nullable=False)
    longitude = Column(Numeric(9, 6), nullable=False)
    stop_id = Column(String(50), nullable=True)
```

2. ТАБЛИЦЯ 2: Поточний стан у реальному часі

```
class VehicleOccupancy(Base):
```

```
    __tablename__ = "vehicle_occupancy"
```

```
    vehicle_id = Column(String(50), primary_key=True, index=True)
```

```
    route_id = Column(String(50), nullable=False)
```

```
    current_passengers = Column(Integer, default=0, nullable=False)
```

```
    last_updated = Column(DateTime, default=datetime.utcnow,
onupdate=datetime.utcnow)
```

```
    last_latitude = Column(Numeric(9, 6), nullable=False)
```

```
    last_longitude = Column(Numeric(9, 6), nullable=False)
```

```
Base.metadata.create_all(bind=engine)
```

```
app = FastAPI(title="Passenger Flow Real-Time API")
```

```
def get_db():
```

```
    db = SessionLocal()
```

```
    try:
```

```
        yield db
```

```
    finally:
```

```
        db.close()
```

```
class TrackerPayload(BaseModel):
```

```
    vehicle_id: str = Field(..., example="BUS-42")
```

```
    route_id: str = Field(..., example="ROUTE-10A")
```

```
    event_type: str = Field(..., example="BOARDING")
```

```
    count_delta: int = Field(..., example=1)
```

```
    latitude: float = Field(..., ge=-90, le=90, example=50.4501)
```

```
    longitude: float = Field(..., ge=-180, le=180, example=30.5234)
```

```
stop_id: Optional[str] = Field(None, example="STOP-77")
```

```
# ЕНДПОІНТ 1: Прийом даних та оновлення балансу пасажирів
```

```
@app.post("/api/v1/telemetry", status_code=status.HTTP_201_CREATED)
```

```
def receive_telemetry(payload: TrackerPayload, db: Session = Depends(get_db)):
```

```
    if payload.event_type not in ["BOARDING", "ALIGHTING"]:
```

```
        raise HTTPException(status_code=400, detail="Некоректний тип події.")
```

```
    # Визначаємо математичний знак зміни кількості пасажирів
```

```
    # Вхід (BOARDING) додає людей, вихід (ALIGHTING) — віднімає
```

```
    change = payload.count_delta if payload.event_type == "BOARDING" else -
```

```
    payload.count_delta
```

```
    current_time = datetime.utcnow()
```

```
# 1. Записуємо подію в історичний лог
```

```
db_event = PassengerFlowEvent(
```

```
    vehicle_id=payload.vehicle_id,
```

```
    route_id=payload.route_id,
```

```
    event_type=payload.event_type,
```

```
    count_delta=payload.count_delta,
```

```
    latitude=payload.latitude,
```

```
    longitude=payload.longitude,
```

```
    stop_id=payload.stop_id,
```

```
    event_timestamp=current_time
```

```
)
```

```
db.add(db_event)
```

```
try:
```

```
    # 2. Оновлення балансу в реальному часі
```

```
occupancy =
db.query(VehicleOccupancy).filter(VehicleOccupancy.vehicle_id ==
payload.vehicle_id).first()
```

```
if occupancy:
```

```
    # Оновлюємо існуючий транспорт
    occupancy.route_id = payload.route_id
    occupancy.current_passengers += change
    # Захист від від'ємних значень (на випадок збоїв датчиків)
    if occupancy.current_passengers < 0:
        occupancy.current_passengers = 0
    occupancy.last_latitude = payload.latitude
    occupancy.last_longitude = payload.longitude
    occupancy.last_updated = current_time
```

```
else:
```

```
    # Створюємо новий запис для транспорту, якого ще не було в базі
    initial_passengers = max(0, change)
    occupancy = VehicleOccupancy(
        vehicle_id=payload.vehicle_id,
        route_id=payload.route_id,
        current_passengers=initial_passengers,
        last_latitude=payload.latitude,
        last_longitude=payload.longitude,
        last_updated=current_time
    )
    db.add(occupancy)
```

```
db.commit()
```

```
return {
    "status": "success",
```

```

        "current_occupancy": occupancy.current_passengers
    }

except Exception as e:
    db.rollback()
    raise HTTPException(status_code=500, detail=f"Помилка оновлення даних:
{str(e)}")

# ЕНДПОІНТ 2: Отримання поточного стану конкретного транспорту
@app.get("/api/v1/vehicle/{vehicle_id}/status")
def get_vehicle_status(vehicle_id: str, db: Session = Depends(get_db)):
    occupancy = db.query(VehicleOccupancy).filter(VehicleOccupancy.vehicle_id
== vehicle_id).first()
    if not occupancy:
        raise HTTPException(status_code=404, detail="Транспортний засіб не
знайдено")

    return {
        "vehicle_id": occupancy.vehicle_id,
        "route_id": occupancy.route_id,
        "current_passengers": occupancy.current_passengers,
        "last_coordinates": {
            "latitude": float(occupancy.last_latitude),
            "longitude": float(occupancy.last_longitude)
        },
        "last_updated": occupancy.last_updated
    }

```