КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра інформаційних систем та технологій

Спеціальність 126 – Інформаційних систем та технологій,

програма “Програмні технології інтернет речей”

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему:

“Інформаційно-аналітична IoT система освітлення в Smart City”

Студента 2-го курсу групи ІРм-21

Ярослава ГОЗАКА

(прізвище, ім’я, по батькові)

Науковий керівник:

д.е.н., професор

(науковий ступінь, вчене звання)

Андрій ОНИЩЕНКО

(прізвище, ім’я, по батькові)

_____ _____ _____

(підпис студента) (дата) (підпис)

Попередній захист:

_____

(Висновок: “До захисту в Державній екзаменаційній комісії”)

В.о завідувача кафедри
Інформаційних систем та технологій _____ Олексій КОЛЕСНІКОВ _____

(підпис) (прізвище, ініціали) (дата)

Київ 2021

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра Інформаційних систем та технологіййй
Освітньо-кваліфікаційний рівень Магістр
Спеціальність 126 – Інформаційних систем та технологій
Програма Програмних технологій інтернет речей

ЗАТВЕРДЖУЮ
В.о завідувача кафедри
д.т.н., доцент Олексій КОЛЕСНІКОВ.

_____
“____” _____20___року

ЗАВДАННЯ

НА ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Студент	Ярослав ГОЗАК
Група	ІРм-21

1. Тема кваліфікаційної роботи: “Інформаційно-аналітична IoT система в Smart City”
Затверджена наказом по від “___”_____20__ р. № ___.
2. Строк подання студентом готової роботи - “20” травня 2021 р.
3. Цільова установка та вихідні дані до роботи: дослідження в області рішень для розумного освітлення. Програмно-апаратні рішення для оптимізації роботи розумних світильників. Дані стану навколишнього середовища з давачів, методи їх контролю та передачі даних з використанням систем IoT.
4. Зміст роботи
РОЗДІЛ 1. КОНЦЕПЦІЯ РОЗУМНОГО ОСВІТЛЕННЯ (аналіз аналогів, метод автоматичного управління режимами роботи розумних світильників);
РОЗДІЛ 2. РОЗРОБКА МЕТОДІВ ОБРОБКИ ДАНИХ ДЛЯ РОЗУМНИХ СВІТИЛЬНИКІВ (Розробка проекту IoT рішення, методи та засоби обробки даних, використання хмарних технологій для реалізації IoT рішень);
РОЗДІЛ 3. ПРОЕКТУВАННЯ АРХІТЕКТУРИ ІОТ СИСТЕМИ РОЗУМНОГО ОСВІТЛЕННЯ (архітектура проекту IoT рішення, комунікаційні технології та системи, аналіз захищеності інформації та методів і способів (протоколів) для її кіберзахисту);
РОЗДІЛ 4. РЕАЛІЗАЦІЯ І ВПРОВАДЖЕННЯ ІНФОРМАЦІЙНО-АНАЛІТИЧНОЇ СИСТЕМИ НА ОСНОВІ ASP.NET WEB API ТА МІКРОСЕРВІСІВ (розробка програмного забезпечення прогнозування

інтенсивності руху та графічного інтерфейсу, аналіз результатів досліджень)
5. Перелік графічного матеріалу (слайдів): Концептуальна модель архітектури інформаційно-аналітичної IoT системи розумного освітлення; алгоритм роботи системи та структурна схема; графічний інтерфейс системи для ОС Windows; зображення прототипу системи.
6. Календарний план виконання роботи

№ п/п	**Назва частин роботи**	**Виконання роботи за планом**
1	Вибір тематики кваліфікаційної роботи магістра	20.10.2020
2	Вивчення літературних джерел з предмету дослідження	07.11.2020
3	Збір і вивчення матеріалів досліджуваного підприємства	02.12.2020
4	Складання розгорнутого плану кваліфікаційної роботи	26.12.2020
6	Підготовка розділу 1	25.01.2021
7	Підготовка розділу 2	20.02.2021
8	Підготовка розділу 3	07.03.2021
9	Підготовка розділу 4	20.03.2021
10	Оформлення кваліфікаційної роботи	02.04.2021
11	Передача кваліфікаційної роботи рецензенту для рецензування	11.04.2021
13	Попередній захист кваліфікаційної роботи	20.04.2021
14	Захист роботи	26.05.2021

Дата видачі завдання “10” листопада 2020 р

Керівник роботи д.е.н., проф. Андрій ОНИЩЕНКО _____

(посада, прізвище, ім’я, по батькові) (підпис)

Завдання прийняв до виконання студент групи ІРм-21

Ярослав ГОЗАК _____

(прізвище, ім’я, по батькові) (підпис)

АНОТАЦІЯ

Гозак Я.Д. ІНФОРМАЦІЙНО-АНАЛІТИЧНА ІОТ СИСТЕМА ОСВІТЛЕННЯ В SMART CITY

Кваліфікаційна робота магістра на здобуття наукового ступеня магістр за спеціальністю 126 – інформаційні системи та технології. – Київський національний університет імені Тараса Шевченка, Київ, 2021.

Тема роботи: “Інформаційно-аналітична IoT система освітлення в Smart City”

Актуальність роботи: Кваліфікаційна робота магістра присвячена вирішенню важливого наукового завдання з розробки методу регулювання режиму роботи розумних світильників вуличних ліхтарів для підвищення економічної ефективності шляхом досягнення оптимального співвідношення енергоефективності та строку служби світильників а також позитивного впливу на соціо-психологічний стан людей з оточення, проектування і програмна реалізація на його основі високоефективної IoT системи розумного освітлення.

Метою кваліфікаційної роботи магістра є подальший розвиток систем розумного освітлення, що включає модернізацію алгоритмів роботи світильників з метою зменшення дратівливого фактору впливу на оточуючих в окремих випадках.

Об’єктом дослідження є система розумного освітлення, що включає в себе розумні світильники та застосунок керування освітленням.

Предметом дослідження є методи оптимізації алгоритмів роботи світильників, що націлені на зменшення дратівливих факторів у вигляді частої зміни інтенсивності освітлення в окремих випадках.

Методи дослідження – використано теоретичні та емпіричні методи дослідження: аналіз та узагальнення наукової літератури, проектування програмного забезпечення та графічного інтерфейсу користувача в середовищі Visual Studio, верифікація даних за структурою SWOT аналізу.

Практичним результатом кваліфікаційної роботи магістра є інформаційно-аналітична IoT система освітлення в Розумному Місті. Запропоновано концептуальну модель архітектури системи, розроблено схематичне рішення, спроектовано та реалізовано прототип системи. Одержані результати рекомендовано впровадити у існуючі системи розумного освітлення.

Кваліфікаційна робота магістра складається зі змісту, вступу, основної частини, яка включає чотири розділи, висновків та списку використаних джерел. Всього _____ сторінок.

Ключові слова: модель, архітектура IoT системи, інформаційно-аналітична IoT система, програмний комплекс, схематичне рішення.

ABSTRACT

Hozak Ya. D. INFORMATION-ANALYTICAL IOT LIGHTING SYSTEM IN SMART CITY

Qualifying work of the master for obtaining the scientific degree of master in the specialty 126 - information systems and technologies. - Taras Shevchenko National University of Kyiv, Kyiv, 2021.

Work topic: “Information-analytical Iot lighting system in Smart City”

Relevance. The master's thesis is devoted to solve an important scientific task to develop a method of regulating the workflow of smart street lights in order to enhance the economical efficiency by achieving optimal ratio between electricity consumption and the lifetime of street lights and also achieving a positive influence on pcycho-social state of surrounding people.

The purpose of master’s qualification work is further development of a smart lighting system that includes enhancements of street lights workflow algorithms in order to reduce negative psycho-social impact on surrounding people.

The object of research is the smart lighting system that includes smart lights and a user application.

The subject of research is the methods of street light workflow algorithm optimization that address issues of frequent light intensity changes in some cases.

Research methods - used theoretical and empirical research methods: analysis and generalization of scientific literature, development of software code for microcontroller in Arduino IDE, software design and graphical user interface in Visual Studio, data verification by SWOT analysis.

The practical significance of the obtained result of the master's qualification work is an information-analytical IoT lighting system in Smart City. A conceptual model of the system architecture is proposed, a prototype of the system is designed and implemented. It is recommended to implement the obtained results in existing smart lighting systems.

The master's qualification work consists of the content, introduction, main part, which includes four sections, conclusions and a list of sources used. Total _____ pages.

Keywords: model, IoT solution architecture, information-analytical IoT system, software package, schematic solution.

ЗМІСТ

ВСТУП

Нещодавнє дослідження Європейської комісії показало, що основне збільшення споживання енергії буде головним чином стимулюватись містами, а не сільськими районами [1]. Насправді, до 80% виробленої енергії використовується для постачання електромереж міст. У цьому відношенні, незважаючи на велику кількість наявних ресурсів та пов’язаних технологій, вуличне освітлення представляє одну з найбільш актуальних державних витрат на електроенергію. Однак останні досягнення у відновлюваній енергетиці, енергетичні установки, технології управління та вбудовані системи зробили можливим дизайн сучасних систем освітлення, наділений розумними технологіями, де питання економії енергії може бути ефективно вирішено. Розумне освітлення [2] - це мультидисциплінарна зона в межах контексту Smart City, яка забезпечує інтеграцію між датчиками, технологіями управління, з метою зробити ефективним управління вуличними світильниками. Коли технологічний аспект прийнятий до уваги, парадигма "Розумне місто" глибоко пов'язана із концепцією Інтернету речей (IoT), і для цього це може бути розглянуте як «технологія, що сприяє» наданню послуг у міських районах.

Кілька робіт були представлені в літературі, що охоплюють різні аспекти розумних систем вуличного освітлення. Серед них [3], є перевагами використання світлодіодних ламп постійного струму порівняно з традиційними вуличними ліхтарями змінного струму; насправді вони характеризуються більшим терміном служби, меншим рівнем витрат на обслуговування, вищою ефективністю, і, головне, вони не містять ртуті і виходять легко одноразовими. Відповідне енергозбереження може бути набагато вище, якщо кожен вуличний ліхтар наділений вбудованою сонячною панеллю [4]. Подальше зменшення споживання енергії може бути досягнуто, якщо інтенсивність світлодіодних ламп управляється за допомогою інтелектуальної системи, яка здатна розрахувати оптимальні профілі освітлення на основі дорожнього руху та

інформація про погоду. Цей аспект широко розглядався в [5], [6] де бездротова мережева система передачі даних на основі вуличного освітлення спроектована таким чином, що здатна до контролю і моніторингу на основі системи, що містить ряд вуличних ліхтарів за допомогою протоколу ZigBee. Розглядаються подібні підходи в [7], [8], де світлодіодна система оснащена датчиками та технологіями управління з метою використання виявлення руху для регулювання інтенсивності світла.

У цій роботі зосереджені результати проекту, що сфокусований на дизайн інтелектуального управління освітленням. Система заснована на використанні локальних датчиків, локальних контролерів, обробці інформації х датчиків руху та віддаленого керування через інтернет. Він забезпечує наступні переваги:

- економія енергії;
- зменшення світлового забруднення та викидів CO2;
- поліпшення загального самопочуття.

Метою роботи є подальший розвиток систем розумного освітлення, що включає модернізацію алгоритмів роботи світильників з метою зменшення дратівливого фактору впливу на оточуючих в окремих випадках.

Об’єктом дослідження є система розумного освітлення, що включає в себе розумні світильники та застосунок керування освітленням.

Предметом дослідження є методи оптимізації алгоритмів роботи світильників, що націлені на зменшення дратівливих факторів у вигляді частої зміни інтенсивності освітлення в окремих випадках.

Завданням роботи є розрахунок режимів функціонування розумного світильнику виходячи з аспектів створення комфортних соціо-психологічних умов оточуючого середовища людей з урахуванням мінімізації економічних витрат пов’язаних з функціонуванням та експлуатацією досліджуваної системи, за яких буде визначено оптимальний варіант зміни режиму освітлення, який би задовольняв прийнятну зазначену сукупність параметрів.

Під режимами функціонування мається на увазі інтервали увімкнення та вимкнення освітлення та його інтенсивність.

РОЗДІЛ 1. КОНЦЕПЦІЯ РОЗУМНОГО ОСВІТЛЕННЯ

1.1 Розумне освітлення

Розумне освітлення - це сфера, яка охоплює різні твердотільні технології як світлодіоди та OLED для освітлення внутрішнього та зовнішнього середовища. Системи розумного освітлення в основному залучають цифрові датчики, драйвери виконавчих механізмів та інтерфейси комунікації. Ці освітлювальні системи програмуються з використанням вдосконалених алгоритмів управління і можуть бути організовані в освітлювальні мережі для віддаленої роботи. Деякі з найбільш популярних рішень призначені для зміни світлового спектру або кольору. Вони також можуть контролювати рівень освітленості, коли відбувається зовнішня подія, наприклад, коли відбулася подія, така як виявлення транспортних засобів або людей на дорозі.

Розумна система освітлення позбавляє потреби керувати загальною системою в ручному режимі. Мережа світильників запрограмована з початковим налаштуванням, проте кожен незалежний світильник можна перепрограмувати на реакцію на команди користувачів і ситуації протягом дня. Для розумного освітлення призначені різноманітні цифрові інтерфейси зв'язку,- це цифровий адресний інтерфейс освітлення (DALI), Ethernet, Wi-Fi, ZigBee або Bluetooth для програмування заздалегідь визначених областей. В таких системах, як правило, ділянки сегментовані або згруповані на відповідне поводження залежно від людей або подій, які можуть виникнути. Це дозволяє системі розрахувати рівень необхідного освітлення в залежності від різних завдань користувачів враховуючи енергоспоживання в режимі реального часу. Розумні системи освітлення, організовані як освітлювальні мережі, часто допускають різні типи ідентифікаторів світильників для взаємодії один з одним, щоб їх можна було синхронізувати. Це також дає можливість індивідуально

керувати освітлювальним приладом через мережу віддалено, наприклад, за допомогою веб-додатку із графічним інтерфейсом або мобільного телефону.

1.2 Джерела освітлення: біле світло на основі світлодіодів

Для того, щоб зрозуміти, як працює твердотільне освітлення, нам потрібно спочатку детально пояснити як біле світло може генеруватися для освітлення та / або для спілкування. Люди пристосовані до роботи в здорових умовах, які імітує сонячне денне світло. З цієї причини ми завжди прагнемо висвітлити простір білим світлом, яке імітує сонячний спектр. Існує два різних способи забезпечити штучне біле світло з використанням світлодіодів і обидва методи представляють інтерес для людино-орієнтованих програм (HCA та

оптичні комунікації, де сприйняття кольорового і світлового спектру є дуже

важливими параметрами. Перший спосіб отримання білого світла для цілей освітлення або оптичного зв'язку використовує комбінацію червоного, зеленого та синього світлодіодів (RGB) [20].

Світлодіодні випромінювачі RGB складаються з трьох світлодіодів різних кольорів: червоного, зеленого та синього, які випромінюють спільно для отримання в середньому білого світла. Суміш різної інтенсивності світлодіодів пов'язана з отриманням різних відтінків білого світла. Наприклад, біле світло з більшим вмістом синього світла називається холодним білим світлом, тоді як нейтральне біле світло включає однорідну суміш і тепле біле світло вказує на більшу відносну інтенсивність червоного випромінювача.

Другий спосіб отримання білого світла використовує фосфорне покриття в сукупності зі смолою для покриття світлодіодів.

Значимою характеристикою є той факт, що зазначене покриття може бути як на поверхні кожного синього світлодіода або як покриття у зовнішній структурі. В цьому випадку буде декілька синіх світлодіодів без покриття на їх поверхнях, але із широким загальним фосфорним покриттям на певній відстані, що робить той самий ефект над усіма ними одночасно.

В обох випадках можливо отримати різні температури, що корелюють за кольором (ККТ) кельвіна (K) між (2700 K до 6500 K) шляхом зміни складу люмінофорного покриття або налаштування різних світлодіодів RGB для імітації сонячного спектра, а також налаштування індексу кольоропередачі (ІК) джерела освітлення від 80 до 100. ІК - це кількісне значення для визначення якості кольору, що сприймається очима, коли світильник розташований в приміщенні або на відкритому повітрі. ІК можна змінити, змінивши відсоток фосфору на етапі виготовлення світильників

1.3 Енергоефективні світлодіодні драйвери

Електронні драйвери для освітлення - це пристрої, які регулюють потужність світлодіодів та забезпечують змінний вихідний струм для збігу характеристик джерела світла. Більшість систем освітлення, підготовлених для екосистеми IoT [21], включають:

- етап перетворення потужності зі світлодіодним драйвером;
- декілька світлодіодів, організованих в масиви;
- та датчики та комунікаційний інтерфейс.

Перші два етапи роботи світлодіодного драйвера пов'язані з джерелом живлення, що включає корекцію коефіцієнта потужності та модуль перетворення постійного струму в постійний , тоді як третій етап стосується поточного управління, заснованого на лінійних постійних струмах світлодіодних драйверів або комутації світлодіодних драйверів постійного струму. Такі регулятори як Boost, Buck, Buck-Boost або SEPIC є популярними архітектурами для комутації світлодіодних драйверів постійного струму. Різні типи світлодіодних драйверів здатні досягти високої ефективності перетворення від 85% до 95% при широкому діапазоні напруг. Світлодіодні драйвери схильні до генерації надлишкового шуму, спричиненого періодичним перемиканням на високих частотах, коли вони працюють у режимі широтно-імпульсної модуляції (ШІМ) [10]. Крім того, щоб уникнути видимого та невидимого мерехтіння на

низьких частотах, світлодіодні драйвери включають можливості динамічного затемнення, щоб уникнути набридливості

мерехтіння, одночасно підвищуючи надійність загальної системи. Каскад світлодіодів містить випромінювачі, класифіковані відповідно до вимог до вхідного струму, як низька потужність (5 мА - 20 мА), середня потужність (30 мА - 150 мА) та потужні світлодіоди (більше 150 мА). Випромінювачі світлодіодів із низьким енергоспоживанням використовуються в портативних та низькопотужних драйверах, тоді як світлодіоди середньої потужності більш корисні для внутрішнього освітлення або оптичних комунікацій, особливо у драйверах зв'язку з видимим світлом (ЗВС) та Li-Fi трансиверах. Також потужні світлодіоди призначені для світлодіодних драйверів, що працюють у зовнішніх умовах, наприклад, при вуличному освітленні та автомобільному освітленні. Рядки світлодіодів організовані в послідовному з'єднанні, паралельному з'єднанні або суміші обох, коли кілька світлодіодів з'єднані послідовно як ланцюги, а потім кілька ланцюгів з'єднані паралельно.

1.4 Архітектурні елементи

Інтелектуальні рішення для освітлення включають в себе різні типи пристроїв, систем та типів мереж. Пристрої - це переважно світильники, що містять датчики, виконавчі механізми та удосконалені алгоритми. Кілька вдосконалених алгоритмів дозволяють контролювати рівні денного світла, спектр світла або заняття користувача для прийняття остаточної дії. Алгоритми працюють всередині пристроїв, або для полегшення навантаження пристрою алгоритм може працювати безпосередньо в хмарі, що виконується як веб-служба, для відправки командних повідомлень для виконання різних дій управління.

Кілька алгоритмів інтелектуального освітлення пов'язані з передовими операціями, такими як налаштування відтворення кольору в режимі реального часу. Ця особливість нав’язується на ринку завдяки досягненню інтелектуальних систем освітлення, які можуть впливати на циркадні ритми

людини для покращення настрою та концентрації користувачів. Циркадні системи освітлення застосовують динамічне спектральне відтворення, щоб налаштувати сприйняття білого кольору відповідно до людських ритмів у режимі реального часу.

Автономні алгоритми людських факторів в основному намагаються адаптувати штучне освітлення до змін поведінки людини відповідно до віку, уподобань користувачів та статі. В останні роки було доведено, як світло може модулювати безліч зорових функцій, таких як біологічний годинник, який регулює наші цілодобові циркадні ритми, нашу увагу, гормональну секрецію, температуру тіла та сон. Це пояснює, чому вміст спектра світла, а не колір або ККТ, є найважливішою характеристикою при оцінці циркадних світильників. Дійсно, маючи контроль над спектральними властивостями через систему можна модулювати та регулювати багато фізіологічних властивостей. Окрім освітлення, зосередженого на людині, інші сфери, такі як садівництво, музейне освітлення або сценічне мистецтво, отримують користь від повного контролю над спектром. На системному рівні пристрої можуть бути зображені у фізичній та логічній ієрархіях для спільної взаємодії в освітлювальній мережі. Ця мережа може бути розроблена для підтримки різних фізичних топологій, таких як кільце, зірка, дерево лінія, шина, сітка або гібридне розташування, щоб забезпечити високий ступінь надійності. Як звичайні комунікаційні мережі, різні вузли можуть бути з'єднані за допомогою кабелю або бездротового зв'язку, що охоплюють різні зони у фізичній установці. Мережеве управління може бути локальним або віддаленим і включати різні дротові або бездротові інтерфейси.

Зараз різні освітлювальні прибори на ринку розробляються з різними варіантами підключення, щоб охопити кілька дротових або бездротових інтерфейсів зв'язку, щоб нормально працювати в екосистемі IoT для освітлення. Загальні інтерфейси дротового зв'язку використовують протоколи 0-10В, RS485, DALI, DMX, LON, KNX, BACnet, лінію електропередачі та Ethernet. Крім того, з точки зору бездротових стандартів, світильники для цифрового освітлення мають деякі інтерфейси бездротового зв'язку, такі як субГГц, IEEE802.15.4,

Bluetooth, Wi-Fi. Рішення для освітлення, орієнтовані на зв’язок короткого діапазону та низьке енергоспоживання засновані на фізичному рівні IEEE 802.15.4, такому як ZigBee Light Link або 6loWPAN.

1.5 Передача даних між світильниками за допомогою протоколу LoRaWAN

Базові станції LoRa, підключені до одного мережевого сервера, працюють як єдиний механізм. Оскільки більшу частину часу кінцеві пристрої мовчать, то колізії в ефірі - випадок вкрай рідкісний. Зазвичай, коли датчик виходить на зв'язок, його чують відразу кілька БС. Але відповість тільки одна. Це не обов'язково сама ближня до датчика БС, але завжди та, у якої кращі якісні характеристики каналу зв'язку.

Мережу дуже легко наростити - потрібно просто підключати налаштовану БС до мережевого сервера через Ethernet або мобільні мережі. Але захоплюватися занадто великою щільністю станцій на одиницю площі не можна:

1. Це економічно недоцільно. Станції є дороговартісними, занадто висока щільність розташування приведе до необґрунтованих витрат.
2. Станція завжди повинна знаходитися на домінуючій висоті. Якщо станція знаходиться на 16-поверховому будинку і добре покриває мікрорайон, то велика частина інформації піде через неї. Якась додаткова БС у неї під боком, швидше за все, буде більшу частину часу простоювати.
3. Кожна БС, так чи інакше, може займати в ефірі дорогоцінне місце. Занадто висока щільність БС підвищує ризик колізій.

Оптимальним радіусом роботи вважається 900-1000 метрів. Такої дистанції достатньо для радіомодулів, які знаходяться в підвалах. Там, де не найкращі радіоумови, доводиться розташовувати БС ближче, щоб покриття було

стабільним. В особливо складних випадках можна зменшити цей радіус, хоча така потреба вважається винятковою.

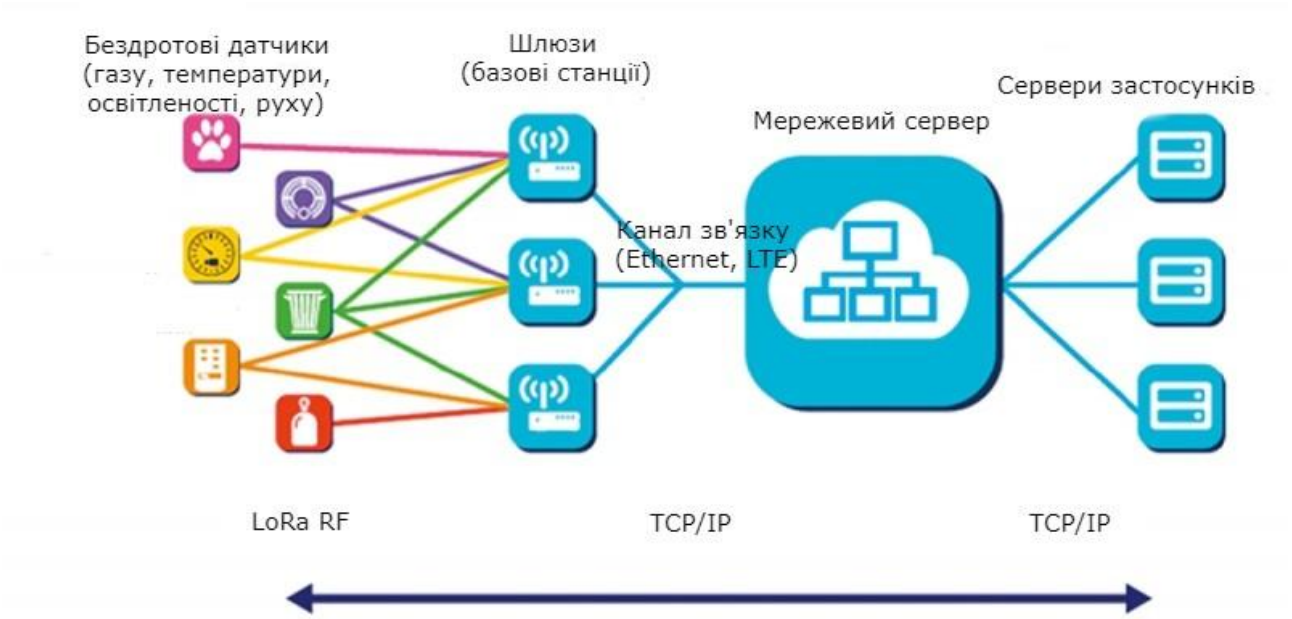


Рисунок 1.1. Схема зв’язку по протоколу LoRaWAN

Пакети приймаються базовою станцією (в архітектурі LoRa її частіше називають шлюзом), однак обробляє їх наступна ланка ланцюга - мережевий сервер. Цей сервер відповідає за управління всіма шлюзами, він вирішує через який шлюз спілкуватися з датчиком (якщо датчик чутно через кілька шлюзів) і визначає ще ряд важливих параметрів (рис. 1.1).

Однак мережевий сервер не обробляє корисну інформацію з пакетів. Це робить наступна і найважливіша ланка - сервер додатків. Саме на сервері додатків відбувається розшифровка показань від датчиків, вони в зрозумілій формі передаються до системи на обробку і далі на інтерфейс споживача або в інше задане місце.[7]

1.6 Зміна освітленості та її наслідки

Для зменшення електровитрат розумні світильники запрограмовані на зменшення яскравості у години неактивності, коли поруч немає рухомих об’єктів. Розумні світильники запрограмовані на збільшення освітленості при

появі рухомих об’єктів і, в залежності від алгоритмів, зменшенні яскравості з часом. Популярним алгоритмом є алгоритм зменшення яскравості із затримкою.

Алгоритм на основі затримки (DEL): ліхтарні стовпи залишаються активними та працюють з повною інтенсивністю, поки датчик руху виявляє присутність користувачів. Світильник вимикається, якщо поблизу немає рухомих об’єктів і протягом часового вікна поруч ніхто не проходить. Потім, коли пішохід (або автомобіль) наближається до ліхтарного стовпа ближче відстані R, відбувається повторне активування лампи. Ця методологія може використовувати як світлодіоди, так і галогенові лампи. Для галогенових ламп, які в даний час є найбільш широко прийнятими в наших містах, може знадобитися близько 15 хвилин, щоб лампа досягла максимальної яскравості після увімкнення, оскільки натрій всередині колби повинен бути повністю нагрітим. Як результат, DEL повинен бути найкращим методом для світлодіодних ламп.

За умов, що розумний світильник знаходиться у енергозберігаючому стані в години неактивності і знижує інтенсивність освітлення при відсутності людей і транспорту і збільшує інтенсивність освітлення при наближенні рухомого об’єкту виникає ефект мирехтіння. Ефект виникає за умов, коли час незгасання світильника після проходження рухомого об’єкту приблизно співпадає із часом появи нового рухомого об’єкту. В такому випадку після зменшення освітлення світильник скоро знову збільшує освітленість для того, щоб освітити простір для нового рухомого об’єкту. За таких умов виникає ефект мерехтіння, коли світильник постійно вмикається і вимикається.

Такий ефект негативно впливає на строк роботи освітлювального приладу а також негативно впливає на психічний стан людей, що проживають неподалік. Блимаюче світло впливає на психічний стан людей і може призводити до підвищення агресії та дратівливості.

1.7 Огляд існуючих рішень

Розумне освітлення є перспективним напрямком у сфері розвитку IoT. Це послугувало причиною виходу на ринку багатьох компаній, що надають свої технічні рішення для розумного освітлення.

1.7.1 Рішення від компанії Tvilight

Лідуючою компанією із встановлення розумного вуличного освітлення на європейському ринку є компанія Tvilight. Компанія спеціалізується на датчиках вуличного освітлення, бездротовому контролі вуличного освітлення та підключеному програмному забезпеченні для управління вуличним освітленням. Інтелектуальні освітлювальні рішення Tvilight створюють незалежну відкриту мережу, яка дозволяє інтегрувати безліч сторонніх додатків і, таким чином, є надійною базою для Smart City та IoT. Tvilight має встановлену базу з понад 100 проектів у всьому світі та розгорнула тисячі інтелектуальних вуличних ліхтарів у відомих містах та критичній інфраструктурі по всьому світу. Міжнародні проекти Tvilight включають, серед іншого, аеропорт Амстердам Шипхол, Нідерландські залізниці, понад 30 голландських містечок і міст, Сеул, а також німецькі міста Дюрен, Кельн, Мюнстер та Берлін.

Компанія пропонує бездротові освітлювальні рішення, які є надійними, маштабованими та відповідають потребам енергоспоживання та навколишнього середовища міст усього світу. Продукція служить відмінною платформою для додаткових пристроїв та додатків Smart City.

Рішення щодо вуличного освітлення працюють за запатентованою технологією виявлення присутності. Підключивши інші датчики поверх освітлювальної мережі, користувач (орган управління освітленням/інфраструктурою міста) може використовувати потужність великих даних для кращого управління трафіком, контролю навколишнього середовища, покращеної безпеки тощо.

Компанія постійно розробляє нові сенсорні програми для кращого освітлення вулиць та розумних міст. Компанія ставить безпеку на перше місце і контролює якість обладнання та програмне забезпечення, що поставляється замовнику.

1.7.2 Рішення від компанії ДТЕК

Лідуючою компанією із сучасної електроенергії в Україні та Києві є компанія ДТЕК. Компанія переважно займається електропостачанням у містах. У проектах компанії є розробка мережі станцій зарядки електроавтомобілів, системи безпеки транспортування великогабаритної продукції та енергоефективні світильники у містах. ДТЕК має декілька проектів із встановлення вуличного освітлення, що використовує сонячні панелі в якості джерела енергії та використовує енергоефективні світлодіодні світильники.

1.7.3 Рішення від компанії Schreder

Ще одна компанія в Україні Schreder - компанія, що спеціалізується на розумному освітленні на основі IoT. Проектами цієї компанії є невеликі ділянки в різних містах України. Продуктами компанії є енергоефективні світильники, що мають інтерфейс підключення IoT модулів, таких як контроллер світильника, модуль дистанційного управління або модуль моніторингу навколишнього середовища. Продукт Schreder Shuffle є основним продуктом компанії, який являє собою освітлювальну опору вуличного ліхтаря із самим ліхтарем. Schreder Shuffle має інтерфейс для кріплення п’яти модулів IoT таких, як: Schreder Luco-ADP (модуль автономної системи контролю освітлення), Schreder Luco-NXP (модуль дистанційного контролю освітлення), Schreder Luco P7 (комбінований модуль керування).

1.7 Методи вирішення проблеми зміни освітленості

В світі розглядаються 2 основні методи зниження частоти зміни освітленості розумних світильників, а саме: переведення світильників на режим

освітлення із постійною яскравістю (зазвичай, це 90-100% яскравості, оскільки повинна виконуватись умова достатньої освітленості простору) та автоматичне підстроювання розумних світильників до інтенсивності руху та вибір оптимальної схеми режиму освітлення.

Перший метод являється простим, але не ефективним, оскільки суттєва частка переваг від розумних світильників пропадає. За умов постійного освітлення з константною яскравістю розумний світильник поводить себе як звичайний світильник.

Другий метод (метод підстроювання) засновується на аналізі статистичних даних по інтенсивності руху в певний час доби, беручи до уваги місцевість розташування та підлаштовуванні розумних світильників. Існує декілька загальних методологічних підходів до прогнозування інтенсивності руху.

1. Методи, які ґрунтуються на використанні даних щодо зміни інтенсивності руху за минулі роки (методи екстраполяції):

– лінійний закон зростання інтенсивності руху, який описується таким рівнянням (прогнозування терміном до 5 років) [1–3]:

$$N_i = N_0 \cdot (1 + \frac{P}{100} \cdot i), \quad (1.1)$$

де N_0 – інтенсивність руху у вихідному році, об./год; N_t – інтенсивність руху в розрахунковому році, об./год; p – середні темпи зростання інтенсивносі за останні 10- 15 років, %; t – розрахунковий період.

– рівняння складних відсотків, яке ґрунтується на застосуванні формули (прогнозування терміном 7 – 10 років) [1–3]:

$$N_i = N_0 \cdot (1 + \frac{P}{100})^{i-1}, \quad (1.2)$$

– експоненціальні й степеневі рівняння, які ґрунтуються на застосуванні таких рівнянь (термін прогнозування до 5 років) [1–3]:

$$N_i = N_0 \cdot e^{\frac{p}{100} \cdot i}, \tag{1.3}$$

$$N_i = N_0 \cdot \left(\frac{P}{100}\right)^i, \tag{1.4}$$

$$N_i = N_0 \cdot i^a, \tag{1.5}$$

де a – показник ступеня.

Для вирішення задачі прогнозування інтенсивності руху на обраних ділянках достатньо використовувати усереднені статистичні значення, зібрані за минулі роки в сукупності із оновленням усереднених значень по мірі надходження нової інформації із сенсорів, розташованих на світильниках. Разом із цим доцільно виконувати корекцію середньотермінового прогнозування інтенсивності руху екстраполяційними методами із уточненням коефіцієнтів.

2. Методи, які ґрунтуються на багатофакторному аналізі факторів впливу на інтенсивність руху.

1.8 Висновки

Ресурси світильника можуть бути збільшені шляхом нормалізації роботи і зменшити економічне навантаження на витрати на обслуговування. Нормалізація роботи світильника разом із контролем плавності зміни інтенсивності освітлення позитивно впливає на психічно-соціальний стан людей поруч.

Більшість рішень, представлених на ринку не беруть до уваги прогнозовану та реальну інтенсивність руху об’єктів повз світильники з метою забезпечення більш стабільного стану освітленості навколишньої території.

Було окреслено основні методи прогнозування інтенсивності руху на базі статистичних даних багатьох показників шляхом статистичного аналізу та багатофакторного аналізу.

РОЗДІЛ 2. РОЗРОБКА МЕТОДІВ ОБРОБКИ ДАНИХ ДЛЯ РОЗУМНИХ СВІТИЛЬНИКІВ

2.1 Огляд вхідних параметрів

Дані відправляються на сервер для подальшої обробки. Сервер збирає та агрегує дані інтенсивності руху для кожного світильника протягом доби.

Алгоритм бере до уваги:

- групу, в якій знаходиться світильник;
- час доби;
- погодні умови;
- тип рухомого об’єкту (пішоходи, автотранспорт).

Група світильників, до якої відноситься поточний є важливим фактором, оскільки група може представляти вулицю, район, тощо, тобто деякий ареал із подібною інтенсивністю руху вздовж усієї області. Наприклад, світильники на одній вулиці можуть відноситися до 2 груп: по одній групі на напрям руху, оскільки інтенсивність руху на одній і тій самій вулиці в певний час доби може суттєво відрізнятися від напряму руху. Так, наприклад, в ранковий час інтенсивність руху до центру міста зазвичай набагато вища, ніж від центру, а в вечірній час навпаки.

Отже, світильники в IoT системі повинні бути правильно згруповані, бо від згрупованості світильників результат передбачень може суттєво змінюватись.

Час доби э фактором, що прямо впливає на інтенсивність руху транспорту та пішоходів, при чому для пішоходів залежність може бути дещо іншою ніж для транспорту.

На інтенсивність руху впливають деякі погодні умови, такі як: опади, вітер, тощо. Деякі погодні умови варто брати до уваги при прогнозуванні інтенсивності руху. Так під час опадів інтенсивність руху пішоходів

зменшується і збільшується інтенсивність руху транспорту. Наявність сильного вітру (варто враховувати силу вітру) так само негативно впливає на рух пішоходів.

Для вирішення задачі прогнозування інтенсивності руху базуючись на заданих факторах і тому факті, що більшість факторів не має точно визначеного впливу на рух доцільним є використання багатофакторного аналізу.

Отже для запропонованої моделі прогнозу інтенсивності руху варто заздалегідь вибрати параметри впливу, на яких буде базуватися прогноз руху.

Для методу багатофакторного аналізу було обрано наступні параметрами рівняння, що впливають на рух:

1. Місяць року (1-12);
2. група світильника, що визначає місце розташування та напрямок руху транспорту (унікальний номер для кожної групи);
3. час доби (16:00 - 8:00);
4. погодні умови:
    a. вітер:
        i. сила вітру;
    b. опади:
        i. кількість опадів;
    c. температура.

2.2 Методи багатофакторного аналізу

Статистичний аналіз може бути досить точним за наявності відповідної кількості статистичних даних.

У загальному вигляді ці методи базуються на регресійному і кореляційному аналізах і описуються такими рівнянням [1–3]:

$$N = A_1 x_1 + A_2 x_2 + A_3 x_3 + \ldots + A_n x_n + B, \quad (2.1)$$

де $A_1, A_2, A_3, \ldots, A_n$ – коефіцієнти регресії; $x_1, x_2, x_3, \ldots, x_n$ – чинники руху, які впливають на інтенсивність.

За допомогою кореляції і регресії можна виявити, які чинники і наскільки впливають на інтенсивність руху та визначити, чи існує зв'язок, а якщо існує, то якої форми (прямий, зворотний, лінійний або нелінійний) і яким рівнянням його можна описати; також в якій мірі інтенсивність руху схильна до коливання (зміни), незалежно від дії чинників на неї. До числа таких чинників включають: час доби; чисельність населення в досліджуваному районі;тип рухомих об’єктів; погодні умови. Методи, які ґрунтуються на багатофакторному аналізі доцільно використовувати лише в тих районах урбанізованого простору, де були зібрані дані про інтенсивність руху та чинники, що впливають на неї.

Ще одним методом з прогнозування є використання апроксимуючих функцій. Для реалізації цього методу необхідний початковий ряд статистичних даних, який вирівнюють шляхом графоаналітичного або математичного підбору аналітичної функції, що дозволяє в максимальній степені наблизити теоретичні і статистичні дані. Наявність такої функції, апроксимуючої значення статистичного ряду (даних), є простою математико-статистичною моделлю показника інтенсивності руху. [5]

2.2.1 Багатофакторний аналіз з використанням критерію Фішера

Критерій Фішера застосовується під час перевірки гіпотези про рівність дисперсій двох генеральних сукупностей, розподілених за нормальним законом. Він є параметричним критерієм.

F-критерій Фішера називають дисперсійним відношенням, так як він формується як відношення двох порівнюваних незміщених оцінок дисперсій.

Нехай в результаті спостережень отримані дві вибірки. За ними обчислені дисперсії S_1^2 і S_2^2, що мають f_1 та f_2 ступенів свободи. Вважаємо, що першу вибірку взято з генеральної сукупності з дисперсією σ_1^2, а другу – з генеральної сукупності з дисперсією σ_2^2. Висувається нульова гіпотеза про рівність двох

дисперсій, тобто H_0: $\sigma_1^2 = \sigma_2^2$. Для того, щоб відкинути цю гіпотезу потрібно довести значимість відмінності при заданому рівні значимості α.

Значення критерію обчислюється за формулою:

$$F = \frac{S_1^2}{S_2^2}, \qquad (2.2)$$

Очевидно, що при рівності дисперсій величина критерію буде дорівнює одиниці. В інших випадках вона буде більше (менше) одиниці.

Критерій має розподілення Фішера $F(\alpha, f_1, f_2)$. Критерій Фішера – двосторонній критерій, і нульова гіпотеза $H_0 : S_1^2 = S_2^2$ відкидається на користь альтернативної $H_1 : S_1^2 \neq S_2^2$, якщо

$$F(1-\alpha/2, f_1, f_2) < F < F(\alpha/2, f_1, f_2), \qquad (2.3)$$

тут $f_1 = n_1 - 1; \quad f_2 = n_2 - 1,$

де $n_1,\ n_2$ – об’єм першої та другої вибірки відповідно.

2.2.2 Метод найменших квадратів

При розв’язанні великої кількості різноманітних задач у процесі досліджень одержують статистичні ряди вимірів двох (або декількох) значень, об'єднаних функцією $y = f(x)$.

Кожному значенню функції $y_1, y_2, \ldots, y_n$ відповідає певне значення аргумента $x_1, x_2, \ldots, x_n$. На основі експериментальних даних можна підібрати алгебраїчні вирази, які називають емпіричними формулами. Такі формули підбирають тільки в межах обмірюваних значень аргумента $x_1 \ldots x_n$.

Емпіричні формули мають тим більшу цінність, чим більше вони відповідають результатам експерименту, і вони є незамінними для аналізу вимірюваних величин.

До емпіричних формул ставлять дві основні вимоги: по можливості вони повинні бути найбільш простими й точно відповідати експериментальним даним.

Процес підбору емпіричних формул складається з двох етапів. Спочатку дані вимірів наносять на сітку координат, з'єднують експериментальні точки плавною кривою й вибирають орієнтовно вигляд формули. На другому етапі обчислюють параметри формул, які б щонайкраще відповідали взятій формулі. Підбір емпіричних формул треба починати з найпростіших виразів.

Одним із найпоширеніших методів визначення параметрів емпіричних формул є метод найменших квадратів, який дає найкращі за точністю результати.

Нехай за експериментальними даними побудована емпірична крива, за виглядом якої підібрана теоретична крива, рівняння якої є відомим. Наприклад, змінні x, y, z зв'язані між собою рівнянням.

$$a \cdot x + b \cdot y + c \cdot z = N, \tag{2.4}$$

Для визначення значень коефіцієнтів a, b та c необхідно знайти дослідним шляхом значення їх величин при трьох різних комбінаціях змінних:

$$\begin{aligned} a \cdot x_1 + b \cdot y_1 + c \cdot z_1 &= N_1, \\ a \cdot x_2 + b \cdot y_2 + c \cdot z_2 &= N_2, \\ a \cdot x_3 + b \cdot y_3 + c \cdot z_3 &= N_3. \end{aligned} \tag{2.5}$$

Після розв’язання системи з трьох рівнянь знайдемо шукані коефіцієнти. Проте, якщо змінним додати деякі четвертні значення x_4, y_4, z_4 і визначити

дослідним шляхом N_4, то виявиться, що рівність (2.4) при коефіцієнтах, обчислених за рівняннями (2.5), не буде виконано:

$$a \cdot x_4 + b \cdot y_4 + c \cdot z_4 \neq N_4 . \qquad (2.6)$$

Пояснюється це тим, що коефіцієнти, визначені з рівнянь (2.5) за дослідними даними, мають деяку помилку експериментів (помилки дослідів, вимірів і т.п.). Тому значення невідомих *a*, *b* і *c* варто знаходити не з трьох рівнянь, а значно більшої кількості даних досліджень, щоб одержати більш надійні значення цих коефіцієнтів:

$$\begin{aligned} &a \cdot x_1 + b \cdot y_1 + c \cdot z_1 = N_1, \\ &a \cdot x_2 + b \cdot y_2 + c \cdot z_2 = N_2, \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ &a \cdot x_n + b \cdot y_n + c \cdot z_n = N_n . \end{aligned} \qquad (2.7)$$

Таким чином, ми одержали "надлишкову" систему рівнянь, які називають "умовними", тому що вони не цілком сумісні. Тобто значення невідомих, визначені, наприклад, з перших трьох рівнянь, не будуть дорівнювати значенням цих самих невідомих, визначених з інших трьох рівнянь цієї системи. Потрібно визначити, які повинні бути значення коефіцієнтів *a*, *b* й *c*, щоб "надлишкова" система рівнянь задовольнялася щонайкраще, тобто помилки $E_i = N - N_i$, де N – розрахункові, а N_i – дослідні дані, або

$$\begin{aligned} &E_1 = a \cdot x_1 + b \cdot y_1 + c \cdot z_1 - N_1, \\ &E_2 = a \cdot x_2 + b \cdot y_2 + c \cdot z_2 - N_2, \\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ &E_3 = a \cdot x_n + b \cdot y_n + c \cdot z_n - N_n . \end{aligned} \qquad (2.8)$$

повинні бути найменшими.

Для розв'язання цієї задачі користуються принципом, запропонованим Лежандром й удосконаленим Лапласом і Гаусом. Відповідно до зазначеного принципу з усіх можливих величин *a*, *b* й *c* найбільш задовільними будуть ті, при яких сума квадратів помилок буде найменша:

$$E_1^2 + E_1^2 + ... + E_n^2 = \min .$$

Піднесення до квадрата рівнянь системи (2.8) та їх додавання дає:

$$\begin{aligned}\sum_1^n E_i^2 = a^2 \sum_1^n x_i^2 + b^2 \sum_1^n y_i^2 + c^2 \sum_1^n z_i^2 + 2ab \sum_1^n x_i y_i + \\ + 2ac \sum_1^n x_i z_i + 2bc \sum_1^n y_i z_i - 2a \sum_1^n x_i N_i - \\ - 2b \sum_1^n y_i N_i - 2c \sum_1^n z_i N_i + \sum_1^n N_i^2\end{aligned} \qquad (2.9)$$

Для дотримання умови $\sum_1^n E_i^2 = \min$ необхідно, щоб сума частинних похідних по *a*, *b* та *c* рівняння (2.9) перетворювалася в нуль, а отже, і кожна з цих частинних похідних повинна перетворитися в нуль:

$$\frac{\partial \sum_1^n E_i^2}{\partial a} = 0, \quad \frac{\partial \sum_1^n E_i^2}{\partial b} = 0, \quad \frac{\partial \sum_1^n E_i^2}{\partial c} = 0.$$

Диференціюючи рівняння (2.9) послідовно по *a*, *b* та *c*, одержимо:

$$\begin{aligned}a\sum x^2 + b\sum xy + c\sum xz - \sum xN = 0, \\ b\sum y^2 + a\sum xy + c\sum yz - \sum yN = 0, \\ c\sum z^2 + a\sum xz + b\sum yz - \sum zN = 0.\end{aligned} \qquad (2.10)$$

Ці рівняння є “нормальними”. Вирішуючи їх щодо *a*, *b* та *c*, одержимо найкращі значення коефіцієнтів, що задовольняють принцип найменших квадратів.[8]

2.3 Метод обробки даних в IoT системі

Дані, зібрані з датчиків а також статистичні дані, зібрані з інших джерел обробляються застосунком, тобто на сервері застосунків.

Базовий алгоритм засновується на статистичних даних про рух пішоходів та транспорту за попередні роки з використанням статистичного аналізу. При ініціюванні системи застосунок аналізує статистичні дані та визначає інтенсивність руху для кожної групи світильників. Визначивши попередню прогнозовану інтенсивність руху застосунок визначає оптимальні часові затримки вимкнення світильників після проходження об’єкту та за умови відсутності рухомих об’єктів у зоні дії. Таким чином визначається початкова схема роботи світильників у системі.

За наявності достатньої кількості статистичних даних значень інших параметрів, таких як погодні умови, є можливість проведення багатофакторного аналізу цих даних з метою уточнення коефіцієнтів рівняння.

Далі через кожен визначений проміжок часу виконується обробка нових вхідних даних: дані з датчиків руху від світильників, нові погодні дані - та виконується уточнення коефіцієнтів рівняння. Після визначення нового рівняння виконується прогнозування інтенсивності руху на визначені періоди та перерахунок схеми роботи світильників.

Часто неможливо зібрати достатню кількість даних до початку роботи розумних світильників. В такому разі пропонується встановлювати схему роботи світильників за замовчуванням. Час затримки за замовчуванням може відрізнятися, проте пропонується встановлювати його близьким до 10 секунд як стандартний час затримки для світильників [22] .

Починаючи з першого дня роботи світильників і отримання даних з сенсорів руху можна виконувати початковий аналіз. Для виконання

регресійного аналізу потребується вибірка даних, кількість записів у якій принаймні в 10 разів більша за кількість параметрів регресійного рівняння [11]. Тож, для аналізу рівняння знадобиться близько 470 вимірювань.

Вимірювання відбуваються на погодинній основі, проте значення групуються погодинно у 24 групи, кожна з яких аналізується окремо. Це значить, що фактично, отримується один результат вимірювання в день. Звідси випливає, що потребується близько 470 днів для достатньої кількості даних для аналізу рівняння із 47 змінними.

Тож, у перші місяці роботи системи за недостатньої кількості даних кращий ефект буде мати звичайне зведення до середнього значення. Тобто дані про рух пішоходів групуються по обраних інтервалах часу, наприклад, погодинно, агрегуються та обраховуються як середнє значення інтенсивності руху за проміжок часу.

2.4 Визначення схеми роботи світильників

Стандартна схема роботи світильників в енергозберігаючому режимі виглядає як наступний набір фаз:

1. Детекція руху в радіусі R від світильника.
2. Увімкнення світильнику.
3. Об’єкт перебуває у радіусі роботи світильника R рухаючись зі швидкістю v (протягом часу *t*, обчисленим за формулою (2.11)).
4. Детекція відсутності руху в радіусі R від світильника.
5. Очікування часу T для комфортного перебування користувача (об’єкту) поза зоною дії світильника .
6. Вимкнення світильнику.

$$t = \frac{R}{v}, \tag{2.11}$$

де R - радіус зони роботи світильнику;

V - швидкість об’єкту, що рухається повз світильник.

Звідси видно, що якщо середній інтервал руху об’єктів повз світильники I складає трохи більше t + T, то світильники будуть вимикатися після проходження об’єкту і невдовзі вмикатися знову для підсвічування зони для проходження наступного об’єкту. Таким чином виникає той ефект мерехтіння, який може негативно впливати на оточуючих.

Для зменшення такого ефекту метод адаптації роботи світильників повинен адаптувати схему роботи світильників змінюючи час очікування T таким чином, щоб зменшити кількість мерехтіння. Зазвичай, алгоритм повинен дещо збільшувати час очікування T, щоб інтервал I був трохи менший за t + T, при тому T може змінюватися у фіксованому діапазоні $T_{min} \leq T \leq T_{max}$.

Для прикладу розглядається деякий змодельований рух пішоходів повз деякого світильника і робота світильника виглядає як на рис. 2.4.1

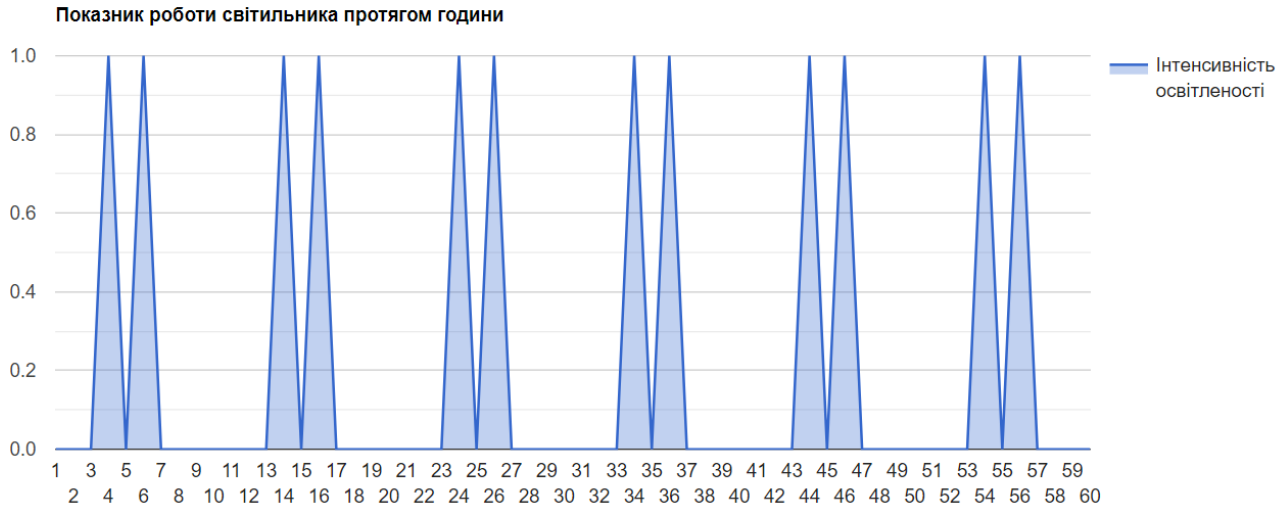


Рисунок 2.4.1. Показники роботи світильника протягом години

Застосовуючи вищезазначені методи прогнозування інтенсивності руху (методи, засновані на статистичному аналізі або методи, засновані на багатофакторному аналізі) прогноз середньої інтенсивності руху для обраної ділянки виглядає як на рис. 2.4.2. Методи прогнозування дозволяють оцінити середню інтенсивність руху на прогнозований відрізок часу. Середня інтенсивність не відображає хаотичного характеру реальної інтенсивності руху,

проте дає змогу оцінити інтервали часу між людьми або автомобілями, що проїжджають повз.

Рисунок 2.4.2. Прогнозована середня інтенсивність руху протягом години для обраної ділянки

Використання методу адаптації схеми роботи світильників веде до зміни часу очікування T таким чином, що час очікування T трохи більший за середню інтенсивність руху I, що дозволяє не перемикати світильник між проходячими об’єктами. Проте такий алгоритм роботи може вестидо майже постійного освітлення території, що не є ефективним.

Для збільшення ефективності роботи кожен світильник визначає чи були присутні пішоходи у радіусі дії протягом часу очікування T. Якщо пішоходи з’являлися у зоні дії світильнику протягом часу очікування T, то час очікування T підібрано правильно, що зменшило кількість перемикань світильнику. Окрім того, якщо пішоходи були у радіусі світильнику протягом часу T, то світильник можна вимикати, оскільки наступне прогнозоване збудження світильнику відбудеться через інтервал I.

Таким чином алгоритм роботи світильнику дещо змінюється, а точніше змінюється пункт 5, який розширюється наступними перевірками:

1. Якщо під час очікування протягом T було помічено нові об’єкти, то алгоритм працює коректно;

2. Якщо протягом часу очікування T об’єктів не було помічено, то алгоритм спрацював не правильно і час T не є задовільним.

При детекції руху алгоритм світильника враховує, чи світильник вже увімкнений, чи ні. Якщо світильник вже увімкнений, то при детекції руху час очікування T не починає відраховуватись заново, а продовжує відрахунок від першого збудження. Проте береться до уваги мінімальний час затримки для комфортного проходження. Тобто, якщо після проходження другого об’єкту час затримки T менший за T_{min}, то світильник працює протягом T_{min} для комфортного проходження об’єкту. При наступній детекції руху світильник знову вмикається і очікує протягом часу T на появу наступного об’єкту. В такий спосіб освітлення відбувається не для окремого об’єкту, а для групи об’єктів, що рухаються із певним передбаченим інтервалом.

Така зміна схеми освітлення може зменшити кількість перемикань світильнику до 2 разів (рис. 2.4.3.) Це дозволяє досягти певного компромісу між енергозбереженням та кількістю перемикань світильників.

Рисунок 2.4.3. Адаптована схема освітлення

Якщо для підбору часу T інтервал руху I занадто великий, такий що $I > T_{max}$, то немає сенсу збільшувати час очікування, оскільки це не призведе до бажаних результатів. В такому випадку можна зменшити час очікування T

для того, щоб зменшити світлове забруднення у нічний час. При тому T не може бути меншим за мінімальний час очікування світильнику для комфортного проходження повз нього. $T_{min} \leq T$

Якщо інтервал руху I занадто малий, такий, що $I < T_{min}$, то відповідно час очікування T може бути зменшений лише до T_{min}, але це свідчить про те, що інтенсивність руху в даний проміжок часу для обраної групи світильників занадто висока і потребується постійне освітлення.

Алгоритм вибору T може бути представлений наступним чином (псевдокод):

```
Tcur = I + t;
if(Tcur <= Tmin  OR Tcur >= Tmax)
        T = Tmin;
else
        T = Tcur
```
, (2.12)

де t - деякий буферний час, щоб запобігти ефекту мерехтіння при маленьких відхиленнях від середнього інтервалу I у більшу сторону.

У наведеному псевдокоді можна побачити, що коли T більше ніж T_{max}, то T встановлюється до найменшого допустимого часу затримки у значення T_{min}. Це обумовлено тим, що при великих інтервалах між рухом пішоходів немає допустимого часу затримки T, щоб забезпечити зменшення кількості перемикань світильників. Отже, розумний ліхтар переходить до найменшого часу затримки, щоб зменшити загальний світловий потік протягом доби.

Оскільки розумні світильники працюють у системі і, відповідно, можуть надсилати і отримувати команди від сусідніх світильників, то кожний світильник має брати до уваги спосіб, яким було спричинено увімкнення цього світильника. Так, якщо світильник був спровокований рухом у зоні дії, то після проходження об’єкту та по закінченні часу очікування світильник повертається

у вихідне положення. Проте світильник може бути збудженим командою від сусідніх світильників, що означає про наближення об’єкту. Отже, світильник ще не зафіксував рух, проте є сповіщеним про наближення об’єкту. В такій ситуації пропонується вмикати світильник на деякий фіксований час. Якщо в цей період з’являється об’єкт, то після його проходження вмикається час очікування. Порівняно із алгоритмом коли спрацювання відбувається через детекцію руху коли об’єкт з’являється у зоні світильнику під час роботи в режимі освітлення, тобто під час очікування T, цей час очікування T не оновлюється по завершенні проходження об’єкту, а залишається таким самим. То якщо об’єкт з’.вився у зоні дії світильнику, коли той був збуджений командою від сусідніх світильників, цей час T оновлюється так, якби перший об’єкт був збуджувачем світильника. В такому випадку зберігається час освітлення між об’єктами, а ввімкнення світильника по команді лише надає додаткового рівня комфорту при проходженні повз.

Проте, об’єкт може не з’явитися у зоні дії світильнику. Тоді світильник вимикається після часу очікування і переходить в режим ніби він (світильник) і не вмикався

Важливий вплив має те, яким чином світильник змінює стан. Із звичайними лампами світильник вмикається і вимикається різко, тобто світловий потік змінюється від 0 до максимуму і навпаки за дуже короткий срок. Важливим з точки зору психічної дії на оточуючих є плавна зміна світлового потоку або діммiювання. Плавна зміна світлового потоку має на увазі увімкнення або вимкнення світильнику протягом деякого часу. Наприклад, увімкнення світильнику протягом 3 секунд має ефект плавного ввімкнення, що суттєво знижує дратівливий фактор.

2.5 Висновки

Було запропоновано метод передбачення інтенсивності руху з метою поправки схеми освітлення для оптимізації роботи світильників у нічний час доби, яка передбачає собою балансування між енергоефективністю за рахунок

зниження освітлення у години неактивності та за малою інтенсивністю руху та кількістю перемикань світильнику з метою зменшення дратівливого впливу на оточуючих.

Для передбачення інтенсивності руху було запропоновано використання середнього значення показника інтенсивності за малої вибірки даних та використання регресійно-кореляційного аналізу за достатньої кількості даних.

Було запропоновано алгоритм підбору часу очікування світильнику після проходження об’єкту таким чином, щоб час очікування був максимально ефективним з точки зору кількості перемикань світильнику, коли таке можливо.

РОЗДІЛ 3. ПРОЕКТУВАННЯ АРХІТЕКТУРИ IOT СИСТЕМИ РОЗУМНОГО ОСВІТЛЕННЯ

Інтернет речей (IoT) включає цілу екосистему інструментів та послуг, які повинні об’єднуватися, щоб забезпечити цілісне рішення. Знання ключових компонентів архітектури IoT та способи їх інтеграції може бути проблемою. Проте, за останні кілька років з’явилися репрезентативні архітектури для конкретних випадків використання IoT.

3.1 Складові архітектури IoT

Незалежно від варіанту використання, майже кожне IoT рішення включає однакові чотири компоненти: пристрої, підключення, платформу та додаток. Деякі випадки використання можуть включати додаткові шари, але ці чотири компоненти є основою кожного IoT рішення як на рис. 3.1.1.

Рисунок 3.1.1. Основні складові IoT рішення

3.1.1 Пристрої IoT

Пристрої IoT - це фізичні або віртуальні речі, які надсилають події та отримують команди від програм IoT. Терміни річ і пристрій означають підключений пристрій у рішенні IoT [10].

Пристрій IoT має одну або кілька з таких характеристик:

- Володіє унікальною ідентичністю, яка відрізняє його в межах рішення (ідентифікатор);
- Має властивості або стан, до якого мають доступ програми;
- Надсилає події на платформу IoT, щоб додатки могли реагувати;
- Отримує команди від програм для виконання.

Пристрої IoT складають фізичний апаратний компонент рішення. У випадку використання моніторингу промислового обладнання це такі речі, як двигуни та контролери двигуна. Для випадків використання розумного середовища це можуть бути датчики руху або зчитувачі знаків. Для відстеження активів або майна це GPS-трекери. Для використання у розумному міському освітленні це датчики руху, вологості, якості повітря, контролери світильників, тощо.

У багатьох випадках використання промислового IoT та при проектуванні розумних будинків клієнти вважають за краще використовувати готові пристрої, які можна додати до існуючого середовища або обладнання. Однією з проблем при виборі готового обладнання може бути отримання доступу до даних. Багато постачальників пропонують приховані рішення, які можуть добре працювати для вирішення дуже конкретних проблем, але вони не працюють достатньо добре для використання даних як частини більш широкого додатку IoT. Досліджуючи апаратні рішення потрібно переконатись, що можна отримати доступ до даних через локальні протоколи, такі як Modbus, Serial, тощо. Деякі постачальники можуть також мати хмарний сервіс, де можна отримати доступ до даних через API.

3.1.2 Підключення IoT

У більшості рішень IoT пристрої надсилають дані про свій стан та отримують команди від централізованої платформи IoT. Існує багато варіантів того, як здійснюється підключення пристрою до платформи, і це сильно залежить від середовища, в якому знаходиться пристрій та обмежень самого

пристрою. Якщо пристрій знаходиться зовні і рухається, як у випадках відстеження активів, мобільний зв’язок є хорошим вибором. Якщо пристрій перебуває в приміщенні в домашньому або промисловому середовищі, зазвичай підключення відбувається через Ethernet або Wi-Fi. Якщо пристрій живиться від акумулятора, то доводиться досліджувати варіанти низького енергоспоживання, такі як Bluetooth Low Energy або LPWAN. Розумне освітлення має стабільне живлення, а встановлення нових світильників передбачає можливість прокладання додаткового кабелю, який може передавати дані. Наприклад, Ethernet кабель.

3.1.3 IoT Gateways та Edge Computing

Пристрої IoT можуть підключатися до платформи IoT безпосередньо або через шлюзи IoT Edge, що реалізують інтелектуальні можливості. Edge шлюзи (gateways) включають такі функції:

1. Агрегування або фільтрація подій пристрою до їх надсилання на платформу IoT.
2. Локалізоване прийняття рішень.
3. Переклад протоколів та ідентичності (унікальності) від імені пристроїв.

Існує два типи крайових шлюзів, польові або IoT Edge, і хмарні або протокольні шлюзи (рис. 3.1.2).

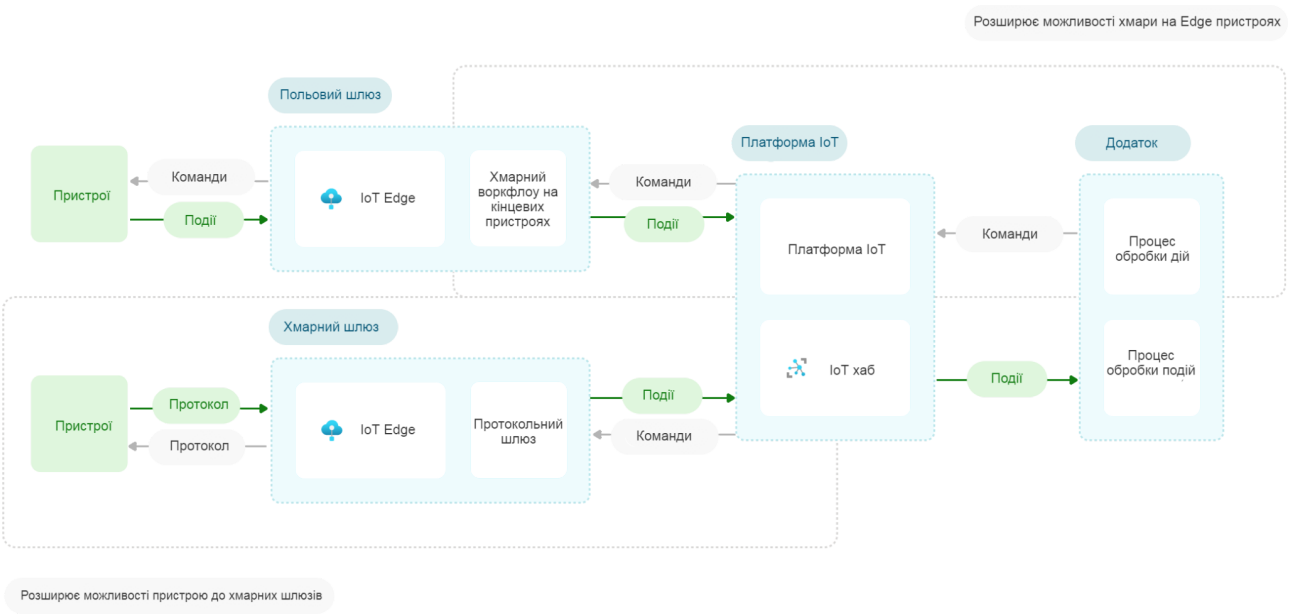


Рисунок 3.1.2. Типи IoT шлюзів

- Польові шлюзи IoT Edge розташовані поблизу локальних пристроїв і підключаються до платформи IoT, щоб розширити хмарні можливості на пристрої. Пристрої IoT Edge можуть виступати засобами зв'язку, локальними системами управління пристроями та процесорами даних для платформи IoT. Пристрої IoT Edge можуть запускати хмарні робочі процеси локально за допомогою модулів Edge і можуть спілкуватися з пристроями навіть у автономних сценаріях.
- Протокольні або хмарні шлюзи дозволяють підключати наявні різноманітні групи пристроїв до рішень IoT, розміщуючи екземпляри пристроїв та забезпечуючи зв'язок між пристроями та платформою IoT. Хмарні шлюзи можуть виконувати переклад протоколів та посвідчень на платформу IoT та з неї, а також можуть виконувати додаткову логіку від імені пристроїв.

Деякі пристрої не можуть підключитися безпосередньо до центральної платформи і вимагають використання шлюзу IoT для подолання розриву між середовищем розташування та платформою платформою. Це характерно для промислових середовищ, зазвичай взаємодія з наявним обладнанням

відбувається через локальні протоколи, такі як Modbus, OPC UA або Serial. Шлюзи також потрібні при використанні бездротових технологій, таких як BLE та LPWAN, оскільки вони не забезпечують прямого підключення до відведеної мережі або хмари.

У цих ситуаціях пристрій підключається до шлюзу. Шлюз зчитує необхідну інформацію, а потім відправляє дані на платформу за допомогою з'єднання “зворотній зв’язок”, наприклад стільникового або Wi-Fi з’єднання, яке може надати доступ шлюзу до встановленої мережі або хмари.

Шлюзи також дозволяють ввести Edge Compute у свою архітектуру. Edge Compute переносить обробку та управління з хмари та розміщує їх на обладнанні або поруч. Хмара є важливою складовою IoT архітектури, але вона має обмеження, які є поза контролем користувачів, такі як надійність з’єднання з Інтернетом та затримка зв'язку. Якщо рішення потрібно приймати в реальному часі, або пристрої системи генерують занадто багато даних для надсилання в хмару, введення Edge Compute у архітектуру IoT може бути хорошим рішенням.

3.1.4 Платформа IoT

Платформа IoT - це сукупність послуг, що дозволяють пристроям та програмам підключатися та взаємодіяти між собою.

Платформа IoT щонайменше виконують наступні дії:

- Забезпечують зв’язок, автентифікацію та зв’язок між пристроями та програмами.
- Генерують контекстуальну інформацію про вхідні події для визначення маршрутизації подій до кінцевих отримувачів.

Платформа IoT є центральним сховищем даних та механізмом організації рішення. Створення безпечної та масштабованої платформи є непростим завданням. Вибір правильної платформи може бути складним процесом. Нижче наведено контрольний список для оцінки платформи.

- Знання галузі.

- Знання IoT.
- Якість та рівень підтримки.
- Обмеження на пристрої, корисне навантаження або дані.
- Стабільність.
- Функціональність відповідає потребам використання.
- Безпека.
- Висока доступність / аварійне відновлення.
- Безперервна освіта.
- Документація.

3.1.5 Додаток IoT

Додатки - це сукупність служб та компонентів, які є унікальними для рішення IoT. Додатки IoT зазвичай мають:

- Об’єднання сервісів для обчислень, зберігання та кінцевих точок події в поєднанні з унікальною бізнес-логікою додатків.
- Процеси подій для отримання та обробки вхідних подій з пристроїв.
- Робочі процеси для надсилання команд на пристрої чи до інших процеси.

Додаток IoT забезпечує взаємодію з кінцевими користувачами або те, як він або клієнти взаємодіють з даними, зібраними з пристроїв IoT. Це може бути мобільний додаток, веб-сайт, настільний додаток або навіть пасивна система, з якою ніхто не взаємодіє безпосередньо. Їх побудова може бути складною і трудомісткою. Досліджуючи платформи IoT, потрібно звернути увагу на сумісність із додатками, оскільки вони пропонують інструменти, які можуть значно прискорити розробку додатку.

Додаток IoT є важливою складовою архітектури IoT і саме тут реалізується фактична цінність IoT рішення. Розгортання обладнання та збір даних з датчиків не має сенсу, якщо ці дані не представлені корисними

способами та не вирішують реальних проблем для клієнтів. У контексті розумного освітлення IoT додаток є панеллю управління розумними світильниками а таком інформаційною системою всього розумного освітлення.

Рішення IoT вимагають зовсім іншої архітектури, яка може бути некомфортною для багатьох компаній. Поки компанії рухаються вперед зі своїми стратегіями цифрової трансформації, розуміння цієї архітектури важливо для досягнення успіху. Деякі компанії, як промислові виробники, продають мільйони пристроїв клієнтам, розподіленим по всьому світу. Створення надійного, безпечного та масштабованого додатка з широко розподіленою системою, подібною до цієї, вимагає міцної основи для того, щоб надати своїм клієнтам винятковий досвід користування [9].

3.1.5 Атестація, автентифікація та забезпечення

Підключення пристроїв IoT до платформи IoT включає три процеси атестації, автентифікації та забезпечення.

Механізм атестації представляє собою метод, вибраний для пристрою, щоб підтвердити себе унікально при підключенні до служби платформи IoT. IoT Хаб підтримує симетричне шифрування, відбиток пальця X.509 та методи атестації X.509 CA.

Аутентифікація - це спосіб, яким пристрій ідентифікує себе. IoT Хаб надає доступ до пристрою на основі його здатності підтвердити себе, використовуючи унікальну ідентифікацію пристрою в поєднанні з механізмом атестації.

Забезпечення - це акт реєстрації пристрою в IoT Хаб. Забезпечення надає IoT Хабу інформацію про пристрій та механізм атестації, який використовує пристрій.

Надання пристроїв може відбуватися через службу надання послуг пристроїв або безпосередньо через API інтерфейсу менеджера реєстру IoT Хабу. Використання DPS (Devise Provisioning Service) надає переваги пізнього

зв’язування, що дозволяє видаляти та повторно реєструвати польові пристрої до IoT Хабу без зміни програмного забезпечення пристрою.

У наступному прикладі показано, як реалізувати робочий процес переходу з тестового середовища на продакшн середовище за допомогою DPS.

1. Розробник рішення пов'язує хмари тестування та продакшену IoT із службою надання послуг;
2. Пристрій реалізує протокол DPS для пошуку IoT Хабу, якщо він більше не надається. Спочатку пристрій під’єднаний до тестового середовища;
3. Оскільки пристрій зареєстровано в тестовому середовищі, він там підключається і відбувається тестування;
4. Розробник повторно надає пристрій продакшн середовищу та видаляє його з тестового хабу. Тестовий хаб відхиляє доступ пристрою під час наступного повторного підключення;
5. Пристрій підключається та повторно проходить під’єднання. Тепер DPS направляє пристрій у продакшн середовище, і пристрій підключається та аутентифікується там.

3.2 Архітектура інформаційно-аналітичної системи розумного освітлення

Світильники мають вбудований годинник реального часу. Це забезпечує світильники можливістю роботи в офлайн режимі, якщо відсутнє мережеве з’єднання з груповим контролером або із сервером. Годинник реального часу може працювати довгий час без налаштувань і цим підвищує стійкість системи в умовах вимкнення живлення чи перебоїв у мережах.

Кожен світильник має зовнішню постійну пам’ять для збереження налаштування. Зовнішня пам’ять монтується або окремим модулем або у складі контролеру світильника. Для управління світильником потребується дуже малий об’єм пам’яті для зберігання налаштувань схеми освітлення, тож витрати на цей модуль є незначними. Постійна пам’ять потрібна для роботи світильника

в автономному режимі, наприклад, при першому підключенні або у разі виникнення непередбачуваних ситуацій із мережею живлення або зв’язком із сервером. Також, у постійній пам’яті зберігаються особливі налаштування, призначені для певного світильника. Наприклад, світильник повинен працювати постійно для підсвічування небезпечних зон таких як перехрестя.

Кожен світильник оснащений GSM модулем для екстреної комунікації із платформою якщо основне з’єднання не працює. Наприклад, повідомлення про несправність (світильника чи мережі), запит на оновлення схеми освітлення, тощо. за нормальних умов коли світильник працює справно GSM модуль не використовується і тим самим не витрачає свої ресурси але коли світильник опиняється у несправному стані, наприклад, світло не вмикається або не вимикається, то світильник надсилає зворотне повідомлення до диспетчерської для повідомлення хоста (повідомляє додаток). У разі несправності вуличний світильник надсилає повідомлення про помилку до диспетчерської, щоб повідомити оператора про несправність. Оператор може вжити подальших заходів для проведення ремонтних робіт. У порівнянні зі звичайною системою вуличного освітлення, інтелектуальна вулична система пропонує високу надійність та низький рівень витрат на обслуговування. Система зворотного зв'язку дозволяє вуличному світлу комунікувати з платформою, повідомляючи про свій щоденний стан [19].

3.2.1 Координація світильників через шлюз

Розумні світильники встановлюються групами. Група світильника визначається відповідно до місця встановлення світильника. Або, навпаки, на плані зони розподіляються по групах і до груп заносять світильники.

До кожної групи встановлюється контролер, що виконує роль Edge шлюзу. Edge шлюз координує дані зі світильників та направляє іх на сервер і отримує команди із серверу та направляє світильникам.

Світильники об’єднуються у групи дротовим зв’язком, адже мережа вуличного освітлення це дозволяє. Об’єднання виконується за допомогою

Ethernet кабелю. Такий вибір обумовлений високою пропускною здатністю, що дозволяє додавати нові IoT пристрої до вуличного освітлення без проблем із передачею даних. Таке з'єднання дозволяє передавати потокове відео з камер відеоспостереження, що можуть бути розташовані на стовпах вуличного освітлення, без затримок у передачі даних до інших IoT пристроїв чи самого світильнику.

Шлюз виконує аутентифікацію світильнику або IoT пристрою на світильнику використовуючи комбінацію сертифікату X.509 та ідентифікатору пристрою. Верифікувавши пристрій шлюз може агрегувати дані для зменшення навантаження на сервіси IoT. Шлюз далі передає дані до IoT платформи через брокер повідомлень за протоколом MQTT чи AMQP.

Брокер повідомлень - це окремий сервіс, який може горизонтально розширюватися для забезпечення опрацювання всіх запитів і команд від і до IoT пристроїв. Завдання брокеру повідомлень це направлення подій від датчиків на світильниках до передбачених ендпоінтів платформи та відправлення команд від платформи до світильників чи інших пристроїв, розташованих на світильниках.

Протоколи передачі даних мають передбачати ідентифікатор світильника незалежно від типу пристрою, що відправляє або приймає команди. Ідентифікатор світильнику має бути присутнім у командах до інших пристроїв, розташованих на світильнику для того, щоб спростити маршрутизацію команди (рис. 3.2.1).

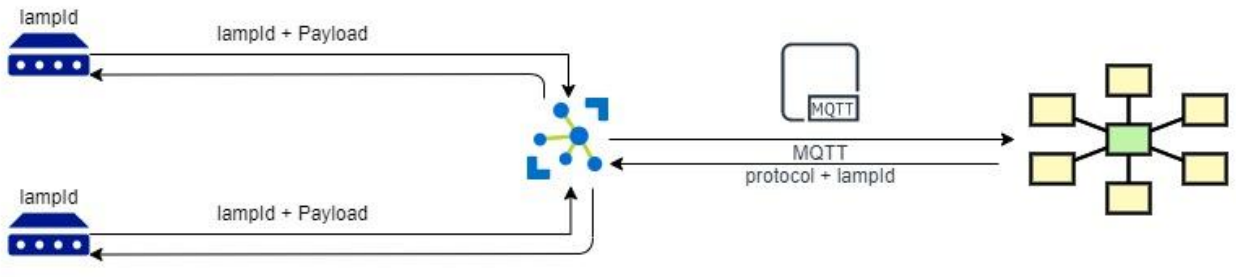


Рисунок 3.2.1. Протокол передачі даних між світильниками і платформою IoT

3.2.2 Платформа Розумного Освітлення

IoT Платформа передбачає в собі 3 основні складові, а саме: Обробники подій, Обробники дій та виконавчі сервіси для обробки даних.

Сервіси обробки даних виконують статистичні алгоритми обробки даних для передбачення інтенсивності руху або інші аналітичні функції. З метою подальшого розширення та розвитку системи аналітичні сервіси варто відокремлювати від додатку у платформі таким чином, щоб можна було додавати нові аналітичні сервіси без загрози змін у існуючих. Отже, видно, що аналітичні сервіси, наприклад, сервіс аналізу інтенсивності руху варто відокремлювати в окремий мікросервіс, зв'язок із яким відбувається через обмін повідомленнями або через HTTP протокол.

Сервіс аналізу даних є виділеним мікросервісом, який має свою базу знань, що поповнюється кожний день та надає API для отримання прогнозованих даних як показано на рисунку 3.2.2.

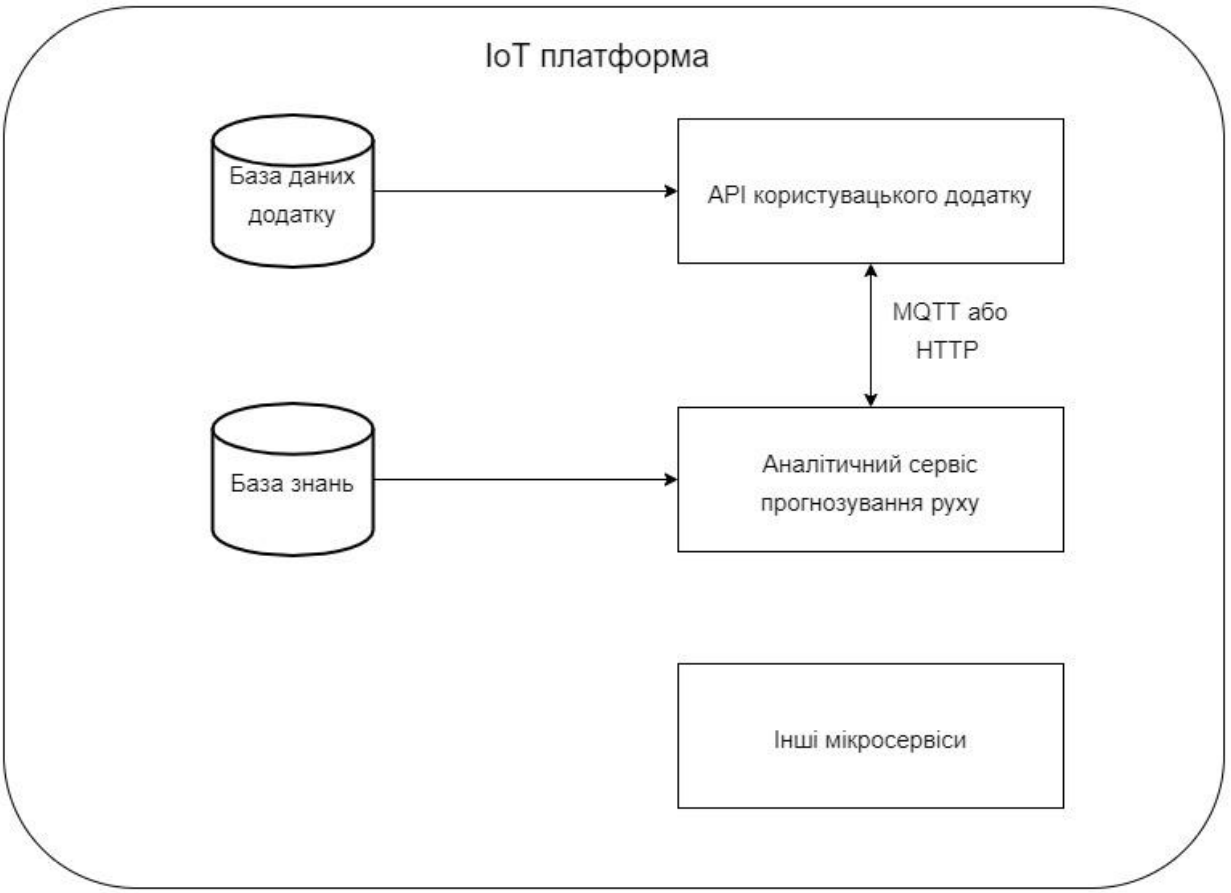


Рисунок 3.2.2. Аналітичний сервіс як виділений мікросервіс у системі

Сервіс аналізу даних періодично у зазначені інтервали часу виконує аналіз статистичних даних для визначення формули залежності інтенсивності руху від зовнішніх чинників.

Оскільки кожний день додаються нові дані до бази знань, то вибірка збільшується з часом і визначена формула інтенсивності руху уточнюється. Сервіс виконує регресійно-кореляційний аналіз для кожної групи світильників окремо за такими параметрами:

- сила дощу;
- хмарність;
- температура;
- сила вітру;
- день тижня;
- місяць;
- час доби.

За цими параметрами відбувається пошук коефіцієнтів для рівняння, результат якого є середній інтервал часу між рухомими об’єктами.

Рівняння має вигляд:

$$I = a \cdot x_1 + b \cdot x_2 + c \cdot x_3 + d \cdot x_4 + e \cdot x_5 + f \cdot x_6, \qquad (3.1)$$

де x_1 - час доби, x_2 - день тижня, x_3 - сила дощу, x_4 - хмарність, x_5 - температура, x_6 - сила вітру, тощо.

Проте, оскільки день тижня, місяць та час доби є параметрами якісними, а не кількісними, то потрібно перевести їх у кількісний вид. Оскільки обидва параметри мають скінченний набір значень, то достатньо для кожного можливого значення кожного параметру ввести свою змінну, яка буде приймати значення 0 або 1 в залежності від значення якісного параметру. Таким чином для параметру дня тижня отримуються 7 нових змінних по одній до кожного

дня відповідно. І параметр для тижню замінюється на нові 7 змінних. наприклад

$$x_2 = a \cdot x_{11} + b \cdot x_{12} + c \cdot x_{13} + d \cdot x_{14} + e \cdot x_1 + f \cdot x_{16} + g \cdot x_{17}, \quad (3.2)$$

де при тому, що тиждень починається з понеділка і закінчується неділею для середи рівняння буде мати вигляд

$$x_2 = a \cdot 0 + b \cdot 0 + c \cdot 1 + d \cdot 0 + e \cdot 0 + f \cdot 0 + g \cdot 0, \quad (3.3)$$

аналогічно змінюються параметри місяця та часу доби. Рівняння набуває великої кількості параметрів, а точніше 47. Як було зазначено вище, для аналізу моделі кількість вимірювань повинна бути приблизно в 10 разів більша за кількість параметрів рівняння [11]. Отже для достовірного визначення параметрів даного рівняння знадобиться близько 470 вимірювань. Це значить, що задовільний результат може бути отриманий через 470 днів після початку вимірювань.

3.2.3 Мікросервіс аналізу руху

Мікросервіс аналізу руху повинен мати спрощений хоча і менш ефективний спосіб прогнозування інтенсивності руху поки кількість вибірки не буде достатньою для регресійного аналізу.

Мікросервіс виконує обчислення середнього значення інтенсивності руху, розрахованого для кожної групи світильників у певний проміжок часу. Тобто мікросервіс також надає інформацію по середньостатистичному руху об’єктів, порахованому на основі отриманих даних. Обрахування середніх значень не потребує великої кількості даних для достовірного значення. Таким чином мікросервіс надає API для наступних дій:

- визначення середнього інтервалу руху об’єктів для певної групи світильників у певний час (час доби, день тижня, місяць) за відповідних погодних умов методом регресійного аналізу;
- визначення середнього інтервалу руху об’єктів для певної групи світильників у певний час (час доби, день тижня, місяць) методом оцінки середнього значення.

Сервіс бере на себе відповідальність за збір потрібних параметрів для короткострокового (один день) прогнозування, при тому дозволяє передавати параметри окремо, якщо потрібно. API виглядає наступним чином:

- int GetInterval(DateTime time);
- int GetInterval(DateTime time, byte rainIntensity, byte cloudy, byte windIntensity, byte temperature).

Цей API автоматично визначає який метод прогнозування використовувати. Також надає окремі API для обох методів окремо: GetRegressionCalculatedInterval(...) та GetAverageCalculatedInterval(...).

Як згадувалось раніше для аналізу потрібні дані. Дані зберігаються на накопичуються у відповідній базі знань. База знань (БЗ; англ. Knowledge base, KB) - база даних, що містить правила виведення і інформацію про людський досвід і знання в деякій предметній області (ISO / IEC / IEEE 24765-2010 [14], ISO / IEC 2382-1: 1993 [14]). У самонавчающихся системах база знань також містить інформацію, що є результатом рішення попередніх завдань [14].

Сучасні бази знань працюють спільно з системами пошуку та вилучення інформації. Для цього потрібна деяка модель класифікації понять і певний формат представлення знань. Ієрархічний спосіб представлення в базі знань набору понять і їх відносин називається онтологією.

Онтологію деякої області знань разом з відомостями про властивості конкретних об'єктів часто називають «базою знань». Разом з тим повноцінні бази знань [15] (на відміну від звичайної бази даних) містять в собі не тільки фактичну інформацію, а й правила виведення, що дозволяють робити

автоматичні умовиводи про вже наявні або нововведені факти і тим самим виробляти семантичну (осмислену) обробку інформації.

Область наук про штучний інтелект, що вивчає бази знань і методи роботи зі знаннями, називається інженерією знань [16].

Дані з датчиків маршрутизуються брокером повідомлень. Брокер повідомлень маршрутизує дані з датчиків до додатку, додаток зберігає дані у своїй базі даних та реплікує їх шляхом черги повідомлень до бази знань аналітичного мікросервісу. Реплікуються тільки потрібні дані, наприклад, дані з датчиків руху або агреговані дані з камер відеоспостереження, а такі дані, як якість повітря або вологість, не реплікуються.

Дані з датчиків надходять щогодинно для можливості аналізу на погодинній основі.

Інша інформація, така як погодні умови, отримується через API інтерфейс сторонніх сервісів. Ця інформація також оновлюється на погодинній основі. Якщо інформація не може бути отримана на погодинній основі, то погодинний аналіз ускладнюється і доведеться або обирати інші більші інтервали часу, або змінювати склад параметрів.

3.2.4 Мікросервіс контролю світильників

Мікросервіс контролю світильників відповідає за команди, що надсилаються до світильників. Коли закінчується чергова обробка алгоритму та уточнення параметрів мікросервісом аналізу нові дані надаються мікросервісу контролю світильників для подальшої оробки та передачі даних світильникам.

Базуючись на параметрах рівняння мікросервіс визначає інтенсивність руху для кожної групи світильників на погодинній основі. Після цього мікросервіс визначає інтервали затримки T для розумних світильників базуючись на алгоритмі, зазначеному вище. Нові значення надсилаються світильникам для зміни їх схеми освітлення.

Також мікросервіс є відповідальним за відправку команд у ручному режимі. Мікросервіс пов’язаний з іншими, наприклад, з користувацьким додатком через чергу повідомлень. Мікросервіс реалізує обробник повідомлень для ручних команд до світильників.

Команда - це запит на виконання системою дії, що змінює стан системи. Команди повинні оброблятися лише один раз.

Оскільки команди є імперативами, вони, як правило, називаються дієсловом у наказовому способі (наприклад, "створити" або "оновити"), і вони можуть включати сукупний тип, наприклад ChangeStreetLightIntensityCommand. На відміну від події, команда не є фактом із минулого; це лише прохання, і таким чином може бути відхилена.

Команди можуть надходити з інтерфейсу користувача в результаті ініціювання користувачем запиту або менеджер процесів, коли менеджер процесу спрямовує агрегат на виконання дії.

Важливою характеристикою команди є те, що вона повинна бути оброблена лише один раз одним одержувачем. Це пояснюється тим, що команда - це одна дія або транзакція, яку потрібно виконати у програмі. Наприклад, одна і та ж команда зміни освітлення не повинна оброблятися більше одного разу. Це важлива різниця між командами та подіями. Події можуть оброблятися кілька разів, оскільки багато систем або мікросервісів можуть бути зацікавлені у події.

Крім того, важливо, щоб команда була оброблена лише один раз, якщо команда не є ідемпотентною. Команда є ідемпотентною, якщо її можна виконати кілька разів, не змінюючи результат, або через характер команди, або через те, як система обробляє команди.

Це хороша практика - робити ваші команди та оновлення ідемпотентними, коли це має сенс відповідно до ділових правил та інваріантів певного домену. Наприклад, використовуючи той самий приклад, якщо з якоїсь причини (логіка повтору спроби, злому тощо) одна і та ж команда ChangeStreetLightState досягає системи кілька разів вона (система) повинна мати можливість це ідентифікувати та гарантувати, що світильники не

змінюють свій стан декілька разів. Для цього потрібно приєднати якусь ідентифікацію до операцій та визначити, чи була команда або оновлення вже оброблено [17].

Команда для ручного увімкнення або вимкнення світильнику має наступний вигляд:

```
ChangeStreetLightStateCommand {
  Guid groupId;
  Guid[] lampIds;
  byte intensity;
  DateTime initiationTime;
  Guid commandId;
}
```

Як видно, команда несе в собі інформацію про світильники, які потрібно змінити та інформацію про групу світильників, в якій знаходяться світильники, які потрібно змінити. *GroupId* є обов'язковим параметром, оскільки визначає отримувача повідомлення. *lampIds* є додатковим параметром, який визначає набір конкретних світильників, стан яких потрібно змінити. Для зміни одного окремого світильника у масиві *lampIds* є один елемент, який представляє собою ідентифікатор світильника. Для зміни стану усіх світильників у групі *lampIds* не вказується, тобто залишається порожнім. Тобто groupId використовується двічі - для визначення отримувача повідомлення та для визначення набору світильників, які знаходяться в цій групі.

Значення параметру intensity визначає яскравість, яку мають прийняти світильники. Значення параметру є відсотковим і варіюється від 0 до 100. Крок інтенсивності пропоновано обирати не менше 5%, адже менші зміни не є помітними для ока оточуючих і не мають впливу. Проте сам мікросервіс не накладає ніяких обмежень до кроку зміни інтенсивності (окрім меж).

Значення параметру *initiationTime* визначає день і час коли має відбутися зміна освітлення. Значення параметру є у форматі DateTime - структурі, яка може зберігати дані про час. Мікросервіс перевіряє, чи не є час у минулому і,

якщо є, то генерує відповідне повідомлення про помилку. Параметр *initiationTime* є опціональним. При значенні цього параметру рівному 0 команда виконується одразу. Якщо *initiationTime* не 0 і визначає час у майбутньому, то зміна освітленості планується у планувальнику подій.

Параметр *commandId* є обов'язковим і визначає команду унікально. Це дозволяє робити 2 речі:

1. Уникнути дублювання команд, наприклад, при поганому мережевому з'єднанні. При поганому з'єднанні додаток відправляє команду і може не отримати підтвердження про отримання команди отримувачем. В такому разі відправник надсилає команду повторно із тим самим *commandId* . Проте, бувають випадки коли команда дійшла до отримувача, проте підтвердження надходження не повернулось відправнику. В такому випадку, якщо відправник спробує надіслати команду і та дійде, то відбудеться дублювання команди. *commandId* цьому запобігає - отримувач при отриманні команди перевіряє оброблені команди із таким *commandId* . І якщо знаходить, то значить команда була дубльована і вже була виконана.
2. Мікросервіс може асинхронно відправляти відповіді (події) на виконані команди. І для кореляції команд і відповідей може використовуватися *commandId.* Тобто у відповіді (події) присутній параметр *commandId,* який визначає команду, до якої належить ця подія.

Функціонал мікросервісу дещо розширюється за допомогою обробника команди, яка інкапсулює в собі колекцію команд на зміну стану групи.

Команда ChangeStreetLifghtsStateCommand виглядає наступним чином:

```
ChangeStreetLightsStateCommand {
  ChangeStreetLightStateCommand [] commands;
}
```

Цей обробник разом із командою дозволяють маніпулювати багатьма світильниками у різних районах одночасно. Наприклад, для увімкнення всіх

підконтрольних світильників *ChangeStreetLightsStateCommand* у параметрі *commands* містить колекцію всіх зареєстрованих груп світильників, кожна з яких виглядає як

```
ChangeStreetLightStateCommand {
  Guid groupId = streetLightUniqueGroupId;
  Guid[] lampIds = null;
  byte intensity = 100;
}
```

Команди після верифікації надсилаються до черги повідомлень, яка відправить команди до світильників, а точніше, до Edge хабу, який вже розподілить команди по своїх світильниках.

3.2.5 IoT Хаб

IoT Хаб - це керована служба, розміщена в хмарі, яка виконує роль центру обміну повідомленнями для двонаправленого зв'язку між додатком IoT та пристроями, якими він керує. Наприклад, можна використовувати Azure IoT Hub для створення рішень IoT із надійним та безпечним зв’язком між мільйонами пристроїв IoT та серверним рішенням, розміщеним у хмарі. Можна підключити практично будь-який пристрій до IoT Хабу.

IoT Хаб підтримує зв'язок як із пристрою в хмару, так і з хмари на пристрій. IoT Hub підтримує декілька моделей обміну повідомленнями, таких як телеметрія між пристроями та хмарою, завантаження файлів із пристроїв та методи запиту-відповіді для управління пристроями з хмари. Моніторинг IoT Хабу допомагає підтримувати роботу платформи, відстежуючи такі події, як створення пристрою, несправності пристроїв та підключення пристроїв.

Можливості IoT Хабу допомагають створювати масштабовані повнофункціональні рішення IoT, такі як управління промисловим обладнанням, що використовується у виробництві, відстеження цінних активів у галузі охорони здоров’я, моніторинг використання офісних будівель та керування вуличним освітленням [12].

IoT Хаб масштабується до мільйонів одночасно підключених пристроїв і мільйонів подій в секунду, щоб підтримувати IoT навантаження від пристроїв. Azure IoT Хаб пропонує два рівні, базовий та стандартний, які відрізняються кількістю функцій, які вони підтримують. Якщо рішення IoT базується на зборі даних з пристроїв та їх централізованому аналізі, то базовий рівень підходить. Якщо планується використовувати більш досконалі конфігурації для віддаленого управління пристроями IoT або розподілу деяких робочих навантажень на самі пристрої, тоді слід розглянути стандартний рівень.

Кожен рівень IoT Хабу доступний у трьох розмірах залежно від того, скільки даних вони можуть обробити за будь-який день. Ці розміри ідентифікуються як 1, 2 та 3. Наприклад, кожен блок концентратора IoT першого рівня може обробляти 400 тисяч повідомлень на день, тоді як блок 3 рівня може обробляти 300 мільйонів.

Стандартний рівень IoT Хабу забезпечує всі функції та є необхідним для будь-яких рішень IoT, які хочуть використовувати двонаправлені можливості зв'язку. Базовий рівень забезпечує підмножину функцій і призначений для рішень IoT, які потребують лише односпрямованого зв'язку від пристроїв до хмари. Обидва рівні пропонують однакові функції захисту та аутентифікації.

Для IoT Хабу можна вибрати лише один тип видання на одному рівні. Наприклад, можна створити IoT Хаб з кількома блоками S1, але не можна об’єднувати блоки з різних видань, таких як S1 та S2 [13].

IoT Хаб надає захищений канал зв'язку для передачі даних пристроями.

- Аутентифікація на кожен пристрій дає можливість кожному пристрою надійно під’єднуватися до Хабу IoT і безпечно керувати кожним пристроєм.
- Повний контроль над доступом до пристрою та можете керувати з’єднаннями на рівні кожного пристрою.
- Служба забезпечення пристроями IoT Хабу автоматично надає пристрої правильному Хабу IoT при першому завантаженні пристрою.

- Кілька типів аутентифікації підтримують різноманітні можливості пристрою:
    - Аутентифікація на основі токенів SAS для швидкого початку роботи з рішенням IoT.
    - Індивідуальна аутентифікація сертифіката X.509 для безпечної стандартної аутентифікаці.
    - Аутентифікація X.509 CA для простої стандартної реєстраці.

Вбудована функція маршрутизації повідомлень надає гнучкість налаштування автоматичної роздачі повідомлень на основі правил:

- Використовується маршрутизація повідомлень, щоб контролювати, куди Хаб надсилає телеметрію пристрою.
- Немає додаткових витрат на маршрутизацію повідомлень до кількох кінцевих точок.
- Правила маршрутизації без коду замінюють спеціальний код диспетчера повідомлень.

Можна інтегрувати IoT Hub з іншими службами платформи або хостингу для створення повних наскрізних рішень. Наприклад:

- Azure Event Grid, щоб організація могла швидко реагувати на критичні події надійним, масштабованим та безпечним способом.
- Програми Azure Logic для автоматизації бізнес-процесів.
- Машинне навчання моделей ШІ до свого рішення.
- Запуск аналітичних обчислень у режимі реального часу на потоковому передаванні даних із ваших пристроїв.

Можна керувати пристроями, підключеними до IoT Хабу, за допомогою безлічі вбудованих функціональних можливостей.

- Зберігати, синхронізовувати та отримувати метадані та інформацію про стан усіх пристроїв.
- Встановлювати стан для кожного пристрою або на основі загальних характеристик пристроїв.

- Автоматично реагувати на повідомлення стану пристрою за допомогою інтеграції маршрутизації повідомлень.

3.2.6 Додаток

Додатки - це сукупність служб та компонентів, які є унікальними для рішення IoT. Додатки IoT зазвичай мають:

- Об’єднання сервісів для обчислень, зберігання та кінцевих точок події в поєднанні з унікальною бізнес-логікою додатків.
- Процеси подій для отримання та обробки вхідних подій з пристроїв.
- Робочі процеси для надсилання команд на пристрої чи до інших процеси.

Додаток IoT забезпечує взаємодію з кінцевими користувачами або те, як він або клієнти взаємодіють з даними, зібраними з пристроїв IoT. Це може бути мобільний додаток, веб-сайт, настільний додаток або навіть пасивна система, з якою ніхто не взаємодіє безпосередньо. Їх побудова може бути складною і трудомісткою.

Додаток IoT є важливою складовою архітектури IoT і саме тут реалізується фактична цінність IoT рішення. Розгортання обладнання та збір даних з датчиків не має сенсу, якщо ці дані не представлені корисними способами та не вирішують реальних проблем для клієнтів. У контексті розумного освітлення IoT додаток є панеллю управління розумними світильниками а також інформаційною системою всього розумного освітлення.

Додаток є програмою, з якою взаємодіють користувачі. Додаток є веб-додатком, який користувачі бачать як веб-сайт. Веб-додаток надає користувацький інтерфейс для управління освітленням, моніторингу стану світильників та для отримання аналітичної і статистичної інформації, в тому числі, по показниках з датчиків.

Веб-додаток є точкою входу для користувачів або точкою входу із зовнішнього світу. Він виконаний у форматі мікросервісу із власною базою даних, що абстрагує веб-додаток від аналітичних мікросервісів та іншої інфраструктури. Бекенд веб-сайту спілкується з іншими мікросервісами через чергу повідомлень, що передбачає асинхронний зв’язок, який лише у сукупності із реплікацією даних збільшує швидкодію системи.

Команди керування світильниками, отримані від користувачів, надсилаються до мікросервісу керування освітленням через чергу повідомлень, як було зазначено вище. Решта дій стосовно керування освітленням виконується мікросервісом. Коли зміна світильників відбулася то мікросервіс керування світильниками повідомляє додаток про успішність операції відправляючи відповідну подію через чергу повідомлень.

Подія дуже схожа на відповідну команду:

```
StreetLightStateChangedEvent {
  Guid groupId;
  Guid[] lampIds;
  byte intensity;
  DateTime eventTime;
  Guid commandId;
}
```

Ця подія означає, що світильники у певній групі змінили свою інтенсивність освітлення (ввімкнулися, вимкнулися або змінили інтенсивність освітлення). Подія також має поле *eventTime,* яке визначає час, коли подія сталася. Це потрібно через те, зо черга повідомлень є асинхронною і повідомлення про подію може бути доставлене через деякий проміжок часу. Тож для зберігання коректної інформації у повідомленні про подію має бути визначений час, коли подія сталася. Параметр *commandId* визначає команду, до якої належить ця відповідь. Тобто за цим параметром можно співвідносити команди та події і визначати які команди були виконані, а які ще ні.

Додаток несе в собі складову візуалізації інформаційної системи. Через веб-сайт користувачі отримують інформацію про стан системи, агреговані зібрані дані, візуалізацію та представлення аналітичних результатів системи.

Результати роботи аналітичних мікросервісів копіюються до бази даних додатку. Це абстрагує мікросервіси один від одного. Додаток може агрегувати дані з бази даних у довільному вигляді і навіть змінювати формат збереження даних без впливу на інші мікросервіси [18].

Додаток надає інформацію відповідно до кожного світильнику окремо, наприклад: кількість пішоходів, що пройшли мимо за годину, температуру, якість повітря, вологість, рівень зв'язку зі світильником, рівень GSM сигналу у світильнику, кількість часу, що світильник працював і час неактивності або неповної активності. Ця інформація також може бути представлена для групи світильників, проте дещо агрегована. Так, наприклад, для групи світильників може надаватися найнижча і найвища температура, помічена світильниками групи, найменший, найбільший та середній час роботи світильників на повній потужності та навпаки у режимі спокою (Рис.3.2.6.1).

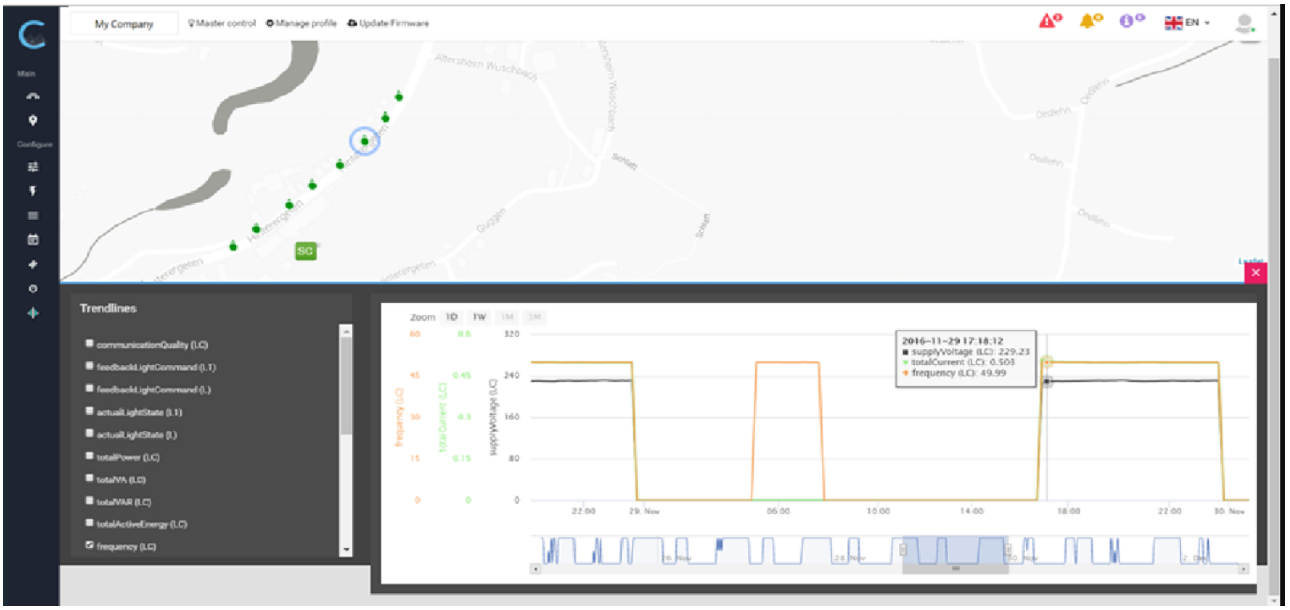


Рис. 3.2.6.1. Інформаційна функція додатку

Через додаток користувачі можуть переглядати стан світильників та управляти ними. Додаток надає інтерфейс для цього. Для виконання перегляду

інформації про стан світильників чи їх оточення додаток використовує власну базу даних, оскільки більшість інформації зі світильників та супутніх датчиків зберігається у базі даних додатку. Додаток відображає ліхтарі на мапі, це найзручніший спосіб відображення для сприйняття інформації користувачами. Додаток виводить інформацію про світильники туди ж на мапу, таку інформацію як: стан світильнику. Також додаток візуально групує світильники, наприклад об’єднуючи світильники лініями певного кольору відповідно до групи. Коли користувач викликає деяку дію над світильником чи світильниками (наприклад, використовуючи кнопки керування, так звані CTA - call to action) додаток попередньо перевіряє правильність дії, генерує команду і через брокер повідомлень відправляє команду до відповідного мікросервісу (керуваня світильниками).

Також додаток відповідний за повідомлення користувачів про певні події, наприклад про те, що світильник вийшов з ладу і потрібно відправити працівників на перевірку і заміну світильнику або повідомлення про те, що відсутнє з’єднання зі світильником і потрібно виявити та усунути причину. (Рис.3.2.6.2)

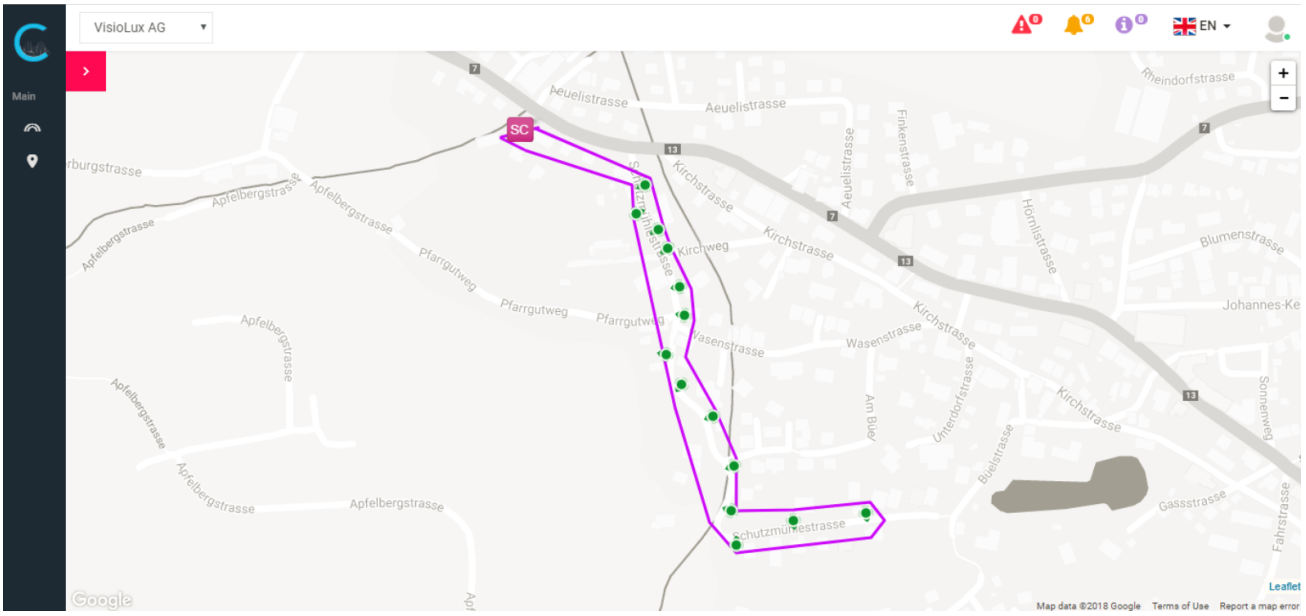


Рис. 3.2.6.2. Відображення стану світильників

3.3 Висновки

Було запропоновано архітектурне рішення для впровадженні інформаційно-аналітичної IoT системи розумного освітлення, яка передбачає горизонтальне функціональне та фізичне розширення.

Перелічено складові мікросервіси, які займають центральне місце IoT платформи. Центральними мікросервісами є мікросервіс аналізу руху та мікросервіс керування світильниками.

Запропоновано алгоритм взаємодії системи із зовнішнім середовищем (датчиками, користувачами) та визначено місце для алгоритмів обробки даних з метою пошуку закономірностей інтенсивності руху та прогнозування на визначені терміни.

РОЗДІЛ 4. РЕАЛІЗАЦІЯ І ВПРОВАДЖЕННЯ ІНФОРМАЦІЙНО-АНАЛІТИЧНОЇ СИСТЕМИ НА ОСНОВІ ASP.NET WEB API ТА МІКРОСЕРВІСІВ

4.1 Загальний опис застосунку

Застосунок “Smart Street Lighting” є прикладом написання IoT-застосунку, тобто такого, який дозволяє користувачам отримувати інформацію із пристроїв і керувати пристроями у режимі реального часу, застосунок має більш менш зручний інтерфейс користувача і прийнятну швидкодію. Застосунок буде складатися з двох основних сторінок: дашборд та сторінка реєстрації/авторизації. На дашборді авторизований користувач може бачити мапу зі розташованими світильниками, переглядати статус світильника чи групи світильників, керувати групою світильників, переглядати аналітичну інформацію по роботі світильників і статистику по зібраним даним. Неавторизований користувач направляється на сторінку авторизації, де він може ввести свої ідентифікаційні дані.

Користувач може бути зареєстрований після попереднього запрошення адміністратором. Адміністратор надсилає запрошення користувачеві на електронну адресу. Лист має посилання на сторінку авторизації та унікальний ідентифікатор, який засвідчує особу, яка перейшла на сторінку авторизації. Після реєстрації користувач автоматично авторизується, після авторизації користувач перенаправляється на головну сторінку.

Стосовно інтерфейсу користувача, він простий. Головна сторінка складається з трьох вікон: колонка меню зліва і центральна частина розділена горизонтально на верхню і нижню. Центральна компонента це мапа із зображенням всіх розташованих світильників, нижня частина вікна є зрізаною аналітичною секцією. Секція Меню знаходиться зліва, у секції відображені посилання на інші сторінки застосунку.

Центральна секція має шапку (хедер). У шапці справа відображається інформація про користувача: повне ім’я та іконка аватару. Також у підсекції із профілем користувача справа є кнопка виходу. У хедері зліва відображена інформація про організацію, до якої належить користувач. По центру у хедері розташовані іконки повідомлень: повідомлень-застережень та повідомлень про помилки. При натисканні на відповідну іконку застосунок виводить всі відповідні повідомлення.

Середня секція є секцією головної інформації про стан системи освітлення і є найбільшою з трьох, адже є головним компонентом застосунку.. Внизу секції є роздільна грань, яка відділяє мапу зі світильниками від аналітичної та статистичної секції. Третя секція являється інформаційною, в ній можна побачити інформацію про окремий світильник, статистичні дані зібрані зі світильника, агреговану інформацію для світильнику. Також можна вибрати групу світильників і бачити аналітичну і статистичну інформацію для всієї групи. Секція надає графіки для візуалізації даних: графік інтенсивності освітлення світильником протягом доби, графік інтенсивності руху повз світильник протягом доби, додаткові графіки, що візуалізують дані з інших додаткових датчиків, встановлених на світильниках.

Сторінка авторизації має два поля: поле для логіну та поле для пароля. І кнопка для входу. Якщо логін не знайдено чи пароль не співпадає, то користувач інформується про це шляхом отримання відповідного повідомлення. Для реєстрації нового користувача в системі адміністратор повинен перейти на сторінку користувачів та натиснути кнопку Запрошення нового користувача. Сторінка запрошення нового користувача складається з чотирьох полів. Всі поля досить стандартні: поле електронної адреси— має бути унікальним у системі, є ключем/ідентифікатором користувача, поля для імені та прізвища, поле організації, до якої належить майбутній користувач, поле підрозділу у організації. Після натискання на кнопку Запросити генерується лист та відправляється на вказану адресу.

Новий користувач переходить за посиланням та опиняється на сторінці реєстрації, яка містить 2 поля для паролю та підтвердження паролю. Якщо паролі не задовольняють умовам паролів, чи паролі не співпадаютьто користувач отримує відповідне повідомлення.

4.2 Проектування бази даних застосунку

Для початку потрібно створити модель пристрою та датчику, оскільки пристрої та їх датчики є основними компонентами системи. Модель створюється із основних полів, що потрібні системі для роботи із світильниками та датчиками.

Модель для датчику має наступні поля: Id - унікальний ідентифікатор датчику у вигляді Guid; AddedOn - часова позначка коли датчик було додано до системи; AddedBy - поле ідентифікатора користувача, який додав датчик до системи; TypeId - тип датчику, наприклад, 1 - датчик руху, 2 - датчик температури, 3 - датчик вологості, тощо; LastCommunicatedOn - часова позначка, коли датчик останній раз надавав інформацію. Так можна відслідкувати коли датчик вийшов з ладу; HealthStatus - статус датчику, що визначає чи є проблеми із датчиком; DeviceId - пристрій, до якого під’єднано датчик (датчики у системі не встановлюються окремо, а є модулями розумних пристроїв, наприклад, розумних світильників). Модель зображено на Рис.4.2.1

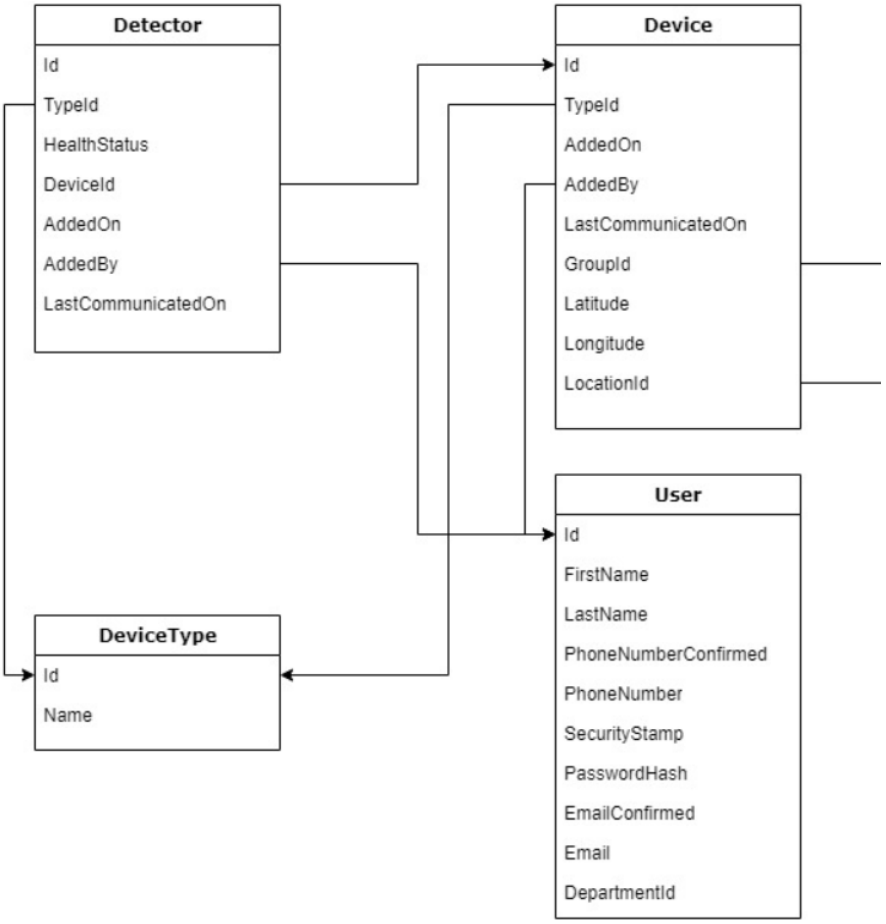


Рисунок 4.2.1. Модель датчику у БД

Модель для пристрою має наступні поля: Id - унікальний ідентифікатор пристрою (світильнику) у вигляді Guid; AddedOn - часова позначка коли пристрій було додано до системи; AddedBy - поле ідентифікатора користувача, який додав пристрій до системи; TypeId - тип пристрою, наприклад, 11 - світильник, тощо; LastCommunicatedOn - часова позначка, коли пристрій останній раз надавав інформацію. Так можна відслідкувати коли пристрій вийшов з ладу; HealthStatus - статус пристрою, що визначає чи є проблеми із пристроєм; GroupId - група пристроїв, до якої відноситься пристрій (пристрої у системі встановлюються у групах і група є обов’язковим атрибутом пристрою). Latitude та Longitude - поля, що визначають географічне розташування світильнику. LocationId - ідентифікатор локації, де розміщений пристрій, локація несе інформацію про місто, вулицю, на якій розташовано пристрій, тощо. Модель зображено на Рис.4.2.2

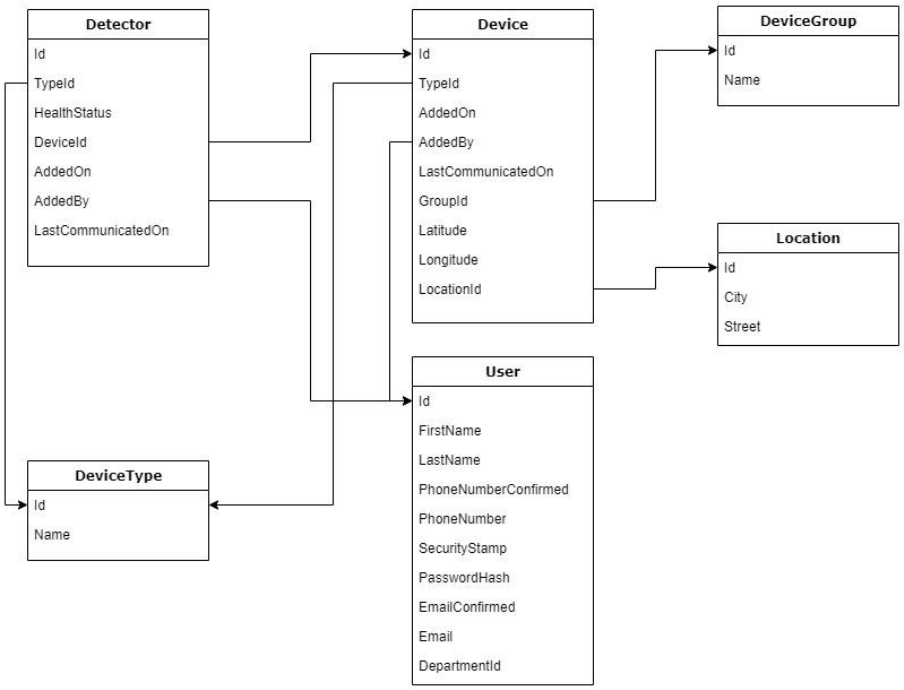


Рисунок 4.2.2. Модель пристрою у БД

Базова модель користувача створюється при підключенні Identity Framework [30]. Одразу після підключення та увімкнення міграцій першою міграцією створиться модель користувача та додаткові моделі, що використовуються при авторизації (рис. 4.2.3).

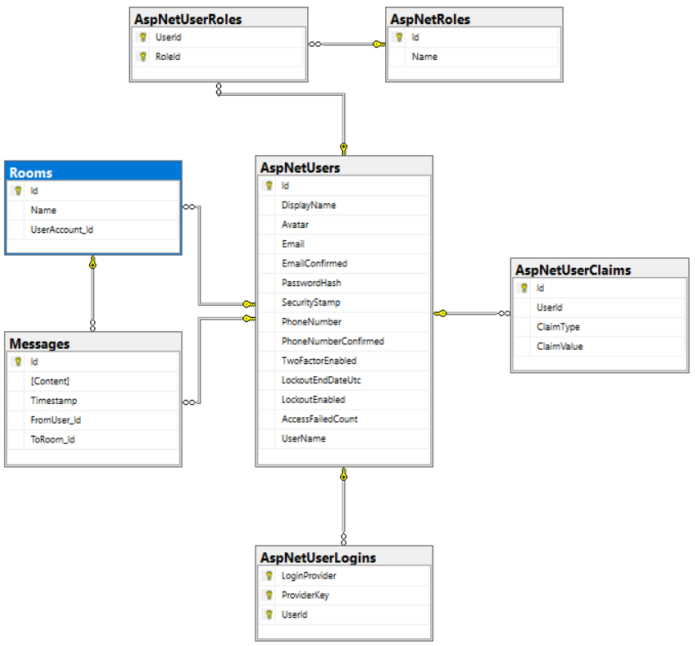


Рисунок 4.2.3. Схема бази даних користувачів

Спершу проектується модель користувача для збереження його персональних даних. DisplayName – ім’я користувача, що буде відображене на сайті, Avatar – посилання на картинку з аватаром користувача, Email – електронна адреса користувача (використовується для авторизації користувача а також у випадках відновлення паролю та інших, якщо потрібно зв'язатися з користувачем), UserName – системне ім'я користувача, що той обирає власноруч, також слугує для ідентифікації користувача, може використовуватись замість email під час авторизації, EmailConfirmed – поле, що зберігає дані чи електронна адреса підтверджена, чи ні.

Під час реєстрації нового користувача вводиться нова електронна адреса, проте система не може гарантувати, що той, хто ввів адресу дійсно має до неї доступ. Для підтвердження власності електронної адреси на неї надсилається повідомлення підтвердження із посиланням на підтвердження адреси. Якщо користувач перейшов по посиланню, то адреса підтверджена і в полі EmailConfirmed ставиться помітка, що адреса підтверджена. В базовому випадку це поле є бітовим (аналог до типу Boolean у C#) і може приймати два значення: 0 – не підтверджена, 1 – підтверджена.

Для збереження пароля використовується його хеш, тобто зберігається не пароль, а хеш. Цьому є принаймні одна вагома причина: якщо дані з бази даних розкриються, то ті, хто отримав дані не отримають паролей від аккаунтів користувачів – лише хеш. Отже для збереження паролю використовується поле PasswordHash, що є типу VARCHAR(max).

Поле PhoneNumber зберігає номер телефону користувача, якщо є, а поле PhoneNumberConfirmed зберігає інформацію про підтвердження номеру телефону. Підтвердження номеру телефону відбувається таким же чином, як і підтвердження електронної адреси, лише повідомлення підтвердження надсилається через смс.

TwoFactorEnabled – поле, що визначає чи активована двофакторна аутентифікація для користувача. Двофакторна аутентифікація (також відома як 2FA) - це тип або підмножина багатофакторної аутентифікації. Це метод

підтвердження ідентичності користувачів за допомогою комбінації двох різних факторів: 1) те, що вони знають; 2) щось у них є, або 3) те, що вони є. Хорошим прикладом двофакторної аутентифікації є отримання грошей з банкомату; тільки правильне поєднання банківської картки (що користувач має) і PIN-код (що користувач знає) дозволяє здійснити транзакцію. Двоетапна перевірка або двоступенева автентифікація - це метод підтвердження заявленої ідентичності користувача, використовуючи те, що він знає (пароль), і другий чинник: що він має або чим є. Прикладом другого етапу є користувач, який повторює щось, що було надіслано йому через третій механізм. Або другий крок може бути шестизначним числом, створеним програмою, яка є спільною для користувача та системи аутентифікації

Поля LockoutEndDateUtc, LockoutEnabled, AccessFailedCount – є допоміжними полями, що допомагають зконфігурувати авторизацію, кількість помилково введених паролей до того, як користувач буде заблокований, час блокування користувача, чи така функція включена для даного користувача.

Доступ до деяких функцій застосунку, наприклад: панель адміністратора, має бути обмеженим. Для обмеження доступу (авторизації) використовуються ролі. Таблиця AspNetRoles зберігає всі ролі у форматі Id – ідентифікатор ролі, Name – назва ролі. Користувач може мати декілька ролей, а до одної ролі можуть належати багато користувачів. Отже для зв’язку користувачів з ролями використовується проміжна таблиця (Junction Table), що називається AspNetUserRoles. Ця таблиця лише зв’язує користувачів і ролі, отже має поля UserId – вказує на ідентифікатор користувача, RoleId – вказує на ідентифікатор ролі. Отже один запис у таблиці відображає приналежність одного користувача до однієї ролі.

4.3 Архітектурне рішення застосунку

Інформаційно-аналітична IoT система розумного освітлення є складною системою, яка має горизонтально та вертикально розширюватись у

майбутньому. Отже система являє собою мікросервісне рішення, ключовими мікросервісами якого є:

- Мікросервіс прогнозування інтенсивності руху.
- Мікросервіс керування світильниками.
- Користувацький застосунок.

Передбачається подальший розвиток в тому числі користувацького застосунку із додаванням графічних елементів для аналітики даних, тощо. З огляду на такі запити розумним буде використання фреймворку React або Angular для UI застосунку. Таким чином кожен мікросервіс системи може бути написаний на будь-якій мові програмування, яка найкраще задовольняє потреби домену, а користувацький застосунок буде написаний на ASP.NET Core WEB API та React/Angular фреймворках.

Отже, розглянемо use-case діаграму (рис. 4.3.1) для того, щоб зрозуміти основні компоненти, що нам потрібні.

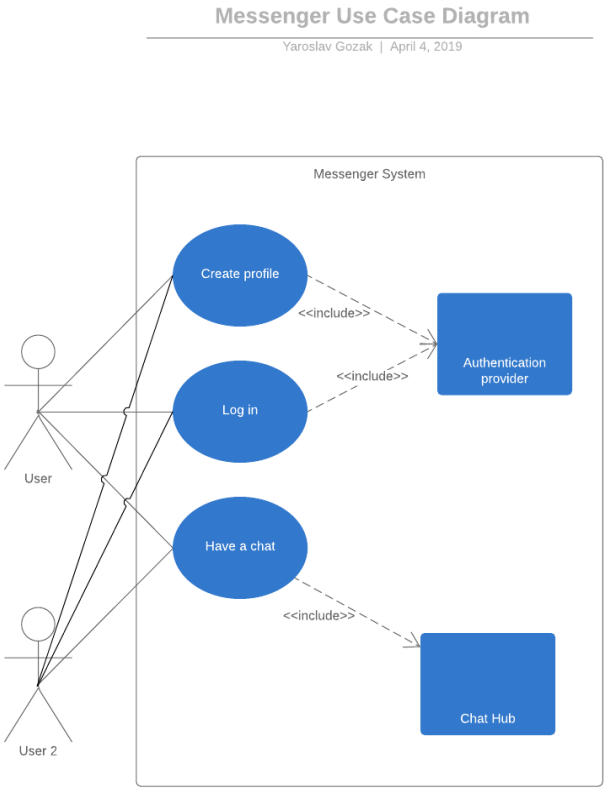


Рисунок 4.3.1. – Use-Case діаграма

При переході на сторінку веб-застосунку система визначає чи авторизований користувач за допомогою атрибута [Authorize] та сервісу авторизації, що на малюнку зображений як Authentication Provider. Якщо користувач не авторизований, то він буде перенаправлений на сторінку Login, де зможе ввести свої дані.

Коли користувач введе дані і натисне кнопку «Увійти», то сторінка надішле запит до Authentication Provider, який перевірить коректність введених даних. Якщо спроба успішна, то користувач буде направлений на сторінку Home (на малюнку Information-Analytical system). Якщо спроба неуспішна, то користувач повернеться на сторінку авторизації, де додатково отримає повідомлення, що описує помилку.

Якщо користувач ще не має профілю, то зможе перейти на сторінку реєстрації отримавши попередньо відповідний електронний лист. Ввівши необхідні дані та натиснувши кнопку «Зареєструватись» користувач зініціює запит сторінки на сервер через AccountController до Authentication Provider (що представлений AppSigninManager).

Якщо всі дані правильні, то користувач буде авторизований і направлений на сторінку HomePage (на малюнку Information-Analytical system). Якщо під час реєстрації користувача виникне помилка, то користувач буде заново направлений на сторінку реєстрації де додатково побачить повідомлення, що описує помилку.

На головній сторінці користувач може переглядати схему розміщення всіх розумних світильників на мапі, обирати групу світильників або певний світильник для перегляду інформації про них, бачити кількість повідомлень про помилку або попередження, переходити на сторінку повідомлень, переглядати коротку аналітичну інформацію по світильниках та датчиках, що там встановлені.

При переході на головну сторінку запускається JavaScript-застосунок, який веде себе як SPA застосунок. Надалі всі дії із сервісом (отримання

інформації по світильниках, управління світильниками) відбувається через веб застосунок, всі запити надходять до бекенд-застосунку, написаного на C# і надалі з нього же надсилаються сповіщення JavaScript застосунку у браузері клієнта.

Технічно при заході на сторінку інформаційно-аналітичної IoT системи застосунок має верифікувати користувача чи він авторизований, чи ні. У разі якщо користувач не авторизований, то він має бути направлений на сторінку авторизації.

Отже, одним з перших компонентів буде сервіс авторизації і аутентифікації. Оскільки авторизація і аутентифікація є задачами, із якими стикається майже кожен веб застосунок і оскільки шляхи вирішення авторизації і аутентифікації є досить стандартними, то платформа ASP.NET Core представляє свій сервіс авторизації і аутентифікації користувачів Identity Framework**.** Identity Framework надає досить широкі можливості роботи з користувачами, розглянемо деякі з них:

Підтримка авторизації і аутентифікації користувачів через кукі, в тому числі claims. Claims – це дані про користувача у кукі, які зберігають в собі такі дані як хто видав ці дані, яка роль користувача, чи, навіть, яке ім’я користувача. Насправді claims є найбільш вільною у використанні системою ідентифікації користувачів. Взагалі, користувача можна аутентифікувати за ім’ям, що має бути унікальним, токеном чи за клеймами. Проте, авторизація може бути проведена також за допомогою ролей, які в свою чергу є заздалегідь створеними системою клеймами.

Для маніпулювання аккаунтами користувачів є адміністратор. Звісно, адміністратором не може бути кожен користувач, проте нам потрібно декілька адміністраторів, які зможуть це робити. І, звісно, нам потрібні самі користувачі. Тож доступ до контролеру створення і видалення користувачів повинні мати лише адміністратори. Для цього чудово підходять ролі. Користувачі, що є адміністраторами наділяються роллю Адміністратор і контролер менеджменту користувачів конфігурується бути доступним лише користувачам із роллю

Адміністратор. Це чудове і елегантне рішення схожих задач. Із Identity Framework доступ до контролера чи окремого його метода (екшену) може бути сконфігурований через C# атрибути. На контролер, що має бути обмежений у доступі навішується атрибут [Authorize()] і туди передається список ролей чи/та імен, яким дозволений доступ до контролера.

Також Identity Framework надає зручний спосіб керування користувачами, а саме: авторизація та створення нового аккаунту. Створення та авторизація відбувається через методи класу AppSigninManager, а управління аккаунтом відбувається через сервіс AppUserManager, які є вбудованими. Зазвичай для отримання можливості використання цих сервісів потрібно успадкувати ці класи і використовувати, власне, новостворені класи.

Вся інформація про користувачів, ролі та інші допоміжні дані зберігаються у базі даних, яка створюється автоматично за допомогою Entity Framework та механізму міграцій. Для того, щоб створити перший раз базу даних для Identity Framework потрібно виконати команди Allow-Migrations та New-Migration і потім Commit-Changes у консольному менеджері. Цей вбудований функціонал Entity Framework та Identity Framework дозволяють не створювати всю базу даних вручну.

Застосунок має наступні сторінки: дашборд, аналітика, користувачі, сторінка реєстрації та авторизації. Відповідно UI-застосунок матиме щонайменше такі компоненти як DashboardComponent, UsersComponent, AnalyticsComponent, RegistrationComponent, AuthenticationComponent. C# застосунок має щонайменше наступні контролери: DashboardController, AnalyticsController, AuthenticationController, оскільки сторінки авторизації і реєстрації мають дуже схожу та у більшості спільну логіку. До речі, нормальним є відношення екшенів до контролерів як 5 до 1.

Усі операції із маніпуляції даних будуть обробляти вище зазначені сервіси: AppUserManager та AppSigninManager.

Для додавання функціоналу прогресивного веб-застосунку визначено файл *manifest.json у* кореневій папці проекту, що визначає поведінку та вигляд

застосунку, коли той використовується як нативний застосунок. Також визначено service-worker.js, який містить в собі логіку кешування даних та обробки запитів.

Також, в основному файлі скрипту визначений фрагмент коду, що додає сервісного робітника до документу вікна браузеру, який надалі буде виконувати всі визначені функції. Сервісний робітник сконфігурований при першому завантаженні сайту завантажити три основні сторінки і файли, що ті використовують.

4.4 Інструкція користувача

Застосунок є великим, проте основний його функціонал зосереджений на головній сторінці. Отже, коли користувач вперше спробує перейти на головну сторінку застосунку, то будучи неавторизованим користувач буде направлений на сторінку авторизації

На сторінці авторизації (рис. 4.4.1) є три стандартні поля для введення даних користувача:

1. Ім'я користувача (Username).
2. Пароль (Password).
3. Маркер чи запам'ятати цього користувача на даному пристрої (Remember me).

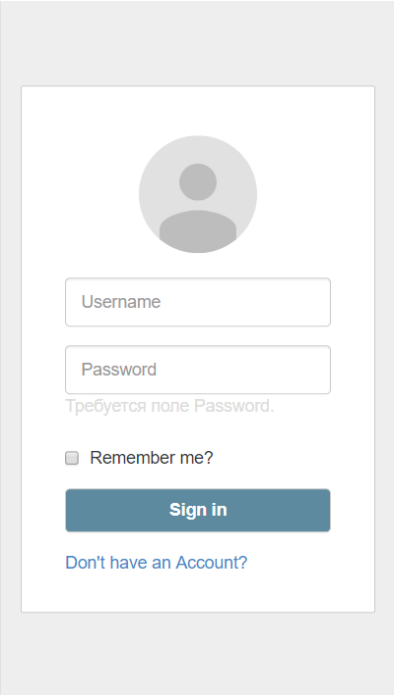


Рисунок 4.4.1. Сторінка авторизації. Мобільна версія

Дані валідуються, щоб ім'я користувача не повторювалось у системі – було унікальним і щоб пароль задовольняв політиці паролей застосунку. Як зазначалося вище, політика паролей конфігурується через Identity Framework.

Після вводу даних користувач натискає на кнопку Sign in і, якщо дані введені правильно, користувач успішно авторизований і направляється на основну сторінку. Якщо користувач не має власного профілю у застосунку, то може перейти за посиланням «Don’t have an Account?» і перейти на сторінку запиту доступу.

На сторінці запиту доступу(рис. 4.4.2) присутнє єдине поле - поле Email, застосунок має отримати мінімальну достатню кількість інформації про користувача для продовження роботи з ним.

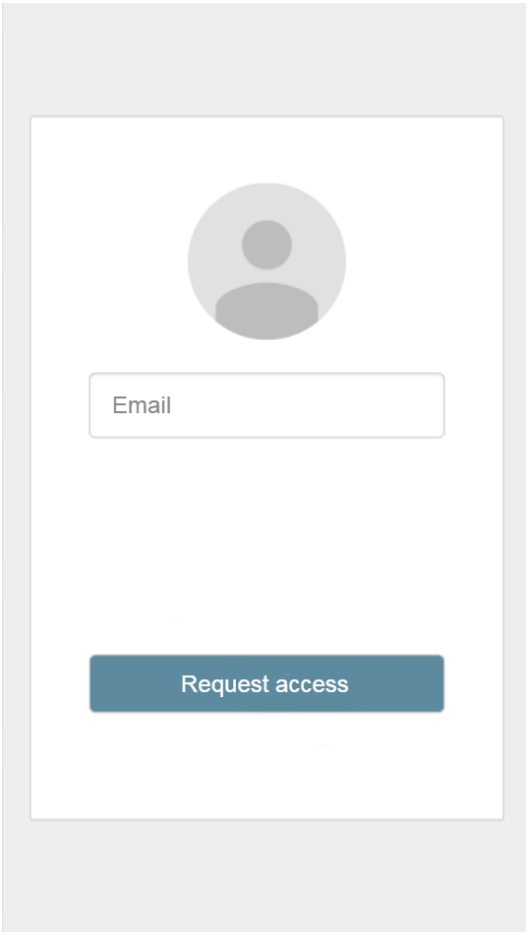


Рисунок 4.4.2. Сторінка запиту доступу. Мобільна версія

Отже, ввівши електронну адресу, користувач натискає кнопку «Запросити доступ». Якщо електронна адреса заповнена правильно, то користувача буде повідомлено про те, що його запит успішно надісланий і адміністрація розпочне роботу по запиту.

На головній сторінці в першу чергу буде відображено список світильників і зображено мапу з їх розташуванням(рис. 4.4.3), до яких користувач має доступ, та буде підсвічено назву міста, світильники або пристрої якого користувач зараз бачить. Натискання на секцію меню зліва направляє користувача на відповідну сторінку. На головній сторінці користувач може натиснути на світильник на мапі, тоді в нижній секції відобразиться інформація про обраний світильник. Також користувач може натиснути на ланку між розумними світильниками на мапі і цим обрати всю групу світильників, до якої належать ті, ланку між якими було натиснуто.

У хедері основної секції показано місто, мака якого зображена. Для того, щоб обрати інше місто потрібно натиснути на назву міста і у дропдауні обрати потрібне місто. Якщо у системі наявне лише одне місто, то дропдаун не відкриється і ніяких дій не відбудеться.

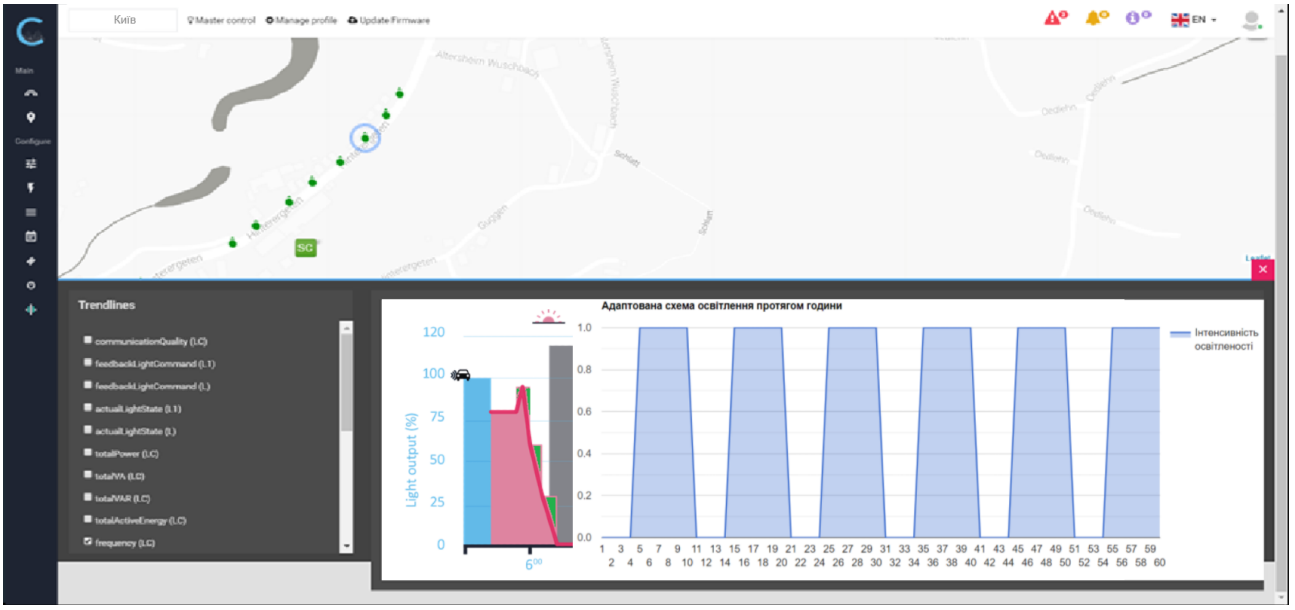


Рисунок 4.4.3. Головна сторінка. Десктопна версія

При натисканні на мапу вона збільшується і займає всю площу екрану. Це полегшує навігацію і дозволяє бачити більш велику площу міста. У режимі збільшеної мапи світильники виділяються групами при наведенні на них курсором. Кліком миші вибирається виділена група світильників і карта маштабується і центрується відносно групи таким чином, щоб вся група займала всю мапу. Теперь можна обирати окремі світильники наводячи на них курсором та натискаючи на них (рис. 4.4.4).

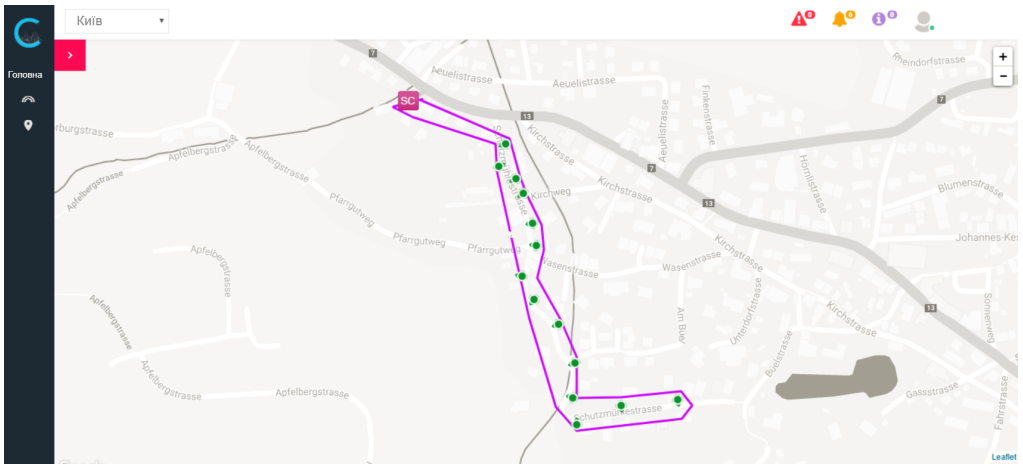


Рисунок 4.4.4. Мапа розумних світильників. Десктопна версія

Аналітична секція спроектована бути гнучкою у відображенні, що дозволяє у налаштуваннях системи обирати вигляд та те, яку інформацію буде надано у аналітичній секції. Аналітична секція може містити багато графіків тому при вертикальному скролі сторінки будуть з’являтися приховані графіки, а мапа із розміщенням світильників зникне.

Також у шапці сторінки є іконки із повідомленнями про помилки системи або попередження. Біля кожної іконки також присутня цифра, що означає кількість повідомлень відповідного типу. При натисканні на відповідну іконку відкривається нове вікно зверху над основною сторінкою і у вікні зображено список повідомлень. При натисканні на повідомлення користувач перенаправляється на сторінку із детальною інформацією додатково до повідомлення. Повідомлення про помилку, наприклад, може інформувати користувача про те, що деякий світильник вийшов з ладу або що пропало з’єднання із групою світильників. Додаткова інформація є часом повідомлення, тощо (рис. 4.4.5).

У аналітичній секції зліва розміщена історія команд та подій, що були надіслані до та від обраного світильника. Обираючи певну команду можна побачити історію всіх команд цього типу, час їх відправки (або отримання, якщо це подія)

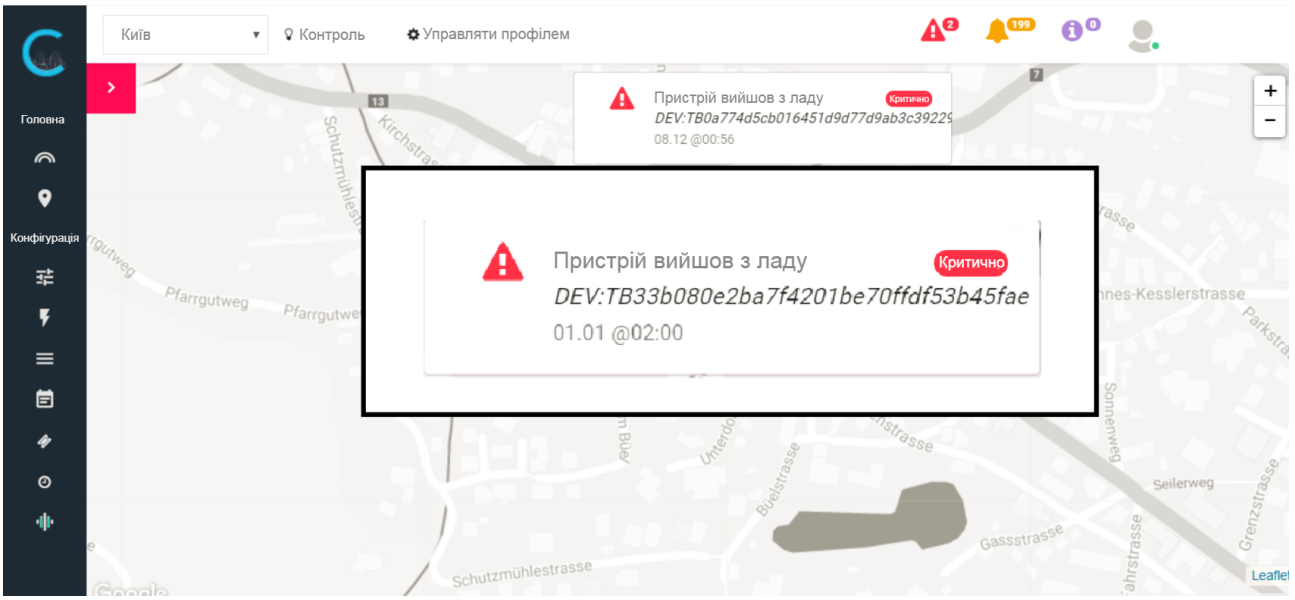


Рисунок 4.4.5. Повідомлення про помилку. Десктопна версія

4.5 Висновки

На основі ASP.NET MVC було реалізовано застосунок для доступу до інформаційно-аналітичної IoT системи розумного освітлення. Для досягнення цих цілей було використано технологію .NET для веб-сокетів SignalR. Наявність більшості основного функціоналу вже у готовому вигляді, в тому числі: реалізація технології веб-сокетів, фреймворк авторизації та аутентифікації користувачів – значно прискорює розробку основного функціоналу, базового каркасу застосунку. Завдяки наявності автоматизованої системи генерації таблиць у базі даних для збереження інформації про користувача, а також автоматичного мапінгу у моделі користувача можна не витрачати час на розробку функціоналу, що є поширеним серед різних застосунків, тобто можна використовувати уніфікований шаблон, що підходить більшості проектів.

ВИСНОВКИ

Було окреслено проблему регулювання режимів роботи розумних світильників у системах міста таку, яка описує деяку властивість руху об'єктів в певні періоди роботи освітлення, яка спричиняє часті зміни в освітленості середовища, що негативно можуть впливати на соціо-психологічний стан людей поблизу та на економічну складову системи.

Сучасні системи розумного освітлення, які присутні на світовому ринку не оцінюють інтенсивність руху для налаштування режимів роботи світильників.

Запропоновано метод передбачення інтенсивності руху з метою поправки схеми освітлення для оптимізації роботи світильників у нічний час доби, яка передбачає собою балансування між енергоефективністю за рахунок зниження освітлення у години неактивності та за малою інтенсивністю руху та кількістю перемикань світильнику з метою зменшення дратівливого впливу на оточуючих. Для передбачення інтенсивності руху було запропоновано використання середнього значення показника інтенсивності за малої вибірки даних та використання регресійно-кореляційного аналізу за достатньої кількості даних. Було запропоновано алгоритм підбору часу очікування світильнику після проходження об’єкту таким чином, щоб час очікування був максимально ефективним з точки зору кількості перемикань світильнику, коли таке можливо.

Запропоновано архітектурне рішення для впровадженні інформаційно-аналітичної IoT системи розумного освітлення, яка передбачає горизонтальне функціональне та фізичне розширення. Було запропоновано складові мікросервіси, які займають центральне місце IoT платформи. Центральними мікросервісами є мікросервіс аналізу руху та мікросервіс керування світильниками. Було запропоновано алгоритм взаємодії системи із зовнішнім середовищем (датчиками, користувачами) та визначено місце для алгоритмів обробки даних з метою пошуку закономірностей інтенсивності руху та прогнозування на визначені терміни.

Реалізовано концепт застосунку для доступу до інформаційно-аналітичної IoT системи розумного освітлення на основі ASP.NET WEB API. Для досягнення цих цілей було використано технологію .NET для веб-сокетів SignalR. Наявність більшості основного функціоналу вже у готовому вигляді, в тому числі: реалізація технології веб-сокетів, фреймворк авторизації та аутентифікації користувачів – значно прискорює розробку основного функціоналу, базового каркасу застосунку. Завдяки наявності автоматизованої системи генерації таблиць у базі даних для збереження інформації про користувача, а також автоматичного мапінгу у моделі користувача можна не витрачати час на розробку функціоналу, що є поширеним серед різних застосунків, тобто можна використовувати уніфікований шаблон, що підходить більшості проектів.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Автономна система вуличного освітлення: пат. 95106 Україна, опубл. 10.12.2014

2. Алюмінієва паркова опора освітлення ROSA SAL-55, URL: https://stolb.com.ua/ru/alumnva-parkova-opora-osvtlennya-rosa-sal-55-dz/ (дата звернення 15.06.2020)

3. Баришнікова Н.Ю. Обучение языку запросов на основе использования Базы Знаний шаблонов программного кода (рус.) // сборник трудов : сборник. — 2018.

4. Гозак Я.Д., Гладка М.В.Залежність яскравості освітлення від інтенсивності трафіку за допомогою IoT // Матеріали 86 Міжнародної наукової конференції молодих учених, аспірантів і студентів Наукові здобутки молоді – вирішенню проблем харчування людства у XXI столітті, Київ: НУХТ, 2–3 квітня 2020 р., Ч.1.

5. Гужков С., Поліщук А., Туркин А. Мережі вуличного освітлення : Напівпровідникова світлотехніка, 2019, С.42-46

6. ДБН В.2.5-28:2018 Природне і штучне освітлення. URL: https://dbn.co.ua/load/normativy/dbn/dbn_v_2_5_28/1-1-0-1188 (дата звернення 15.06.2020)

7. Капітальний ремонт мереж зовнішнього освітлення у м.Києві, URL: https://prozorro.gov.ua/tender/UA-2019-08-16-000390-b?fbclid=IwAR1CCYhc9Y4tleLqC22hrewWk9f8CQpCQghgOk4a3NbsPo2o08KONm6QlQU (дата звернення 15.06.2020)

8. Контроллер Schreder LUCO P7 для дистанционной системы управления освещением Schreder Owlet, URL:

https://stolb.com.ua/ru/kontroler-schreder-luco-p7-dlya-distantsiynoyi-sistemi-keruvannya-osvitlennyam-schreder-owlet/ (дата звернення 15.06.2020)

9. Кравець О. С. Статистика: Навчальний посібник / О. С. Кравець. – О.: Пальміра, 2008. С. 266

10. Ройко Ю., Санько Я. Прогнозування зміни інтенсивності руху в урбанізованому просторі: зб. матеріалів Транспортне планування міст та керування дорожнім рухом / Львів, 2019, С.3-5

11. Светодиодный уличный консольный светильник LED 20Вт 5000K 1800 Lm LEDEX, URL: https://nazarenko.dp.ua/p699595824-svetodiodnyj-ulichnyj-konsolnyj.html (дата звернення 15.06.2020)

12. Чистов Д.В. Новые информационные технологии в образовании: Применение технологий "1С" для развития компетенций цифровой экономики (рус.) // научно-практическая конференция : Сборник. — 2018.

13. Що таке LoRaWAN : веб-сайт. URL: https://habr.com/ru/company/nag/blog/371067/ (дата звернення: 12.01.2021)

14. A. Gil-de-Castro, Antonio Moreno-Munoz, Anders Larsson, Math Bollen LED street lighting: A power quality comparison among street light technologies, Lighting Research and Technology, 2017, №45, С.710-728, URL: https://www.researchgate.net/publication/275451135_LED_street_lighting_A_power_quality_comparison_among_street_light_technologies (дата звернення 15.05.2020)

15. A Smart City Adaptive Lighting System: зб. матеріалів доп. учасн. конференції “Third International Conference on Fog and Mobile Edge Computing (FMEC)”, 2018 p

16. Azure IoT Hub : веб-сайт. URL: https://docs.microsoft.com/en-us/azure/iot-hub/about-iot-hub

17. Azure IoT Hub Scaling : веб-сайт. URL: https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-scaling

18. BQLZR DC 12V~24V 8A Automatic LED PIR Motion Sensor Switch Light Lighting: веб-сайт. URL: https://www.amazon.com/BQLZR-12V-24V-Automatic-Lighting-Staircase/dp/B00EQ20C6K

19. C.A. Cheng, C.H. Chang, T.Y. Ching, F.L. Yang. Design and implementation of a Single-Stage Driver for supplying an LED Streetlighting Module with Power Factor Corrections. : IEEE Transactions on Power Electronics, Vol 30, 2015.

20. Castro, M., Jara, A. J., and Skarmeta. Smart lighting solutions for smart cities : зб. матеріалів доп. учасн. 27th IEEE International conference on Advanced information networking and applications workshops (WAINA) : Варшава, 2015. С. 1374-1379

21. Cesar de la Torre, Bill Wagner .NET Microservices: Architecture for Containerized .NET Applications, 5th edition: 2021, URL: https://docs.microsoft.com/en-us/dotnet/architecture/microservices

22. Daisuke Namihira, Tomohiro Nakajima IoT Platform for Comprehensive Coordination of Iot Systems, URL: https://www.fujitsu.com/global/documents/about/resources/publications/fstj/archives/vol53-1/paper05.pdf (дата звернення 15.06.2020)

23. IoT solution architecture - Azure Example Scenarios : веб-сайт. URL: https://docs.microsoft.com/en-us/azure/architecture/example-scenario/iot/devices-platform-application

24. Jagadeesh Y.M, Akilesh S Intelligent Street Lights, ResearchGate, 2015, URL: https://www.researchgate.net/publication/284226969_Intelligent_Street_Lights - (дата звернення 15.06.2020)

25. Jorge Mario Avella Ruiz, Teófilo Miguel de Souza, José Luz Silveira. A comparative analysis between fluorescent and LED illumination for improve energy

efficiency at IPBEN building, XI - Latin-American Congress on electricity generation and transmission, 2015, №1. T.24

26. Joseph F., Hair Jr. Multivariate Data Analysis / Joseph F., Hair Jr. // New Deli, 2016. P.266-272

27. LED lighting efficacy: Status and directionsEfficacité de l'éclairage LED : état de l'art et directions - Comptes Rendus Physique - Milan, 2018. P. 134-145 [https://www.sciencedirect.com/science/article/pii/S1631070517300932]

28. Moreno I., Avendano-Alejo M Modeling LED street lighting, Applied Optics - Standford, 2014. №53

29. Nenad Petrovic, Milorad Tosic, Valentina Nejkovic, Nenad Milosevic Formalizing Device Coordination in IoT Systems: The SCOR Case Study, YuInfo 2019, 2019, URL: https://www.researchgate.net/publication/332900856_Formalizing_Device_Coordination_in_IoT_Systems_The_SCOR_Case_Study (дата звернення 15.06.2020)

30. Palumbo M.L. Architettura Produttiva: Principi di progettazione ecologica / Palumbo M.L // Maggioli S.p.A - 2019 - №3 - C.11-17.

31. Prince Yadav. Smart Street Lighting System, April 2020, URL: https://www.researchgate.net/publication/348096383_Smart_Street_Lighting_System

32. R. Barraza Garcia, G. Velazquez Angulo, J. R. Gonzalez, E. F. Tavizon, J.I. Huertas Cardozo. LED Street Lighting as a Strategy for Climate Change Mitigation at Local Government Level. : зб. матеріалів Proceedings of IEEE 2014 Global Humanitarian Technology, Варшава, 2014, С. 345-349

33. R. Zambare, P. Pawar. Street light controller with GSM technology : International Journal of Engineering Applied Sciences and Technology. Baku, 2020, Vol. 4, Issue 10, P.110-112

34. Smart Street Lighting System : веб-сайт. URL: https://www.citintelly.com/intelligent-street-lighting-products/street-lighting-control-

system/ (дата перегляду: 03.03.2021)

35. Solar Light Malaysia - Alpha 1080X Solar Powered Street Light. URL: https://www.solarlight-mart.com/alpha_1080x_street_light (дата звернення 15.06.2020)

36. Systems and software engineering : Vocabulary / Christopher J.D. - Bratislava, 2017. P.87-88

37. The fundamental IoT architecture : веб-сайт. URL: https://www.losant.com/blog/the-fundamental-iot-architecture#:~:text=The%20Components%20of%20an%20IoT,foundation%20of%20every%20IoT%20solution (дата звернення: 20.01.2021)

38. Warner Lonsing. Smart Street Light, Conference: ASK.the.Conference, Warsaw, 2020. URL: https://www.researchgate.net/publication/335889236_Smart_Street_Light (дата перегляду: 05.03.2021)

ДОДАТКИ

Додаток А

Стаття на тему “Система освітлення у розумному місті”

СИСТЕМА ОСВІТЛЕННЯ У РОЗУМНОМУ МІСТІ

Гозак Ярослав Дмитрович

студент Факультету Інформаційних Технологій

Київського національного університету імені Тараса Шевченка

Україна, м. Київ

Анотація

Система розумного освітлення набирає популярність в останні роки і стає все більш розповсюдженою, оскільки таке освітлення підвищує комфорт та якість життя, заощаджує ресурси та кошти. Система розумного освітлення є найбільш готовою платформою для впровадження наступних IoT рішень завдяки своїй мережі у всьому місті.

Було досліджено концепцію розумного освітлення як базової платформи для впровадження подальших IoT проектів. Було сформульовано концепцію та спроектовано архітектурне рішення системи розумного освітлення, що передбачає модернізацію у майбутньому з ціллю використання надбудованих пристроїв IoT для покращення функціонування системи.

У цій роботі представлено рішення для системи розумного освітлення із використанням модулю управління як абстрактного шару між центром управління та розумними світильниками. Спроектовано систему розумного освітлення з застосування технологій Інтернету речей, хмарних обчислень та туманних обчислень, а також алгоритмів машинного навчання.

Ключові слова: розумне освітлення, IoT, хмарні обчислення, туманні обчислення, розширювана система.

SMART STREET LIGHTING IN SMART CITY

Yaroslav Gozak
student of Faculty Of Informational Technologies
Taras Shevchenko National University of Kyiv
Kyiv, Ukraine

Abstract

The smart lighting system is gaining popularity in recent years and is becoming more common, as such lighting increases comfort and quality of life, saves resources and money. The smart lighting system is the most ready platform for the implementation of subsequent IoT solutions thanks to its network throughout the city.

The concept of smart lighting as a basic platform for the implementation of further IoT projects was explored. The concept and architectural solution of the smart lighting system were formulated, which envisages future modernization in order to use add-on IoT devices to improve the functioning of the system.

This paper presents a solution for a smart lighting system using the control module as an abstract layer between the control center and smart lights. A system of intelligent lighting with the use of Internet of Things technologies, cloud computing and fog computing, as well as machine learning algorithms has been designed.

Keywords: smart street light, IoT, cloud computing, fog computing, extensible system

Вступ

Система розумного освітлення набирає популярність в останні роки і стає все більш розповсюдженою, оскільки таке освітлення підвищує комфорт та якість життя, заощаджує ресурси та кошти. Система розумного освітлення є найбільш готовою платформою для впровадження наступних IoT рішень завдяки своїй мережі у всьому місті [23]. Ця мережа, якщо правильно спроектована, дозволяє надбудовувати IoT прилади і легко інтегрувати із існуючою системою.

Виявлення присутності та контроль вуличного освітлення

Блок виявлення (блок датчиків) складається з інфрачервоних датчиків і пасивних інфрачервоних датчиків з кожної сторони дороги, які визначають інтенсивність руху і передають дані в мікроконтролер для обробки. Ці пристрої виявлення розміщуються по обидва боки дороги для індивідуального освітлення з кожного боку. Пристрої виявлення спілкуються один з одним під час роботи. Коли рухомий об’єкт виявляється на дорозі, світильник вмикається на повну потужність, а також інформує сусідні світильники, щоб простір навколо рухомого об’єкту міг бути підсвічений. Це досягається завдяки зв’язку між датчиками. Таке динамічне управління вуличними ліхтарями зберігає витрати електричної енергії, яка є найнеобхіднішою в сьогоднішній енергетичній кризі.

У цьому динамічному керуванні блок виявлення відіграє важливу роль для моніторингу інтенсивності руху та виконання відповідних дій через мікроконтролер.

З блоку сенсорів дані передаються в мікроконтролер, в якому виконується динамічне управління вуличним світлом, схема роботи якого наведена на Рис.2.3.

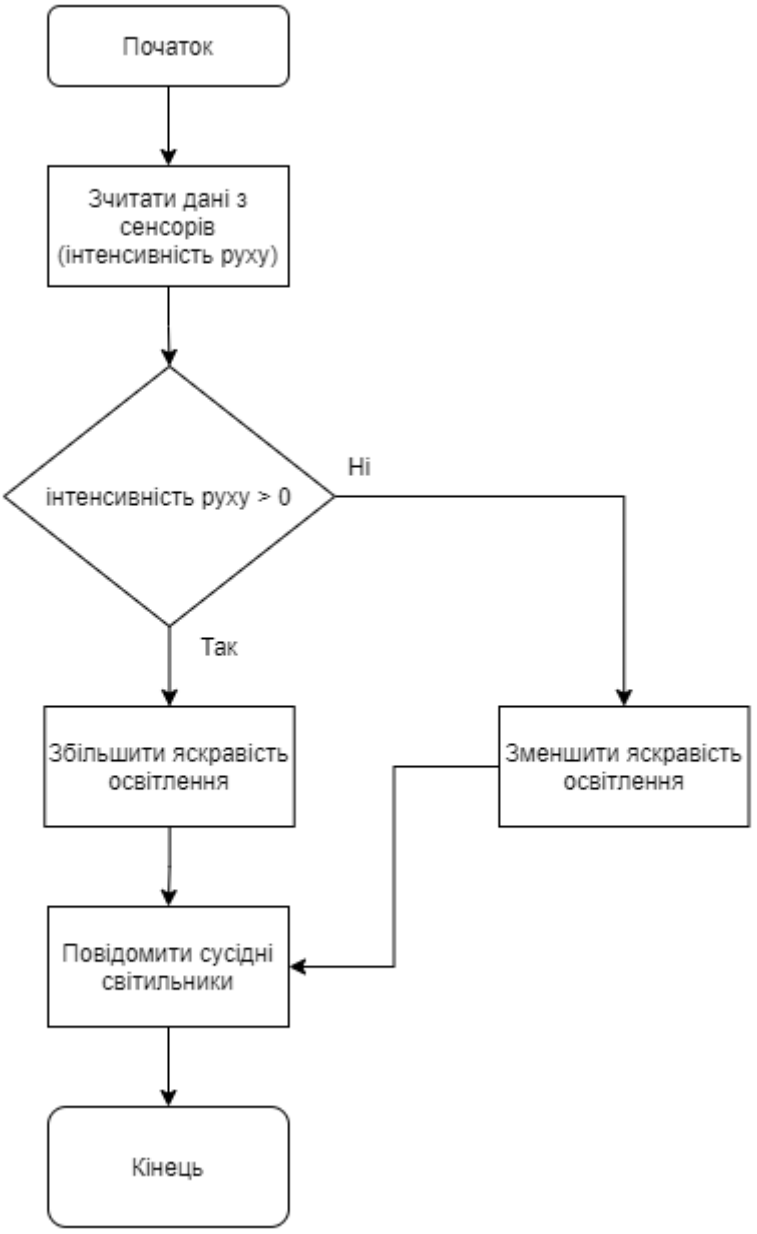


Рисунок 2.3 - Схема роботи динамічного вуличного освітлення

Схема роботи описує контроль яскравості в системі динамічного освітлення вулиць. Спочатку всі значення щодо інтенсивності руху зчитуються через модуль виявлення [24]. Спочатку контроллер перевіряє, чи є рух в зоні видимості, потім він виконує одну з наступних дій.

1. Якщо є якийсь рух, то світлова установка збільшує інтенсивність освітлення та передає інформацію сусіднім світильникам про наявність руху, щоб ті також збільшили інтенсивність освітлення.

2. Інакше зменшує інтенсивність освітлення до 25% та надсилає команду сусіднім світильникам, щоб також понизили яскравість.

Тому система забезпечує високу яскравість для більш інтенсивного руху та меншу яскравість для менш інтенсивного руху. У цю систему також включено зустрічне управління вуличними ліхтарями. Впровадження системи розумного освітлення в місті знижує споживання електроенергії до 60% і тим самим зменшує кількість викидів CO2 [25].

Покращення системи розумного освітлення із додаванням камер спостереження

Увімкнення фіксованої кількості світильників для освітлення простору попереду рухомого об'єкту має певні недоліки, пов'язані із регулюванням кількості світильників, що вмикаються та змінною швидкістю об'єкта. Об'єкти у полі дії світильників можуть рухатися із дуже різною швидкістю, отже потрібно передбачити можливість підстроювання увімкнення світильників в залежності від швидкості об'єкта. Проте виконати це завдання не є можливим доки немає можливості визначити швидкість об'єкту. Тож зазвичай обирається деяка середня швидкість для об'єкту, яка враховується при увімкненні світильників. Це не є ефективним, адже, якщо об'єкт рухається швидше, то ділянка попереду освітлюється недостатньо, що може підвищувати рівень небезпеки [26]. А якщо об'єкт рухається повільніше, то ділянка попереду освітлюється занадто сильно, що веде до зниження ефективності використання вуличного освітлення за всіма показниками.

Потрібно зауважити, що деколи фіксована швидкість увімкнення розумних світильників є бажаним рішенням, наприклад, на дорогах фіксована швидкість увімкнення світильників спонукає до дотримання швидкісного режиму дороги.

Коли до встановленої системи розумного освітлення додаються камери спостереження, то система може не лише передавати дані з камер до дата-центру, але також використовувати дані з камер для покращення алгоритмів управління освітленням.

Дані з камер проходять через модуль управління, який слугує зв'язуючою ланкою між розумними світильниками і серверами центру. Модуль управління являється комп'ютером із визначеними характеристиками. Для покращення алгоритму контролю яскравості світильників можна використовувати дані зображень, що передаються з камер спостереження.

Модуль управління використовує зображення з камер для ідентифікації об'єктів. Ідентифікувавши об'єкт на зображеннях, зроблених на певній відстані модуль визначає швидкість рухомого об'єкту. Враховуючи швидкість і маючи дані про відстань ділянки, яку потрібно висвітлити модуль має можливість визначити кількість світильників, що потрібно увімкнути завчасно.

Ідентифікація об’єктів виконується за ознаками Хаара або іншим методом, що задовольняє умови. Після розпізнавання об’єкту на різних ділянках визначається швидкість об’єкту (визначена відстань між ділянками, на яких зроблені зображення і різниця у часі між зображеннями). Кількість світильників, що потрібно увімкнути визначається за простою формулою.

$$N = \frac{V \cdot t}{Lc} \tag{1.1}$$

де: V - швидкість об’єкту;

t - час, що світильник має бути увімкненим до моменту, коли об’єкт мине світильник;

Lc - середня відстань між світильниками.

Потрібно врахувати, що в один момент часу можуть пройти декілька об’єктів повз одну камеру спостереження, отже потрібно визначити швидкість і напрям кожного об’єкта. Псевдокод для визначення світильників, які мають працювати на повну потужність наведено у Додатку В. Наступним кроком модуль управління передає інформацію світильникам про те, які світильники мають бути ввімкнені, а які вимкнені.

При такому підході, коли визначення світильників, які мають бути ввімкнені виконується виділеним комп’ютером (модулем управління) роль світильників як розумних зменшується, адже світильники просто виконують команди від модулю управління. Хоча світильники також можуть визначати яскравість свічення [27].

Віддалене управління розумним освітленням

Віддалене управління є звичайним підходом до увімкнення і вимкнення вуличного освітлення. Зазвичай, віддалене управління освітленням виконується подачею струму в електромережі вуличних світильників.

Для розумних світильників, які можуть вмикатися при настанні темної пори сутінок і вимикатися вранці, віддалене управління також необхідне. Воно слугує для ручного керування світильниками за необхідності, також зв’язок світильників із центром управління відкриває можливості збору даних із датчиків, що розміщені на освітлювальних приборах. Зібрані дані можуть слугувати для виявлення статистичних закономірностей використання світла, моніторингу стану освітлювальних пристроїв чи моніторингу споживання світильниками ресурсів [28].

У подальшому може знадобитися передача даних для додаткових модулів, що будуть встановлені пізніше. Тож, з'єднання розумних вуличних світильників у мережі є важливим кроком у створенні IoT-мережі у місті.

Пов'язувати кожний окремий розумний світильник із сервером через мережу інтернет не є доцільним через кілька причин [29]:

· кожний окремий світильник повинен мати свій власний модуль для підключення до мережі інтернет або локальної мережі, що збільшує ціну кінцевого продукту;

· велика кількість світильників, поєднаних у мережу інтернет може призвести до проблеми із недостатністю IP-адрес;

· велика кількість запитів від світильників до сервера буде утворювати велике навантаження на сервер.

Для вирішення питання з'єднання світильників із сервером використовується модуль управління. Цей блок виконує роль комутатора між розумними світильниками та серверами центру управління (Рис.2.4).

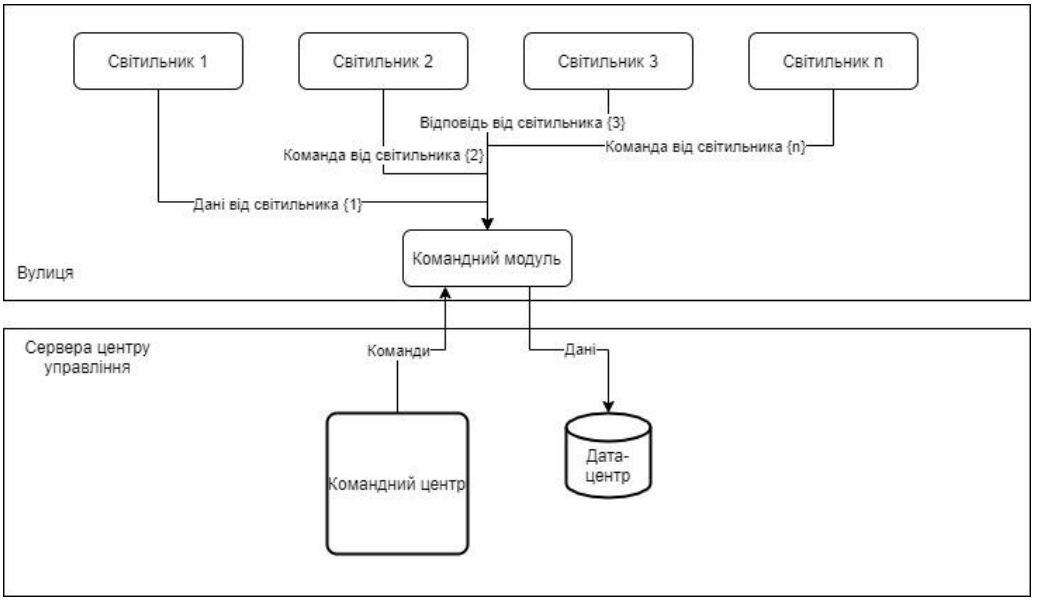


Рисунок 2.4 - Координація розумного освітлення із командного центру

Модуль управління - це пристрій, що призначений для збору інформації від окремих світильників, агрегації даних, збереження та відправки даних на сервер, а також для приймання команд від центру управління та направлення команд до відповідних світильників [30].

Використання командного модуля несе в собі такі переваги як зменшення навантаження на сервер - замість сотень чи тисяч запитів від світильників щохвилини

відправляється лише кілька запитів по одному від кожного командного модуля. Також зменшується кількість пристроїв, що безпосередньо під’єднані до мережі інтернет [31]. Також зменшення кількості модулів, які виконують нетривіальну роботу спрощує оновлення програмного забезпечення шляхом зменшення пристроїв, яким це потрібно.

Модуль управління потрібно приєднувати до визначеної кількості світильників, щоб забезпечити зниження трафіку з одного боку і забезпечити надійність відправки даних з іншого боку, оскільки модуль управління як пристрій IoT має обмежені ресурси.

Модуль управління виконує задачі направлення команд до відповідних світильників, а також опитування світильників і агрегування даних з них як показано на Рис.2.5

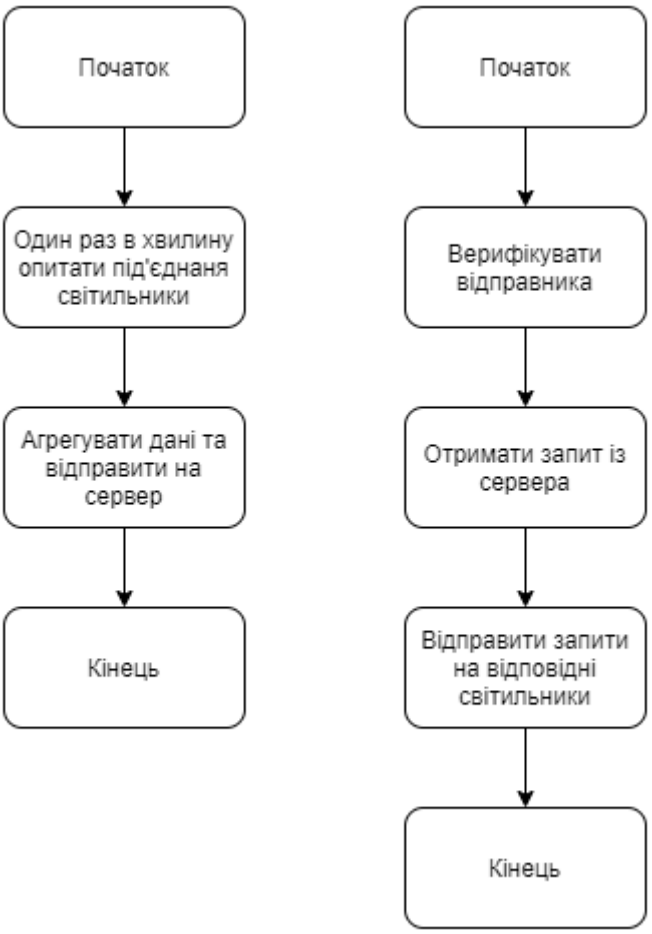


Рисунок 2.5 - Блок-схема роботи командного модулю

Схема роботи описує організацію збору даних від світильників, агрегування та відправку на сервер. Зібрана інформація також включає в себе інформацію про несправні світильники. Однією з ознак несправності є відсутність відповіді від світильника. Також схема описує маршрутизацію команди від центру управління до світильників: модуль

отримує команду чи запит по мережі інтернет, верифікує відправника та, якщо верифіковано, відправляє команду до відповідних світильників.

Тобто структура запиту та схема роботи командного модулю передбачають адресацію команд до частини світильників, наприклад за:

- ідентифікаторами світильників
- деякою умовою належності світильника (наприклад, парність/непарність порядкового номера)

Розумні світильники з’єднуються з модулем через провідний кабель інтернет. Такий підхід зменшує електроспоживання світильників бо не потребує енергії на передачу даних безпровідним шляхом а також зменшує вірогідність помилок у роботі через вихід з ладу модулю безпровідного зв’язку, а також зменшує кількість компонент, які потенційно повинні бути обслужені.

Додаток Б

Тези доповіді на конференції. Тема “Залежність яскравості освітлення від інтенсивності трафіку за допомогою IoT речей”

Залежність яскравості освітлення від інтенсивності трафіку за допомогою IoT речей

Ярослав Гозак[1], Мирослава Гладка[1]
1 - Київський національний університет ім. Тараса Шевченка, Київ, Україна

Вступ. Яскравість освітлення вуличними лампами може регулюватися датчиками освітлення та датчиками руху, що націлені на рух людей або транспорту.

Матеріали та методи. Експериментальні дослідження були проведені на наборі світлодіодів (світлодіоди SMART PIXEL D-12MM WS-2811), чия яскравість змінювалася відповідно до показників датчиків світла (фотодіоду GL5528) та руху (інфрачервоний датчик руху HC-SR501). Споживання електроенергії вимірювалось та розраховувалось стандартними методами в місті Київ на Дарницькій площі.

Результати. В процесі дослідження та моніторингу інтенсивності руху на основі даних з датчиків, що виконувалось протягом 2019 року встановлено залежність інтенсивності руху від часу доби та періоду року (рис.1)

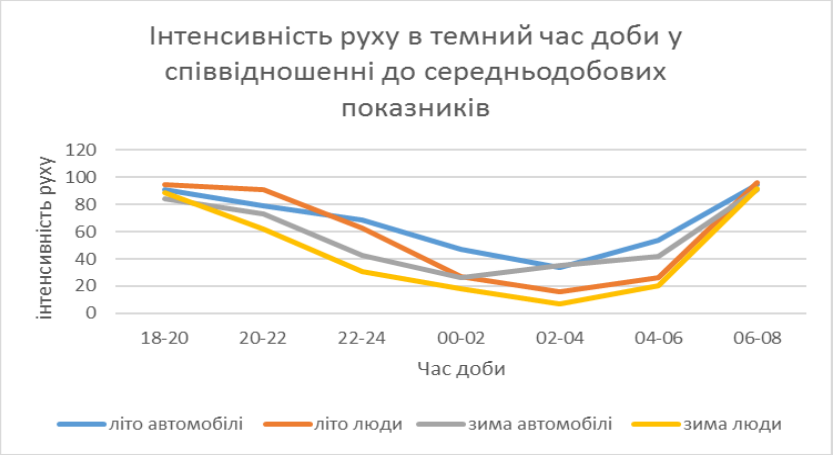


Рис.1. Інтенсивність руху в темний час доби у співвідношенні до середньодобових показників

Визначено можливу економію електроенергії за рахунок зменшення інтенсивності освітлення вночі та за відсутності людей чи транспорту. Вважаючи, що в період з 12 години ночі до 6 години ранку інтенсивність трафіку та наявність людей на вулицях міста значно нижча, ніж ввечері та зранку цей проміжок часу (у 6 годин) може бути використаний для зменшення інтенсивності освітлення міста.

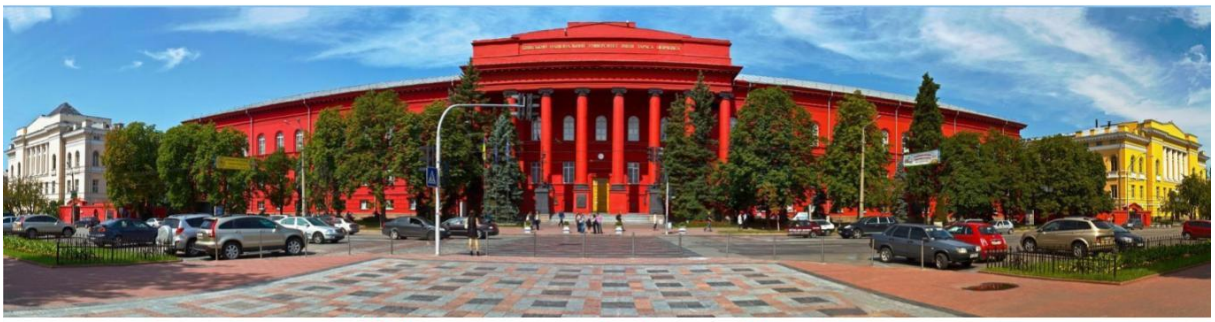
Споживання енергії світлодіодними лампами на 85-90% нижче, ніж у ламп розжарювання. Час неактивності людей у темну пору доби складає 66% влітку та 50% взимку. Якщо зменшувати інтенсивність освітлення вночі до 30% від стандартної яскравості, то, враховуючи, що трафік зменшиться до 20% від вечірнього, можна досягти економії електроспоживання до 73.22%

Висновки. Зменшення інтенсивності освітлення в нічний час та варіювання освітлення в залежності від інтенсивності трафіку вночі позитивно впливає декілька факторів міста. Таке впровадження пропонує суттєву економію електроенергії, більшу толерантність міста до екології та покращення біологічних ритмів людей та тварин у місті за рахунок суттєвого зменшення яскравості міста в цілому.

Додаток В

Презентація кваліфікаційної роботи магістра

Київський національний університет
імені Тараса Шевченка



Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка
Факультет інформаційних технологій
Кафедра інформаційних систем та технологій

Кваліфікаційна робота магістра

Підготував студент 2-го курсу, групи ІРм-21
Гозак Ярослав Дмитрович
Науковий керівник: доктор економічних наук, професор, Онищенко Андрій Михайлович
Київ - 2021

www.knu.ua

Тема

Інформаційно-аналітична IoT система освітлення в Smart City

Київський національний університет
імені Тараса Шевченка

Актуальність дослідження

Інформаційно-технічний прогрес, який зумовлений активним розвитком глобальної мережі Інтернет, впливає і на розвиток інфраструктури міст та змінює характер і методи проектування інфраструктури. Регулювання режимів роботи розумних світильників є актуальним завданням, результати якого можуть позитивно впливати на соціо-психологічний стан оточуючих людей та на економічну складову системи.

Впровадження алгоритмів регулювання режимів роботи світильників у системі має наступні переваги:

- Позитивний вплив на оточення через зменшення кількості вмикання і вимикання світильників
- Зниження витрат на обслуговування шляхом збільшення ресурсу розумного світильнику

Об’єктом дослідження є система розумного освітлення, що включає в себе розумні світильники та застосунок керування освітленням.

Предметом дослідження є методи оптимізації алгоритмів роботи світильників, що націлені на зменшення дратівливих факторів у вигляді частої зміни інтенсивності освітлення в окремих випадках.

www.knu.ua

Мета і завдання дослідження

Метою роботи є подальший розвиток систем розумного освітлення, що включає модернізацію алгоритмів роботи світильників з метою зменшення дратівливого фактору впливу на оточуючих в окремих випадках.

Завданням роботи є:

1. Огляд та вивчення існуючих на ринку рішень.
2. Розрахунок режимів функціонування розумного світильнику виходячи з аспектів створення комфортних соціо-психологічних умов оточуючого середовища людей з урахуванням мінімізації економічних витрат пов'язаних з функціонуванням та експлуатацією досліджуваної системи, за яких буде визначено оптимальний варіант зміни режиму освітлення, який би задовольняв прийнятну зазначену сукупність параметрів.
3. Проектування архітектури інформаційно-аналітичної IoT системи розумного освітлення.
4. Реалізація концептуального застосунку.

Київський національний університет
імені Тараса Шевченка

Існуючі аналоги

1. Tvilight - фінська компанія, лідер на Європейському ринку, проекти у Фінляндії, Німеччині, Швейцарії. Вуличне розумне освітлення, система адаптивного освітлення.
2. Schreder - українська компанія-дистриб'ютор, що надає контролери розумного освітлення, контролери для віддаленого контролю освітлення, контролери для інтелектуального управління освітленням.
3. Citintelly - латвійська компанія, яка надає продукти для системи розумного освітлення, включаючи програмне забезпечення і IoT пристрої.

До недоліку наведених систем належить: не беруть до уваги прогнозовану та реальну інтенсивність руху об' єктів повз світильники з метою забезпечення більш стабільного стану освітленості навколишньої території.

www.knu.ua

Метод розрахунку режимів функціонування світильників

Прогнозування інтенсивності руху

Запропоновані методи:

- Ковзного середнього (є ефективним в умовах початково відсутнього масиву даних та дозволяє отримувати прогнозні значення з незначними похибками)
- Багатофакторний аналіз (дозволяє враховувати вплив кожного досліджуваного фактору на прогнозоване значення моделі)

До розрахунку беруться наступні параметри:

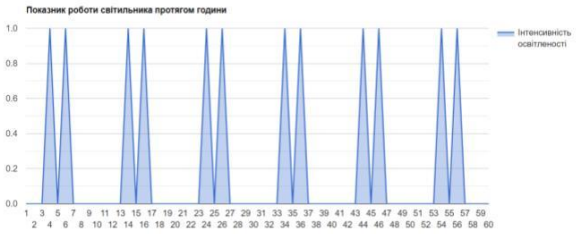
1. День тижня
2. Місяць
3. Погодні умови:
   a. Температура повітря
   b. Сила опадів
   c. Сила вітру

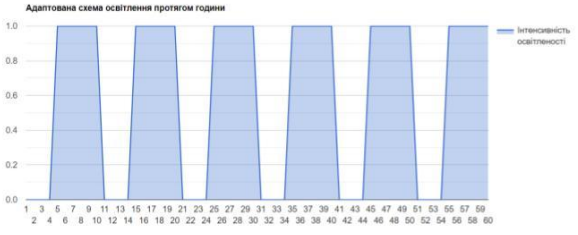
Київський національний університет
імені Тараса Шевченка

Метод розрахунку режимів функціонування світильників

Зміна довжини інтервалу очікування світильнику

```
Tcur = I + t;
if(Tcur <= Tmin  OR Tcur >= Tmax)
        T = Tmin;
else
        T = Tcur
```

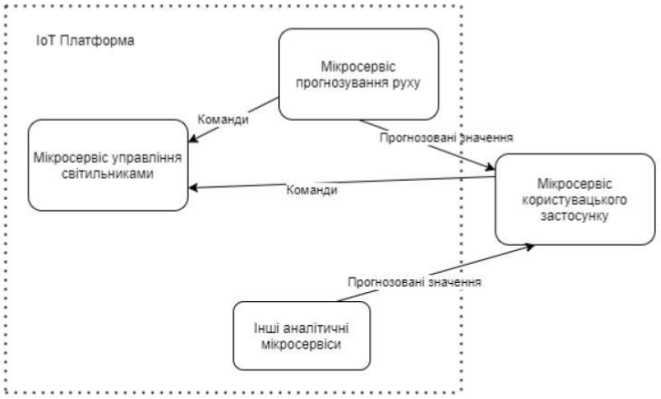




www.knu.ua

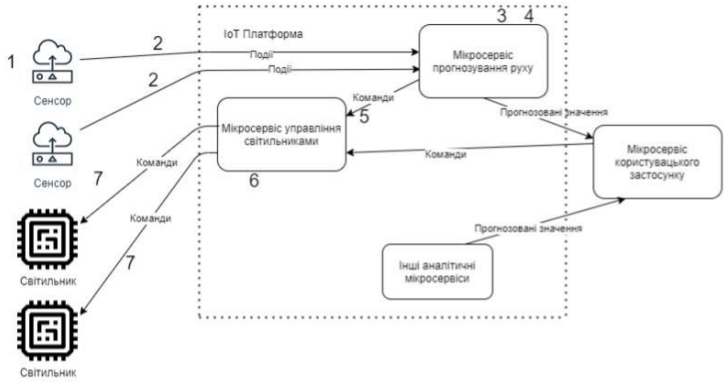
Архітектура інформаційно-аналітичної IoT системи

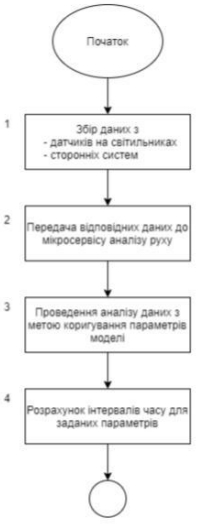
Структурна модель системи



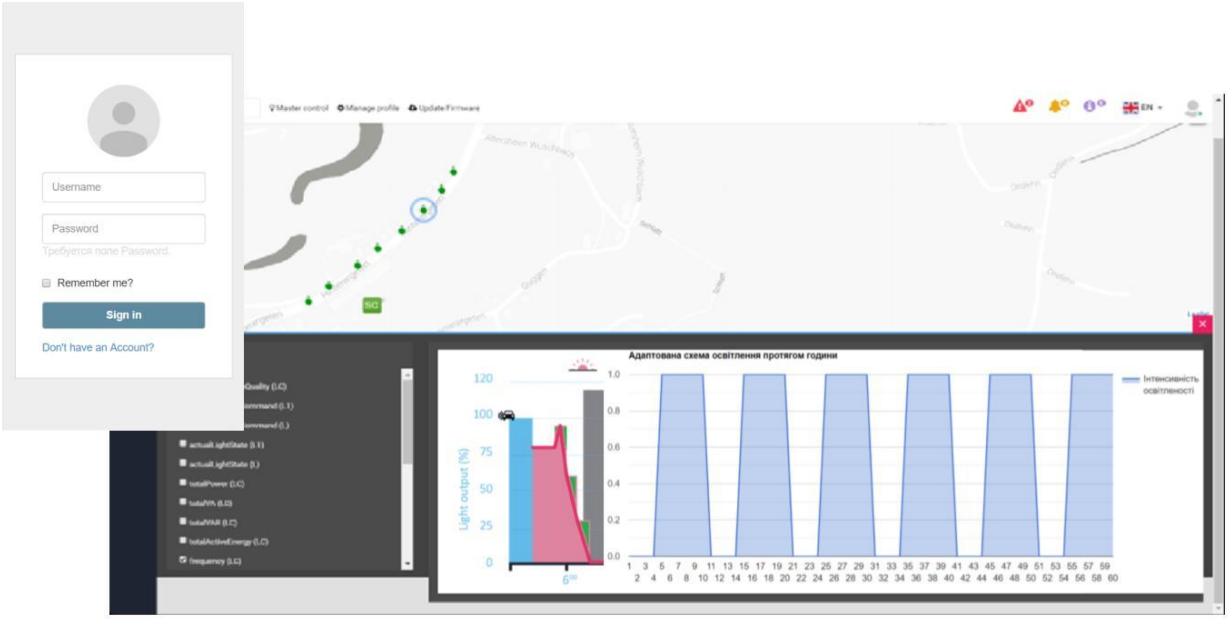
Архітектура інформаційно-аналітичної IoT системи

Діаграма алгоритму та діаграма мікросервісів





Концепт додатку



Київський національний університет
імені Тараса Шевченка

Висновки

1. У ході дослідження розумного освітлення розглянуто існуючу схему управління розумною освітлювальною установкою, її особливості та недоліки. Також досліджено існуючі на світовому та українському ринках компанії, їх продукти та особливості.

2. Запропоновано метод передбачення інтенсивності руху з метою поправки режимів освітлення для оптимізації роботи світильників у нічний час доби, яка передбачає собою балансування між енергоефективністю за рахунок зниження освітлення у години неактивності та кількістю перемикань світильнику з метою зменшення дратівливого впливу на оточуючих та подовження строку служби пристроїв.

3. Запропоновано структурну модель інформаційно-аналітичної IoT системи розумного освітлення, яка передбачає горизонтальне функціональне та фізичне розширення.

4. Реалізовано концепт застосунку для доступу до інформаційно-аналітичної IoT системи розумного освітлення на основі ASP.NET WEB API. Для досягнення цих цілей було використано технологію серверних застосунків .NET, веб-сокетів SignalR та JavaScript фреймворк Angular.

www.knu.ua