

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ
СІКОРСЬКОГО»**

Навчально-науковий інститут атомної та теплової енергетики

Кафедра цифрових технологій в енергетиці

«До захисту допущено»

Завідувач кафедри

_____ Наталія АУШЕВА

“ ” _____ 2026 р.

**Дипломна робота
на здобуття ступеня бакалавр**

За освітньою програмою “Цифрові технології в енергетиці”

Спеціальності 122 “Комп’ютерні науки”

на тему: “ІоТ-система моніторингу та контролю умов фармацевтичних приміщень”

Виконала: студентка 4 курсу, групи ТР-23

_____ Петренко Аліна Андріївна

(прізвище, ім’я, по батькові)

_____ (підпис)

Керівник _____ асистент Антон ПАСТІЧНЮК

(посада, науковий ступінь, вчене звання, ім’я, ПРІЗВИЩЕ)

_____ (підпис)

Рецензент _____ к.т.н., доц., Віталій ПЕШКО

(посада, науковий ступінь, вчене звання, ім’я, ПРІЗВИЩЕ)

_____ (підпис)

Н.контроль _____ асистент Артем МЕЛЬНИЧЕНКО

(посада, ім’я, ПРІЗВИЩЕ)

_____ (підпис)

Засвідчую, що у цій дипломній роботі немає
запозичень з праць інших авторів без
відповідних посилань.

Студент _____

Київ - 2026

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”**

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ

Кафедра ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ

Рівень вищої освіти – перший (бакалаврський)

спеціальність 122 “Комп’ютерні науки”

Освітньо-професійна програма “Цифрові технології в енергетиці”

ЗАТВЕРДЖУЮ

Завідувач кафедри ЦТЕ

Наталія АУШЕВА

(підпис)

“ ” 2026 р.

ЗАВДАННЯ

на дипломну роботу студентці

Петренко Аліні Андріївні

(прізвище, ім’я, по батькові)

1. Тема роботи “ІоТ-система моніторингу та контролю умов фармацевтичних приміщень”

Науковий керівник Пасічник Антон Олексійович, асистент

(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від “28” травня 2026 року № 1992-с

2. Термін подання студентом роботи 04.06.2026 р.

3. Вихідні дані до роботи мови програмування Python та JavaScript, брокер повідомлень Eclipse Mosquitto (протокол MQTT), СУБД часових рядів InfluxDB, аналітична платформа Grafana, середовище імітаційного моделювання Cisco Packet Tracer, інструмент сповіщення Telegram Bot API, середовище розробки PyCharm.

4. Перелік питань, які потрібно розробити _____

1) провести аналіз предметної області, нормативних вимог GDP та наявних комерційних аналогів систем екологічного моніторингу;

2) обґрунтувати вибір методів сегментації корпоративної мережі, протоколів передачі даних ІоТ-пристроїв та стеку відкритих технологій;

- 3) розробити архітектуру мережевої інфраструктури об'єкта та схему логічного розподілу зон складу;
- 4) розробити програмну логіку роботи периферійних IoT-датчиків і актуаторів;
- 5) реалізувати серверний стек для асинхронного збору, довготривалого збереження, візуалізації телеметрії та фільтрації тривожних сповіщень персоналу;
- 6) виконати комплексне тестування розробленої системи та підтвердити працездатність алгоритмів безпеки й автономного керування.
5. Орієнтований перелік ілюстративного матеріалу структурна схема мережевої інфраструктури фармацевтичного складу, схема логічної сегментації та адресації (VLAN), діаграма варіантів використання (Use Case), графічні панелі візуалізації Grafana, інтерфейс отримання тривожних сповіщень.
6. Дата видачі завдання 11.03.2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1.	Вибір теми роботи	11.03.2026	Виконано
2.	Аналіз методів та засобів розв'язання задачі	09.04.2026 - 14.04.2026	Виконано
3.	Розробка архітектури та загальної структури системи	15.04.2026 - 22.04.2026	Виконано
4.	Розробка окремих підсистем	23.04.2026 - 30.04.2026	Виконано
5.	Програмна реалізація системи	01.05.2026 - 11.05.2026	Виконано
6.	Оформлення пояснювальної записки	18.05.2026 - 31.05.2026	Виконано
7.	Захист програмного забезпечення	13.05.2026	Виконано
8.	Передзахист	03.06.2026	Виконано
9.	Захист	15.06.2026 - 19.06.2026	Виконано

Студент

(підпис)

Аліна ПЕТРЕНКО

(прізвище та ініціали)

Керівник

(підпис)

Антон ПАСІЧНЮК

(прізвище та ініціали)

АНОТАЦІЯ

Дипломна робота виконана на 70 сторінках, містить 31 ілюстрацію, 2 таблиці, 2 додатки та 20 джерел в переліку посилань.

Метою роботи є розробка IoT-системи автоматизованого моніторингу та контролю умов зберігання лікарських засобів у фармацевтичному приміщенні.

Основний зміст роботи включає аналіз нормативних вимог до умов зберігання лікарських засобів, проєктування мережевої інфраструктури складу з VLAN сегментацією п'яти функціональних зон, програмування IoT датчиків та актуаторів, реалізацію серверного стеку для збору та зберігання даних, розробку дашборду візуалізації та системи сповіщень.

Методи та засоби: трійарусна ієрархічна модель мережі, VLAN сегментація, протокол MQTT; Cisco Packet Tracer, Python, Eclipse Mosquitto, InfluxDB, Grafana, Telegram Bot API.

Результат: IoT-система моніторингу мікрокліматичних умов фармацевтичного складу з автоматичним керуванням кліматичним обладнанням та сповіщеннями про відхилення від норм.

Розроблена система може бути впроваджена на фармацевтичних складах та підприємствах з дистрибуції лікарських засобів для автоматизації контролю умов зберігання та забезпечення відповідності вимогам EU GDP Guidelines та Наказу МОЗ України №584.

Ключові слова: ІНТЕРНЕТ РЕЧЕЙ, КОМП'ЮТЕРНІ МЕРЕЖІ, ФАРМАЦЕВТИЧНИЙ СКЛАД, МОНІТОРИНГ МІКРОКЛІМАТУ, VLAN, MQTT, CISCO PACKET TRACER, INFLUXDB, GRAFANA, АВТОМАТИЗАЦІЯ.

ABSTRACT

This thesis consists of 70 pages, contains 31 figures, 2 tables, 2 appendices and 20 references in the bibliography.

The aim of this thesis is to develop an IoT system for the automated monitoring and control of storage conditions for pharmaceuticals in a pharmaceutical facility.

The main content of the thesis includes an analysis of regulatory requirements for the storage conditions of medicinal products, the design of a warehouse network infrastructure with VLAN segmentation of five functional zones, the programming of IoT sensors and actuators, the implementation of a server stack for data collection and storage, and the development of a visualization dashboard and a notification system.

Methods and tools: three-tier hierarchical network model, VLAN segmentation, MQTT protocol; Cisco Packet Tracer, Python, Eclipse Mosquitto, InfluxDB, Grafana, Telegram Bot API.

Result: An IoT system for monitoring microclimatic conditions in a pharmaceutical warehouse with automatic control of climate equipment and notifications of deviations from standards.

The developed system can be implemented in pharmaceutical warehouses and drug distribution companies to automate storage condition monitoring and ensure compliance with the EU GDP Guidelines and Order No. 584 of the Ministry of Health of Ukraine.

Keywords: INTERNET OF THINGS, COMPUTER NETWORK, TOPOLOGY, PHARMACEUTICAL WAREHOUSE, MICROCLIMATE MONITORING, VLAN, MQTT, CISCO PACKET TRACER, INFLUXDB, GRAFANA, AUTOMATION.

ЗМІСТ

ВСТУП.....	8
1 ПОСТАНОВКА ЗАДАЧІ ТА ОГЛЯД МЕТОДІВ РОЗРОБКИ.....	10
1.1 Задача автоматизованого моніторингу фармацевтичних складів.....	10
1.2 Аналіз існуючих систем моніторингу	11
1.2.1 Комплексна система моніторингу Comet System.....	12
1.2.2 Система моніторингу Testo Saveris.....	12
1.2.3 Система моніторингу Vaisala viewLine	13
1.2.4 Результати аналізу існуючих систем	14
1.3 Функціональні та нефункціональні вимоги до IoT-системи моніторингу	14
2 МЕТОДИ ТА ТЕХНОЛОГІЇ РЕАЛІЗАЦІЇ IoT-СИСТЕМИ МОНІТОРИНГУ.....	16
2.1 Огляд методів побудови IoT-систем моніторингу.....	16
2.2 Організація та сегментація мережі складу	17
2.2.1 Ієрархічна модель мережевої інфраструктури	18
2.2.2 Технологія VLAN для ізоляції зон зберігання	20
2.3 Забезпечення безпеки мережевого контуру	21
2.3.1 Криптографічний захист та контроль фізичних портів	21
2.3.2 Фільтрація трафіку за допомогою списків доступу (ACL).....	22
2.4 Асинхронний обмін даними за протоколом MQTT	23
2.5 Збір телеметрії та обробка часових рядів.....	25
2.5.1 Архітектурні принципи обробки телеметрії	25
2.5.2 Обґрунтування застосування баз даних часових рядів.....	26
2.6 Оперативне сповіщення та візуалізація даних	27
2.6.1 Алгоритм фільтрації сповіщень у Telegram	27
2.6.2 Побудова інтерфейсів та контроль доступу в Grafana.....	29
3 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ	30
3.1 Варіанти використання системи	30
3.2 Огляд та обґрунтування програмних засобів реалізації	31
3.3 Архітектура та організація проєкту	32

3.4 Налаштування мережевої інфраструктури в Cisco Packet Tracer	35
3.5 Налаштування безпеки мережевої інфраструктури	37
3.6 Програмування логіки IoT-датчиків та актуаторів	38
3.7 Реалізація серверної логіки обробки телеметрії	40
3.8 Проєктування логічної моделі бази даних InfluxDB	41
3.9 Побудова аналітичного інтерфейсу в середовищі Grafana	42
3.10 Розробка модуля сповіщень Telegram Bot API	43
4 ТЕСТУВАННЯ ТА ДЕМОНСТРАЦІЯ РОБОТИ СИСТЕМИ	45
4.1 Системні вимоги та інсталяція програмного забезпечення	45
4.2 Результати комплексного тестування компонентів	48
4.2.1 Тестування мережевої інфраструктури та маршрутизації	49
4.2.2 Тестування системи безпеки мережі	52
4.2.3 Тестування IoT-датчиків та актуаторів	55
4.2.4 Тестування серверного стеку та Grafana дашборду	57
4.2.5 Тестування системи сповіщень Telegram	59
4.3 Аналіз результатів та порівняльна характеристика	60
ВИСНОВКИ	62
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	64
ДОДАТОК А	66
ДОДАТОК Б	70

ВСТУП

Фармацевтична галузь належить до найбільш регульованих секторів промисловості, оскільки якість лікарських засобів безпосередньо впливає на здоров'я та життя пацієнтів. Одним із ключових факторів, що визначають стабільність і безпечність препаратів, є умови їх зберігання, насамперед температура та відносна вологість повітря. Відхилення цих параметрів навіть на короткий проміжок часу може призвести до втрати терапевтичних властивостей лікарських засобів, псування цілих партій продукції та значних фінансових втрат.

Актуальність теми зумовлена вимогами міжнародних і національних нормативних документів, які передбачають необхідність безперервного моніторингу умов зберігання з обов'язковою фіксацією показників та оперативним реагуванням на відхилення. Незважаючи на це, на практиці багато фармацевтичних складів, особливо малого та середнього розміру, все ще використовують ручний контроль із періодичними вимірюваннями та паперовими журналами. Такий підхід не забезпечує безперервності спостереження, є вразливим до людського фактора та не дозволяє своєчасно реагувати на критичні зміни мікроклімату.

Існуючі комерційні системи моніторингу, забезпечують високий рівень автоматизації та відповідність нормативним вимогам, однак їх широке впровадження обмежується високою вартістю, ліцензійними платежами та залежністю від виробника або хмарної інфраструктури. У результаті це робить подібні рішення малодоступними для більшості підприємств та обмежує можливості їх гнучкої адаптації до конкретних умов експлуатації. Таким чином, актуальною є задача розробки доступної та масштабованої IoT-системи моніторингу умов зберігання лікарських засобів на основі відкритих технологій, яка б забезпечувала виконання вимог GDP без додаткових ліцензійних витрат.

Метою роботи є розробка IoT-системи автоматизованого моніторингу та контролю умов зберігання лікарських засобів у фармацевтичному складі з використанням VLAN-сегментації мережі, протоколу MQTT для передачі даних та стеку відкритих технологій Python, InfluxDB і Grafana для обробки, зберігання та

візуалізації даних у реальному часі. Обраний підхід дозволяє забезпечити розподілену архітектуру системи, високу масштабованість та автономність роботи всіх її компонентів.

У межах виконання роботи передбачається аналіз підходів до побудови IoT-систем моніторингу та нормативних вимог до умов зберігання лікарських засобів, дослідження існуючих програмних рішень у цій сфері, а також обґрунтування вибору технологій для реалізації кожного з компонентів системи. Окрему увагу приділено розробці програмної та мережевої архітектури системи, яка включає IoT-пристрої, серверну обробку даних, базу даних часових рядів, систему візуалізації та механізм сповіщень. Завершальним етапом є тестування розробленого рішення та перевірка його відповідності нормативним вимогам.

Було прийнято участь у Міжнародній науково-практичній конференції «Science at the Frontier of Progress». Апробацію результатів роботи наведено в додатку Б.

Дипломна робота складається зі вступу, чотирьох розділів основної частини, висновків, списку використаних джерел та додатків. Загальний обсяг роботи становить 70 сторінок, вона містить 31 рисунок, 2 таблиці та 2 додатки, список використаних джерел налічує 20 найменувань.

1 ПОСТАНОВКА ЗАДАЧІ ТА ОГЛЯД МЕТОДІВ РОЗРОБКИ

Метою цього розділу є обґрунтування необхідності розробки автоматизованої системи контролю мікроклімату у фармацевтичних приміщеннях. У межах розділу здійснюється детальний аналіз специфіки предметної області, формалізуються вимоги до функціональних компонентів системи, а також проводиться системний аналіз наявних архітектурних підходів, мережевих методів та програмних інструментів, які стануть основою для практичної реалізації інженерного рішення.

1.1 Задача автоматизованого моніторингу фармацевтичних складів

Фармацевтична галузь потребує суворого регулювання, оскільки умови зберігання лікарських засобів напряду впливають на їхню безпеку та ефективність. Порушення температурного чи вологісного режимів призводить до деградації препаратів, втрати їхньої терапевтичної дії або появи токсичних властивостей. Через це фармацевтичні склади є об'єктами обов'язкового ліцензування та регулярних інспекцій, а їхня діяльність в Україні регламентується Законом «Про лікарські засоби» та нормативними актами Міністерства охорони здоров'я (МОЗ) [3, 4].

Залежно від фізико-хімічних властивостей, медикаменти поділяють на кілька категорій зберігання. Термолабільні препарати, такі як вакцини чи інсулін, потребують безперервного підтримання температури від +2°C до +8°C, де навіть короткочасне порушення «холодового ланцюга» є критичним. Термочутливі форми вимагають прохолодних умов від +8°C до +15°C, а більшість твердих форм

(таблетки, капсули) зберігаються за кімнатної температури від +15°C до +25°C, але є чутливими до підвищеної вологості.

Застарілий ручний контроль із фіксацією показників у паперових журналах кілька разів на добу не забезпечує належної безпеки через велику дискретність вимірів, людський фактор і відсутність оперативного сповіщення про аварії. Сучасні вимоги автоматизації визначаються міжнародними стандартами EU GDP Guidelines 2013/C 343/01 [1] та WHO Technical Report Series No. 1025, а також національним Наказом МОЗ України №584 і вимогами Належної практики зберігання [2, 4]. Ці документи зобов'язують впроваджувати системи безперервного моніторингу з автоматичним документуванням усіх відхилень для забезпечення аудиторського сліду. Відповідно до зазначених нормативних документів були реалізовані діапазони температури і вологості для кожної зони складу, які наведені в таблиці 1.1.

Таблиця 1.1 – Діапазони температури і вологості

Зона складу	Температура	Вологість
Холодна зона	+2 – +8°C	< 60% RH
Прохолодна зона	+8 – +15°C	40–60% RH
Загальний склад	+15 – +25°C	40–60% RH
Карантинна зона	+15 – +25°C	40–60% RH
Зона відпуску	+15 – +25°C	40–60% RH

Метою роботи є розробка IoT-системи автоматизованого моніторингу та контролю умов зберігання лікарських засобів, яка забезпечує безперервний збір метрик із інтервалом опитування не більше 30 секунд, автоматичне керування кліматичними актуаторами без участі людини та миттєве сповіщення про аномалії.

1.2 Аналіз існуючих систем моніторингу

Перед початком розробки власної системи проведено аналіз існуючих комерційних рішень у сфері моніторингу умов зберігання лікарських засобів.

Метою аналізу є визначення переваг та недоліків наявних систем для обґрунтування архітектурних рішень розробленої системи та уникнення виявлених недоліків.

1.2.1 Комплексна система моніторингу Comet System

Comet System є спеціалізованою системою моніторингу мікроклімату, орієнтованою на фармацевтичну галузь [5]. Рішення включає апаратні датчики, програмне забезпечення для збору та аналізу даних, а також інструменти для централізованого контролю умов зберігання лікарських засобів.

Основними перевагами системи є висока точність вимірювань, підтримка автоматичного формування звітів для проведення аудиту та наявність веб-інтерфейсу з можливістю налаштування сповіщень. Система відповідає вимогам фармацевтичної галузі щодо контролю температури та вологості й забезпечує збереження історичних даних для подальшого аналізу.

Водночас рішення має ряд недоліків. До них належать висока вартість впровадження, залежність від ліцензійного програмного забезпечення та обмежені можливості інтеграції з іншими платформами. Тому при розробці власної системи доцільно використати підхід до формування звітів і моніторингу, але реалізувати його на базі відкритих технологій без залежності від виробника.

1.2.2 Система моніторингу Testo Saveris

Testo Saveris є системою моніторингу умов зберігання, орієнтованою на фармацевтичну та харчову галузі [6]. До її складу входять бездротові датчики, базова станція та хмарна платформа для централізованого збору й аналізу інформації.

Основною перевагою системи є використання бездротових каналів зв'язку, що спрощує монтаж та подальше масштабування. Крім того, датчики підтримують локальне буферування даних у разі втрати зв'язку, що дозволяє уникнути втрати вимірювань. Хмарна платформа забезпечує доступ до інформації з будь-якого пристрою та підтримує систему автоматичних сповіщень.

Разом із тим система має суттєву залежність від хмарної інфраструктури та стабільного доступу до мережі Інтернет. Також використання сервісу супроводжується регулярними витратами на обслуговування та обмеженими можливостями адаптації під специфічні потреби підприємства. У розробленій системі збережено переваги бездротового збору даних, проте всі програмні компоненти розгортаються локально на сервері підприємства.

1.2.3 Система моніторингу Vaisala viewLinc

Vaisala viewLinc є корпоративною системою моніторингу параметрів навколишнього середовища, що широко застосовується у фармацевтичній промисловості та лабораторіях [7]. Система поєднує мережеві пристрої збору даних, датчики контролю мікроклімату та централізоване серверне програмне забезпечення.

До переваг рішення належать підтримка детального аудиту дій користувачів, розвинуті механізми розмежування доступу, резервування даних та відповідність вимогам GMP і 21 CFR Part 11. Такі можливості забезпечують високий рівень надійності та придатність системи для використання на великих фармацевтичних підприємствах. Основними недоліками є висока вартість впровадження, складність налаштування та надлишковість функціоналу для невеликих складських комплексів.

Водночас аналіз системи підтвердив важливість ведення історії подій та збереження архівних даних, що також було реалізовано у власному проєкті на базі відкритих програмних засобів.

1.2.4 Результати аналізу існуючих систем

Проведений аналіз показав, що сучасні системи моніторингу фармацевтичних складів мають спільний набір функціональних можливостей, до яких належать безперервний збір телеметричних даних, зберігання історичних записів для проведення аудиту, веб-інтерфейс для моніторингу параметрів середовища та система автоматичних сповіщень про аварійні ситуації. Саме ці можливості були враховані під час проєктування розробленої системи.

Разом із тим більшість комерційних рішень характеризуються високою вартістю впровадження, залежністю від виробника та ліцензійними обмеженнями. Деякі системи додатково потребують використання хмарної інфраструктури, що створює ризики для безперервності моніторингу та збільшує експлуатаційні витрати.

З урахуванням результатів аналізу було прийнято рішення побудувати систему на основі відкритого програмного забезпечення з локальним розгортанням усіх ключових компонентів. Такий підхід дозволяє реалізувати функціональність, характерну для комерційних рішень, зберігаючи незалежність від постачальників, можливість подальшої модернізації та мінімальні витрати на експлуатацію.

1.3 Функціональні та нефункціональні вимоги до IoT-системи моніторингу

На основі аналізу нормативних документів, а також результатів дослідження існуючих систем моніторингу було сформовано перелік вимог до розроблюваної IoT-системи контролю умов зберігання лікарських засобів.

Основною функціональною вимогою є забезпечення безперервного автоматичного моніторингу температури та вологості в усіх зонах складу. Система повинна регулярно збирати дані з датчиків [8], перевіряти їх відповідність нормативним значенням та визначати поточний стан середовища. У разі виходу

параметрів за допустимі межі необхідно автоматично формувати та надсилати сповіщення відповідальним працівникам із зазначенням проблемної зони та характеру відхилення.

Важливою вимогою є збереження всієї зібраної інформації для подальшого аналізу та проходження аудитів. Дані про вимірювання та аварійні події повинні накопичуватися в єдиній базі даних і бути доступними для перегляду за будь-який період часу. Для контролю роботи системи необхідний графічний інтерфейс, який забезпечує відображення поточних показників, історичних графіків та статусів роботи обладнання в режимі реального часу. Також система повинна підтримувати рольову модель доступу, що розмежовує права звичайних користувачів і адміністраторів.

Серед нефункціональних вимог особливе значення мають надійність, безпека та масштабованість архітектури Інтернету речей [9]. Відмова окремого компонента не повинна призводити до зупинки всієї системи. Мережева інфраструктура має забезпечувати ізоляцію технологічних зон, захист від несанкціонованого доступу та контроль міжмережевого трафіку. Крім того, система повинна підтримувати підключення нових датчиків і розширення функціональності без суттєвої зміни архітектури. Для зменшення вартості впровадження та уникнення залежності від постачальників програмного забезпечення доцільним є використання відкритих технологій і програмних засобів з відкритим вихідним кодом.

2 МЕТОДИ ТА ТЕХНОЛОГІЇ РЕАЛІЗАЦІЇ ІОТ-СИСТЕМИ МОНІТОРИНГУ

У цьому розділі наведено теоретичне та практичне обґрунтування технологій і рішень, використаних під час розробки системи автоматизованого контролю мікроклімату фармацевтичного складу. Розглянуто організацію мережевої інфраструктури, обмін даними в ІоТ-системах, структуру баз даних часових рядів, методи захисту від спаму повідомлень та засоби візуалізації даних.

2.1 Огляд методів побудови ІоТ-систем моніторингу

Технологічна концепція інтернету речей передбачає побудову мереж із фізичних об'єктів, які за допомогою сенсорів та комунікаційних модулів здійснюють збір і обмін даними в автоматичному режимі. Під час проєктування архітектури таких систем зазвичай спираються на три підходи (рисунок 2.1).

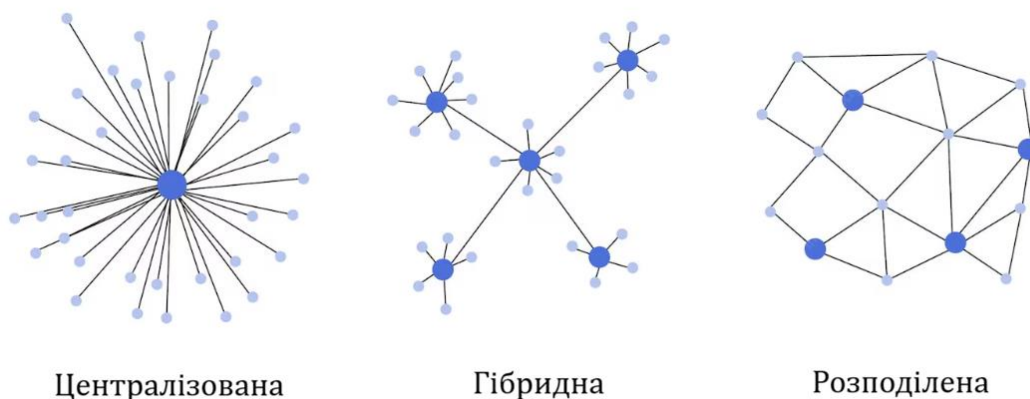


Рисунок 2.1 – Архітектурні підходи до організації топології ІоТ-систем

Централізована архітектура передбачає передачу необроблених даних на єдиний сервер, що є простим, але створює ризик єдиної точки відмови системи.

Розподілена архітектура переносить логіку первинної обробки на периферійні пристрої, зменшуючи мережевий трафік, але ускладнюючи координацію. Найбільш збалансованою є гібридна архітектура, обрана для цього проєкту, де базовий рівень підключення й первинного пакування телеметрії забезпечується локальним IoT-шлюзом, а довготривала обробка й аналітика покладаються на центральний сервер [9].

Для побудови мережевої інфраструктури складу застосовано методи ієрархічного моделювання Cisco Three-Tier (ядро, розподіл, доступ) та суворого логічного ізолювання трафіку через VLAN-сегментацію [10]. Це дозволяє надійно розділити зони зберігання на канальному рівні й заблокувати несанкціоновану міжмережеву комунікацію.

Важливим етапом є вибір прикладного протоколу передачі даних. Протокол HTTP не є оптимальним для систем інтернету речей через великі текстові заголовки, а CoAP функціонує поверх транспортного протоколу UDP, що не гарантує надійності доставки повідомлень. Найбільш оптимізованим рішенням є протокол MQTT, який працює за асинхронною моделлю «видавець-підписник» поверх надійного TCP, має мінімальний розмір заголовка у два байти та підтримує механізми контролю якості обслуговування.

2.2 Організація та сегментація мережі складу

Стабільна робота системи автоматизованого моніторингу мікроклімату залежить від правильної організації мережевої інфраструктури. Для забезпечення надійної передачі телеметричних даних, зменшення затримок і розмежування технологічних зон у проєкті використано ієрархічну модель мережі та логічну сегментацію на VLAN. Такий підхід підвищує відмовостійкість системи, спрощує масштабування мережі та забезпечує ефективну взаємодію між усіма компонентами інфраструктури.

2.2.1 Ієрархічна модель мережевої інфраструктури

Проектування мережевого контуру для об'єктів фармацевтичного призначення вимагає особливого підходу до вибору топології, оскільки система повинна забезпечувати безперервний збір даних без затримок та втрат пакетів. Використання простих плоских мереж, де всі пристрої підключені до одного комутаційного поля без функціонального розподілу, є неефективним. У таких конфігураціях виникає велика кількість службового широкомовного трафіку, який змушені обробляти всі підключені пристрої. Для периферійних IoT-датчиків, які мають обмежені обчислювальні ресурси, це створює надлишкове навантаження і може призвести до збоїв у виконанні їхньої основної логіки.

Щоб усунути ці недоліки та забезпечити високу стабільність інфраструктури, у цьому проєкті застосовано метод ієрархічного моделювання корпоративних мереж Cisco [11]. Дана концепція передбачає чіткий поділ мережі на три незалежні функціональні рівні, що дозволяє оптимізувати обробку трафіку, локалізувати потенційні збої та значно спростити процедури адміністрування і пошуку несправностей.

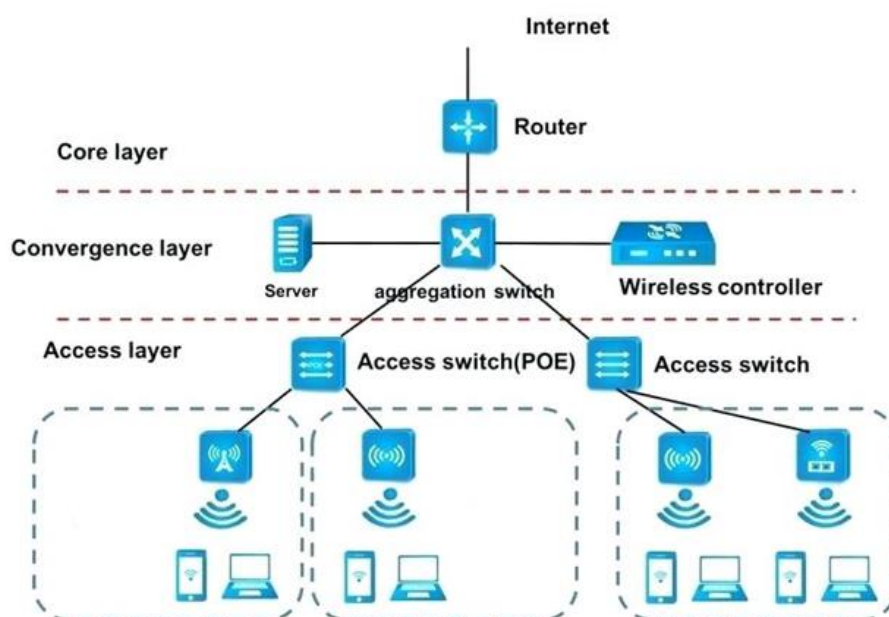


Рисунок 2.2 – Ієрархічна трирівнева модель мережі

Нижнім шаром архітектури є рівень доступу, який безпосередньо взаємодіє з кінцевими вузлами системи. Він реалізується за допомогою комутаторів другого рівня, які розташовуються безпосередньо у функціональних зонах складу. Головним завданням цього рівня є забезпечення надійного фізичного підключення бездротових IoT-шлюзів, датчиків моніторингу та актуаторів контролю мікроклімату до загальної мережі. Окрім цього, рівень доступу виконує функцію первинного захисту, контролюючи легітимність підключень на рівні окремих портів обладнання.

Вся зібрана інформація з рівня доступу передається на рівень розподілу, який виступає інтелектуальним центром мережевої інфраструктури об'єкта. Цей шар побудований на базі комутатора третього рівня, який об'єднує всі локальні сегменти в єдину систему. Саме на рівні розподілу виконуються ключові операції з міжмережевої маршрутизації трафіку, агрегації каналів зв'язку, а також реалізуються основні політики безпеки та правила фільтрації потоків даних [10]. Розподіл функцій дозволяє розвантажити кінцеві комутатори та зосередити управління політиками доступу в одному центральному вузлі.

Найвищим в ієрархії є рівень ядра, представлений центральним маршрутизатором. Його єдиною і головною функцією є забезпечення максимально швидкісної передачі пакетів між внутрішньою інфраструктурою складу та зовнішніми цифровими сервісами. Через рівень ядра здійснюється взаємодія серверної частини проєкту із зовнішніми прикладними інтерфейсами, зокрема з серверами Telegram API для надсилання екстрених сповіщень. На цьому рівні навмисно не виконуються складні та обчислювально важкі операції фільтрації, щоб не знижувати пропускну здатність усієї системи.

Така трирівнева структура забезпечує високу масштабованість, бо за необхідності розширення складу або додавання нових технологічних приміщень достатньо розгорнути новий комутатор на рівні доступу, не змінюючи при цьому глобальні налаштування ядра чи розподілу мережі.

2.2.2 Технологія VLAN для ізоляції зон зберігання

Суворі вимоги міжнародних та національних стандартів належної практики дистрибуції лікарських засобів зобов'язують підприємства забезпечувати чітке логічне, а в деяких випадках і фізичне розділення зон зберігання різних категорій медичної продукції. Для відображення цих нормативних вимог на рівні мережевої інфраструктури в межах проєкту було впроваджено технологію віртуальних локальних мереж стандарту IEEE 802.1Q [12]. Метод сегментації дозволяє розділити одне фізичне середовище комутаторів на кілька логічно ізольованих ширококомовних доменів, що суттєво підвищує безпеку й продуктивність мережі. Базовий концептуальний принцип логічного розділення трафіку та його передачі через спільні магістральні канали наочно представлено на рисунку 2.3.

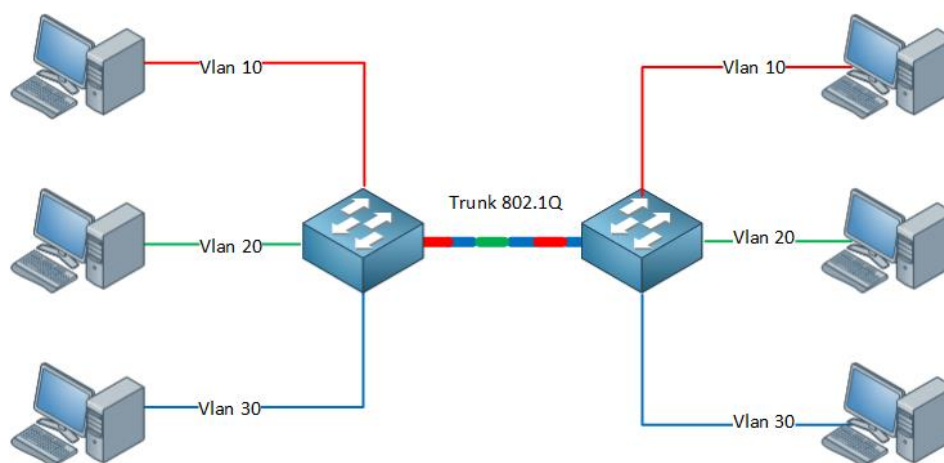


Рисунок 2.3 – Схема сегментації трафіку та організації магістрального каналу

Кожна функціональна зона фармацевтичного складу виділена в окремий віртуальний сегмент зі своєю унікальною адресацією. Зокрема, створено ізольовані мережі для холодильної камери, де зберігаються термолабільні вакцини, для прохолодної зони термочутливих препаратів, для загального складу твердих лікарських форм, а також для зон карантину та відпуску продукції. Завдяки такому підходу весь ширококомовний трафік, який генерують датчики під час пошуку шлюзу

або встановлення з'єднання, залишається виключно в межах одного приміщення. Датчики з різних зон складу повністю позбавлені можливості безпосередньо обмінюватися даними між собою, що ліквідує надлишковий рух пакетів і підвищує загальну стабільність бездротових каналів зв'язку.

2.3 Забезпечення безпеки мережевого контуру

Безпека мережевої інфраструктури складу є важливою складовою надійної роботи системи моніторингу мікроклімату. Для захисту даних і запобігання несанкціонованому доступу в проєкті реалізовано багаторівневу систему захисту, що охоплює як мережевий, так і фізичний рівні безпеки. Такий підхід дозволяє мінімізувати ризики кібератак, забезпечити цілісність даних та підтримувати безперервність роботи системи.

2.3.1 Криптографічний захист та контроль фізичних портів

Перший рівень захисту мережевої інфраструктури спрямований на забезпечення безпечного адміністрування мережевого обладнання. Для цього на всіх комутаторах і маршрутизаторі використовується протокол SSH версії 2, тоді як служба Telnet повністю відключена [11]. Такий підхід обумовлений тим, що Telnet передає дані у відкритому вигляді, включаючи логіни, паролі та команди керування, що створює ризик їх перехоплення. Натомість SSH використовує механізми шифрування та автентифікації, завдяки чому весь адміністративний трафік передається захищеним каналом. Це дозволяє запобігти несанкціонованому доступу до конфігурації мережевого обладнання та підвищує загальний рівень безпеки системи.

Другий рівень захисту реалізовано на комутаторах рівня доступу за допомогою технології Port Security. Її призначення полягає в обмеженні кількості

пристроїв, які можуть бути підключені до конкретного фізичного порту комутатора. У межах проєкту для кожного порту встановлено ліміт в одну MAC-адресу, що відповідає підключеному IoT-шлюзу, серверу або іншому легітимному мережевому пристрою. Після першого підключення обладнання його апаратна адреса автоматично реєструється комутатором і використовується для подальшої перевірки автентичності підключень.

Якщо до порту буде підключено пристрій з іншою MAC-адресою, комутатор зафіксує порушення політики безпеки та застосує визначену реакцію. У результаті порт буде автоматично заблокований, а передача мережевого трафіку через нього припиниться до усунення причини спрацювання захисту. Такий механізм дозволяє запобігти підключенню стороннього обладнання, зменшує ризик несанкціонованого доступу до внутрішніх ресурсів мережі та підвищує захищеність системи від фізичних атак на мережеву інфраструктуру.

Поєднання використання SSH та Port Security забезпечує багаторівневий підхід до захисту мережі. Перший механізм гарантує безпечне віддалене адміністрування обладнання, тоді як другий контролює фізичний доступ до мережевого середовища. Завдяки цьому підвищується надійність функціонування системи моніторингу мікроклімату та забезпечується захист критично важливих даних від несанкціонованого втручання.

2.3.2 Фільтрація трафіку за допомогою списків доступу (ACL)

Третій рівень захисту реалізовано на межі віртуальних мереж і призначено для контролю обміну даними між різними сегментами інфраструктури складу. Незважаючи на те, що технологія VLAN забезпечує логічне розділення пристроїв за функціональними зонами, сама по собі вона не обмежує міжмережеву взаємодію. Для керування такими з'єднаннями в проєкті використовуються розширені списки контролю доступу (ACL), налаштовані на комутаторі третього рівня [12].

Розширені ACL дозволяють фільтрувати мережевий трафік за кількома параметрами одночасно, зокрема за IP-адресами джерела та призначення, типом транспортного протоколу і номером порту. Завдяки цьому можна детально контролювати маршрути передачі даних між окремими підмережами та дозволяти лише ті з'єднання, які необхідні для роботи системи моніторингу.

Політика фільтрації побудована за принципом мінімальних привілеїв. Це означає, що всі мережеві взаємодії за замовчуванням заборонені, а доступ надається лише для визначених сервісів і вузлів. У межах розробленої системи датчикам та IoT-шлюзам дозволено передавати телеметричні дані виключно до центрального MQTT-сервера, який виконує роль єдиної точки збору повідомлень. Будь-які спроби прямого обміну даними між пристроями різних технологічних зон складу блокуються правилами ACL.

Під час маршрутизації кожен пакет перевіряється на відповідність встановленим правилам. Якщо пакет відповідає дозволеному сценарію взаємодії, він передається до пункту призначення. У випадку невідповідності хоча б одному з правил пакет автоматично відкидається. Такий підхід виключає можливість несанкціонованого доступу між сегментами мережі та зменшує площу потенційного поширення атак.

2.4 Асинхронний обмін даними за протоколом MQTT

Одним із ключових завдань під час розробки системи є забезпечення надійного обміну даними між IoT-пристроями та серверною частиною. У процесі роботи датчики постійно генерують телеметричну інформацію про температуру та вологість у різних зонах складу, тому механізм передачі даних повинен забезпечувати мінімальні затримки, низьке мережеве навантаження та можливість одночасної роботи великої кількості пристроїв. Для реалізації цих вимог у проєкті використано протокол MQTT та брокер повідомлень Eclipse Mosquitto [14].

На відміну від протоколу HTTP, який працює за схемою «запит–відповідь», MQTT використовує асинхронну модель взаємодії «видавець–підписник». У такій архітектурі датчики не взаємодіють із серверними модулями напряму, а передають повідомлення до брокера, який виконує їх подальшу маршрутизацію між усіма зацікавленими компонентами системи. Завдяки цьому джерела даних та їхні споживачі залишаються незалежними один від одного, що спрощує масштабування системи та дозволяє додавати нові сервіси без зміни логіки роботи датчиків [9]. Загальну схему інформаційної взаємодії наведено на рисунку 2.4.

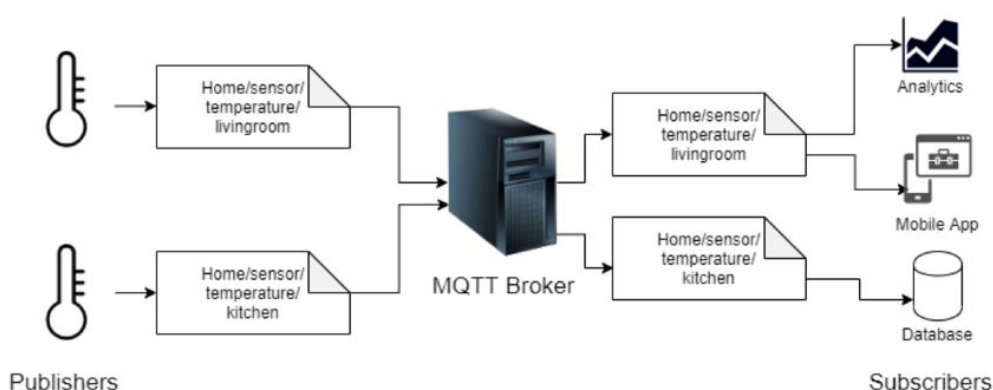


Рисунок 2.4 – Схема розподілу інформаційних потоків в архітектурі MQTT

Передача повідомлень у MQTT організована за допомогою системи тем (topics). У розробленій системі структура тем відображає логічну структуру фармацевтичного складу та містить інформацію про конкретну зону й контрольований параметр. Такий підхід дозволяє гнучко організувати обробку даних та забезпечує можливість підписки як на окремі потоки повідомлень, так і на всі дані складу загалом. У результаті серверні модулі отримують лише ту інформацію, яка необхідна для виконання їхніх функцій.

Важливою перевагою MQTT є також низькі накладні витрати на передачу даних. Завдяки компактному формату повідомлень і мінімальному службовому заголовку протокол добре підходить для IoT-систем, у яких необхідно регулярно передавати невеликі обсяги телеметричної інформації від великої кількості

пристроїв. Це дозволяє зменшити навантаження на мережу та забезпечити стабільну роботу системи навіть при збільшенні кількості датчиків.

У межах розробленої системи MQTT-брокер виступає центральною ланкою взаємодії між модулем генерації даних, підсистемою аналізу телеметрії, базою даних InfluxDB та сервісом Telegram-сповіщень. Використання асинхронного обміну повідомленнями забезпечує незалежність окремих компонентів, підвищує відмовостійкість системи та спрощує її подальше розширення.

2.5 Збір телеметрії та обробка часових рядів

Після налаштування мережевого контуру та організації передачі даних через MQTT-брокер наступним етапом є проєктування серверного рівня системи. Даний рівень забезпечує безперервний прийом, обробку та збереження телеметричних даних. Основна увага приділяється перетворенню сирих вимірювань параметрів мікроклімату у структуровану модель часових рядів, придатну для подальшого аналізу, візуалізації та формування регуляторної звітності. Серверна частина системи реалізує асинхронну обробку повідомлень, їх первинну валідацію на коректність і відповідність встановленим правилам безпеки, а також підготовку до подальшого запису в базу даних.

2.5.1 Архітектурні принципи обробки телеметрії

Серверна архітектура системи побудована на основі постійно запущених фонових процесів, що забезпечують безперервну готовність до прийому даних від периферійних пристроїв. Після ініціалізації та встановлення стабільного з'єднання з MQTT-брокером сервіс виконує підписку на відповідні ієрархічні теми та переходить у режим асинхронного очікування повідомлень.

Обробка телеметрії реалізується за подійною моделлю, за якої кожне нове повідомлення опрацьовується одразу після надходження через функцію-обробник. Такий підхід виключає необхідність постійного опитування пристроїв і дозволяє ефективно використовувати обчислювальні ресурси, задіюючи їх лише в момент появи нових даних.

Важливим архітектурним рішенням є відокремлення логіки обробки даних від конфігураційних параметрів системи. Усі змінні середовища, мережеві налаштування, а також нормативні діапазони температури та вологості для окремих зон винесені в окремий конфігураційний файл. Такий підхід забезпечує можливість гнучкого налаштування системи без внесення змін до програмного коду, що спрощує її супровід та масштабування.

Життєвий цикл кожного повідомлення включає кілька етапів. Спочатку виконується десеріалізація даних та виділення параметрів вимірювання і метаданих, таких як ідентифікатор зони. Далі здійснюється порівняльний аналіз фактичних значень із нормативними межами, на основі чого формується статус стану – нормальний або аварійний. Отримані дані разом із часовою міткою передаються до підсистеми збереження.

2.5.2 Обґрунтування застосування баз даних часових рядів

Для довготривалого зберігання та аналізу телеметричних даних у системах моніторингу мікроклімату використано спеціалізовану базу даних часових рядів InfluxDB [15]. Застосування традиційних реляційних СКБД у таких умовах є неефективним через високу інтенсивність записів, що призводить до значного навантаження на індексні структури та зниження продуктивності. Крім того, реляційна модель даних є менш придатною для горизонтального масштабування при роботі з потоковими даними.

Бази даних часових рядів оптимізовані для обробки безперервних потоків вимірювань, де ключовим елементом є часова мітка. У таких системах дані

організуються у вигляді вимірювань, тегів та полів, що забезпечує ефективне поєднання високої швидкості запису та гнучких можливостей аналітичної обробки.

У межах розробленої системи числові значення температури та вологості зберігаються у полях, тоді як метадані, зокрема ідентифікатор приміщення та статус стану, реалізовані у вигляді тегів. Така структура дозволяє виконувати швидко фільтрацію та агрегацію даних безпосередньо на рівні бази даних. Використання мови Flux забезпечує виконання аналітичних запитів, агрегацій та трансформацій без необхідності передачі великих обсягів сирих даних на сторонні сервіси, що підвищує продуктивність системи та ефективність аналізу історичних даних.

2.6 Оперативне сповіщення та візуалізація даних

Фінальний рівень архітектури системи забезпечує взаємодію з кінцевими користувачами, зокрема персоналом складу та інспекторами якості. Для підвищення ефективності контролю мікроклімату реалізовано поєднання механізмів оперативного сповіщення та інтерактивних інтерфейсів візуалізації, що дозволяє користувачам у режимі реального часу відстежувати стан приміщень і швидко реагувати на відхилення від нормативних параметрів. У разі виникнення критичних ситуацій система автоматично формує та надсилає сповіщення відповідальним особам, забезпечуючи безперервний контур моніторингу та управління.

2.6.1 Алгоритм фільтрації сповіщень у Telegram

Оперативне інформування персоналу про порушення мікроклімату є критично важливим компонентом системи моніторингу фармацевтичного складу, оскільки дозволяє мінімізувати ризики псування лікарських засобів у разі

аварійних ситуацій. У проєкті взаємодія з користувачами реалізована за допомогою Telegram Bot API, який забезпечує доставку повідомлень майже в реальному часі після фіксації відхилення серверною частиною. Передача даних здійснюється через HTTP-запити у форматі JSON, що містять ідентифікатор отримувача та текстове повідомлення [20].

Під час експлуатації подібних IoT-систем виникає проблема надмірної кількості сповіщень у разі тривалого аварійного стану. Оскільки датчики передають телеметрію з високою частотою, система без додаткової логіки формувала б повторні повідомлення на кожному циклі оновлення даних. Це призводить до ефекту інформаційного перевантаження, коли користувач отримує сотні ідентичних сповіщень, що суттєво знижує їхню ефективність і може викликати ігнорування критичних сигналів або вимкнення повідомлень. Тому для забезпечення коректної роботи каналу інформування необхідно було впровадити механізм обмеження частоти відправки повідомлень.

У серверній частині системи для вирішення цієї проблеми реалізовано алгоритм Cooldown, який контролює інтервали між відправленням сповіщень. Його логіка базується на використанні структури даних в оперативній пам'яті сервера, де для кожної унікальної комбінації технологічної зони та параметра моніторингу зберігається час останнього успішного відправлення повідомлення в Telegram. Таким чином формується своєрідний тимчасовий реєстр активних тривог, який дозволяє відстежувати частоту сповіщень у межах кожного окремого каналу.

При виникненні аварійного стану серверний модуль не виконує негайну відправку повідомлення, а спочатку запускає перевірку часових обмежень. Алгоритм обчислює різницю між поточним часом та збереженою часовою міткою для відповідної зони. Якщо ця різниця перевищує заданий у конфігурації поріг (у межах проєкту встановлено 5 хвилин), система дозволяє відправку нового повідомлення до Telegram-каналу. Після успішної доставки оновлюється значення часової мітки, що фіксує момент останнього інформування користувача.

Якщо ж час від останнього повідомлення ще не перевищив встановлений інтервал, система блокує повторне надсилання сповіщення. При цьому подія не

ігнорується повністю – вона може бути зафіксована у внутрішньому лог-файлі сервера для подальшого аналізу або аудиту. Таким чином забезпечується баланс між оперативністю інформування та уникненням надмірного дублювання повідомлень.

2.6.2 Побудова інтерфейсів та контроль доступу в Grafana

Для забезпечення довготривалого моніторингу мікроклімату, аналізу історичних тенденцій та формування звітності для регуляторних перевірок у проєкті використано аналітичну платформу Grafana [16]. Вона виконує роль графічного інтерфейсу користувача та інтегрується з базою даних часових рядів InfluxDB через вбудовані конектори. На відміну від статичних звітів, Grafana забезпечує динамічне оновлення інформації в режимі реального часу, регулярно отримуючи дані зі сховища та відображаючи їх у вигляді інтерактивних візуалізацій.

Розроблений у межах системи дашборд моніторингу базується на поєднанні кількох типів візуальних панелей. Панелі часових рядів використовуються для відображення змін температури та вологості в динаміці, що дозволяє оцінювати стабільність роботи кліматичного обладнання та виявляти довгострокові тенденції. Для оперативного контролю застосовуються панелі миттєвих значень, які відображають актуальні показники у числовому форматі.

Важливим аспектом є забезпечення безпеки та цілісності даних через систему розмежування доступу. У Grafana реалізовано рольову модель, яка поділяє користувачів на адміністративний та користувацький рівні. Адміністратори мають повний доступ до налаштувань системи, у той час як користувачі з обмеженими правами можуть лише переглядати інформацію та формувати звіти, без можливості змінювати конфігурацію або впливати на роботу системи. Такий підхід забезпечує контрольованість середовища, захист від несанкціонованих змін та відповідність вимогам безпеки.

3 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

У цьому розділі наведено детальний опис практичної реалізації IoT-системи моніторингу та контролю умов зберігання лікарських засобів у фармацевтичному складі. Послідовно розглянуто процес побудови мережі в Cisco Packet Tracer, програмування логіки пристроїв, розробку серверного стеку на Python, а також налаштування дашборду в Grafana та Telegram-бота для сповіщень.

3.1 Варіанти використання системи

Розроблена інженерна IoT-система передбачає чітко визначений розподіл функцій та прав доступу між різними категоріями користувачів та автоматичними процесами. Для формалізації та наочного представлення функціональних можливостей комплексу було побудовано діаграму варіантів використання (Use Case), яка описує взаємодію трьох основних акторів: завідувача складу, операційного персоналу та самої автоматизованої системи.

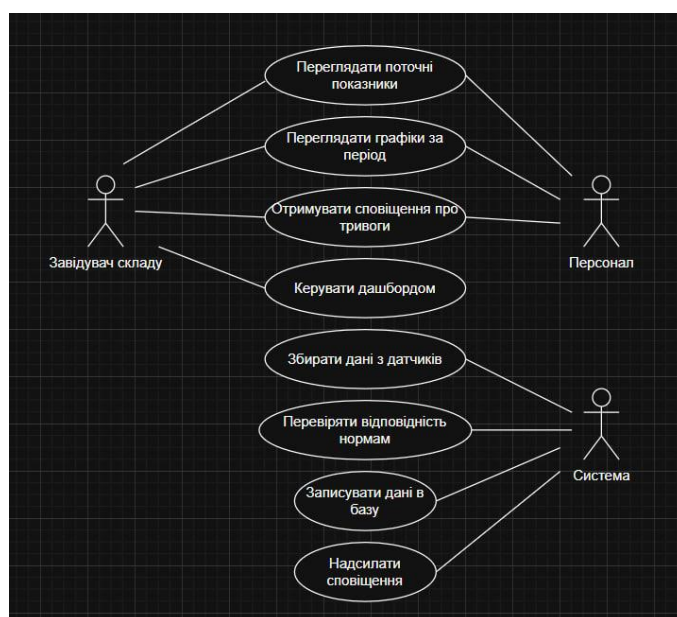


Рисунок 3.1 – Use Case діаграма системи

Завідувач складу – виступає в ролі адміністратора і володіє повним набором прав. Він може переглядати поточні показники мікроклімату, аналізувати історичні графіки за обраний період, отримувати екстрені сповіщення про тривоги у месенджер, а також має ексклюзивне право керувати дашбордом (налаштовувати віджети, змінювати межі зон та керувати доступом).

Персонал – об’єднує уповноважених осіб з якості та чергових технічних співробітників. Цей актор має доступ до базових функцій щоденного контролю: перегляду поточних значень, аналізу графіків за період для ведення журналів та отримання повідомлень про аварії, але не може змінювати конфігурацію самої платформи.

Окремим автономним актором виступає Система, яка функціонує у фоновому режимі без безпосередньої участі людини. Вона відповідає за виконання чотирьох послідовних технологічних процесів: безперервний збір даних із периферійних датчиків, автоматичну перевірку відповідності отриманих значень нормам законодавства, довготривалий запис інформації в базу даних часових рядів та миттєве надсилання сповіщень у разі фіксації критичних відхилень.

3.2 Огляд та обґрунтування програмних засобів реалізації

Для практичного втілення спроектованої архітектури було проведено аналіз та обрано оптимальний стек програмного забезпечення. Як середовище моделювання мережевої інфраструктури розглядалися емулятори GNS3 та EVE-NG, проте вони орієнтовані на класичну маршрутизацію реальних образів ОС і не підтримують компоненти інтернету речей. Оптимальним вибором стало інтерактивне середовище Cisco Packet Tracer [13], яке містить вбудований IoT-модуль із підтримкою програмування датчиків та актуаторів.

Оцінюючи варіанти мови програмування для розробки серверних модулів, розглядалися Go, Node.js та Python. Попри високу продуктивність Go та Node.js, для завдань цього проєкту ключовим критерієм стала наявність зрілих бібліотек.

Мова Python [17] повністю задовольняє цю потребу завдяки готовим пакетам для побудови підключень до брокера [18], для пакетного запису телеметрії [19] та для взаємодії з зовнішніми веб-інтерфейсами [20].

Як MQTT-брокер аналізувалися рішення HiveMQ, RabbitMQ та Eclipse Mosquitto. Оскільки перші два орієнтовані на великі промислові кластери та є занадто складними, було обрано Eclipse Mosquitto [14], який є легковажним, демонструє мінімальні затримки та стабільно працює у вигляді системної служби.

Під час вибору бази даних реляційні рішення та документоорієнтована MongoDB були відхилені через відсутність оптимізації під інтенсивні безперервні потоки даних із часовими мітками. Для вирішення цього завдання обрано спеціалізовану СКБД часових рядів InfluxDB [15], яка забезпечує ефективний запис великої кількості метрик, оптимізоване зберігання даних та підтримує мову запитів Flux для аналізу історичних даних і ведення аудиторського журналу.

Для візуалізації накопиченої інформації обрано аналітичну платформу Grafana [16], яка є індустріальним стандартом моніторингу, має вбудовану підтримку підключення до InfluxDB, дозволяє створювати динамічні інтерактивні дашборди з кодуванням статусів та підтримує розмежування прав доступу.

Для підсистеми тривожних сповіщень було обрано Telegram Bot API, який безкоштовно гарантує доставку структурованих повідомлень на мобільні пристрої персоналу протягом однієї секунди з моменту фіксації аномалії.

3.3 Архітектура та організація проєкту

Розроблена система побудована за багаторівневою архітектурою, яка складається з чотирьох взаємопов'язаних рівнів: периферійного, мережевого, серверного та клієнтського. Такий підхід забезпечує чіткий розподіл функцій між окремими компонентами системи, спрощує її супровід і подальший розвиток, а також дозволяє масштабувати окремі підсистеми без суттєвого впливу на роботу

всього комплексу. Загальну структуру системи та взаємозв'язки між її компонентами наведено на рисунку 3.2.

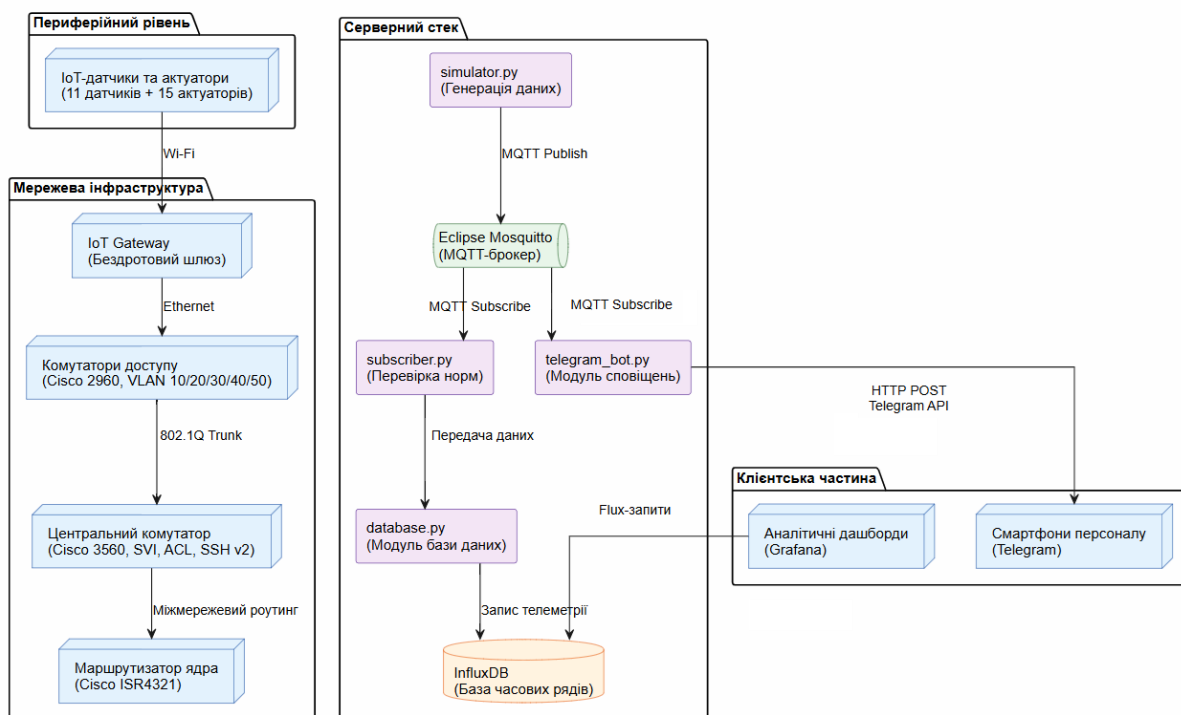


Рисунок 3.2 – Загальна архітектура системи

Периферійний та мережевий рівні реалізовані у середовищі Cisco Packet Tracer. Периферійний рівень включає одинадцять IoT-датчиків і п'ятнадцять виконавчих пристроїв, зокрема кондиціонери, обігрівачі та зволожувачі повітря, які забезпечують моніторинг і підтримання нормативних параметрів мікроклімату в різних зонах складу. Передача даних між пристроями здійснюється через бездротове підключення до шлюзу IoT Gateway, який виконує роль центрального вузла збору інформації.

Мережева інфраструктура побудована за тривірневою ієрархічною моделлю Cisco та включає комутатори рівня доступу Cisco 2960, центральний комутатор третього рівня Cisco 3560 і маршрутизатор Cisco ISR4321. Для логічного розділення технологічних зон використовується механізм VLAN-сегментації, а контроль взаємодії між мережевими сегментами реалізований за допомогою

списків контролю доступу ACL [12]. Таке рішення дозволяє ізолювати окремі зони складу одна від одної, обмежити несанкціонований обмін даними та підвищити загальний рівень безпеки мережі. Використання Cisco Packet Tracer дало можливість змодельовати роботу мережевої інфраструктури, перевірити коректність маршрутизації та протестувати впроваджені механізми захисту.

Серверний рівень розгорнуто на окремому комп'ютері та реалізовано з використанням технологій Python, Eclipse Mosquitto та InfluxDB. Оскільки середовище Cisco Packet Tracer не підтримує передачу телеметричних даних до зовнішніх програмних компонентів у режимі реального часу, для імітації роботи фізичних датчиків було створено програмний симулятор. Його використання дозволило відтворити процес генерації телеметрії та забезпечити функціонування серверного стеку в умовах, максимально наближених до роботи реальної системи.

Взаємодія між серверними компонентами організована за допомогою MQTT-брокера Eclipse Mosquitto, який забезпечує асинхронний обмін повідомленнями між незалежними програмними модулями. Усі налаштування системи, зокрема параметри підключення та нормативні межі температури і вологості для різних зон складу, зберігаються у конфігураційних файлах, що спрощує адміністрування та подальше розширення функціональності системи.

Серверна частина програмного забезпечення складається з кількох взаємодіючих модулів. Модуль `simulator.py` відповідає за генерацію телеметричних даних і їх передачу через MQTT-брокер. Модуль `subscriber.py` приймає повідомлення, виконує перевірку отриманих значень на відповідність встановленим нормативам та формує статуси стану середовища. Для довготривалого зберігання інформації використовується модуль `database.py`, який записує отримані дані до бази часових рядів InfluxDB, забезпечуючи можливість подальшого аналізу та перегляду історичних показників.

Клієнтський рівень забезпечує взаємодію користувачів із системою та представлений двома основними компонентами. Перший компонент реалізований на базі платформи Grafana і призначений для візуалізації поточних та історичних даних у вигляді інтерактивних графіків і інформаційних панелей. Другий

3.4 Налаштування мережевої інфраструктури в Cisco Packet Tracer

35

На робочому полі було розгорнуто маршрутизатор рівня ядра, центральний комутатор третього рівня для розподілу трафіку та п'ять периферійних комутаторів для рівня доступу. Усі двадцять шість пристроїв складу та актуатори були згруповані по своїх функціональних зонах і підключені по бездротовій мережі до шлюзу. Фізичний зв'язок між обладнанням виконано відповідними типами кабелів: для однотипних пристроїв використано перехресні з'єднання, а для підключення шлюзу та сервера до центрального комутатора – прямі.

Налаштування маршрутизатора ядра виконувалось через командний інтерфейс. На пристрої було конфігуровано інтерфейс з'єднання з центральним комутатором, що дозволило виділити ізольовану службову підмережу, а також прописано статичні маршрути для передачі трафіку до внутрішніх підмереж зон складу.

Центральний комутатор рівня розподілу забезпечує міжмережеву взаємодію між усіма сегментами системи. Для реалізації маршрутизації між VLAN на пристрої активовано функцію між VLAN-роутингу та створено шість віртуальних мереж, що відповідають основним зонам складу [12]. Для кожної VLAN налаштовано логічний інтерфейс з унікальною IP-адресою, який виконує роль шлюзу за замовчуванням для відповідного сегмента. Під час конфігурації транкових портів увімкнено передачу VLAN-трафіку з визначенням дозволених віртуальних мереж, що забезпечує ізоляцію сегментів і контроль міжмережових з'єднань.

На комутаторах доступу було обов'язково створено локальні бази даних, що містили як номер власної зони, так і управлінську мережу. Це дозволило уникнути поширених помилок із втратою віддаленого доступу до керування обладнанням. Контроль стану магістральних каналів зв'язку між комутаторами рівня розподілу та доступу здійснювався шляхом налаштування активних портів у режим транзиту з чітким переліком дозволених віртуальних мереж.

3.5 Налаштування безпеки мережевої інфраструктури

Система захисту даних моніторингу фармацевтичного складу реалізована на кількох рівнях відповідно до концепції ешелонованої оборони (Defense in Depth). Для захисту розгорнутих каналів віддаленого адміністративного доступу до обладнання було повністю вимкнено вразливий протокол Telnet. Замість нього на маршрутизаторі та центральному комутаторі активовано криптографічний протокол SSH версії 2 [11]. Під час налаштування було визначено доменне ім'я пристроїв, згенеровано криптографічні ключі RSA розміром 2048 біт та примусово обмежено лінії підключення VTU виключно безпечними сесіями. Додатково налаштовано автоматичний тайм-аут для скидання неактивних сесій адміністрування тривалістю десять хвилин.

Для захисту фізичного рівня доступу на портах комутатора розподілу, де підключені бездротовий шлюз IoT Gateway та центральний сервер, застосовано технологію Port Security [10]. В її конфігурації встановлено суворе обмеження на підключення лише однієї апаратної адреси на порт. Режим протидії загрозам налаштовано на повне вимкнення інтерфейсу (режим Shutdown), що автоматично переводить порт у заблокований стан при спробі несанкціонованої заміни пристрою стороннім комп'ютером. Завдяки активованій функції Sticky MAC комутатор автоматично фіксує легітимні MAC-адреси обладнання та зберігає захист у своїй конфігурації.

Логічна фільтрація трафіку між віртуальними мережами складу реалізована за допомогою розширеного списку контролю доступу (ACL) під назвою IOT-POLICY, який налічує 26 послідовних правил [12]. Цей список закріплений на вхідному напрямку логічних інтерфейсів віртуальних мереж зон зберігання. Логіка правил дозволяє датчикам та актуаторам надсилати IP-пакети виключно на фіксовану адресу MQTT-сервера. Водночас наступний великий блок інструкцій списку повністю блокує будь-яку пряму міжмережеву комунікацію між різними кімнатами складу, наприклад, трафік між холодильною камерою та загальним складом чи зоною карантину. Додаткова безпека бездротового сегмента посилена

шифруванням WPA2-PSK на точці доступу шлюзу, а магістральні канали захищені обмеженням дозволених номерів VLAN на Trunk-портах.

3.6 Програмування логіки IoT-датчиків та актуаторів

Периферійний рівень системи містить двадцять шість індустріальних пристроїв інтернету речей, які взаємодіють з інфраструктурою через бездротову мережу шлюзу IoT Gateway. Одинадцять модулів виконують функції пасивного збору метрик середовища, а п'ятнадцять пристроїв є актуаторами активного контролю, розподіленими по три одиниці на кожне складське приміщення: кондиціонер, обігрівач та зволожувач повітря. Програмне забезпечення для кожного мікроконтролера написано на мові JavaScript з використанням внутрішнього прикладного інтерфейсу середовища моделювання Cisco Packet Tracer.

Внутрішня прошивка кожного пристрою ініціалізується за допомогою функції `IoEClient.setup()`, яка реєструє модуль на локальному шлюзі та визначає структуру його станів для відображення температури чи вологості. Основний робочий цикл, що безперервно імітує поведінку реального мікроконтролера, реалізований всередині функції `loop()`. У цьому циклі датчик зчитує поточні фізичні показники мікроклімату через метод `Environment.get()`, порівнює отримане значення з нормативними пороговими константами своєї зони та транслює результат на шлюз за допомогою методу `IoEClient.reportStates()`. Для оптимізації навантаження на бездротові канали зв'язку функція `delay()` встановлює фіксований інтервал опитування, який становить 30 секунд для сенсорів моніторингу та 15 секунд для виконавчих актуаторів. Блок-схема розробленого алгоритму функціонування IoT-пристроїв наведена на рисунку 3.4.

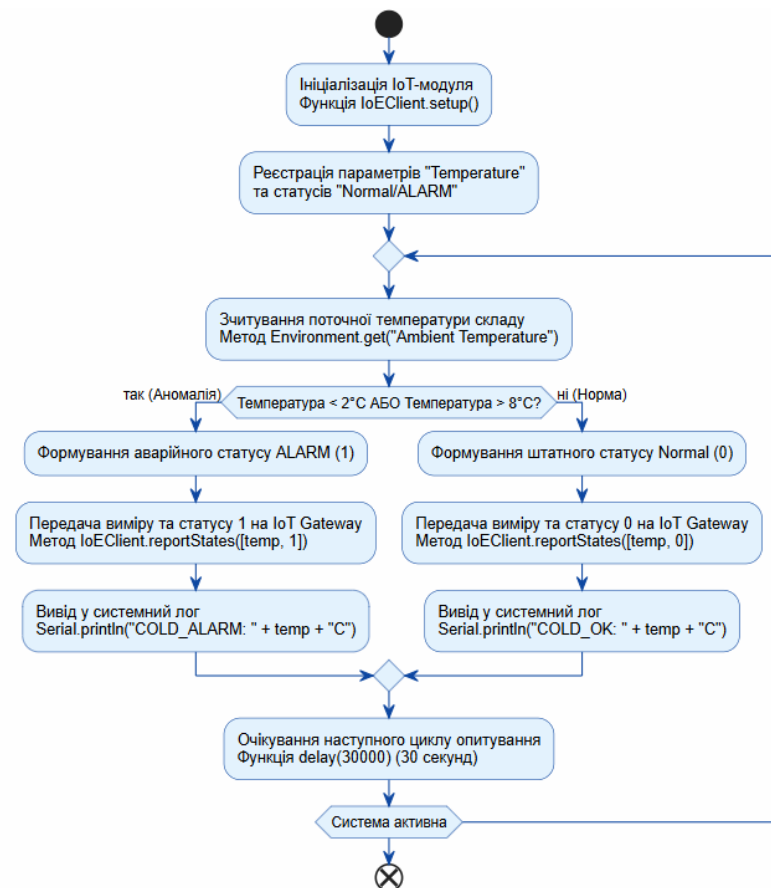


Рисунок 3.4 – Блок-схема алгоритму функціонування програмної логіки IoT-датчиків

Логіка функціонування виконавчих акuatorів побудована на базі автоматичного реагування на відхилення показників. Кондиціонер активується виключно тоді, коли фактична температура перевищує максимальну межу кімнати, а обігрівач вмикається лише при падінні температури нижче мінімального порогу, що виключає ризик помилкового спрацювання обладнання. Зволожувач аналогічним чином контролює рівень вологості повітря.

Після запуску програм на мікроконтролерах усі периферійні пристрої проходять реєстрацію у веб-інтерфейсі шлюзу IoT Gateway за адресою 192.168.200.1, який доступний через браузер смартфона або планшета у внутрішній бездротовій мережі WiFi під назвою PHARMA-IOT для відображення актуального стану кожного пристрою в реальному часі.

3.7 Реалізація серверної логіки обробки телеметрії

Серверна частина системи реалізована у вигляді набору незалежних програмних процесів, написаних мовою Python, які виконуються паралельно та обмінюються даними через централізований MQTT-брокер Eclipse Mosquitto. Така архітектура забезпечує слабку зв'язаність компонентів системи, що дозволяє кожному модулю працювати автономно, спрощує масштабування та підвищує загальну відмовостійкість рішення. MQTT-брокер розгорнуто як локальний сервіс у межах внутрішньої мережі із використанням стандартного порту 1883, що забезпечує стабільний обмін повідомленнями між усіма компонентами системи.

Для підвищення ефективності обміну даними використано механізм retained-повідомлень, який дозволяє зберігати останні актуальні значення для кожної теми. Завдяки цьому нові підписники отримують поточний стан системи одразу після підключення, без необхідності очікування наступного циклу передачі даних від датчиків. Такий підхід є особливо важливим для компонентів візуалізації та моніторингу, оскільки забезпечує миттєве відображення актуального стану системи в дашбордах Grafana.

Модуль `simulator.py` виконує роль джерела телеметричних даних і моделює роботу IoT-датчиків фармацевтичного складу. Він забезпечує формування значень температури та відносної вологості для кожної контрольованої зони. Алгоритм налаштовано так, що близько 95% згенерованих значень знаходяться в межах норми, тоді як 5% навмисно виходять за допустимі межі для перевірки роботи механізмів виявлення аварій. Отримані дані передаються до MQTT-брокера у вигляді структурованих повідомлень із використанням ієрархічної системи тем за шаблоном `pharmacy/{room_id}/{parameter}` [18].

Обробка телеметричних даних реалізована у модулі `subscriber.py`, який підписується на всі повідомлення системи за допомогою універсальної маски `pharmacy/#`. Після отримання даних виконується їх розбір, визначається зона та тип параметра, після чого здійснюється перевірка відповідності значень встановленим нормам, які імпортуються з єдиного файлу конфігурації `config/settings.py`. За

результатами перевірки кожному вимірюванню присвоюється статус нормального стану (OK) або аварійного відхилення (ALARM), що використовується іншими компонентами системи.

3.8 Проєктування логічної моделі бази даних InfluxDB

Збереження історичних даних реалізовано через модуль `database.py`, який забезпечує інтеграцію з базою даних часових рядів InfluxDB. Для забезпечення високої швидкодії при формуванні аудиторської звітності була розроблена оптимізована логічна модель даних, яка наочно представлена на рисунку 3.5.

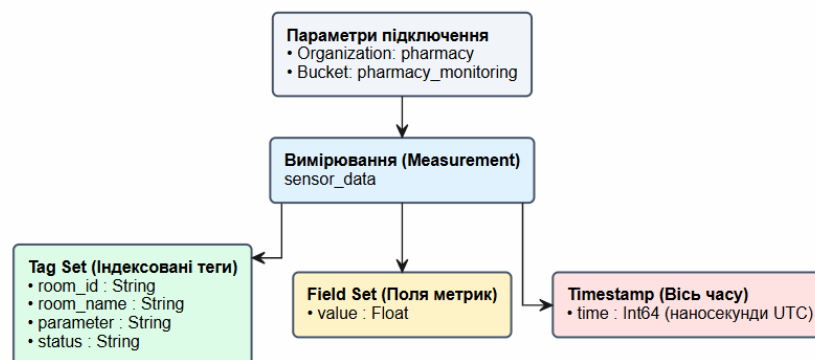


Рисунок 3.5 – Логічна модель структури даних часових рядів в InfluxDB

Відповідно до розробленої схеми, кожне вимірювання трансформується у спеціальну точку даних (Point) всередині загального вимірювання `sensor_data`. Запис має чітко розділену структуру: він містить індексовані метадані (теги `room_id`, `room_name`, `parameter`, `status`), неіндексоване числове поле `value` з фактичним значенням, а також автоматичну часову мітку сервера у наносекундах. Винесення статусу безпеки саме в індексований тег (Tag Set), а не в звичайне поле (Field Set), є ключовим архітектурним рішенням. Як демонструє модель, це дозволяє базі даних виконувати фільтрацію та агрегацію відхилень по індексу в

оперативній пам'яті без повного сканування диска. Такий підхід суттєво пришвидшує формування аудиторської звітності для інспектуючих органів.

3.9 Побудова аналітичного інтерфейсу в середовищі Grafana

Аналітична платформа Grafana виступає основним графічним інтерфейсом користувача для персоналу фармацевтичного складу та надає повну картину поточного стану мікроклімату в реальному часі. Підключення до бази даних InfluxDB виконується через інтегроване джерело даних типу Data Source з автентифікацією за допомогою секретного токена доступу. Взаємодія зі сховищем здійснюється на базі конвеєрних запитів мовою Flux.

Розроблений диспетчерський дашборд містить чотири інтерактивні панелі, розташовані у два функціональні ряди. Верхній ряд сформовано з двох графічних панелей типу Time Series, які відображають динаміку коливань температури та відносної вологості повітря у вигляді лінійних графіків за обраний користувачем проміжок часу, де кожна зона представлена окремим кольором. Вбудований у панелі Flux-запит виконує операцію агрегації вікон даних, яка обчислює середнє значення метрик для очищення графіків від мережевих шумів, та застосовує функцію підстановки нормативних меж безпосередньо в назву легенди ліній.

Нижній ряд містить дві панелі типу Stat, які відображають актуальні показники мікроклімату великим шрифтом окремо для кожного приміщення. На відміну від графіків, ці панелі використовують функцію отримання останнього значення для миттєвого відображення найсвіжішої точки даних, що надійшла від датчиків.

Для забезпечення безпеки даних та захисту конфігурації моніторингу від неавторизованих змін у системі Grafana налаштовано рольову модель доступу, яка розділяє облікові записи на профілі адміністратора з повними правами редагування та профілі рядових спостерігачів (Viewer) з правами виключно на перегляд та генерацію історичних звітів.

3.10 Розробка модуля сповіщень Telegram Bot API

Telegram-бот є фінальним компонентом системи, який забезпечує операційне інформування відповідальних осіб про виникнення позаштатних ситуацій. Модуль реалізований у скрипті `src/telegram_bot.py` і функціонує як третій паралельний серверний процес. Він підключається до загального джерела даних – MQTT-брокера Eclipse Mosquitto – окремим потоком і підписується на потік телеметрії за маскою `pharmacy/#`. Обробка кожного надходячого повідомлення ініціюється через метод-обробник `on_message()`, який виконує первинний парсинг топіку та отриманого значення.

Для наочного представлення логіки роботи модуля, маршрутизації даних та захисту від інформаційного перевантаження було розроблено структурну блок-схему алгоритму, яку наведено на рисунку 3.6.

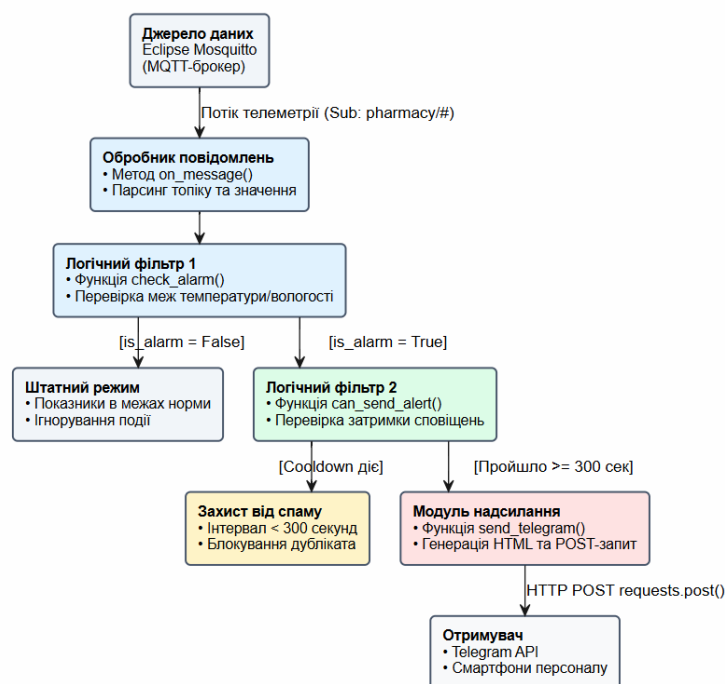


Рисунок 3.6 – Блок-схема алгоритму обробки повідомлень та фільтрації сповіщень

Як видно з діаграми, архітектура модуля базується на послідовній двоступеневій системі фільтрації. Перший рівень (логічний фільтр 1) реалізований

у функції `check_alarm()`. Цей етап алгоритму перевіряє вихід показників за межі норми. Якщо система фіксує штатний режим (показники в нормі), подія просто ігнорується. Реакція програми орієнтована виключно на фіксацію статусів тривоги.

При виявленні відхилення управління передається на другий рівень – логічний фільтр 2, реалізований функцією `can_send_alert()`. Його головним завданням є впровадження захисного механізму `Cooldown` для запобігання спаму під час тривалих аварій. Бот звертається до динамічного словникового масиву в оперативній пам'яті сервера, де зберігається час останнього надісланого сповіщення для кожної пари параметрів і зон складу. Модуль надсилання активується лише у тому випадку, якщо з моменту попереднього інформування минуло 300 або більше секунд. Якщо ж інтервал менший – дублікат блокується алгоритмом захисту від спаму, а подія лише логується всередині сервера.

У разі успішного проходження обох фільтрів викликається функція `send_telegram()`. Екстрені повідомлення форматуються за допомогою вбудованої HTML-розмітки та надсилаються через HTTP POST-запит (метод `requests.post()`) [20] до Telegram API. У структурованому тексті, що доставляється на смартфони персоналу, чітко виділено назву зони, тип порушеного параметра, фактичне значення, нормативний діапазон та точний час аварії. Додатково, при кожному перезапуску сервера бот автоматично надсилає стартове повідомлення, яке підтверджує успішну ініціалізацію контуру сповіщень.

4 ТЕСТУВАННЯ ТА ДЕМОНСТРАЦІЯ РОБОТИ СИСТЕМИ

У цьому розділі наведено системні вимоги до розгортання розробленої системи та покроковий опис процесу інсталяції її програмних компонентів. Представлено діаграму варіантів використання та детальні сценарії роботи автоматики для різних категорій користувачів. Також наведено результати комплексного тестування всіх складових компонентів мережевого та серверного стеку з підтвердженням повної відповідності встановленим нормативним вимогам.

4.1 Системні вимоги та інсталяція програмного забезпечення

Для коректного розгортання та роботи розробленої системи необхідне відповідне апаратне та програмне забезпечення. З точки зору апаратних вимог достатньо комп'ютера з тактовою частотою процесора не менше 2 ГГц, оперативною пам'яттю не менше 8 ГБ та вільним місцем на диску не менше 10 ГБ. Рекомендований об'єм оперативної пам'яті 16 ГБ забезпечує комфортну одночасну роботу Cisco Packet Tracer, InfluxDB, Grafana та трьох Python процесів без деградації продуктивності системи. Наявність стабільного інтернет-з'єднання є обов'язковою вимогою для роботи Telegram бота оскільки надсилання сповіщень здійснюється через HTTPS запити до серверів Telegram. Програмні компоненти системи та їх версії наведені у таблиці 4.1.

Таблиця 4.1 – Програмні компоненти системи

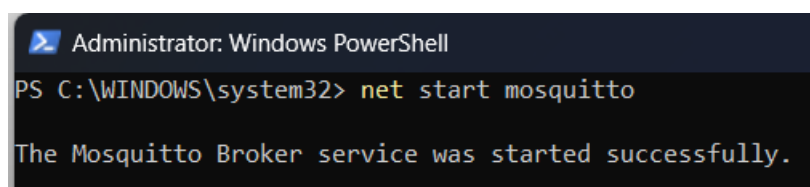
Компонент системи	Версія програмного забезпечення	Функціональне призначення в проєкті
Cisco Packet Tracer	8.x	Моделювання та верифікація мережевої інфраструктури

Продовження таблиці 4.1.

Python	3.11	Розробка серверної логіки обробки та симуляції даних
Eclipse Mosquitto	2.x	MQTT-брокер для асинхронного обміну повідомленнями
InfluxDB	2.9	База даних часових рядів для довготривалого збереження
Grafana	11.x	Інтерактивна система візуалізації та моніторингу

Процес інсталяції та первинного конфігурування компонентів виконується у суворо визначеній послідовності, що забезпечує коректну ініціалізацію залежностей між окремими службами.

На першому етапі здійснюється встановлення MQTT-брокера Eclipse Mosquitto за допомогою офіційного інсталлятора. Під час розгортання в операційній системі активуються опції реєстрації брокера як постійної системної служби та автоматичного завантаження компонентів середовища виконання Visual Studio Runtime, які є обов'язковими для роботи мережесокетів програми. Після завершення копіювання файлів запуск служби виконується через консоль керування з правами адміністратора. Успішний запуск і перехід брокера в робочий стан підтверджуються відповідним сервісним повідомленням у вікні терміналу, що наочно продемонстровано на рисунку 4.1.



```

Administrator: Windows PowerShell
PS C:\WINDOWS\system32> net start mosquitto

The Mosquitto Broker service was started successfully.
  
```

Рисунок 4.1 – Результат успішного запуску служби Eclipse Mosquitto

Другим кроком виконується розгортання бази даних часових рядів InfluxDB шляхом розпакування архіву у цільову системну директорію та запуску

виконуваного файла сервера. Після цього через веб-інтерфейс здійснюється первинне налаштування сховища, де створюються назва організації, bucket для збереження даних та унікальний токен автентифікації API, необхідний для подальшого підключення Grafana та клієнтських Python-скриптів. Початкове вікно успішної ініціалізації бази даних наведено на рисунку 4.2.

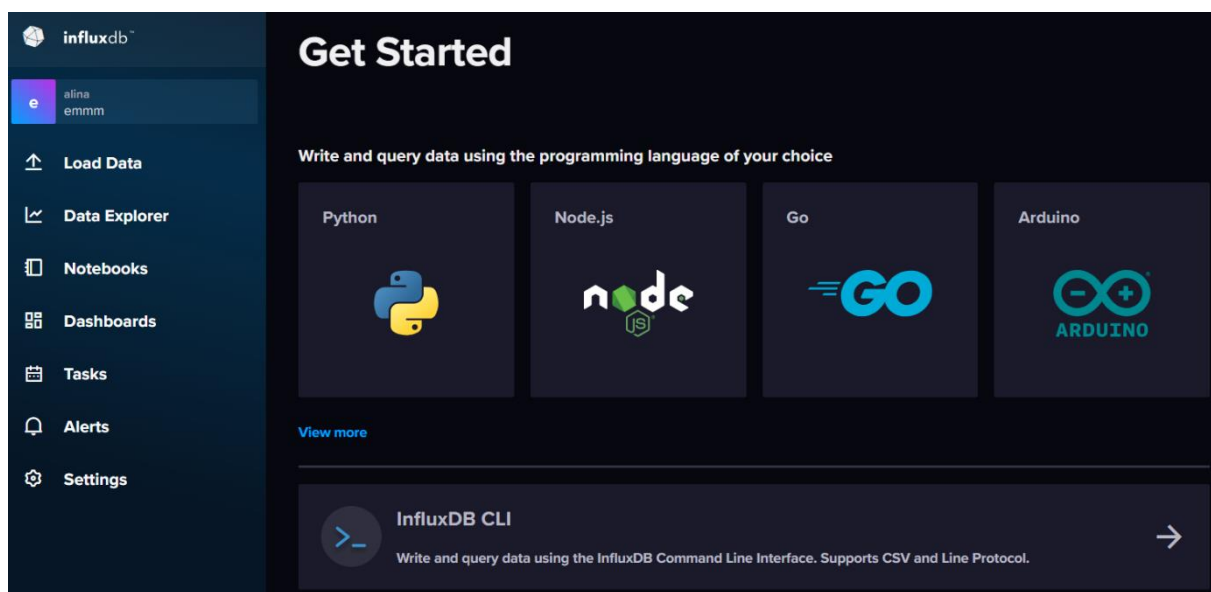


Рисунок 4.2 – Веб-інтерфейс InfluxDB після першого запуску

Встановлення Grafana виконується через офіційний інстальатор з сайту grafana.com. Grafana встановлюється як системна служба Windows і запускається командою `net start grafana` від імені адміністратора. Після запуску веб-інтерфейс доступний за адресою `http://localhost:3000`. При першому вході використовуються облікові дані за замовчуванням `admin/admin` після чого система пропонує змінити пароль адміністратора. Підключення до InfluxDB виконується через розділ `Connections => Data Sources`, де необхідно вказати URL бази даних, API токен, назву організації та bucket.

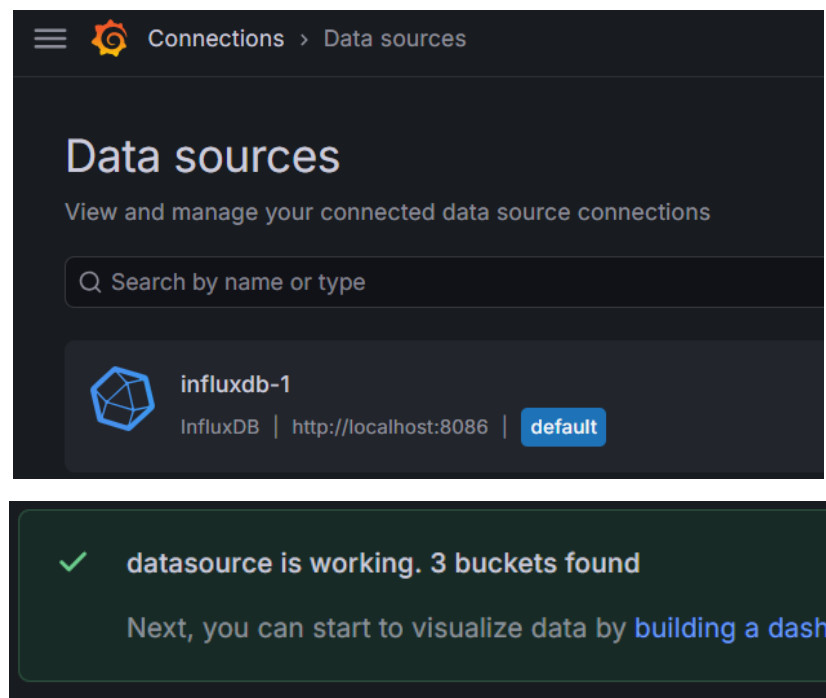


Рисунок 4.3 – Успішне підключення джерела даних InfluxDB у Grafana

Фінальним етапом є встановлення інтерпретатора мови Python з обов'язковою активацією опції додавання шляхів до системних змінних середовища. За допомогою вбудованого менеджера пакетів в один рядок встановлюються сторонні бібліотеки для підтримки протоколу MQTT, клієнта бази даних та обробки HTTP-запитів месенджера.

Після виконання всіх дій залишається відкрити три термінали і запустити `main.py`, `src/subscriber.py` та `src/telegram_bot.py`.

4.2 Результати комплексного тестування компонентів

У цьому підрозділі наведено результати комплексного тестування та верифікації всіх рівнів розробленої IoT-системи. Експериментальні випробування охоплюють перевірку працездатності мережевого контуру, надійності налаштованих засобів безпеки, а також точності функціонування периферійних пристроїв та серверних модулів обробки телеметрії.

4.2.1 Тестування мережевої інфраструктури та маршрутизації

Перевірка працездатності мережевого контуру, налаштованого в Cisco Packet Tracer, виконувалася за допомогою введення команд верифікації в консолі командного рядка (CLI) центрального комутатора третього рівня CORE-SW-L3 та маршрутизатора ядра. Для перевірки успішності створення, активації та логічного розподілу всіх шести віртуальних локальних мереж використовувалася команда `show vlan brief`. Консольний результат її виконання, що підтверджує правильне налаштування баз даних VLAN, наведено на рисунку 4.4.

```
CORE-SW-L3#show vlan brief
```

VLAN	Name	Status	Ports
1	default	active	Fa0/6, Fa0/7, Fa0/8, Fa0/9 Fa0/12, Fa0/13, Fa0/14, Fa0/15 Fa0/16, Fa0/17, Fa0/18, Fa0/19 Fa0/20, Fa0/21, Fa0/22, Fa0/23 Fa0/24, Gig0/2
10	COLD-STORAGE	active	
20	GENERAL-STORAGE	active	
30	COOL-STORAGE	active	
40	QUARANTINE	active	
50	DISPATCH	active	
100	MANAGEMENT	active	Fa0/10, Fa0/11
1002	fddi-default	active	
1003	token-ring-default	active	
1004	fddinet-default	active	
1005	trnet-default	active	

Рисунок 4.4 – Перелік налаштованих VLAN

Контроль стану магістральних транзитних каналів зв'язку між комутаторами розподілу та доступу здійснювався за допомогою перегляду активних інтерфейсів через команду `show interfaces trunk`. Скріншот консолі з підтвердженням увімкнення відповідного режиму транзиту (trunking) та переліком дозволених віртуальних мереж на кожному інтерфейсі представлено на рисунку 4.5.

```

CORE-SW-L3#show interfaces trunk
Port      Mode      Encapsulation  Status        Native vlan
Fa0/1     on        802.1q         trunking      1
Fa0/2     on        802.1q         trunking      1
Fa0/3     on        802.1q         trunking      1
Fa0/4     on        802.1q         trunking      1
Fa0/5     on        802.1q         trunking      1

Port      Vlans allowed on trunk
Fa0/1     10,100
Fa0/2     20,100
Fa0/3     30,100
Fa0/4     40,100
Fa0/5     50,100

Port      Vlans allowed and active in management domain
Fa0/1     10,100
Fa0/2     20,100
Fa0/3     30,100
Fa0/4     40,100
Fa0/5     50,100

Port      Vlans in spanning tree forwarding state and not pruned
Fa0/1     10,100
Fa0/2     20,100
Fa0/3     30,100
Fa0/4     40,100
Fa0/5     50,100

```

Рисунок 4.5 – Демонстрація trunk-портів

Перевірка поточного стану налаштованих віртуальних інтерфейсів комутатора та їхніх IP-адрес виконувалася через коротку зведену таблицю інтерфейсів пристрою за допомогою команди `show ip interface brief`. Підтвердження того, що всі логічні шлюзи для зон складу успішно активовані та знаходяться в робочому стані, відображено на рисунку 4.6.

```

CORE-SW-L3#show ip interface brief
Interface      IP-Address      OK? Method Status        Protocol
FastEthernet0/1 unassigned      YES unset    up            up
FastEthernet0/2 unassigned      YES unset    up            up
FastEthernet0/3 unassigned      YES unset    up            up
FastEthernet0/4 unassigned      YES unset    up            up
FastEthernet0/5 unassigned      YES unset    up            up
FastEthernet0/6 unassigned      YES unset    down          down
FastEthernet0/7 unassigned      YES unset    down          down
FastEthernet0/8 unassigned      YES unset    down          down
FastEthernet0/9 unassigned      YES unset    down          down
FastEthernet0/10 unassigned      YES unset    up            up
FastEthernet0/11 unassigned      YES unset    up            up
FastEthernet0/12 unassigned      YES unset    down          down
FastEthernet0/13 unassigned      YES unset    down          down
FastEthernet0/14 unassigned      YES unset    down          down
FastEthernet0/15 unassigned      YES unset    down          down
FastEthernet0/16 unassigned      YES unset    down          down
FastEthernet0/17 unassigned      YES unset    down          down
FastEthernet0/18 unassigned      YES unset    down          down
FastEthernet0/19 unassigned      YES unset    down          down
FastEthernet0/20 unassigned      YES unset    down          down
FastEthernet0/21 unassigned      YES unset    down          down
FastEthernet0/22 unassigned      YES unset    down          down
FastEthernet0/23 unassigned      YES unset    down          down
FastEthernet0/24 unassigned      YES unset    down          down
GigabitEthernet0/1 172.16.0.2     YES manual   up            up
GigabitEthernet0/2 unassigned      YES unset    down          down
Vlan1          unassigned      YES unset    administratively down down
Vlan10         192.168.10.254 YES manual   up            up
Vlan20         192.168.20.254 YES manual   up            up
Vlan30         192.168.30.254 YES manual   up            up
Vlan40         192.168.40.254 YES manual   up            up
Vlan50         192.168.50.254 YES manual   up            up
Vlan100        192.168.100.2  YES manual   up            up

```

Рисунок 4.6 – Стан логічних SVI-інтерфейсів

Аналіз поточної таблиці маршрутизації комутатора третього рівня дозволив перевірити наявність записів про підключені мережі та маршрут за замовчуванням до ядра мережі за допомогою команди `show ip route`. Отриманий розподіл діючих маршрутів у консолі пристрою наведено на рисунку 4.7.

```
172.16.0.0/30 is subnetted, 1 subnets
C    172.16.0.0 is directly connected, GigabitEthernet0/1
C    192.168.10.0/24 is directly connected, Vlan10
C    192.168.20.0/24 is directly connected, Vlan20
C    192.168.30.0/24 is directly connected, Vlan30
C    192.168.40.0/24 is directly connected, Vlan40
C    192.168.50.0/24 is directly connected, Vlan50
C    192.168.100.0/24 is directly connected, Vlan100
S*   0.0.0.0/0 [1/0] via 172.16.0.1
```

Рисунок 4.7 – Розподіл діючих маршрутів у таблиці маршрутизації комутатора

Тестування мережевої зв'язності виконувалось введенням команд `ping` з центрального комутатора третього рівня CORE-SW-L3 до інтерфейсів інших мережевих пристроїв та підмереж системи.

```
CORE-SW-L3#ping 172.16.0.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.0.2, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 6/7/11 ms

CORE-SW-L3#ping 192.168.100.10
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.100.10, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/0 ms

CORE-SW-L3#ping 192.168.100.11
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.100.11, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/2/12 ms

CORE-SW-L3#ping 192.168.100.12
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.100.12, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/0 ms

CORE-SW-L3#ping 192.168.100.13
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.100.13, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/0 ms

CORE-SW-L3#ping 192.168.100.14
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.100.14, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/2/12 ms

CORE-SW-L3#ping 192.168.100.15
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.100.15, timeout is 2 seconds:
!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 0/0/0 ms
```

Рисунок 4.8 – Лог успішного виконання ping-тестів на комутаторі CORE-SW-L3

Таким чином, проведене тестування підтвердило коректність налаштування мережевої інфраструктури в Cisco Packet Tracer. Всі VLAN були успішно створені та ізольовані, транзитні з'єднання працюють у режимі trunking, а SVI-інтерфейси та маршрутизація налаштовані правильно. Результати ring-тестів засвідчили повну зв'язність між вузлами мережі з показником успішності 100%, що підтверджує стабільну роботу всієї топології.

4.2.2 Тестування системи безпеки мережі

Тестування впроваджених механізмів безпеки мережі підтвердило стійкість контуру до поширених векторів кібератак. Активність SSH версії 2 підтверджується командою `show ip ssh` на маршрутизаторі та L3 комутаторі. Команда відображає версію протоколу, час очікування автентифікації та кількість дозволених спроб входу.

```
CORE-SW-L3#show ip ssh
SSH Enabled - version 2.0
Authentication timeout: 120 secs; Authentication retries: 3

PHARMA-ROUTER#show ip ssh
SSH Enabled - version 2.0
Authentication timeout: 120 secs; Authentication retries: 3
```

Рисунок 4.9 – Підтвердження активності SSH v2

При спробі провести негативний тест – встановити віддалене з'єднання за застарілим протоколом Telnet з будь-якого комп'ютера – комутатор та маршрутизатор миттєво відхиляли запит, видаючи сервісне повідомлення про заборону з'єднання.

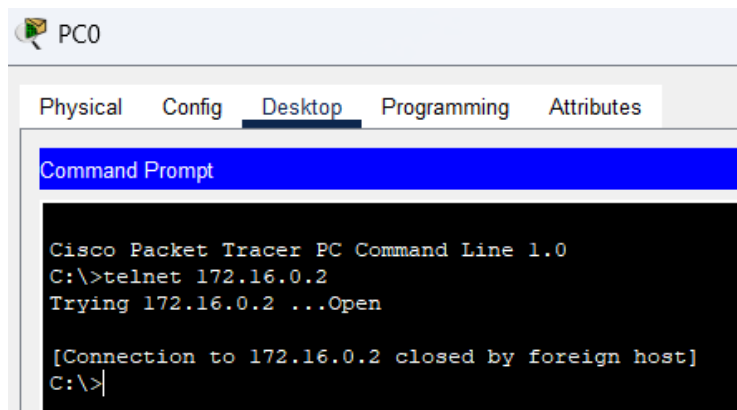


Рисунок 4.10 – Відхилений запит

Захист критичних портів підключення реалізований через механізм Port Security. Його стан перевіряється командою `show port-security interface` для кожного захищеного порту.

```
CORE-SW-L3#show port-security interface FastEthernet0/10
Port Security          : Enabled
Port Status            : Secure-up
Violation Mode         : Shutdown
Aging Time             : 0 mins
Aging Type             : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses  : 1
Total MAC Addresses    : 1
Configured MAC Addresses : 0
Sticky MAC Addresses   : 1
Last Source Address:Vlan : 0000.0000.0000:0
Security Violation Count : 0

CORE-SW-L3#show port-security interface FastEthernet0/11
Port Security          : Enabled
Port Status            : Secure-up
Violation Mode         : Shutdown
Aging Time             : 0 mins
Aging Type             : Absolute
SecureStatic Address Aging : Disabled
Maximum MAC Addresses  : 1
Total MAC Addresses    : 1
Configured MAC Addresses : 0
Sticky MAC Addresses   : 1
Last Source Address:Vlan : 0000.0000.0000:0
Security Violation Count : 0
```

Рисунок 4.11 – Стан Port Security на портах Fa0/10 та Fa0/11 комутатора CORE-SW-L3

З рисунку 4.11 видно, що Port Security активний на обох портах – статус Secure-up підтверджує коректну роботу механізму. Параметр Maximum MAC Addresses зі значенням 1 дозволяє підключення лише одного пристрою на порт.

Режим порушення Shutdown забезпечує автоматичне вимкнення порту при спробі підключення несанкціонованого пристрою. Sticky MAC Addresses зі значенням 1 підтверджує, що MAC-адреса підключеного пристрою автоматично запам'ятована. Лічильник порушень Security Violation Count дорівнює 0, що свідчить про відсутність спроб несанкціонованого доступу.

Логічна фільтрація трафіку між віртуальними мережами складу реалізована за допомогою розширеного списку контролю доступу, який налічує 26 послідовних правил. Вивід налаштованих правил безпеки перевірено командою `show ip access-lists`, наведеною на рисунку 4.12.

```
CORE-SW-L3#show ip access-lists
Extended IP access list IOT-POLICY
 10 permit ip 192.168.10.0 0.0.0.255 host 192.168.100.10
 20 permit ip 192.168.20.0 0.0.0.255 host 192.168.100.10
 30 permit ip 192.168.30.0 0.0.0.255 host 192.168.100.10
 40 permit ip 192.168.40.0 0.0.0.255 host 192.168.100.10
 50 permit ip 192.168.50.0 0.0.0.255 host 192.168.100.10
 60 deny ip 192.168.10.0 0.0.0.255 192.168.20.0 0.0.0.255
 70 deny ip 192.168.10.0 0.0.0.255 192.168.30.0 0.0.0.255
 80 deny ip 192.168.10.0 0.0.0.255 192.168.40.0 0.0.0.255
 90 deny ip 192.168.10.0 0.0.0.255 192.168.50.0 0.0.0.255
100 deny ip 192.168.20.0 0.0.0.255 192.168.10.0 0.0.0.255
110 deny ip 192.168.20.0 0.0.0.255 192.168.30.0 0.0.0.255
120 deny ip 192.168.20.0 0.0.0.255 192.168.40.0 0.0.0.255
130 deny ip 192.168.20.0 0.0.0.255 192.168.50.0 0.0.0.255
140 deny ip 192.168.30.0 0.0.0.255 192.168.10.0 0.0.0.255
150 deny ip 192.168.30.0 0.0.0.255 192.168.20.0 0.0.0.255
160 deny ip 192.168.30.0 0.0.0.255 192.168.40.0 0.0.0.255
170 deny ip 192.168.30.0 0.0.0.255 192.168.50.0 0.0.0.255
180 deny ip 192.168.40.0 0.0.0.255 192.168.10.0 0.0.0.255
190 deny ip 192.168.40.0 0.0.0.255 192.168.20.0 0.0.0.255
200 deny ip 192.168.40.0 0.0.0.255 192.168.30.0 0.0.0.255
210 deny ip 192.168.40.0 0.0.0.255 192.168.50.0 0.0.0.255
220 deny ip 192.168.50.0 0.0.0.255 192.168.10.0 0.0.0.255
230 deny ip 192.168.50.0 0.0.0.255 192.168.20.0 0.0.0.255
240 deny ip 192.168.50.0 0.0.0.255 192.168.30.0 0.0.0.255
250 deny ip 192.168.50.0 0.0.0.255 192.168.40.0 0.0.0.255
260 permit ip any any
```

Рисунок 4.12 – Список контролю доступу IOT-POLICY

Правила 10–50 дозволяють трафік від кожної IoT зони виключно до MQTT сервера з адресою 192.168.100.10 – тобто датчики можуть передавати дані тільки на сервер і нікуди більше. Правила 60–250 забороняють будь-яку пряму комунікацію між зонами складу, наприклад, трафік з холодильної камери (192.168.10.0) до загального складу (192.168.20.0) або карантину (192.168.40.0) повністю заблокований в обох напрямках. Завершальне правило 260 `permit ip any`

ану дозволяє весь інший трафік, що не потрапив під попередні правила – це забезпечує нормальну роботу управлінської мережі та адміністрування пристроїв.

4.2.3 Тестування IoT-датчиків та актуаторів

Верифікація коректності функціонування периферійного рівня пристроїв інтернету речей здійснювалася шляхом штучного моделювання екстремальних температурних умов усередині симуляційних приміщень складу та контролю реакції логіки мікроконтролерів.

Результат роботи датчика відображається в консолі. При значеннях в межах норми датчик виводить повідомлення з позначкою ОК, при відхиленні – ALARM з фактичним значенням параметра. На рисунку 4.13 видно, що датчик коректно виявляє відхилення температури від допустимого діапазону 2–8°C для холодильної камери.

```
Starting cold-temp-1 (JavaScript)...  
COLD_ALARM: 8.701379776000977C  
COLD_OK: 7.993060111999512C
```

Рисунок 4.13 – Консольний вивід датчика cold-temp-1 під час роботи

У відповідь на появу цього аварійного статусу мікроконтролер управління кондиціонером холодильної камери миттєво змінив логічний стан свого вихідного інтерфейсу, запустивши режим активного охолодження повітря. Текст консольних повідомлень актуатора, що підтверджує автоматичне відновлення мікроклімату без участі оператора, представлено на рисунку 4.15.

```
AC OFF - OK: 7.1805500984191895C  
AC OFF - OK: 7.888879776000977C  
AC COOLING ON: 8.555549621582031C
```

Рисунок 4.14 – Консольний вивід кондиціонера під час роботи

Усі двадцять шість розгорнутих периферійних вузлів успішно підтримували бездротовий зв'язок зі шлюзом. Для контролю стану реєстрації модулів було виконано підключення до інтегрованої веб-сторінки адміністрування IoT Gateway за адресою 192.168.200.1.

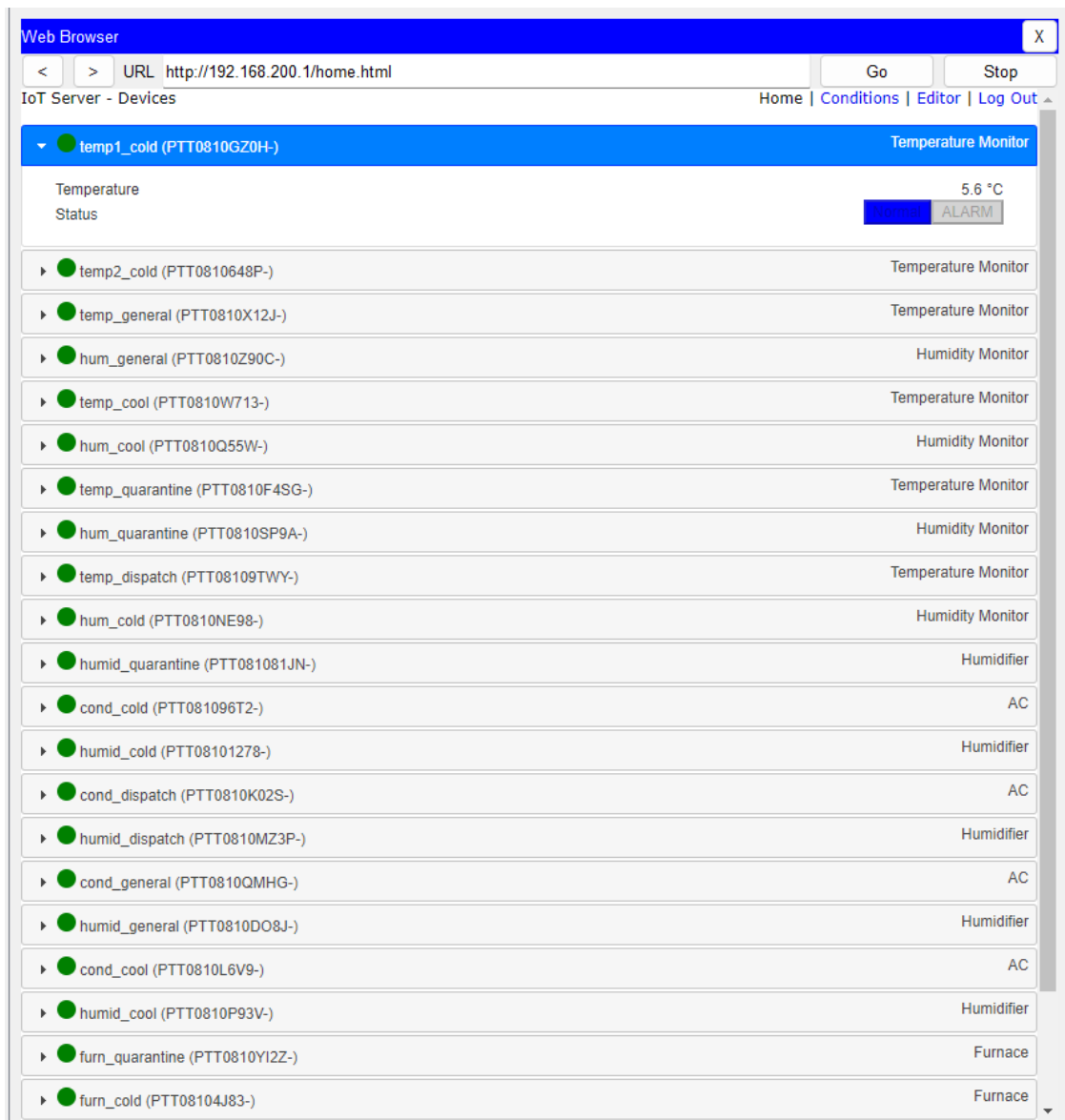


Рисунок 4.15 – Веб-інтерфейс IoT Gateway з переліком підключених пристроїв

Перевірка зведеної таблиці пристроїв через інтегрований веб-інтерфейс за адресою шлюзу підтвердила коректність відображення поточних числових значень та статусів роботи актуаторів для всіх зон складу.

4.2.4 Тестування серверного стеку та Grafana дашборду

Тестування серверного стеку на базі мови Python та платформи Grafana здійснювалося шляхом тривалого паралельного запуску процесів та моніторингу наскрізного руху інформації. Симулятор коректно публікує дані кожні 30 секунд для всіх п'яти зон складу. В терміналі видно чергування рядків зі статусами OK та ALARM, що підтверджує коректну роботу генератора реалістичних даних з вбудованим механізмом відхилень. Subscriber отримує ці дані через MQTT брокер і відображає їх у власному терміналі з кольоровими індикаторами, що дозволяє візуально контролювати потік даних в реальному часі. Відсутність повідомлень про помилки запису в обох терміналах підтверджує, що дані коректно надходять до InfluxDB.

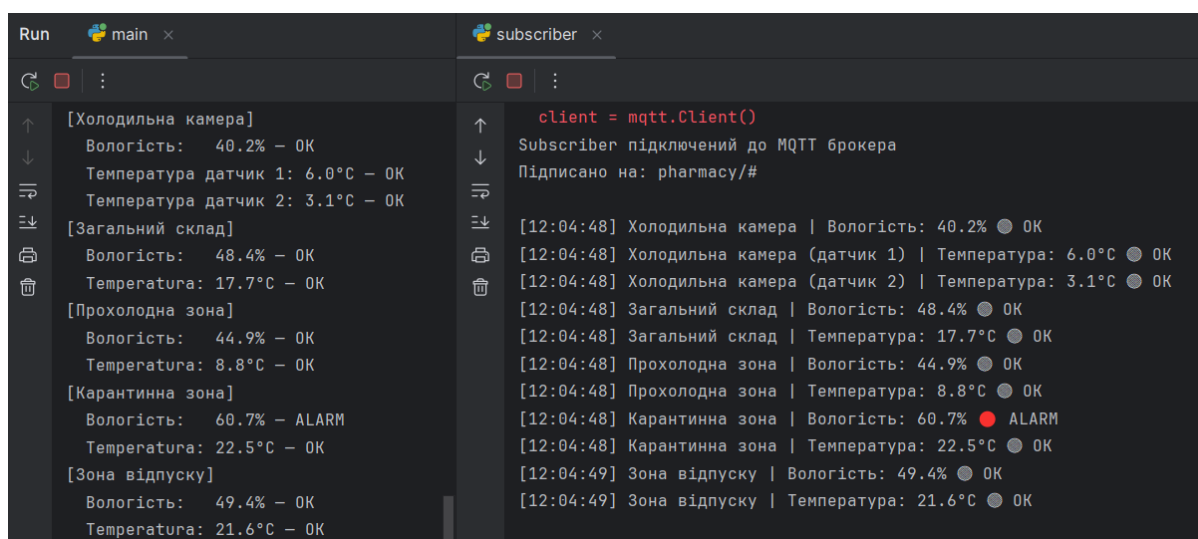
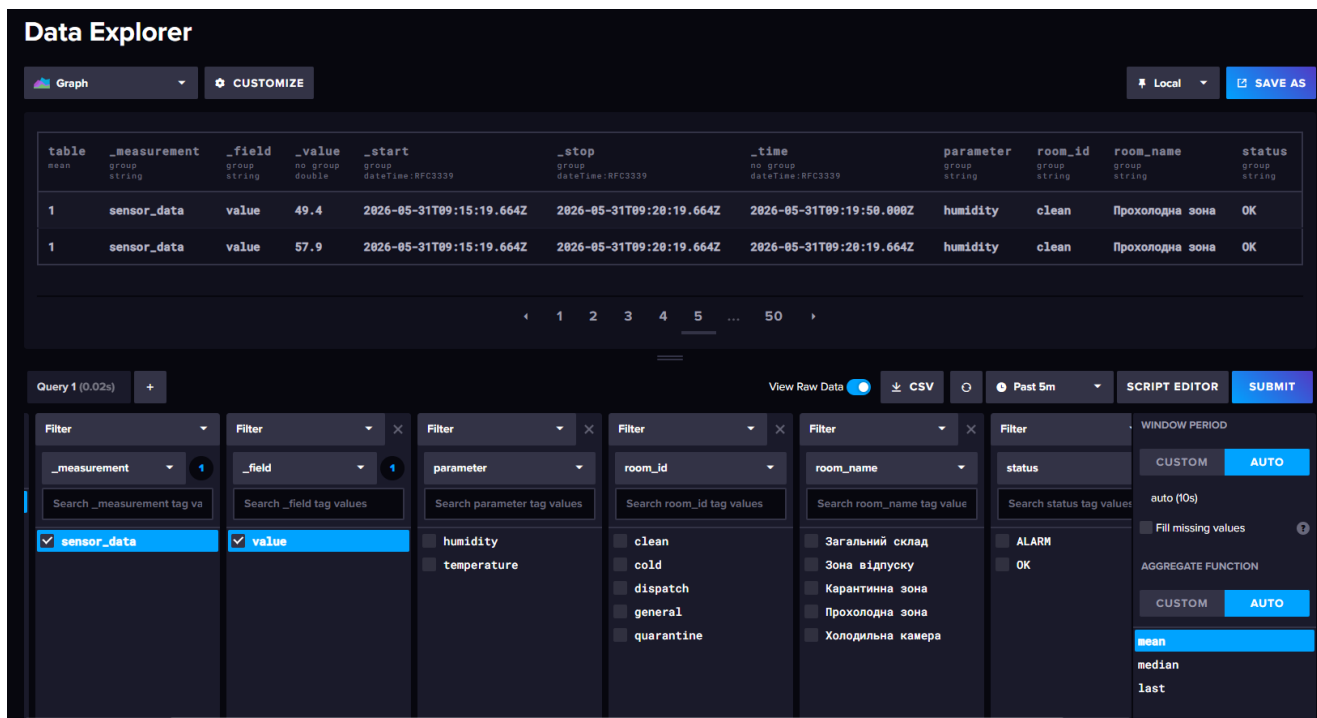


Рисунок 4.16 – Вікна терміналів запущених модулів симулятора та обробника даних

Коректність структури збереження інформації в базі даних часових рядів InfluxDB перевірялася за допомогою виконання тестових аналітичних запитів мовою Flux через інтегрований інструмент Data Explorer. Результати виконання запиту підтвердили, що кожна історична подія записується з повним набором

необхідних текстових тегів кімнат, параметрів та статусів безпеки, а самі числові виміри мають точну часову прив'язку, що відображено на рисунку 4.17.



The screenshot shows the InfluxDB Data Explorer interface. At the top, there's a 'Graph' tab and a 'CUSTOMIZE' button. Below that, a table displays sensor data. The table has columns: table, _measurement, _field, _value, _start, _stop, _time, parameter, room_id, room_name, and status. The data shows two rows of humidity measurements for 'sensor_data' in a 'clean' room, with values 49.4 and 57.9. Below the table, there's a pagination bar showing 1 to 50 records. At the bottom, the 'Query 1' section shows the query editor with filters for _measurement, _field, parameter, room_id, room_name, and status. The 'WINDOW PERIOD' is set to 'Past 5m' and the 'AGGREGATE FUNCTION' is set to 'mean'.

table	_measurement	_field	_value	_start	_stop	_time	parameter	room_id	room_name	status
mean	group	string	no group	group	dateTime:RFC3339	dateTime:RFC3339	dateTime:RFC3339	group	string	group
1	sensor_data	value	49.4	2026-05-31T09:15:19.664Z	2026-05-31T09:20:19.664Z	2026-05-31T09:19:50.000Z	humidity	clean	Проходна зона	OK
1	sensor_data	value	57.9	2026-05-31T09:15:19.664Z	2026-05-31T09:20:19.664Z	2026-05-31T09:20:19.664Z	humidity	clean	Проходна зона	OK

Рисунок 4.17 – Результат запиту до бази даних InfluxDB з переглядом записів датчиків у розділі Data Explorer

Grafana дашборд відображає дані в реальному часі з автоматичним оновленням кожні 30 секунд. Графіки температури та вологості коректно відображають окремі лінії для кожної зони складу.



Рисунок 4.18 – Панелі температури та вологості по зонах

Блоки поточних значень відображають актуальні дані з підписами нормативних діапазонів.

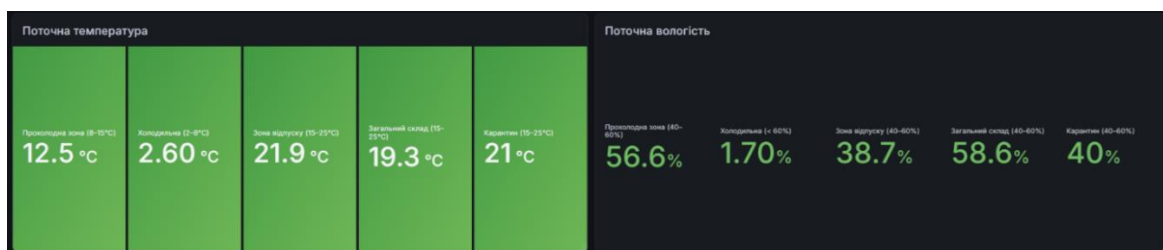


Рисунок 4.19 – Панелі поточної температури та вологості по зонах

Зміна часового діапазону підтверджує, що всі накопичені дані збережені і доступні для перегляду за будь-який проміжок часу.

4.2.5 Тестування системи сповіщень Telegram

Тестування Telegram бота виконувалось паралельно з роботою симулятора та subscriber. При запуску системи на телефон надійшло стартове повідомлення, що підтверджує успішну ініціалізацію всіх компонентів моніторингу.

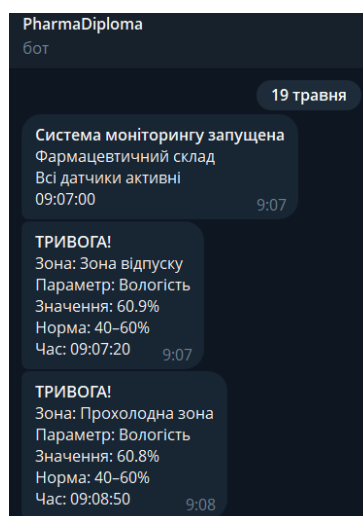


Рисунок 4.20 – Стартове повідомлення та повідомлення тривоги в Telegram

Протягом тестування при генерації симулятором відхилень від норми система надсилала детальні повідомлення тривоги. Час від генерації відхилення до отримання повідомлення на телефоні становив менше 3 секунд, що повністю відповідає встановленій вимозі. Кожне повідомлення містило назву зони, параметр, фактичне значення, нормативний діапазон та точний час що забезпечує повноту інформації для прийняття оперативного рішення.

Механізм cooldown підтвердив коректну роботу – в терміналі telegram_bot.py чітко видно чергування рядків ALARM => Telegram та ALARM (cooldown), що свідчить про блокування повторних сповіщень при тривалих відхиленнях. Журнал роботи модуля представлений на рисунку 4.21.

```
[09:45:21] ALARM → Telegram: Зона відпуску Температура 14.0°C  
[09:45:50] ALARM (cooldown): general temperature  
[09:45:50] ALARM (cooldown): clean humidity  
[09:46:20] ALARM (cooldown): general temperature  
[09:46:20] ALARM (cooldown): dispatch temperature  
[09:46:50] ALARM (cooldown): clean temperature  
[09:46:50] ALARM → Telegram: Карантинна зона Вологість 39.5%  
[09:46:51] ALARM (cooldown): dispatch humidity
```

Рисунок 4.21 – Журнал роботи модуля telegram_bot.py

Для різних комбінацій зона-параметр cooldown працює незалежно – відхилення температури в холодильній камері та відхилення вологості в загальному складі генерують окремі сповіщення незалежно одне від одного. Таким чином, система сповіщень повністю виконує покладені на неї функції і відповідає встановленим вимогам щодо оперативності та захисту від спаму.

4.3 Аналіз результатів та порівняльна характеристика

За результатами проведеного тестування встановлено, що всі компоненти розробленої системи функціонують узгоджено та забезпечують виконання поставлених вимог на всіх рівнях архітектури. Перевірка мережевої

інфраструктури підтвердила коректну сегментацію VLAN, стабільну маршрутизацію між підмережами та відсутність несанкціонованого міжзонального трафіку завдяки налаштованим ACL та механізмам Port Security. Також підтверджено стабільну роботу захищеного адміністративного доступу через SSH v2.

На рівні IoT-пристроїв зафіксовано коректне формування телеметричних даних, своєчасне виявлення відхилень та автоматичне керування виконавчими пристроями без участі оператора. Серверна частина системи забезпечує безперервну обробку даних у режимі реального часу, їх коректне збереження в базі даних часових рядів InfluxDB та подальшу візуалізацію в Grafana з можливістю ретроспективного аналізу.

Окремо підтверджено ефективність реалізованої системи сповіщень через Telegram, яка забезпечує доставку критичних повідомлень у межах декількох секунд, а також механізм обмеження частоти сповіщень, що запобігає перевантаженню користувача при тривалих аварійних станах. Усі функціональні модулі працюють синхронізовано, без втрати повідомлень та збоїв у передачі даних.

У порівнянні з традиційним підходом ручного моніторингу розроблена система забезпечує суттєве підвищення швидкості реагування, точності збору даних та рівня автоматизації процесів. Це дозволяє мінімізувати вплив людського фактору, зменшити ризик втрати або спотворення даних та забезпечити безперервний контроль умов зберігання лікарських засобів відповідно до вимог GDP.

ВИСНОВКИ

У дипломній роботі розроблено IoT-систему автоматизованого моніторингу та контролю умов зберігання лікарських засобів у фармацевтичному складі. Отримані результати є підсумком послідовного виконання поставлених завдань і охоплюють усі основні етапи проєктування та реалізації системи.

На початковому етапі проведено аналіз нормативних вимог та існуючих рішень у даній предметній області. Досліджено положення стандартів належної практики дистрибуції (GDP), а також виявлено основні обмеження комерційних систем, серед яких ключовими є висока вартість впровадження та залежність від закритих платформ постачальників. На основі цього сформовано вимоги до розроблюваної системи, які враховують як функціональні, так і нефункціональні аспекти її роботи.

Було обґрунтовано вибір програмних засобів і технологій, орієнтований на використання відкритого програмного забезпечення з метою зниження вартості рішення та підвищення його гнучкості. Для моделювання мережевої інфраструктури використано Cisco Packet Tracer, передача телеметричних даних реалізована на основі протоколу MQTT із застосуванням брокера Eclipse Mosquitto, серверна логіка побудована на мові Python, а для збереження та візуалізації даних використано базу часових рядів InfluxDB та платформу Grafana.

Спроектовано та реалізовано комплексну програмну і мережеву архітектуру системи. Побудовано трирівневу ієрархічну модель мережі з логічною сегментацією приміщень на основі VLAN, а також впроваджено багаторівневу систему захисту, що включає списки контролю доступу (ACL), захищений протокол віддаленого адміністрування SSH та механізм контролю фізичних підключень Port Security. Розроблено програмну логіку для двадцяти шести периферійних IoT-пристроїв керування мікрокліматом, а також серверні модулі, які забезпечують безперервний збір телеметрії, її обробку в реальному часі та формування журналу подій для подальшого аналізу.

На завершальному етапі виконано комплексне тестування системи, яке підтвердило коректність роботи всіх її компонентів. Випробування показали стабільну передачу даних у мережі, ефективну ізоляцію сегментів, коректну роботу механізмів автоматичного керування кліматичним обладнанням та своєчасне надсилання сповіщень у Telegram. Додатково підтверджено роботу механізму захисту від надлишкових повідомлень, що забезпечує стабільність інформаційного потоку та відповідає вимогам експлуатації систем подібного класу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Guidelines on Good Distribution Practice of medicinal products for human use (2013/C 343/01). Official Journal of the European Union. European Commission, 2013. URL: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=oj:JOC_2013_343_R (дата звернення: 13.05.2026).
2. СТ-Н МОЗУ 42-5.1:2011. Лікарські засоби. Належна практика зберігання / за ред. Міністерства охорони здоров'я України. Київ : МОЗ України, 2011. URL: <https://www.dls.gov.ua/wp-content/uploads/2019/06/Належна-практика-зберігання.pdf> (дата звернення: 13.05.2026).
3. Про організацію зберігання в аптечних закладах різних груп лікарських засобів та виробів медичного призначення : Наказ Міністерства охорони здоров'я України від 16.03.1993 № 44. Верховна Рада України : веб-сайт. URL: <https://zakon.rada.gov.ua/laws/show/v0044282-93> (дата звернення: 13.05.2026).
4. Про затвердження Правил зберігання та проведення контролю якості лікарських засобів у лікувально-профілактичних закладах : Наказ Міністерства охорони здоров'я України від 16.12.2003 № 584. Верховна Рада України : веб-сайт. URL: <https://zakon.rada.gov.ua/laws/show/z1430-11> (дата звернення: 13.05.2026).
5. Environmental Monitoring Systems. Comet System : веб-сайт. URL: <https://www.cometsystem.com/> (дата звернення: 13.05.2026).
6. Testo Saveris 2 – WiFi data logger system. Testo : веб-сайт. URL: <https://www.testo.com/> (дата звернення: 13.05.2026).
7. Continuous Monitoring System viewLinc. Vaisala : веб-сайт. URL: <https://www.vaisala.com/> (дата звернення: 13.05.2026).
8. Квасніков В. П. Метрологічне забезпечення систем екологічного моніторингу : монографія. Київ : НАУ, 2015. 210 с.
9. Internet of Things in Five Days / A. L. Colina et al. Geneva : International Telecommunication Union, 2016. 132 p.
10. Буров Є. В., Митник М. М. Комп'ютерні мережі : підручник. Львів : Магнолія-2006, 2021. 416 с.

11. Kurose J. F., Ross K. W. Computer Networking : A Top-Down Approach. 8th ed. New Jersey : Pearson, 2021. 816 p.
12. Configuring VLANs and IP Access Lists. Cisco Systems : веб-сайт. URL: <https://www.cisco.com/> (дата звернення: 13.05.2026).
13. Cisco Packet Tracer. Cisco Networking Academy : веб-сайт. URL: <https://www.netacad.com/courses/packet-tracer> (дата звернення: 13.05.2026).
14. Eclipse Mosquitto. An open source MQTT broker. Eclipse Foundation : веб-сайт. URL: <https://mosquitto.org/documentation/> (дата звернення: 13.05.2026).
15. InfluxDB 2.0 Documentation. InfluxData : веб-сайт. URL: <https://docs.influxdata.com/influxdb/v2/> (дата звернення: 13.05.2026).
16. Grafana Documentation. Grafana Labs : веб-сайт. URL: <https://grafana.com/docs/grafana/latest/> (дата звернення: 13.05.2026).
17. Python 3.13 Documentation. Python Software Foundation : веб-сайт. URL: <https://docs.python.org/3.13/> (дата звернення: 13.05.2026).
18. Paho MQTT Python Client Library. PyPI : веб-сайт. URL: <https://pypi.org/project/paho-mqtt/> (дата звернення: 13.05.2026).
19. InfluxDB Python Client Library. PyPI : веб-сайт. URL: <https://pypi.org/project/influxdb-client/> (дата звернення: 13.05.2026).
20. Requests : HTTP for Humans. PyPI : веб-сайт. URL: <https://pypi.org/project/requests/> (дата звернення: 13.05.2026).

ДОДАТОК А

Реалізація логіки серверного модуля сповіщень Telegram Bot API

Програмні засоби:

- мова програмування – Python;
- асинхронний обмін даними за протоколом MQTT – бібліотека Paho-MQTT;
- виконання HTTP-запитів до Telegram API – бібліотека Requests;
- робота з системним часом та форматування міток – модулі Time та Datetime.

Фрагмент коду telegram_bot.py:

```
import sys
import os
sys.path.append(os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
)

import paho.mqtt.client as mqtt
from datetime import datetime
import requests
import time
from config.settings import MQTT_BROKER, MQTT_PORT, ROOMS

TELEGRAM_TOKEN = ""
TELEGRAM_CHAT_ID = ""

COOLDOWN = 300
last_alert = {}

def send_telegram(message):
    url = f"https://api.telegram.org/bot{TELEGRAM_TOKEN}/sendMessage"
    data = {
        "chat_id": TELEGRAM_CHAT_ID,
```

```

        "text": message,
        "parse_mode": "HTML"
    }
    try:
        requests.post(url, data=data)
    except Exception as e:
        print(f"Помилка Telegram: {e}")

def check_alarm(room_id, parameter, value):
    room = ROOMS[room_id]
    if parameter == "temperature":
        return not (room["temp_min"] <= value <= room["temp_max"])
    elif parameter == "humidity":
        return not (room["humidity_min"] <= value <= room["humidity_max"])
    return False

def can_send_alert(key):
    now = time.time()
    if key not in last_alert:
        last_alert[key] = now
        return True
    if now - last_alert[key] >= COOLDOWN:
        last_alert[key] = now
        return True
    return False

def on_connect(client, userdata, flags, rc):
    if rc == 0:
        print("Telegram бот підключений до MQTT")
        client.subscribe("pharmacy/#")
    else:
        print(f"Помилка: {rc}")

def on_message(client, userdata, msg):
    topic = msg.topic

```

```

value = float(msg.payload.decode())
time_now = datetime.now().strftime("%H:%M:%S")

parts = topic.split("/")
if len(parts) != 3:
    return

room_id = parts[1]
parameter = parts[2]

if room_id not in ROOMS:
    return

is_alarm = check_alarm(room_id, parameter, value)

if is_alarm:
    alert_key = f"{room_id}_{parameter}"

    if can_send_alert(alert_key):
        room_name = ROOMS[room_id]["name"]
        unit = "°C" if parameter == "temperature" else "%"
        param_ua = "Температура" if parameter == "temperature" else
"Вологість"

        if parameter == "temperature":
            norm = f"{ROOMS[room_id]['temp_min']}-
{ROOMS[room_id]['temp_max']}°C"
        else:
            norm = f"{ROOMS[room_id]['humidity_min']}-
{ROOMS[room_id]['humidity_max']}%"

        message = (
            f"<b>ТРИВОГА!</b>\n"
            f"Зона: {room_name}\n"
            f"Параметр: {param_ua}\n"

```

```

        f"Значення: {value}{unit}\n"
        f"Норма: {norm}\n"
        f"Час: {time_now}"
    )

    print(f"[{time_now}] ALARM → Telegram: {room_name} {param_ua}
{value}{unit}")
    send_telegram(message)
else:
    print(f"[{time_now}] ALARM (cooldown): {room_id} {parameter}")

if __name__ == "__main__":
    print("Telegram бот запущено. Очікую тривоги...\n")

    send_telegram(
        "<b>Система моніторингу запущена</b>\n"
        "Фармацевтичний склад\n"
        "Всі датчики активні\n"
        f"{datetime.now().strftime('%H:%M:%S')}"
    )

    client = mqtt.Client()
    client.on_connect = on_connect
    client.on_message = on_message
    client.connect(MQTT_BROKER, MQTT_PORT)
    client.loop_forever()

```

ДОДАТОК Б

Апробація роботи

Тези подано до друку в збірнику Міжнародної науково-практичної конференції «Science at the frontier of progress» (8-10 червня 2026 р., м. Париж, Франція). Сертифікат учасника представлений нижче.

