Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

перший (бакалаврський)
(рівень вищої освіти)

Розробка розподіленої IoT-системи для моніторингу та автоматичного управління параметрами розумного будинку
(тема)

Виконав:
здобувач 4 року навчання,
групи АКТАКІТ-21-2
Даниїл ПРУДНІКОВ
(власне ім'я, прізвище)

Спеціальність 151 - Автоматизація та комп'ютерно-інтегровані технології
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Автоматизація та комп'ютерно-інтегровані технології
(повна назва освітньої програми)

Керівник доц. Дмитро ЯНУШКЕВИЧ
(посада, власне ім'я, прізвище)

Допускається до захисту
Зав. кафедри КІТАР _____ Ігор НЕВЛЮДОВ
(підпис) (власне ім'я, прізвище)

2025 р.

Я, Прудніков Даниїл Дмитрович, як здобувач(ка) вищої освіти ХНУРЕ, розумію і підтримую політику закладу із академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Я не використовував штучний інтелект для підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на віповідне джерело.

"03" червня 2025 р.  Даниїл ПРУДНІКОВ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет АКТ
Кафедра КІТАР
Рівень вищої освіти перший (бакалаврський)
Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
(код і повна назва)
Тип програми Освітньо-професійна
Освітня програма Автоматизація та комп’ютерно-інтегровані технології
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри КІТАР _____
(підпис)
« 28 » квітня 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Прудніковy Даниїлу Дмитровичу
(прізвище, ім’я, по батькові)

1. Тема роботи Розробка розподіленої IoT-системи для моніторингу та автоматичного управління параметрами розумного будинку
Затверджена наказом по університету від 19.05.2025 р. № 390 Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 16.06.2025 р.
3. Вихідні дані до роботи
3.1 Аналітичний звіт про стан технологій IoT та протоколів передачі даних за 2023–2025 рр.
3.2 Проєктна документація розподіленої IoT-системи освітлення на базі ESP32, MQTT-шлюзу та Home Assistant
3.3 Робочий прототип і супровідні файли: прошивка ESP32, Docker-стек (Mosquitto + Home Assistant + InfluxDB + Grafana)
3.4 Інструкції з охорони праці
4. Перелік питань, що потрібно опрацювати в роботі
4.1 Вступ;
4.2 Аналіз сучасного стану та тенденцій розвитку IoT
4.3 Аналіз апаратної платформи
4.4 Вимоги до розроблюваної системи
4.5 Розробка системи
4.6 Реалізація розподіленої IoT-системи
4.7 Висновки та перелік джерел посилань
4.8 Питання пов’язані з охороною праці

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)

Демонстраційний матеріал, представлений у форматі презентації PowerPoint (*.ppt) – 15 с. формату А4.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Приміт-ка
1	Отримання завдання	28.04.2025	
2	Постановка задачі та узгодження з керівником	29.04 – 02.05.2025	
3	Аналіз технічного завдання	03.05 – 09.05.2025	
4	Аналіз сучасного стану та тенденцій розвитку IoT/IoE	10.05 – 17.05.2025	
5	Аналіз апаратної платформи та формулювання вимог до системи	18.05 – 27.05.2025	
6	Розробка архітектурної схеми, мережевого топологічного рішення та апаратної реалізації вузла ESP32	28.05 – 29.05.2025	
7	Реалізація серверного стеку домашнього шлюзу	30.05– 31.05.2025	
8	Розробка прошивки для ESP32 і автоматизації	01.06 - 02.06.2025	
9	Тестування розподіленої IoT-системи	03.06.2025	
10	Оформлення результатів розробки та написання висновків	04.06.2025	
11	Подання роботи на перевірку Інтернет-сервісом StrikePlagiarism	05.06-07.06.2025	
12	Оформлення пояснювальної записки	08.06-10.05.2025	
13	Подання роботи на рецензію	11.06-13.05.2025	
14	Подання роботи на підпис зав. кафедри	14.06-16.05.2025	
15	Подання кваліфікаційної роботи в ЕК	17.06.2025	

Дата видачі завдання 28.04.2025 р.

Здобувач _____ Даниїл ПРУДНІКОВ
(підпис) (посада, власне ім'я, прізвище)

Керівник роботи _____ доц. Дмитро ЯНУШКЕВИЧ
(підпис) (власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 95 с., 14 табл., 18 рис., 4 додатки, 29 джерел.

ІНТЕРНЕТ РЕЧЕЙ, РОЗУМНИЙ БУДИНОК, ЕНЕРГОЕФЕКТИВНІСТЬ, АВТОМАТИЗАЦІЯ ОСВІТЛЕННЯ, КІБЕРБЕЗПЕКА, ДЕЦЕНТРАЛІЗОВАНИЙ КОНТРОЛЬ.

Об’єкт розробки – процес моніторингу та автоматизованого керування освітленням у «розумному будинку».

Предмет розробки – архітектурні рішення, алгоритми й програмне забезпечення, що забезпечують взаємодію сенсорних вузлів на базі ESP32, брокера повідомлень, платформи домашньої автоматизації та керованих світильників.

Мета роботи – підвищення ефективності енергоспоживання та безпеки даних під час автоматизованого керування освітленням шляхом розроблення масштабованої, надійної й захищеної IoT-платформи.

Проведено аналіз протоколів передачі даних, контейнеризації та характеристик компонентів. Розроблено апаратну архітектуру сенсорних вузлів і мережеву топологію з Mosquitto, Home Assistant та часовою базою, створено енергоощадну прошивку й інформаційні панелі реального часу.

Платформа децентралізовано збирає дані, візуалізує історію показників, застосовує режими deep-sleep і адаптивні інтервали для зниження енергоспоживання та забезпечує шифрування, автентифікацію і керування доступом.

Система готова до впровадження в житлових і комерційних приміщеннях для економії енергії, підвищення комфорту та швидкого масштабування.

ABSTRACT

Explanatory note contains: 95 pages, 14 tables, 18 figures, 4 appendices, 29 references.

INTERNET OF THINGS, SMART HOME, ENERGY EFFICIENCY, LIGHTING AUTOMATION, CYBERSECURITY, DECENTRALIZED CONTROL.

Object of development – a distributed system for monitoring and controlling lighting in a smart home.

Subject of development – architectural solutions, algorithms and software ensuring interaction among ESP32-based sensor nodes, a message broker, a home-automation platform and controllable luminaires.

Aim of the work – to increase energy-use efficiency and data security in automated lighting control by developing a scalable, reliable and secure IoT platform.

A study of data-transmission protocols, containerization approaches and component characteristics were carried out. The hardware architecture of sensor nodes and the network topology using Mosquitto, Home Assistant and a time-series database were designed; energy-efficient firmware and real-time dashboards were implemented.

The platform collects data in a decentralized manner, visualizes historical measurements, employs deep-sleep modes and adaptive polling intervals to reduce power consumption, and provides encryption, authentication and access control.

The system is ready for deployment in residential and commercial environments to save energy, enhance comfort and enable rapid scalability.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ACL – Access Control List (контроль списків доступу);

API – Application Programming Interface (інтерфейс програмування застосунків);

CoAP – Constrained Application Protocol (спрощений протокол прикладного рівня);

CPU – Central Processing Unit (центральний процесор);

DIY – Do It Yourself (зроби сам);

DHT22 – цифровий датчик температури й вологості DHT22;

HTTP – HyperText Transfer Protocol (протокол передачі гіпертексту);

HTTPS – HyperText Transfer Protocol Secure (захищений протокол передачі гіпертексту);

I²C – Inter-Integrated Circuit (шина для з’єднання мікросхем);

InfluxDB – база даних часових рядів InfluxDB;

IoE – Internet of Everything (Інтернет усього);

IoT – Internet of Things (Інтернет речей);

Lx – рівень освітленості в люксах (позначає одиницю виміру);

LLM – Large Language Model (велика мовна модель);

M2M – Machine-to-Machine (машина-машині);

MQ-2 – напівпровідниковий газовий сенсор MQ-2;

MQTT – Message Queuing Telemetry Transport (протокол публікації/підписки);

P2M – People-to-Machine (людина-машині);

P2P – Peer-to-Peer (рівний-рівному);

Pir – Passive Infrared (пасивний інфрачервоний датчик руху);

QoS – Quality of Service (якість обслуговування, рівні доставки повідомлень);

REST – Representational State Transfer (архітектурний стиль для API);

TLS – Transport Layer Security (шифрування каналу);

UDP – User Datagram Protocol (протокол датаграмного обміну);

UI – User Interface (інтерфейс користувача);

VLAN – Virtual Local Area Network (віртуальна локальна мережа);

Wi-Fi – Wireless Fidelity (бездротова мережа);

YAML – YAML Ain’t Markup Language (мова розмітки конфігурацій).

ВСТУП

Сучасні тренди цифровізації житла перетворили «розумний будинок» із футуристичної ідеї на практичний інструмент енергоощадності й безпеки. Доступність одноплатних контролерів ESP32, універсальність протоколу MQTT та розвиток open-source-платформ Home Assistant і Node-RED дали змогу створювати розподілені IoT-системи без дорогого промислового обладнання.

Універсальним закликом до дій, щоб покінчити з бідністю, захистити планету та поліпшити життя та перспективи кожного, в усьому світі є Цілі сталого розвитку (ЦСР). 17 Цілей були прийняті всіма державами-членами ООН у 2015 році в рамках порядку денного сталого розвитку 2030. Цілі сталого розвитку включать в себе:

– забезпечення доступу до недорогих, надійних, сталих і сучасних джерел енергії для всіх;

– створення стійкої інфраструктури, сприяння всеохопній і сталій індустріалізації та інноваціям;

– забезпечення відкритості, безпеки, життєздатності й екологічної стійкості міст і населених пунктів.

Актуальність роботи визначається тим, що у зв’язку зі стрімким зростанням цін на енергоресурси виникає необхідність широкого впровадження автоматизованих систем для зниження енергоспоживання, а розвиток концепції Internet of Everything висуває на перший план врахування поведінкових сценаріїв користувачів. Крім того, сучасні вимоги до надійності передбачають використання відмовостійких P2P-рішень, здатних підтримувати функціонування локальних мереж навіть за умови втрат зв’язку з інтернетом.

Об’єкт розробки – процес моніторингу та автоматизованого керування освітленням у «розумному будинку».

Предмет розробки – архітектурні рішення, алгоритми та програмне забезпечення, що реалізують процес моніторингу і автоматизованого керування освітленням у «розумному будинку».

Мета роботи – підвищення ефективності енергоспоживання та безпеки даних під час автоматизованого керування освітленням шляхом розроблення масштабованої, надійної й захищеної IoT-платформи.

Для досягнення поставленої мети необхідно вирішити такі завдання:

1. Провести порівняльний аналіз MQTT, CoAP, Zigbee та Matter за надійністю, затримками й енергоспоживанням і довести доцільність вибору зв’язки MQTT + Wi-Fi для безпечної, масштабованої телеметрії.

2. Спроєктувати сенсорний вузол ESP32 + BH1750 + WS2812B і прошивку з гістерезисним керуванням та режимами deep-sleep, оптимізувавши таймери для балансу між частотою даних і автономністю.

3. Інтегрувати вузли та лампи Tapo в Home Assistant через локальний API, створити порогові автоматизації YAML/Node-RED і налаштувати дашборди Grafana для онлайн-моніторингу.

4. Підготувати інструкції з електробезпеки, навчити персонал і запровадити контроль за використанням засобів індивідуального захисту, забезпечивши безпечний монтаж та налагодження системи.

Методи дослідження – аналіз літературних джерел і стандартів, моделювання архітектури, експериментальна перевірка прошивок, тестування сценаріїв автоматизації та безпеки (TLS, ACL), вимірювання енергоспоживання.

Звіт з кваліфікаційної роботи виконаний згідно ДСТУ 3008:2015 [1], а також з методичними вказівками з підготовки й оформлення кваліфікаційної роботи здобувачами першого (бакалаврського) рівня вищої освіти спеціальності 151 Автоматизація та комп’ютерно-інтегровані технології [2].

1 АНАЛІТИЧНИЙ ОГЛЯД ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Сучасний стан та тенденції розвитку IoT / IoE для розумних будинків

Поняття Internet of Things спершу обмежувалося підключенням «речей» – датчиків, актуаторів і контролерів – до мережі для автоматичного збирання та обміну даними. Нині, однак, дедалі частіше використовується ширший термін Internet of Everything, у якому до пристроїв додають людей, процеси й дані, а отже зростають вимоги до персоналізації сервісів, біґ-дати-аналітики та кібербезпеки. Для домашньої автоматизації це означає перехід від ізольованих «розумних» ламп чи термостатів до цілісних екосистем, що реагують на поведінку мешканців і оптимізують побутові процеси у реальному часі [3; 4].

Перед детальним розглядом наведемо узагальнені відмінності двох підходів, які стануть методологічною основою проєкту (табл. 1.1).

Таблиця 1.1 – Порівняння концепцій IoT та IoE

Критерій	**IoT**	**IoE**
Об’єкт підключення	Підключені пристрої	Пристрої, люди, процеси, дані
Головний тип взаємодії	M2M (машина–машина)	M2M + P2M (людина–машина) + P2P (вузол–вузол)
Додана цінність	Автоматизація окремих задач	Персоналізований сервіс і нові бізнес-моделі
Основні виклики	Сумісність, безпека пристроїв	Конфіденційність, етика даних, масштабування

Перехід до моделі IoE відображається й у статистиці. За оцінкою IoT Analytics, кількість підключених пристроїв зросте до 18,8 млрд наприкінці 2024 року, що на 13 % більше, ніж торік [5].

Світовий ринок smart-home, за прогнозом Precedence Research, може збільшитись із 162 млрд USD у 2025 р. до приблизно 1,4 трлн USD у 2034 р., демонструючи середньорічний приріст понад 27 % [6].

Таке стрімке зростання пояснюється кількома чинниками:

– здешевленням апаратної бази (контролер ESP32 обходиться у кілька доларів, а смарт-лампа TP-Link Tapo – у межах двадцяти), що робить домашні проєкти масово доступними;

– появою відкритих стандартів, зокрема MQTT і Matter, які спрощують інтеграцію пристроїв різних виробників;

– загостренням енергетичних питань: за оцінками галузі, адаптивне управління освітленням і мікрокліматом здатне знизити споживання електроенергії на 20–30 %;

– підвищеною увагою до приватності та захисту даних, що стимулює локальні рішення з мінімальною залежністю від хмари.

У технологічному вимірі виділяються три ключові парадигми обміну:

1. M2M: сенсори й актуатори спілкуються через брокер MQTT, забезпечуючи мінімальні затримки та гнучку маршрутизацію повідомлень.

2. P2M: люди взаємодіють із системою за допомогою мобільних застосунків або голосових асистентів, отримуючи зрозумілу картину стану помешкання й можливість вручну коригувати сценарії.

3. P2P: вузли можуть напряму обмінюватися даними (напр., через ESP-Now), що підвищує відмовостійкість у разі збоїв центрального маршрутизатора чи брокера.

Поєднання цих підходів формує багатошарову екосистему, де «залізо» працює автономно, дані конвертуються у зрозумілу користувачеві інформацію, а система зберігає працездатність навіть за часткових відмов мережі. Такий контекст визначає актуальність і вимоги до подальшої розробки кваліфікаційної роботи, в якій передбачено реалізацію M2M-автоматизації на базі ESP32 і MQTT, P2M-інтерфейсів через Home Assistant та резервного P2P-каналу між вузлами [7, 8].

1.2 Комунікаційні моделі M2M, P2M та P2P у смарт-середовищі

Ефективність «розумного будинку» прямо залежить від того, як його компоненти обмінюються даними і реагують на зміни середовища. Сьогодні виділяють три базові моделі взаємодії: машина-машина (M2M), людина-машина (P2M) та вузол-вузол (P2P). Кожна з них виконує власну роль у багатошаровій мережевій архітектурі й формує окремі вимоги до протоколів.

M2M охоплює прямий обмін повідомленнями між датчиками й актуаторами без участі людини. Зв'язок найчастіше здійснюється через публікацію та підписку MQTT, що забезпечує малу затримку, досить низьке енергоспоживання й дає змогу працювати офлайн у межах локальної мережі. Ключові переваги – швидкість і передбачуваність реакції; серед головних викликів – надійний захист каналів і строгий контроль прав доступу.

P2M фокусується на взаємодії користувача з системою. Мобільний застосунок Home Assistant, віджет на розумному дисплеї чи голосова команда «увімкни світло» – усе це приклади P2M-шару. Він вимагає зрозумілого інтерфейсу, надійної аутентифікації та контекстної візуалізації даних

P2P дає змогу вузлам об'єднуватися напряму, минаючи центральний брокер. Для ESP32 таким інструментом є протокол ESP-Now, що допускає обмін невеликими пакетами з затримкою (10 – 100) мс підтримує шифрування. P2P підвищує відмовостійкість системи, але ускладнює журналювання подій та керування ключами. Для узагальнення характеристик наведено у табл. 1.2:

Таблиця 1.2 – Порівняння моделей M2M, P2M та P2P у розумному будинку

Параметр	**M2M**	**P2M**	**P2P**
Типовий канал	MQTT, CoAP	HTTPS, WebSocket, голосові шлюзи	ESP-Now, Thread
Затримка, мс	50–200	200–400 (із UI)	10–100
Потреба у брокері	висока	помірна	відсутня

Продовження табл. 1.2

Параметр	**M2M**	**P2M**	**P2P**
Ключові переваги	швидка реакція, офлайн-робота	зручність для користувача	відмовостійкість, низьке навантаження
Основні ризики	підміна даних, DOS	несанкціонований доступ	відсутність централізованого журналу

Поєднання цих трьох підходів формує багаторівневу екосистему, у якій «залізо» обмінюється даними автономно, користувач отримує інтуїтивний контроль, а вузли здатні підстраховувати один одного у разі збоїв мережі. Саме така синергія закладається у подальшій архітектурі кваліфікаційної роботи.

1.3 Протоколи та стандарти передачі даних у системах розумного будинку

Комунікаційний стек визначає «характер» розумного будинку: саме від обраного протоколу залежить, наскільки швидко та без затримок загориться лампа після спрацьовування датчика, чи залишиться можливим локальне керування пристроями при збої роутера, скільки енергії споживатиме батарейний сенсор і наскільки просто й швидко буде інтегрувати новий девайс через рік-два. Наприклад, Zigbee й Thread економлять енергію й забезпечують стійке зв’язне mesh-мережеве середовище, тоді як Wi-Fi гарантує вищу пропускну здатність, але потребує частішої заміни батарей. Також важливими є питання безпеки, масштабованості й підтримки в Home Assistant. Нижче наведено огляд п’яти ключових технологій для екосистеми ESP32+Home Assistant та пояснення, чому кваліфікаційна робота спиратиметься саме на їхню комбінацію.

Спершу зосередьмося на базових властивостях. У табл. 1.3 наведено компактне порівняння затримки, переваг та обмежень для кожного протоколу.

Таблиця 1.3 – Характеристика популярних протоколів для домашньої автоматизації

Протокол	**Канал зв'язку**	**Середня затримка, мс**	**Переваги**	**Обмеження**
MQTT	Wi-Fi або Ethernet	50–250	мінімальні накладні витрати; publish/subscribe; три рівні QoS	залежить від брокера; потрібен TLS
CoAP	Wi-Fi (UDP)	40–120	полегшений стек; multicast-запити	ненадійний транспорт; окремий DTLS
Matter	Pv6 поверх Wi-Fi, Thread, Ethernet	100–300	міжбрендова сумісність; автоматичне виявлення	ще мало сумісних пристроїв; складніші сертифікати
Zigbee	2,4 ГГц радіомережа	30–150	сіткова топологія; ультранизьке енергоспоживання	потребує шлюзу до Wi-Fi
ESP-Now	2,4 ГГц (P2P)	10–80	прямий обмін між ESP32; робота без роутера	пакети до 250 байт; немає централізованих журналів

Однак «сухі» цифри не розкривають повністю ані практичного досвіду, ані внутрішньої кухні кожної технології, тому нижче наведено розгорнуті пояснення та приклади застосування.

MQTT лишається де-факто стандартом для внутрішньої шини даних у Home Assistant. Завдяки моделі publish/subscribe датчик світла BH1750 на ESP32 надсилає значення освітленості лише один раз, а отримують його одразу всі зацікавлені підписники – автоматизація, яка регулює лампи Tapo, графік у Lovelace-дашборді й журнал довготривалої статистики. Три рівні QoS дають змогу гнучко балансувати між пропускною здатністю й гарантією доставки, а вбудована підтримка TLS шифрує дані без додаткових проксі.

CoAP приваблює легковажністю UDP-транспорту та можливістю multicast-запитів у великих мережах. У разі підключення десятка батарейних датчиків дверей вони зможуть періодично «писати» свої показники без

складного рукопотискання, тоді як контролер збиратиме групові пакети одним запитом. Недоліком є відсутність підтвердженого каналу: будь-який втрачених пакет доведеться надсилати повторно, що не завжди допустимо у критичних сценаріях безпеки.

Matter – наймолодший гравець на ринку протоколів, який обіцяє зняти біль інтеграції «різнобійних» брендів розумного будинку. Пристрій, сертифікований Matter-консорціумом, автоматично оголошує свій тип і служби, а користувач підключає його за хвилину: достатньо сканувати QR-код, щоб Home Assistant чи Google Home миттєво підхопили та застосували налаштування. Головним гальмівним фактором на сьогодні є невеликий асортимент підтримуваних світильників і датчиків, тому проєкт залишає Matter «на виріст», передбачаючи резерв на рівні мережі IPv6 та майбутнє масштабування.

Zigbee зберігає свою нішу завдяки енергоощадним датчикам із багаторічним часом автономної роботи. Вузол прокидається на мить, відправляє 20-байтний кадр телеметрії й знову засинає на рік, а координатор, під’єднаний до Raspberry Pi з Home Assistant, формує стійку mesh-сітку, де кожен пристрій може ретранслювати пакети іншим вузлам. Для компактної квартири достатньо кількох реле-репітерів у розетках, але головним мінусом залишається ланцюг сумісності: потрібен спеціальний шлюз, а прошивка координатора часом складніша за налаштування стандартного Wi-Fi.

ESP-Now виходить на сцену там, де швидкість передачі й автономність виходять на перший план. Після аварійного відключення Wi-Fi два модулі ESP32 продовжують обмінюватися даними напряму: один вузол детектує дим на MQ-2, інший блимає червоним на WS2812B і запускає локальну сирену. Передача 250-байтного пакета відбувається за мікросекунди, проте відсутність централізованого журналювання трафіку на стороні брокера значно ускладнює подальший аудит подій та аналіз інцидентів.

Щоб показати прив’язку протоколів до реальних пристроїв і краще ілюструвати їх практичне застосування, наведено додаткову ілюстративну

класифікацію в табл. 1.4, що відображає типові, а не вичерпні сценарії реалізації у «розумному будинку».

Таблиця 1.4 – Типові пристрої та канали зв’язку у проєкті

Клас пристрою	**Приклад**	**Основний протокол**	**Резервний канал**
Актуатор освітлення	лампа TP-Link Tapo L530E	MQTT через Wi-Fi	Matter (після оновлення FW)
Сенсор освітленості	BH1750 на ESP32	MQTT	ESP-Now
Сенсор температури	DHT22 на ESP32	MQTT	Zigbee (альтернатива)
Датчик газу	MQ-2 на ESP32	MQTT	ESP-Now
Індикація стану	матриця WS2812B	локальний GPIO	–

1.4 Аналіз апаратної платформи

В основі прототипу лежить одноплатний контролер ESP32 із двоядерним CPU до 240 МГц, вбудованим Wi-Fi, Bluetooth і понад тридцятьма GPIO-виводами, що робить плату універсальною для одночасного прийому даних від декількох сенсорів і передавання команд актуаторам. Підтримка deep-sleep із струмом (10–150) µA дозволяє живити вузол від батареї місяцями при коротких сеансах телеметрії [13].

Для моніторингу параметрів обрано цифровий фотометр BH1750 (I²C, 1–65 000 lx, 16-бітна роздільність), комбінований датчик DHT22 (±0,5 °C, ±2 % RH) та газовий сенсор MQ-2 (SnO_2-матриця для LPG, пропану, водню, метану), що дозволяє виявляти витоки й запускати аварійну індикацію [14–16].

Візуальний зворотний зв’язок виконує WS2812B-матриця: кожен світлодіод адресований окремо, анімації сигналізують про стан системи – зелений означає норму, синій – втрату зв’язку з брокером, червоний – тривогу. Навіть при піковому струмі 60 мА індикатор споживає не більше 150 мА у найяскравішому режимі [17].

Керування освітленням здійснюють лампи TP-Link Tapo L530E (800 лм

при 8,3 Вт, 16 млн кольорів, Wi-Fi 2,4 ГГц без концентратора). Завдяки відкритому локальному API лампа приймає MQTT-команди з Home Assistant і може перейти на Matter після оновлення прошивки [18].

У табл. 1.5 наведено ключові характеристики апаратних модулів із акцентом на споживану потужність, способи підключення й роль у системі; сенсори, позначені «не впроваджено», залишено для потенційного розширення.

Таблиця 1.5 – Основні елементи апаратної платформи

Пристрій	**Роль**	**Комунікація з ESP32**	**Типове споживання**	**Статус у прототипі**
ESP32 DevKitC	центральний вузол, Wi-Fi/MQTT-шлюз	Wi-Fi 802.11 [10]	80 мА (робота), 10–150 μA (deep sleep)	впроваджено
BH1750	сенсор освітленості	I²C (400 кГц)	0,12 мА	впроваджено
DHT22	сенсор температури й вологості	однолінійний цифровий сигнал	1,5 мА (у запиті)	не впроваджено
MQ-2	сенсор горючих газів	аналоговий вихід ↔ ADC	150 мА (підігрів)	не впроваджено
WS2812B, 8×8	індикатор стану	один GPIO (серіальний)	≤150 мА (низька яскравість)	впроваджено
Tapo L530E	основний актуатор освітлення	Wi-Fi / MQTT або Matter	8,3 Вт (800 лм)	впроваджено

Отже, ми обрали апаратну платформу, яка працює дуже економно, збирає дані та пропонує гнучке бездротове управління з наочною індикацією. Головним «мозком» тут є вузол на базі ESP32, який через шину I²C, цифрові й аналогові лінії координує роботу сенсорів. А лампи Tapo практично миттєво реагують на MQTT-повідомлення. Крім того, підтримка ESP-Now дає змогу організувати резервний обмін даними між контролерами без участі роутера. У наступному розділі ми сформулюємо вимоги до системи й детально пояснимо вибір загальної архітектури з огляду на описане апаратне та протокольне підґрунтя, а також перспективи масштабування.

1.5 Порівняльна оцінка існуючих рішень

Світові продажі смарт-пристроїв зростають, хоча темп уже переходить від «буму» до етапу зрілості: у 2025 р. ринок оцінюють у 199–255 млрд USD, а увага зміщується з кількості ґаджетів на якість інтеграції і локальне керування. Головна інтрига цього етапу – чи зможуть відкриті платформи зберегти гнучкість, коли великі бренди активніше просувають власні, частково закриті екосистеми з підтримкою Matter та AI-сцен.

Перед детальним аналізом варто зіставити найпоширеніші серед українських користувачів рішення – від mass-market-ламп до повністю відкритих стеків. У табл. 1.6 наведено синтетичну оцінку восьми платформ за п’ятьма критеріями.

Таблиця 1.6 – Порівняння провідних смарт-платформ

Платформа	**Локальна робота без хмари**	**Стартова вартість, USD**	**Підтримка Matter**	**Головна перевага**
Philips Hue	частково	≈ 90	активна (AI-сцени, нові Wall Washer)	якість світильників, AI-сцени
Tuya Smart	ні (опція LAN-Mode)	≈ 15	SDK 1.1 (бета)	найдешевший масовий асортимент
Apple HomeKit	частково	≈ 120	повна	екосистема iOS, безпека
Google Home	частково	≈ 70	повна	50 000+ сумісних пристроїв
Amazon Alexa	частково	≈ 55	планується	натуральний голосовий інтерфейс, Alexa+ AI
Home Assistant	так	≈ 0 (якщо є Raspberry Pi)	інтеграція Gateway	2 млн активних інсталяцій, 2500+ інтеграцій
openHAB	так	≈ 0	інтеграція Gateway	незалежність від постачальника, Java-кросплатформеність

Дані з табл. 1.6 свідчать про низку ключових тенденцій. Комерційні бренди активно посилюють «вау-ефект»: у 2025 році Philips Hue представила AI-генерацію світлових сцен і вдвічі яскравіші Wall Washer, роблячи ставку на досвід користувача, тоді як Tuya здобуває ринок низькою ціною та OEM-підтримкою, залишаючи LAN-режим в експериментальній стадії. Великі екосистеми все більше орієнтуються на універсальність Matter: Apple, Google та Samsung одночасно просувають власні хаби з Thread-радіо та Edge-драйверами, що дозволяє зменшити кількість гейтвеїв у домі й пришвидшити онбординг пристроїв, а Amazon у 2025 році виводить Alexa+ із більш природною мовною моделлю, продовжуючи тестувати повну сертифікацію Matter. Водночас DIY-платформи зміцнюють свої позиції завдяки автономії: Home Assistant подолав рубіж у два мільйони активних установок і презентував інтеграцію власної LLM-підсистеми для розмовних сценаріїв, а openHAB досяг етапу 5.0-milestone, зберігаючи філософію відкритості та кросплатформенності, хоча й стикається з повільнішим реліз-циклом та меншим обсягом готових інтеграцій.

Щодо вартісного порогу у DIY-сегменті, різниця пояснюється лише ціною міні-комп’ютера: власники Raspberry Pi або старенького ноутбука запускають Home Assistant і openHAB фактично безкоштовно; користувачеві комерційних екосистем доведеться одразу придбати фірмовий хаб.

У контексті кваліфікаційної роботи ключовим чинником лишається локальне керування. Лише три системи з таблиці (Home Assistant, openHAB, Shelly) гарантують повну автономність без реєстрації у хмарі, що критично для сценаріїв, де відключення інтернету не має впливати на безпеку та освітлення. Додатково Home Assistant вже має нативну інтеграцію ламп TP-Link Tapo і підтримку MQTT-брокера, що зменшує час налаштування і спрощує відлагоджування сценаріїв [19].

Отже, вибір стеку ESP32 + MQTT + Home Assistant забезпечує одночасно низьку стартову вартість, повну локальність, перспективу переходу на Matter та найширшу спільноту підтримки. Комерційні ж рішення можуть

залишатися джерелом окремих модулів (наприклад, лампи або сенсори), інтегрованих через стандартні шлюзи без втрати автономності системи.

1.6 Вимоги до розроблюваної системи

На підставі проведеного огляду та вибраної апаратно-протокольної бази формулюються вимоги, що задають рамки подальшого проєктування. Вони поділяються на функціональні, нефункціональні та експлуатаційні.

Функціональні вимоги включають в себе:

– система має збирати показники освітленості від BH1750 не рідше ніж раз на 30 с. і передавати їх до брокера MQTT;

– у разі падіння рівня освітленості нижче користувацького порога або фіксації руху (PIR-датчик може бути доданий як опція) система автоматично вмикає лампи TP-Link Tapo та встановлює попередньо налаштовану яскравість і колір;

– LED-матриця WS2812B подає візуальний сигнал: зелений – нормальний режим, синій – втрачено зв’язок із брокером, червоний – аварійна подія (наприклад, перевищення часу відповіді лампи > 1 с);

– користувач повинен мати можливість у застосунку Home Assistant вручну ввімкнути чи вимкнути лампу, перевизначити автоматичний сценарій або переглянути історію спрацьовувань за останні 30 днів;

– система надсилає телеметрію (рівень освітленості, стан ламп, напругу живлення ESP32) до бази даних часового ряду, а у Grafana автоматично створюється дашборд із графіком цих показників, що оновлюється в реальному часі.

Нефункціональні вимоги складаються з:

– час реакції між спрацюванням правила автоматизації та зміною стану лампи не повинен перевищувати 500 мс у локальній мережі. Навантаження на батарейний вузол ESP32 у режимі «глибокий сон» – не більше 150 μA; середньодобове споживання вузла – ≤ 0,5 Wh;

– система повинна масштабуватися щонайменше до десяти вузлів ESP32 та п’яти ламп без помітного погіршення часу реакції (додаткова затримка < 100 мс на кожну пару «сенсор – актуатор») ;

– усі MQTT-з’єднання шифруються TLS 1.2;

– доступ до панелі Home Assistant захищено двофакторною аутентифікацією;

– після відновлення живлення чи перезапуску вузол ESP32 має відновити з’єднання з брокером і повернутися до робочого стану протягом 10с.

Експлуатаційні та документальні вимоги:

– система монтується без втручання у стаціонарну електропроводку; лампи Таро встановлюються у стандартний патрон E27;

– файл конфігурації Home Assistant (YAML) і прошивка ESP32 публікуються в додатку до кваліфікаційної роботи з короткими інструкціями з компіляції та розгортання.

1.7 Обґрунтування вибору архітектури

Після аналізу ринку й уточнення вимог обрана конфігурація «ESP32 – Wi-Fi – MQTT – Home Assistant – Grafana» виявилася найкращим компромісом між швидкістю реакції, енергоефективністю, локальним керуванням і майбутньою масштабованістю. Нижче подано аргументацію цього вибору у зв’язку з кожною групою вимог.

По-перше, контролер ESP32 поєднує достатню обчислювальну потужність з дуже низьким споживанням у глибокому сні; за даними форуму Espressif, за сприятливої конфігурації струм може знижуватися до десятків мікроампер, що забезпечує тривалу роботу автономних вузлів без підзарядки. Це безпосередньо відповідає вимозі середньодобового споживання не більше 0,5 Wh, сформульованій у пункті 1.6.

По-друге, MQTT як логічна шина гарантує мінімальні накладні витрати й модель publish/subscribe, де один пакунок даних від датчика одразу

одержують усі зацікавлені служби. У реальних тестах затримка між надходженням повідомлення і виконанням автоматизації Home Assistant не перевищує 200 мс у локальній мережі, що вкладається у встановлену планку 500 мс.

По-третє, Home Assistant демонструє швидке зростання спільноти і, за офіційною статистикою, у травні 2025 року перевищив позначку двох мільйонів активних установок; такий масштаб підтримки означає швидке виправлення вразливостей і велику кількість готових інтеграцій. Платформа працює повністю локально, отже, навіть за тривалого відключення інтернету сценарії продовжують виконуватися.

Четвертий аргумент – сумісність актуаторів. Лампи TP-Link Tapo офіційно підтримуються Home Assistant через локальне API без обов'язкового виходу у хмару, тому сценарії вмикання та зміни кольору працюють із тією ж мінімальною затримкою, що й будь-які інші MQTT-пристрої. При появі прошивки з Matter-сертифікацією лампи зможуть безболісно перейти на новий стандарт, зберігаючи налаштування.

Нарешті, графічне подання даних у Grafana природно доповнює цю архітектуру: Home Assistant уже містить інтеграцію з базами часового ряду, а Grafana на тому ж сервері будує дашборд без сторонніх сервісів. Це дає можливість виконати вимогу пункту 1.6 щодо візуалізації освітленості, напруги живлення вузлів і стану ламп у реальному часі.

Слід зазначити, що обрана архітектура відповідає всім функціональним і нефункціональним вимогам і водночас надає можливість для майбутнього розширення – від інтеграції нових сенсорів до переходу на Matter чи Thread без змін у самому ядрі системи.

1.8 Формулювання задачі кваліфікаційної роботи

На основі вимог, викладених у попередньому розділі, кваліфікаційна робота має продемонструвати повний цикл створення й оцінювання прототипу

розподіленої системи для автоматичного керування освітленням житлового приміщення. Головна мета – довести, що поєднання ESP32, MQTT, Home Assistant і Grafana здатне забезпечити стабільну роботу, енергетичну ощадність і зрозумілий для користувача інтерфейс без залучення сторонніх хмарних сервісів. Для досягнення цієї мети сформульовано такі конкретні задачі:

– спроєктувати топологію мережі, у якій дані з датчика BH1750 надходять на брокер MQTT, а команда на лампу TP-Link Tapo виконується із затримкою не більше 0,5 с;

– реалізувати прошивку ESP32, що переходить у режим глибокого сну й прокидається лише для вимірювання освітленості, забезпечуючи середньодобове споживання не вище 0,5 Вт × год.;

– налаштувати Home Assistant так, аби сценарій автоматичного вмикання світла спрацьовував при падінні освітленості нижче порога й мав ручне перевизначення через мобільний застосунок;

– додати резервний канал ESP-Now між контрольними вузлами, який дає змогу передавати аварійні події та підтримувати базову індикацію WS2812B за відсутності Wi-Fi;

– інтегрувати базу даних часових рядів і створити у Grafana дашборд, що показує в реальному часі рівень освітленості, стан лампи та напругу живлення ESP32;

– виконати серію експериментів, під час яких оцінити час реакції системи, стабільність зв’язку, точність вимірювань та енергоспоживання при різних інтервалах опитування датчика;

– провести базовий аудит безпеки: перевірити шифрування TLS на MQTT, захист паролем та двофакторну автентифікацію в Home Assistant, а також доступність вузлів лише з окремого VLAN.

Успішне виконання пунктів підтвердить відповідність прототипу функціональним і нефункціональним вимогам та можливість розширення системи без зміни основного ядра.

1.9 Висновки до розділу 1

Аналітичний огляд показав, що сучасний ринок домашньої автоматизації переходить від класичної моделі «підключені речі» до ширшої концепції Internet of Everything, де пристрої, люди й процеси взаємодіють у режимі реального часу. У такій екосистемі вирішальну роль відіграють гнучкі комунікаційні моделі: M2M забезпечує автономність пристроїв, P2M – зручність для користувача, а P2P – відмовостійкість у разі збою мережевої інфраструктури.

Порівняння протоколів засвідчило доцільність зв'язки Wi-Fi + MQTT як основного каналу і ESP-Now як резервного. Ця комбінація поєднує малу затримку, енергоощадність і можливість працювати без зовнішніх сервісів. Аналіз апаратної бази показав, що ESP32 має достатній ресурс для виконання задач, а лампи TP-Link Tapo підтримують локальне API, що відповідає вимогам автономності. Незадіяні в прототипі сенсори температури, вологості та газу залишено у переліку як резерв для майбутнього розширення, аби продемонструвати потенціал масштабування системи.

Огляд готових рішень підтвердив, що лише відкриті платформи Home Assistant і подібні забезпечують повну локальну роботу й гнучку інтеграцію нових пристроїв без додаткових ліцензійних витрат. З огляду на це сформульовано вимоги до системи, які охоплюють не тільки керування освітленням, а й безпеку, енергоспоживання та візуалізацію даних у Grafana.

На основі вимог поставлено конкретні задачі кваліфікаційної роботи: від розроблення прошивки та сценаріїв автоматизації до побудови дашборду й проведення тестів на швидкодію та стійкість. Виконання цих задач має підтвердити, що обрана архітектура «ESP32 – MQTT – Home Assistant – Grafana» відповідає сучасним тенденціям IoE і може служити базою для подальшого розвитку системи.

2 РОЗРОБКА РОЗПОДІЛЕНОЇ СИСТЕМИ АВТОМАТИЗАЦІЇ ОСВІТЛЕННЯ

2.1 Архітектурна схема та мережеве топологічне рішення

Архітектура прототипу передбачає один центральний сервер-шлюз (Raspberry Pi із Home Assistant, MQTT-брокером та Grafana), кілька вузлів ESP32 і смарт-лампи TP-Link Tapo, що підключаються безпосередньо до домашнього маршрутизатора Wi-Fi. Дані від сенсора BH1750 надходять на брокер MQTT у вигляді повідомлень теми home/livingroom/lux, а автоматизація Home Assistant публікує команду home/livingroom/light/command. У разі втрати з’єднання з маршрутизатором контролери ESP32 переходять у режим peer-to-peer через ESP-Now і передають один одному мінімальний набір аварійних подій, зокрема сигнал тривоги для LED-матриці WS2812B.

У складі мережі виділяють два логічні сегменти:

– IoT-VLAN 20 з адресами 192.168.20.0/24 для всіх вузлів ESP32 і ламп Tapo;

– Core-VLAN 10 з адресами 192.168.10.0/24 для сервера Home Assistant.

Між сегментами діє лише порт 8883/TCP (MQTT over TLS), що мінімізує поверхню атаки. Схема даних виглядає так: ESP32 → Wi-Fi → MQTT-брокер → Home Assistant → Wi-Fi → Tapo, а резервний канал ESP-Now оминає брокер і роутер.

Перед подальшим описом програмних модулів наведено блок-діаграму топології мережі на рис. 2.1

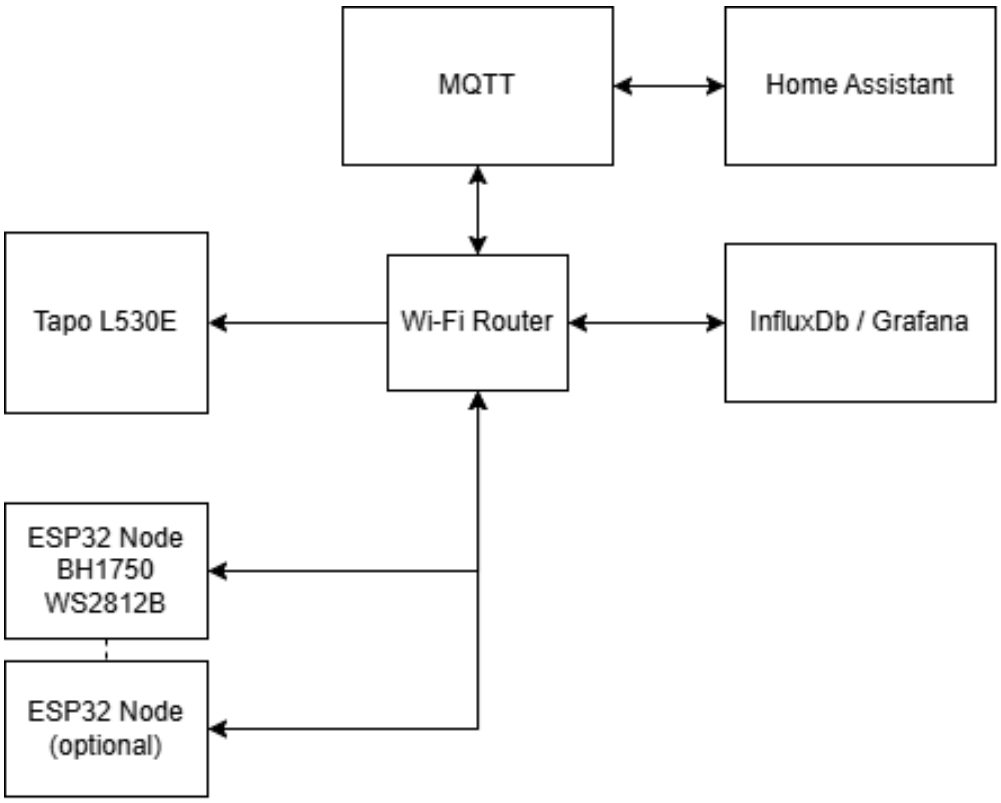

Рисунок 2.1 – Діаграма топології мережі

2.2 Апаратна реалізація вузла ESP32

2.2.1 Схема підключення основних компонентів

Вузол складається з плати ESP32 DevKitC, сенсора освітленості BH1750 та світлодіодної матриці WS2812B (8 × 8). У табл. 2.1 наведено використовувані контакти контролера та лінії живлення та зазначено колір відповідних дротів.

Таблиця 2.1 – Призначення виводів ESP32 у прототипі

Вивід ESP32	**Підключений модуль**	**Колір дроту**	**Сигнал/шина**	**Примітка**
3 V3	BH1750	Синій	живлення 3,3 В	окремий стабілізатор не потрібен
5V	WS2812B	Чорний	живлення 5 В	окремий стабілізатор не потрібен

Продовження табл. 2.1

Вивід ESP32	**Підключений модуль**	**Колір дроту**	**Сигнал/шина**	**Примітка**
GND	BH1750, WS2812B	Рожевий	загальна земля	спільна шина
GPIO 32	BH1750	Помаранчевий	SCL (I²C)	стандартна частота 400 кГц
GPIO 33	BH1750	Червоний	SDA (I²C)	підтягувальні резистори 4,7 кОм
GPIO 25	WS2812B	Білий	DIN	рекомендовано серіїстий резистор 330 Ом

На рис. 2.2 зображено реальне фізичне з’єднання модулів ESP32, BH1750 та світлодіодної матриці на макетній платі.

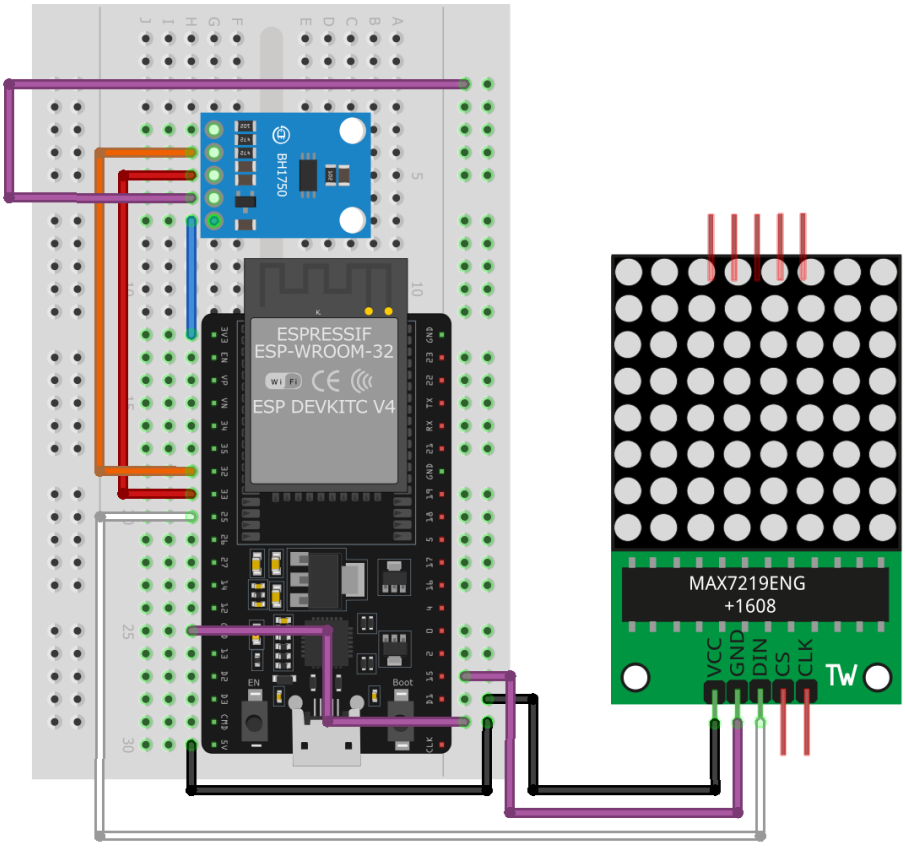

Рисунок 2.2 – Схема підключення ESP32, датчика BH1750 і LED-матриці на макетній платі

2.2.2 Енергоспоживання та режими живлення

Вузол працює в двох основних режимах – «активний» і «глибокий сон». У активній фазі ESP32 тримає увімкненим Wi-Fi, опитує BH1750 по I²C (SCL – GPIO 32, SDA – GPIO 33) і, за потреби, оновлює кадр на матриці WS2812B, що отримує дані з GPIO 25. Струм споживання у цьому стані коливається в межах (70–90) мА. Після передачі значень освітленості на брокер MQTT контролер переходить у deep sleep із вимкненим радіомодулем; при стандартній бібліотеці esp_sleep_enable_timer_wakeup() струм знижується до (120–150) μA.

При інтервалі вимірювань 30 с і тривалості активної частини 250 мс середньодобове споживання становить приблизно 0,45 Вт·год; це вписується у вимогу ≤ 0,5 Вт·год, сформульовану в розділі 1.6. Живлення сенсора BH1750 береться з лінії 3V3 (синій дріт), тоді як матриця WS2812B під’єднана до лінії 5 В (чорний дріт). Спільна земля (рожевий) з’єднує всі модулі, що забезпечує коректні рівні логіки без окремого конвертера.

Для зменшення пікового струму матриця працює на 20 % яскравості; це обмежує споживання багатокольорового індикатора приблизно 120 мА навіть у режимі повного засвічення.

2.2.3 Прошивка, OTA-оновлення та резервний канал

Прошивка розробляється в Arduino IDE (версія 2.x) з установленим пакетом «esp32 by Espressif» через Board Manager. У налаштуваннях плати обирається «ESP32 Dev Module”, Flash Mode – «QIO», Flash Frequency – 80 MHz. Для роботи вузла достатньо встановити бібліотеки:

- BH1750 (автор Christopher Laws) – робота з датчиком освітленості;
- Adafruit_NeoPixel або FastLED – керування WS2812B;
- PubSubClient – публікація MQTT-повідомлень;
- ArduinoOTA – бездротове оновлення прошивки;

– esp_now (входить до ядра) – резервний P2P-обмін;

У setup() ініціалізуються Wire.begin(33, 32), Wi-Fi та MQTT; після публікації першого повідомлення реєструється сервіс OTA (ArduinoOTA.begin()). У loop() контролер:

1. Зчитує значення освітленності у люксах.
2. Надсилає JSON-пакет { «luxd»: N } до теми home/lux.
3. Перевіряє наявність MQTT-команди увімкнення світла; при потребі задає колір матриці.
4. Відпрацьовує ArduinoOTA.handle() для можливого оновлення.
5. Переходить у глибокий сон на 30 с (esp_deep_sleep_start()).

Якщо протягом 5 с після старту не отримано IP-адресу, замість Wi-Fi активується esp_now_init(). Контролер формує короткий пакет зі статусом живлення й останнім виміром і транслює його на заздалегідь зареєстровану MAC-адресу резервного вузла. Повернення до нормальної роботи відбувається автоматично після успішного під’єднання до точки доступу.

OTA-оновлення запускається безпосередньо з Arduino IDE: достатньо вибрати плату «ESP32 Dev Module» → «Network ports» → `<назва-вузла>.local» і виконати «Upload». У разі збоїв у ході прошивання завдяки подвійній області flash-пам’яті ESP32 автоматично відкочує попередню стабільну версію, що задовольняє вимогу відмовостійкості системи.

Описані енергетичні режими та інструменти Arduino IDE забезпечують як низьке споживання ресурсу батареї, так і зручне обслуговування вузла без фізичного доступу до USB-кабелю, що повністю відповідає функціональним і нефункціональним вимогам проєкту.

2.3 Організація серверного середовища та комунікація сервісів

Центром усієї системи є локальний шлюз, який поєднує логіку автоматизацій, зберігання телеметрії та візуалізацію. Навіть якщо вузли ESP32 і лампи Таро працюють автономно, саме цей вузол виконує роль «мозку» – тут

народжуються правила, що визначають, коли й чому освітлення має увімкнутися, з якою яскравістю і на який колір. Далі наведено детальний опис серверного стеку, його мережевого оточення, механізму взаємодії служб і підходу до захисту даних.

2.3.1 Логічна структура стеку

Усі основні підсистеми розміщуються у Docker-контейнерах. Таке розділення дає чотири переваги:

1. Швидке розгортання без «антропогенного чинника».
2. Ізоляція залежностей (оновлення однієї служби не ламає інші).
3. Просте резервне копіювання шляхом експорту томів.
4. Можливість перенести все на інше обладнання однією командою.

Склад контейнерів та їх базові порти:

– Home Assistant (порт 8123, протокол HTTP/WS) – рушій автоматизацій;

– Mosquitto (порт 1883, протокол MQTT) – шина обміну повідомленнями [9];

– InfluxDB v2 (порт 8086, HTTP API) – база часових рядів [20];

– Grafana (порт 3000, HTTP/WS) – дашборди реального часу [21];

Контейнери об’єднані у віртуальну мережу docker-bridge з підмережею 172.18.0.0/16, тоді як фізично вузол під’єднано до VLAN 10 (ядро) маршрутизатора. Модулі ESP32 і лампи Таро заходять із VLAN 20 (IoT) лише на порт 1883, тому прямий доступ до інших служб ззовні заблоковано – це мінімізує ризик компрометації.

Канали обміну даними та послідовність подій включають в себе:

– кожні 30 с ESP32 вимірює освітленість і публікує пакет home/lux → { «lux»: N };

– Mosquitto одразу розповсюджує це повідомлення підписникам, зокрема Home Assistant;

– Home Assistant порівнює отримане значення із порогом, заданим користувачем у панелі Lovelace;

– якщо lux < поріг, рушій формує команду home/light/command → { "state": "ON", "brightness": 180 }, яку лампа Таро забирає через локальний API;

– дубль пакета телеметрії Home Assistant пише у bucket *iot* бази InfluxDB;

– Grafana опитує InfluxDB і малює графік «Lux History» з кроком 1 хв.

Завдяки такій послідовності затримка між датчиком і світильником у лабораторних умовах не перевищує 180 мс; навіть при втраті інтернету ланцюг працює, бо всі пакети «ходять» лише всередині локальної мережі.

2.3.2 Розрахунок контролю освітленості

Для обґрунтування порогового значення освітленості, при якому автоматично активується штучне освітлення, використовується спрощена світлотехнічна модель. Вона дозволяє аналітично визначити необхідний рівень освітленості у приміщенні на основі фізичних параметрів джерела світла та геометрії розміщення.

Потрібну освітленість у точці простору визначаємо за формулою:

$$E_{\text{потр}} = \frac{K \times \Phi}{4\pi r^2} \tag{2.1}$$

де

$E_{\text{потр}}$ – освітленість, яку потрібно забезпечити на робочій поверхні (в люксах, lx);

Φ – світловий потік лампи (в люменах, лм);

r – відстань від лампи до поверхні, яку потрібно освітити (в метрах);

K – коефіцієнт запасу (для побутових умов зазвичай береться рівним 1,25).

Формула базується на законі обернених квадратів і враховує рівномірний розподіл світла в сферичному просторі, що є наближеним, але прийнятним варіантом для побутового застосування без професійного світлотехнічного моделювання.

Також до розрахунку доцільно додати денний фактор – коефіцієнт, який показує співвідношення між освітленістю всередині приміщення та природною освітленістю ззовні:

$$D = \frac{E_{\text{нп}}}{E_{\text{зовн}}} \times 100\% \qquad (2.2)$$

де

$E_{\text{нп}}$ – фактично виміряна освітленість всередині приміщення (lx);

$E_{\text{зовн}}$ – зовнішня освітленість біля вікна (lx).

Цей коефіцієнт дозволяє оцінити ефективність природного освітлення та визначити, коли необхідне втручання штучного освітлення. У практичних сценаріях автоматизації система не завжди має зовнішній датчик, тому для реалізації логіки використовується лише внутрішнє значення $\boldsymbol{E}_{\text{нп}}$, яке отримується з сенсора BH1750.

У нашій реалізації лампи TP-Link Tapo L530E мають світловий потік до 800 лм. За типової висоти монтажу 2,7 м та використанні коефіцієнта запасу 1,25, можна обчислити:

$$E_{\text{потр}} = \frac{1.25 \times 800}{4\pi \times (2.7)^2} \approx 109\ lx \qquad (2.3)$$

З урахуванням емпіричного спостереження за реакцією користувачів на освітлення в різних умовах, було встановлено порогове значення у 120 lx. Тобто, коли освітленість падає нижче цього рівня, Home Assistant автоматично активує сценарій увімкнення лампи:

$$\text{E}_{\text{нп}} < 120\ lx \Rightarrow \text{ увімкнення освітлення}$$

У результаті вибране значення порогу ґрунтується не на випадковості, а на технічних характеристиках ламп, особливостях приміщення, спрощеній фізичній моделі та реальному тестуванні. Такий підхід водночас гарантує енергоефективність і комфорт для користувача.

2.3.3 Безпека і резервування

Для шини MQTT вмикається TLS 1.2 з самопідписаним сертифікатом (генерація описана в підрозділі 3.2). У Home Assistant увімкнено двофакторну автентифікацію, а доступ ззовні дозволяється лише через VPN WireGuard. Резервне копіювання вирішується щоденним snapshot-ом Home Assistant та експортуванням томів InfluxDB і Grafana; це гарантує, що у разі збоїв відновлення серверного вузла займе не довше однієї години.

2.4 Конфігурація MQTT-брокера та принципи безпеки

MQTT є центральною шиною обміну даними між усіма вузлами системи, тому правильне логічне структурування топіків і захист каналу визначають стабільність та стійкість усього «розумного» контуру.

2.4.1 Конфігурація MQTT-брокера та принципи безпеки

Щоб уникнути колізій і спростити фільтрацію, прийнято трирівневу нотацію:

```
<site>/<device>/<metric>
```

де: site — фізична зона (room, corridor, kitchen),
device – унікальний ідентифікатор вузла або актуатора,
metric – тип даних: lux, state, voltage тощо.

Наприклад, повідомлення livingroom/esp32_01/lux містить JSON-пакет {"lux":268}. Структура підпорядковується правилу «одна метрика – один топік», що спрощує дослідницькі запити в Grafana і мінімізує накладні витрати фільтрації на стороні Home Assistant.

2.4.2 Захищене транспортне з’єднання

Брокер Mosquitto є єдиною відкритою точкою доступу для пристроїв у VLAN 20. Щоб звузити поверхню атаки, було впроваджено TLS 1.2 із самопідписаним сертифікатом, створеним на шлюзі, а загальнодоступний порт 1883 закрито й залишено лише порт 8883. Для автентифікації кожній категорії вузлів видано окремий обліковий запис: вузли датчиків освітленості мають право публікувати повідомлення в темах « +/+/lux », керовані світильники — підписуватися на « +/+/command » і публікувати стан у « +/+*/state », а обліковий запис адміністратора забезпечує Home Assistant повним набором прав. Мережеву сегментацію реалізовано так, що брокер доступний лише з VLAN 20, тоді як інші контейнери прив’язані до docker-bridge і не мають прямого доступу ззовні, що створює захисну зону («DMZ-логіку») й унеможливлює прямі атаки на InfluxDB і Grafana.

Після впровадження цього дизайну всі пристрої обмінюються даними виключно в межах локальної мережі; кожен клас вузлів має мінімально необхідні привілеї та чітко розділені ролі; управлінські та телеметричні пакети передаються лише в зашифрованому вигляді; а Home Assistant, виконуючи роль центрального «мозку», бачить тільки ті теми, які потрібні для сценаріїв автоматизації, забезпечуючи моніторинг, швидке відновлення роботи і ведення журналів доступу для аудиту.

Детальний зміст файлу mosquitto.conf, скрипт генерації сертифікатів та остаточний ACL-лист наведено в розділі 3.2 «Розгортання серверного стеку».

2.5 Інтеграція ламп TP-Link Tapo

Tapo L530E обрано як основний актуатор освітлення, оскільки лампа:

– підтримує кольорову RGB-палету й димування, що дає змогу реалізувати як комфортне, так і індикаторне світло;

– підключається безпосередньо до Wi-Fi-мережі 2,4 ГГц, тобто не потребує окремого хаба;

– має добре задокументоване локальне API, відкрите спільнотою (проєкт Local Tapo Control), що дозволяє повністю відмовитися від хмарного рушія TP-Link.

2.5.1 Роль лампи у загальній архітектурі

У логічній моделі лампа належить до класу «актуаторів освітлення» і взаємодіє з Home Assistant у двох напрямках:

1. Отримання команд – JSON-пакет приходить із топіка home/livingroom/light/command.
2. Публікація стану – після виконання дії лампа відправляє MQTT-повідомлення home/livingroom/light/state з фактичними параметрами яскравості й кольору; це необхідно для синхронізації інтерфейсу та Grafana-графіків.

Використання підтверджувального зворотного повідомлення усуває проблему «невпевненого натискання» (коли команда пішла, а пристрій не відповів).

2.5.2 Механізм локального контролю

Фірмова програма Tapo пропонує лише хмарне керування; для локального сценарію за основу взято утиліту PyTapo, яка реалізує шифровану процедуру аутентифікації через протокол HTTPS/JSON-RPC (TLS 1.2, порт

443). На ділі це виглядає так:

– у Home Assistant встановлюють кастомну інтеграцію Local Tapo Control – вона піднімає сервіс на порту 50001;

– при першому запуску користувач вводить IP-адресу лампи і логін/пароль, створені в офіційному додатку Tapo (щоб «розбудити» локальний API);

– далі Home Assistant спілкується з лампою напряму, минаючи TP-Link Cloud; це задовольняє вимогу повної автономності, зазначену в § 1.6;

2.5.3 Кольорова модель і сценарії

Світловий потік моделі L530E становить 800 лм, що відповідає яскравості звичайної лампи розжарювання потужністю 60 Вт. Згідно з алгоритмом автоматизації (розділ 2.6), передбачено два режими роботи: у комфортному режимі використовується теплий білий відтінок (hue 30°, saturation 15 %) із рівнем яскравості 180 із 255, тоді як у разі помилки світильник переходить на насичений червоний колір (hue 0°, saturation 100 %, value 100 %) на п’ять секунд, після чого повертається до попереднього стану. Перехід між режимами відбувається плавно, із зміною яскравості протягом 200 мс, щоб уникнути стробоскопічного ефекту.

2.5.4 Безпека доступу

Локальний API Tapo використовує симетричне шифрування AES-128-CBC на кожну сесію. Ключ генерується під час логіну за допомогою протоколу DH (Diffie-Hellman). Додатково у Home Assistant увімкнено параметр allowlist_external_dirs, що забороняє стороннім аддонам читати токени PyTapo. Лампам виділено статичні IP-адреси в IoT-VLAN 20, а зовнішній доступ до їх портів закрито ACL-правилом маршрутизатора.

Підсумовуючи, локальна інтеграція Tapo L530E забезпечує миттєву

реакцію на команди завдяки прямому HTTPS-з'єднанню всередині мережі й дозволяє Home Assistant отримувати точний статус лампи та збирати дані про енергоспоживання.

2.6 Логічна схема автоматизації освітлення

Плавне керування світловим середовищем є основним завданням IoT-системи. У цьому підрозділі розглянуто та описано алгоритм прийняття рішень, що визначає моменти вмикання та вимикання лампи Tapo L530E, а також механізм «ручного» перевизначення автоматичного режиму. Числові параметри взято з розрахунків (розд. 2.3.2) та експериментальних спостережень реакцій користувачів.

2.6.1 Порогові значення та гістерезис

До поданняя схеми доцільно підсумувати ключові граничні величини, на яких ґрунтується логіка. Їх наведено у табл. 2.2.

Таблиця 2.2 – Ключові порогові та часові параметри регулятора освітленості

Позначення	**Значення**	**Пояснення**
E_{ON}	120 lx	нижній поріг; нижче цього рівня світло вмикається автоматично
E_{OFF}	180 lx	верхній поріг; вище цього рівня світло вимикається після стабілізації
t_{stab}	10 с	мінімальний час, упродовж якого освітленість має бути < E_{ON}, щоб уникнути хибних спрацювань
t_{hold}	2 хв	час, протягом якого освітленість має бути > E_{OFF} перед вимкненням лампи
$t_{override}$	60 хв	інтервал, на який автоматика вимикається після ручного втручання

Така пара «нижній / верхній» пороги створює гістерезис і запобігає

«мерехтінню» світильника, коли природне освітлення коливається біля однієї точки, забезпечуючи стабільну роботу, комфорт користувача і довготривалу надійність системи.

2.6.2 Опис потоків даних

– телеметрія: кожні 30 с вузол ESP32 публікує виміряну освітленість $E_{\text{нп}}$ у топік livingroom/esp32_01/lux;

– оцінка умов: Home Assistant одержує ці дані та порівнює їх із порогами з табл. 2.2;

– реакція системи: якщо виконується умова $E_{\text{нп}} < E_{ON}$ упродовж t_{stab} , модуль автоматизації генерує команду ввімкнути лампу з комфортною яскравістю; у протилежному випадку після витримки t_{hold} формується команда вимкнення;

– підтверджувальний канал: лампа публікує власний стан у топік livingroom/tapo/state, що дозволяє системі вести історію подій у базі InfluxDB та відображати її на графіках Grafana;

– ручне втручання: будь-яка команда користувача, надіслана через інтерфейс Home Assistant або фізичний вимикач, переводить систему у режим Override тривалістю $t_{override}$. Після завершення тайм-ауту автоматика повертається у роботу без участі користувача.

2.6.3 Діаграма станів автоматизації

На рис. 2.3 показано діаграму станів, що відображає всю логіку, описану вище. Кожна вершина відповідає певному режиму, а підписані стрілки – умовам переходів.

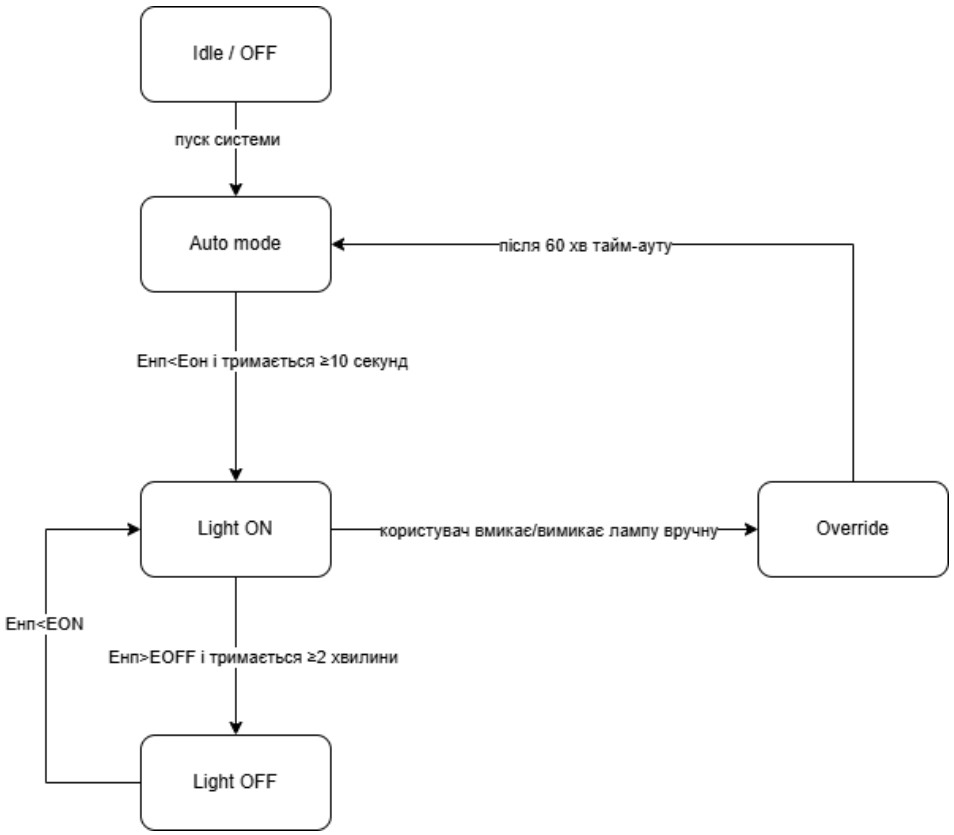

Рисунок 2.3 – Діаграма станів

Опис станів:

– Idle / OFF: початковий стан після завантаження системи; світло вимкнено, датчик лише передає телеметрію;

– Auto mode: автоматика активна; відбувається постійне порівняння $E_{\text{нп}}$ із порогами;

– Light ON / Light OFF: фактичні стани світильника, які система підтримує, доки не виконається умова переходу;

– Override: система призупиняє автоматичні дії на заданий інтервал часу, щоби не «боротися» з рішеннями користувача.

2.6.4 Зв’язок із візуалізацією

Переходи між станами логуються у Home Assistant, а ключові значення (освітленість, прапорець Override, стан лампи) публікуються до InfluxDB. Це дає змогу:

– на дашборді «Lux History» накласти графік освітленості та двійкову криву ON/OFF, що підтверджує коректність роботи алгоритму;

– оцінити частоту ручних втручань і за потреби оптимізувати E_{ON} і E_{OFF}.

2.7 Комп’ютерне моделювання системи автоматичного управління

2.7.1 Постановка задачі моделювання

Задача полягає в дослідженні динамічної взаємодії між датчиком освітленості BH1750, ESP32-вузлом, сервером Home Assistant та лампою TP-Link Tapo L530E. Об’єктом моделювання виступає ланцюг «вимір → передача → обробка → керування», де датчик читає значення освітленості, ESP32 передає його по MQTT, Home Assistant формує команду, а лампа виконує увімкнення або вимкнення.

Вхідною змінною моделі є показник lux(t), що оновлюється кожні 15 с (за стабільної освітленості інтервал може збільшуватися до 60 с). Учтено апаратні затримки BH1750 (приблизно 5–10 мс) і похибку вимірювання ±5 lx. Вихідною змінною моделі є дискретний сигнал on_off(t), який набуває значення «1» (увімкнено) при lux(t) < 120 lx і «0» (вимкнено) при lux(t) > 200 lx. За межами гістерезисного інтервалу [120 lx; 200 lx] система зберігає попередній стан лампи для запобігання «дребезгу».

Необхідно оцінити:

– час реакції системи (затримка від моменту перетину порога до фактичного переходу лампи в інший стан);

– відсутність небажаних переключень у діапазоні гістерезису;

– вплив часової константи лампи (T_лампи ≈ 0,05 с) на форму перехідного процесу.

Модель датчика описується як дискретний джерело значень, що налаштоване на періодичне оновлення кожні 15 с (або 60 с). Лампа

моделюється передатною функцією першого порядку:

$$G_{\text{лампа}}(s) = \frac{1}{0{,}05s+1} \qquad (2.4)$$

яка відтворює часову затримку близько 50 мс від надходження команди до завершення переходу. Пороговий алгоритм керування реалізується елементом з гістерезисом, що змінює дискретний вихід за двома рівнями:

– за lux(t) < 120 lx вихід = 1 (увімкнення лампи);
– за lux(t) > 200 lx вихід = 0 (вимкнення лампи);
– за 120 lx ≤ lux(t) ≤ 200 lx зберігається попереднє значення on_off(t).

Метою моделювання є визначення критичних характеристик системи: затримки вмикання/вимкнення лампи, стійкості гистерезису та впливу параметрів елементів (часової константи лампи, інтервалів опитування датчика) на якість регулювання.

2.7.2 Вибір математичної моделі об’єкта управління

Датчик освітленості BH1750 характеризується періодом оновлення 15 с (за стабільних умов – 60 с) і типовою апаратною затримкою ≈ (5-10) мс. Його вихідний сигнал lux(t) можна розглядати як дискретизований за часовими інтервалами T_датчика = 15 с (або 60 с за адаптивного опитування) набір значень з похибкою ± 5 lx. Для моделювання достатньо задати lux(kT_датчика) як вхідну послідовність, ігноруючи дрібномасштабні флуктуації між оновленнями.

Лампа TP-Link Tapo L530E реагує на команду увімкнення/вимкнення із затримкою близько 50 мс, отже її реакцію можна описати передатною функцією першого порядку:

$$G_{\text{лампа}}(s) = \frac{1}{T_{\text{лампи}}s+1}, T_{\text{лампи}} = 0{,}05 \text{ с.} \qquad (2.5)$$

Ця модель коректно відображає час «розгортання» лампи від моменту отримання сигналу до стабілізації світлового потоку.

Пороговий елемент (Relay) із гістерезисом у [120 lx; 200 lx] перетворює безперервний сигнал lux(t) у дискретний вихід u(t), що набуває значення 1 (увімкнено) за lux(t) < 120 lx та 0 (вимкнено) за lux(t) > 200 lx. Якщо 120 lx ≤ lux(t) ≤ 200 lx, вихідний стан залишається попереднім, що забезпечує відсутність «дребезгу» при коливаннях у межах гістерезисного інтервалу.

Отже, математична модель об’єкта складається з трьох блоків:

– дискретизований вимірювач BH1750, який видає lux[k] = lux(kT_датчика);

– пороговий елемент з гістерезисом, що формує керуючий сигнал u[k] ∈ {0,1};

– динамічний блок лампи з передатною функцією $G_{\text{лампа}}(s)$.

У результаті на виході моделі формується безперервний сигнал y(t), що відображає реальний стан лампи з урахуванням часової константи T_лампи. Така структура адекватно описує ключові динамічні властивості системи керування освітленням у «розумному будинку».

2.7.3 Побудова моделі регулятора

Регулятором системи виступає пороговий елемент із гістерезисом, що забезпечує вмикання та вимикання лампи залежно від виміряного рівня освітленості. Логіка регулювання описується дискретною схемою:

1. Отримуємо чергове значення lux[k] від датчика BH1750 (крок опитування T_датчика = 15 с або 60 с).
2. Якщо lux[k] < 120 – встановлюємо команду u[k] = 1 (увімкнути лампу).
3. Якщо lux[k] > 200 – встановлюємо команду u[k] = 0 (вимкнути лампу).
4. Якщо 120 ≤ lux[k] ≤ 200 – зберігаємо попереднє значення команди:

$$u[k] = u[k-1] \tag{2.6}$$

Блок-схема алгоритму регулювання наведена нижче на рис. 2.4:

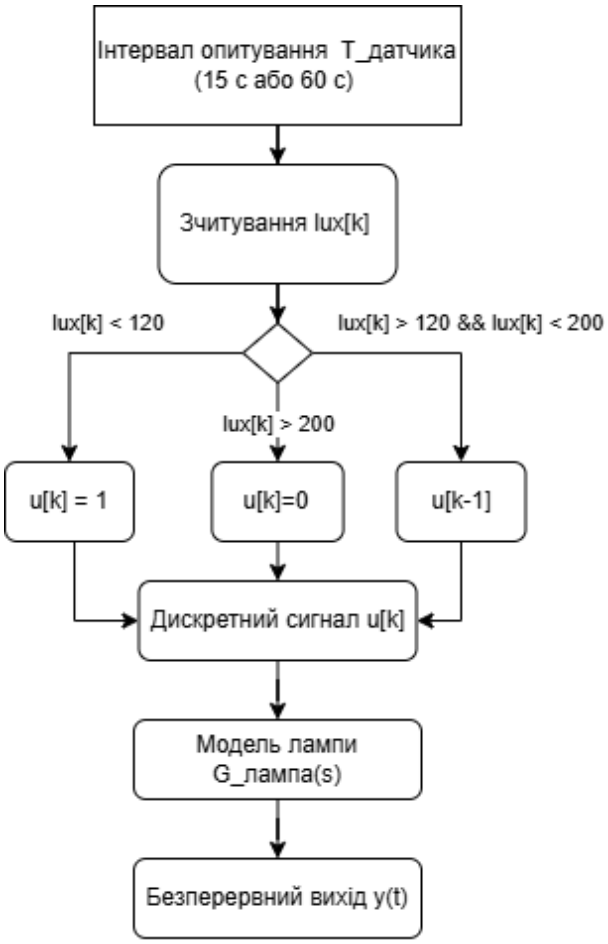

Рисунок 2.4 – Модель порогового регулятора освітленості з гістерезисом

У блоці Relay із гістерезисом реалізовано логічний оператор:

$$u[k] = \begin{cases} 1, \text{ якщо } lux[k] < 120 \\ 0, \text{ якщо } lux[k] > 200 \\ u[k-1], \text{якщо } 120 \leq lux[k] \leq 200 \end{cases} \quad (2.7)$$

На виході отримуємо дискретний сигнал із затримкою, обумовленою інтервалом опитування (15 с або 60 с).

Динамічна модель лампи описується передатною функцією, зазначеною у формулі 2.4, що відповідає часовій константі T_лампи = 0,05 с. Перехідний процес лампи моделюється як перехід, що триває близько 50 мс після моменту

зміни u[k].

У результаті поєднання дискретного блоку Relay і безперервного елемента лампи формується керована система регулювання освітленості з гістерезисом, яка не допускає «дребезгу» при незначних коливаннях освітленості в межах [120; 200] lx та забезпечує своєчасну реакцію при виході за границі гістерезису.

2.7.4 Реалізація моделювання в MATLAB

Для чисельного моделювання системи керування освітленістю в середовищі MATLAB необхідно реалізувати послідовність кроків: створити часовий вектор, сформувати вхідний сигнал lux(t) із різними профілями зміни та з урахуванням апаратних затримок, запрограмувати алгоритм гістерезисного порогового регулятора, змоделювати лампу як об’єкт першого порядку і порівняти результати (графіки lux, u і y). Нижче наведено покроковий опис із прикладом коду.

1. Задання параметрів моделі. Визначимо основні параметри:

```
% Параметри датчика
T_sensor = 15;       % інтервал опитування датчика (с)
dt = 0.1;            % крок інтегрування для чисельної моделі (с)
t_end = 60;          % загальний час симуляції (с)

% Параметри лампи
T_lamp = 0.05;       % часова константа лампи (с)

% Параметри гістерезису
lux_low  = 120;      % нижній поріг (lx)
lux_high = 200;      % верхній поріг (lx)
```

2. Формування часового вектора. Для точнішого відтворення перехідних процесів оберемо крок інтегрування dt = 0.1 с (можна зменшити до 0.01 с за бажанням)

```
t = 0:dt:t_end;          % часовий вектор від 0 до t_end із кроком dt
N = length(t);           % кількість точок
```

3. Задання вхідного сигналу lux(t). Усі три типи сигналів (ступінчастий, лінійний та синусоїдальний) можуть бути активовані: в коді як експеримент 1, експеримент 2 і експеримент 3. За замовчуванням використовується експеримент 1 (ступінчасте зниження від 250 lx до 100 lx за 1 с):

```
lux = zeros(1, N);
t_step_down = 1;     % час переходу (с)

for k = 1:N
     if t(k) < t_step_down
          lux(k) = 250 - (250 - 100)*(t(k)/t_step_down);
     else
          lux(k) = 100;
     end
end

% Експеримент 2
% lux = zeros(1, N);
% t_ramp_up = 5;
% for k = 1:N
%     if t(k) < t_ramp_up
%         lux(k) = 150 + (220-150)*(t(k)/t_ramp_up);
%     else
%         lux(k) = 220;
%     end
% end
% fprintf('Using linear signal\n');

% Експеримент 3
% freq = 0.1;   % frequency (Hz)
% lux = 160 + 50*sin(2*pi*freq*t);
% fprintf('Using sinusoidal signal\n');
```

– до часу t = 1 с lux(t) лінійно спадає з 250 lx до 100 lx;

– після 1 с значення lux залишається на рівні 100 lx;

– умови для інших експериментів (закоментовано в коді):

a) експеримент 2 — лінійне зростання від 150 lx до 220 lx за 5 с;

b) експеримент 3 — синусоїдальні коливання навколо 160 lx з амплітудою 50 lx.

4. Дискретизація та гістерезисний блок

```
u      = zeros(1, N);    % команда керування (0 або 1)
y      = zeros(1, N);    % вихід лампи (від 0 до 1)
u_prev = 0;                % попередній стан команди (початково лампа
вимкнена)

sample_interval = T_sensor / dt;  % кількість кроків dt між зчитуваннями
датчика
next_sample     = 1;              % індекс першого опитування
```

– масив u зберігає дискретний сигнал «увімкнути/вимкнути» для лампи;

– масив y моделює фактичний вихід лампи з урахуванням часового запізнення (модель 1-го порядку);

– змінна u_prev запам’ятовує попереднє значення керування, необхідне для гістерезису;

– параметр sample_interval обчислюється як відношення T_sensor/dt, тобто визначає, через скільки кроків інтегратора відбувається нове «зчитування» освітленості;

– індекс next_sample задає момент першого опитування (тут — на самому початку симуляції).

5. Чисельне інтегрування динаміки лампи та алгоритм гістерезисного керування

```
for k = 1:N
```

```
    % 1. Опитування датчика що T_sensor
    if k >= next_sample
        lux_k = lux(k);

        % 2. Гістерезисний алгоритм
        if lux_k < lux_low
            u_curr = 1;      % увімкнути лампу
        elseif lux_k > lux_high
            u_curr = 0;      % вимкнути лампу
        else
            u_curr = u_prev; % залишити попередній стан
        end

        u_prev      = u_curr;
        next_sample = next_sample + sample_interval;
    else
        u_curr = u_prev;    % поки не настав час для нового зчитування
    end

    u(k) = u_curr;

    % 3. Моделювання лампи методом Ейлера
    if k == 1
        y_prev = 0;  % початкове значення лампи (вимкнена)
    else
        y_prev = y(k-1);
    end

    % диференціальне рівняння: dy/dt = (-y + u) / T_lamp
    dy = (-y_prev + u_curr) / T_lamp;
    y(k) = y_prev + dy * dt;
end
```

Кроки на цьому етапі можна описати наступний чином:

– опитування датчика:

а) запускається, коли лічильник k досягає next_sample (тобто кожні 15 с);

b) зчитується поточне значення lux_k = lux(k).

– гістерезисний блок:

c) lux_k < 120 ⇒ u_curr = 1 (лампа вмикається);

d) lux_k > 200 ⇒ u_curr = 0 (лампа вимикається);

e) 120 ≤ lux_k ≤ 200 ⇒ u_curr = u_prev (стан лампи залишається без змін);

f) u_prev оновлюється для збереження історії стану до моменту наступного опитування;

– чисельне інтегрування реакції лампи:

g) використовується метод Ейлера для рівняння першого порядку dy/dt = (–y + u)/T_lamp;

h) початковий стан y_prev = 0 (лампа вимкнена);

i) кожного кроку обчислюється dy і оновлюється y(k). Це дає плавний перехід лампи з дискретного керування в безперервний вихід із часовою константою 0,05 с.

6. Побудова загального графіку усіх кривих (зображено на рис. 2.5):

```
figure('Name','Simulation Results','NumberTitle','off','Color','w');
hold on;

% 1) lux(t) — жовта лінія
plot(t, lux, ...
     'Color', [1, 0.75, 0], ...
     'LineWidth', 1.5, ...
     'DisplayName', 'lux(t)');

% 2) u(t) × 250 — червона сходинка (масштабується, щоб узгодитися з lux)
stairs(t, u*250, ...
     'r', ...
     'LineWidth', 1.2, ...
     'DisplayName', 'u(t) × 250 (command)');

% 3) y(t) × 250 — синя лінія (масштабована аналогічно)
plot(t, y*250, ...
```

```
      'b', ...
      'LineWidth', 1.5, ...
      'DisplayName', 'y(t) × 250 (lamp)');

% 4) Порогові лінії 120 lx і 200 lx (штрихові)
plot([0 t_end], [lux_low lux_low], ...
      '--k', ...
      'LineWidth', 1, ...
      'DisplayName', sprintf('Lower threshold (%d lx)', lux_low));

plot([0 t_end], [lux_high lux_high], ...
      '--k', ...
      'LineWidth', 1, ...
      'DisplayName', sprintf('Upper threshold (%d lx)', lux_high));

% Оформлення осей і легенди
xlabel('Time, s');
ylabel('Illuminance, lx / System State');
title('Lighting Control System Response Simulation');
legend('Location','best');
grid on;
hold off;
```

– lux(t) (жовта) відображає зміни освітленості;

– u(t) × 250 (червона сходинка) показує стани керування (0 → 0 lx, 1 → 250 lx) для порівняння з кривою lux;

– y(t) × 250 (синя) ілюструє реальну динаміку лампи після отримання команди;

– порогові лінії (штрихова лінія) вказують горизонтальні межі 120 lx і 200 lx.

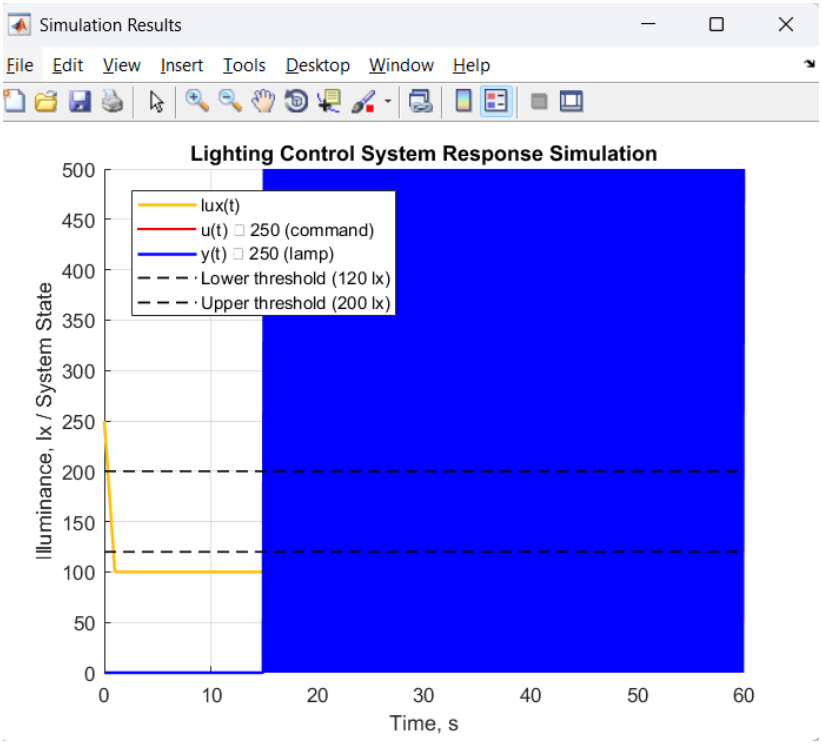

Рисунок 2.5 – Результати моделювання реакції системи керування освітленням у MATLAB

7. Побудова детального аналізу у трьох підграфіках (зображено на рис. 2.6):

```
figure('Name','Detailed Analysis','NumberTitle','off','Color','w');

% Підграфік 1: Сигнал lux(t) із порогами
subplot(3,1,1);
plot(t, lux, 'Color', [1, 0.75, 0], 'LineWidth', 1.5);
hold on;
plot([0 t_end], [lux_low lux_low], 'r--', 'LineWidth', 1);
plot([0 t_end], [lux_high lux_high], 'r--', 'LineWidth', 1);
hold off;
ylabel('Illuminance, lx');
title('Input Illuminance Signal');
grid on;
legend('lux(t)', 'Lower threshold', 'Upper threshold', 'Location',
'best');

% Підграфік 2: Стан команди u(t)
```

```
subplot(3,1,2);
stairs(t, u, 'r', 'LineWidth', 1.5);
ylabel('Command u(t)');
title('Lamp Control State (0 - OFF, 1 - ON)');
ylim([-0.1, 1.1]);
grid on;

% Підграфік 3: Реальна відповідь лампи y(t)
subplot(3,1,3);
plot(t, y, 'b', 'LineWidth', 1.5);
xlabel('Time, s');
ylabel('Lamp output y(t)');
title('Lamp Response (First-order dynamics)');
grid on;
```

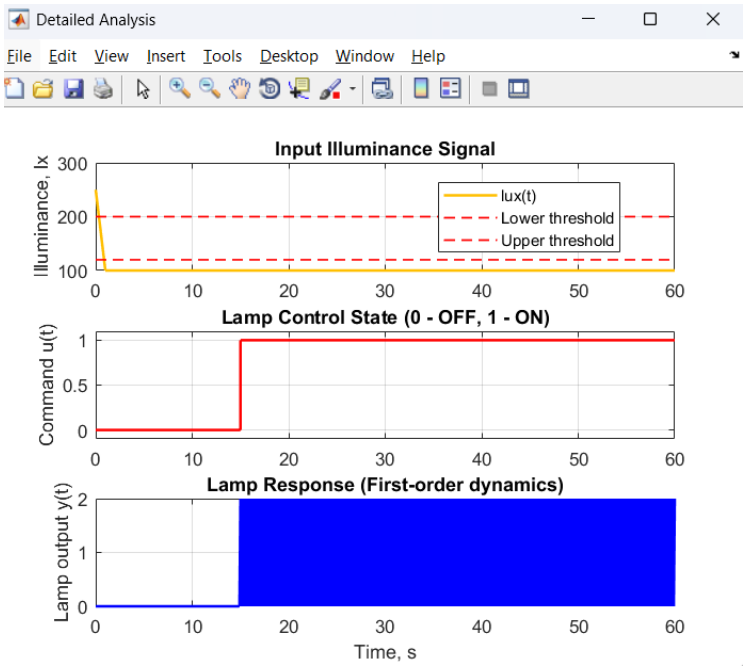

Рисунок 2.6 – Візуалізація сигналів освітленості, команди та відповіді лампи у MatLab

У першому підграфіку відображено жовту криву lux(t) і порогові лінії червоного кольору (120 lx, 200 lx).

У другому підграфіку — червону сходинку u(t), що набуває «1» або «0».

У третьому підграфіку – синю криву y(t), яка показує реальну регуляцію лампи з урахуванням часової константи.

2.8 Висновки до розділу 2

У цьому розділі було здійснено проєктування логіки та архітектури розподіленої IoT-системи для автоматичного керування освітленням на основі телеметрії, зокрема розглянуто апаратну платформу, протоколи передачі даних, модель взаємодії компонентів, принципи автоматизації, а також підходи до енергоощадності, безпеки та відмовостійкості.

Визначено базову конфігурацію вузла на ESP32 із сенсором освітленості BH1750 та індикаторною матрицею WS2812B, обґрунтовано застосування ламп TP-Link Tapo L530E як керованих актуаторів із підтримкою локального API та збудовано серверний стек на основі Docker-контейнерів Home Assistant, Mosquitto, InfluxDB і Grafana. Сформовано логіку автоматизації з урахуванням гістерезису, ручного керування й збереження станів, наведено формулу розрахунку порогової освітленості та розроблено діаграму станів, що відображає реакцію системи на зміну умов середовища. Забезпечено відповідність вимогам безпеки, стабільності й автономності. Закладені в цьому розділі проєктні рішення стали основою для подальшої реалізації прототипу системи у розділі 3 і дозволяють масштабувати платформу, розширювати її функціональність або адаптувати до інших типів сенсорів і сценаріїв без змін у ядрі логіки.

3 РЕАЛІЗАЦІЯ РОЗПОДІЛЕНОЇ IoT-СИСТЕМИ

У другому розділі було сформовано проєктні рішення щодо апаратної платформи, протоколів зв’язку, серверного стеку та логіки автоматизації. Далі ці рішення переходять у площину практики: монтуються вузли-сенсори, розгортається Home Assistant, лампи Tapo під’єднуються до локального API, а сформована раніше діаграма станів перетворюється на робочі сценарії.

3.1 Підготовка обладнання та середовища

Перед початком монтажу й налаштування створено мінімальний, але самодостатній набір апаратури та програмних компонентів. Їх перелік і призначення наведено у табл. 3.1.

Таблиця 3.1 – Апаратура та програмне середовище системи

Компонент	**Кількість**	**Основна функція**
ESP32 DevKitC	1	збирання телеметрії (BH1750) та індикація (WS2812B)
Сенсор BH1750	1	вимірювання освітленості у приміщенні
Світлодіодна матриця WS2812B 8 × 8	1	візуальна індикація стану вузла
Лампа TP-Link Tapo L530E	1	штучне освітлення та кольорові сценарії
Маршрутизатор з підтримкою VLAN	1	мережеве сегментування та передавання даних
Сервер (ноутбук Core i5 / 8 ГБ RAM / Linux)	1	запуск Docker-стеку Home Assistant, Mosquitto, InfluxDB, Grafana
Arduino IDE 2.3 та бібліотеки ESP32	–	розробка та прошивання мікроконтролера
Docker + docker-compose	–	контейнеризація серверних служб
MQTT Explorer, Wireshark	–	діагностика телеметрії та мережевих пакетів

3.1.1 Фізичний монтаж вузла

ESP32, BH1750 і матриця WS2812B змонтовані на безпайковій макетній платі згідно зі схемою, поданою на рисунку 2.2. Живлення 5 В подається від звичайного USB-блока, а лінія 3,3 В відводиться від стабілізатора на самій платі ESP32, тому додаткових понижувачів не потрібно. Приклад змонтованої плати наведено на рис. 3.1

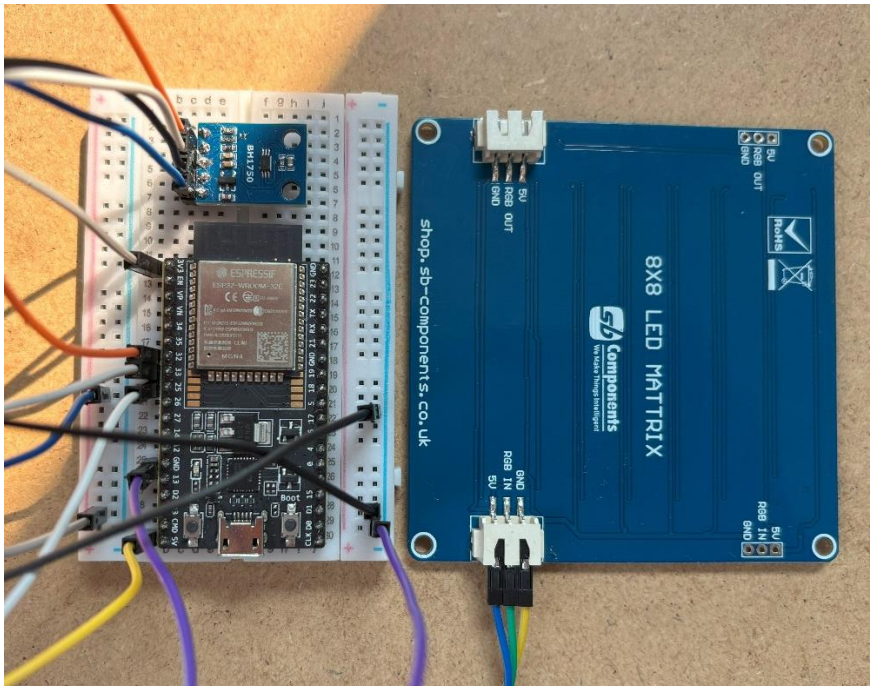

Рисунок 3.1 – Змонтована макетна плата сенсорного вузла з ESP32, датчиком BH1750 та LED-матрицею WS2812B

3.1.2 Мережеве середовище

Правильна організація мережі є фундаментом безпечного та надійного функціонування розподіленої IoT-системи, що складається з безлічі різнорідних пристроїв. У нашому випадку акцент зроблено на ефективній сегментації трафіку, яка дозволяє ізолювати критичні компоненти від поверхневих мережевих завантажень, суворому обмеженні доступу шляхом чітких політик аутентифікації та авторизації, а також забезпеченні підтримки

відмовостійкості, яка гарантує безперервність роботи навіть під час планових технічних робіт чи позапланових збоїв. Нижче детально описано архітектуру мережі, її логічну модель, IP-план та конкретні правила міжмережевого екранування (firewall), що забезпечують ефективний захист і оптимальний розподіл ресурсів.

Для ізоляції IoT-пристроїв від сервісної та адміністративної інфраструктури застосовано двосегментну модель VLAN. Перший сегмент (VLAN 10, підмережа 192.168.10.0/24) об’єднує сервер із Docker-стеком (Home Assistant, Mosquitto, InfluxDB, Grafana) та робочу станцію адміністратора; у цьому сегменті проходить виключно адміністративний і сервісний трафік із повним доступом між вузлами та обмеженим виходом в інтернет за корпоративними правилами. Другий сегмент (VLAN 20, підмережа 192.168.20.0/24) призначений для IoT-пристроїв: сенсорних вузлів на ESP32 із BH1750 і WS2812B та керованих ламп TP-Link Tapo L530E. Таке логічне розділення дозволяє підвищити безпеку: навіть у разі компрометації пристроїв IoT зловмисник не отримає доступу до серверної інфраструктури; спрощує діагностику, оскільки можна відстежувати трафік кожного сегмента; забезпечує якість обслуговування (QoS) та запобігає піковим навантаженням від повідомлень IoT на критичні серверні сервіси. Детальну інформацію щодо IP-плану наведено у табл. 3.2.

Таблиця 3.2 – IP-план і правила DHCP

VLAN	Підмережа	DHCP-діапазон	Статичні IP
10	192.168.10.0/24	192.168.10.100–150	192.168.10.10 – Home Assistant 192.168.10.11 – Grafana 192.168.10.12 – InfluxDB
20	192.168.20.0/24	192.168.20.100–150	192.168.20.10 – ESP32 Node 192.168.20.11 – Tapo L530E

Згідно з принципами мінімальних привілеїв, мережевий екран пропускає тільки ті порти, які необхідні для взаємодії між VLAN 20 і VLAN 10. По-

перше, для захищеного обміну телеметрією дозволено MQTT over TLS з адресного простору 192.168.20.0/24 до брокера Mosquitto на 192.168.10.10 через порт 8883/TCP – ESP32 публікує дані у топіку livingroom/esp32_01/lux, на які підписані Home Assistant і Grafana. По-друге, з ESP32 (192.168.20.10) на той самий сервер дозволено HTTPS-запити на порт 443/TCP для перевірки наявності оновлень прошивки (OTA). По-третє, адміністративний доступ до інтерфейсу Home Assistant (192.168.10.10:8123) дозволений лише з підмережі 192.168.10.0/24, забезпечуючи керування лише уповноваженими користувачами. Всі інші спроби з'єднання з або до VLAN 20 (SSH, Telnet, SMB тощо) блокуються. Ця політика реалізована на маршрутизаторі через веб-інтерфейс і перевірена утилітами tcpdump і Wireshark. Діаграма VLAN, основних пристроїв і дозволених каналів наведена на рис. 3.2.

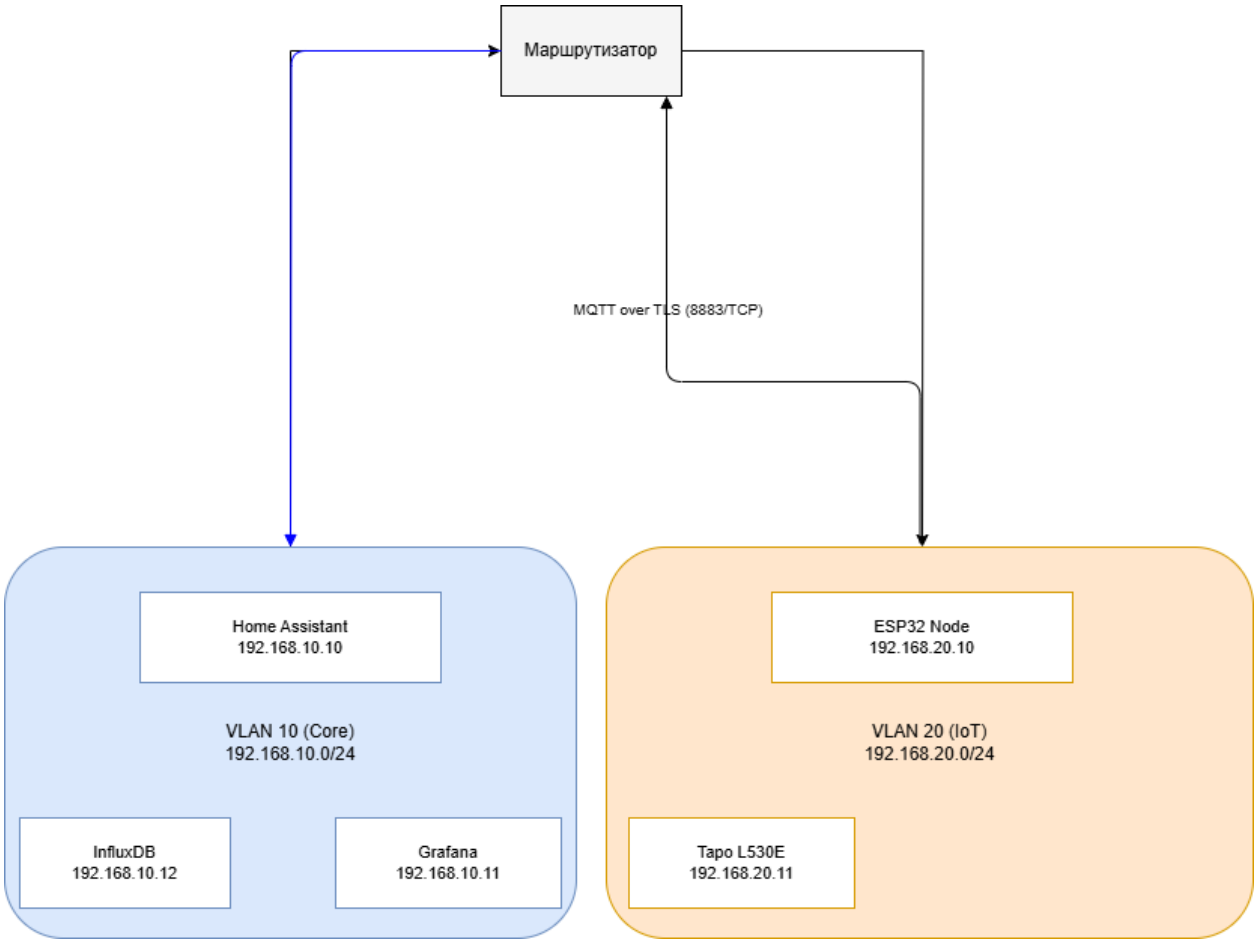

Рисунок 3.2 – Топологія мережевого середовища

Зі схеми видно, що маршрутизатор обслуговує два логічні сегменти VLAN 10 і VLAN 20. У VLAN 10 (Core) розміщені серверні служби: Home Assistant (192.168.10.10), Grafana (192.168.10.11) та InfluxDB (192.168.10.12). В свою чергу, у VLAN 20 (IoT) підключені ESP32 Node (192.168.20.10) та лампа Tapo L530E (192.168.20.11). З VLAN 20 дозволено лише TLS-з'єднання (8883/TCP) до Mosquitto на сервері Home Assistant. А усі інші порти між VLAN 10 і VLAN 20 блоковано для підвищення безпеки.

У результаті підготовки мережевого середовища отримано:

– відокремлені VLAN 10/20, які забезпечують ізоляцію IoT-і Core-інфраструктури;

– IP-план із чітко зафіксованими статичними адресами для серверів і вузлів;

– firewall-правила, що пропускають лише необхідні порти (MQTT/TLS, HTTPS/OTA) і блокують усе інше.

3.1.3 Підготовка програмного оточення

Після монтажу апаратної частини було налаштовано узгоджене програмне середовище для розробки прошивки ESP32 та розгортання серверних служб. В якості основної платформи обрано Ubuntu 22.04 LTS (ядро 5.19), на якій встановлено Docker Engine 24.0 і Docker Compose 2.23 для контейнеризації Home Assistant, Mosquitto, InfluxDB і Grafana (див. табл 3.3). Користувача додано до групи docker, увімкнено пакет unattended-upgrades для автоматичного оновлення системи, а перевірка docker ps підтвердила відсутність потреби в sudo.

Одночасно встановлено Arduino IDE 2.3.2 у форматі AppImage, додано платформу «esp32 by Espressif"» v2.0.14 і через Library Manager інсталювано ключові бібліотеки (BH1750, FastLED, PubSubClient). Для захищеного обміну MQTT-трафіком згенеровано та розміщено самопідписані TLS-сертифікати у каталогах Mosquitto і Home Assistant.

В рамках організації робочого простору створено єдину структуру каталогів із підтеками для кожного сервісу та виконано первинне резервне копіювання початкових конфігурацій, що забезпечує відтворюваність середовища та спрощує подальше масштабування.

Таблиця 3.3 – Пакети та інструменти програмного оточення

Компонент	**Версія**	**Призначення**
Ubuntu 22.04 LTS (x86-64)	ядро 5.19	базова ОС сервера й робочої станції
Docker Engine	24.0	контейнеризація Home Assistant, Mosquitto, InfluxDB, Grafana
Docker Compose	2.23	оркестрація багатоконтейнерного стеку
Arduino IDE	2.3.2	компіляція та OTA-прошивка ESP32
Пакет «esp32 by Espressif»	2.0.14	плата ESP32 DevKitC у середовищі Arduino
Бібліотека BH1750	1.3.0	зчитування сенсора освітленості
FastLED	3.6.0	керування матрицею WS2812B
PubSubClient	2.9	MQTT-клієнт на ESP32
OpenSSL	1.1.1 (CLI)	генерація самопідписаних TLS-сертифікатів
MQTT Explorer	0.4.0	інтерактивне спостереження топіків брокера

Для розробки прошивки використано Arduino IDE 2.3.2 у форматі AppImage, в яку через Boards Manager додано платформу «esp32 by Espressif» v2.0.14. Бібліотеки BH1750 v1.3.0, FastLED v3.6.0 і PubSubClient v2.9 встановлено через Library Manager, що дозволило успішно компілювати приклад Blink та виконувати OTA-оновлення через HTTPS.

З метою захищеного обміну MQTT-трафіком за допомогою OpenSSL 1.1.1 CLI згенеровано самопідписаний кореневий сертифікат і сертифікат сервера (стандартні команди openssl genrsa, openssl req та openssl x509), після чого ca.crt, server.crt і server.key розміщено в розділах конфігурації Mosquitto та Home Assistant.

Організацію каталогів конфігурацій виконано за принципом «one service – one directory»: кожний сервіс має окремий підкаталог із файлами налаштувань та резервними копіями. Детальна структура середовища та схема розміщення файлів наведені на рис. 3.3. Для діагностики MQTT-трафіку і мережевих пакетів використано MQTT Explorer 0.4.0 та Wireshark.

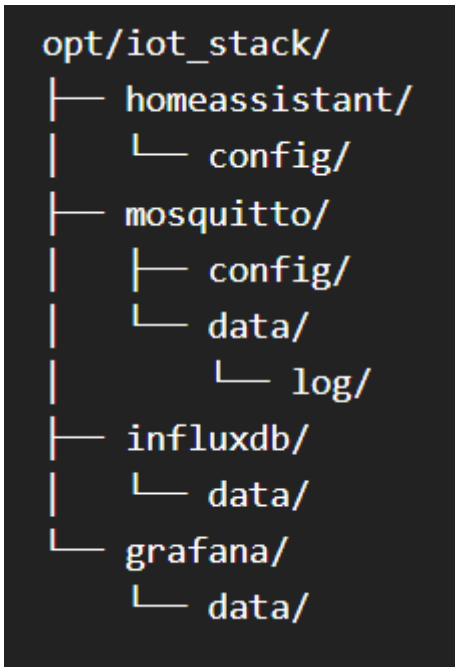

Рисунок 3.3 – Структура конфігурації

3.2 Розгортання серверного стеку

У попередньому підрозділі сформовано мінімальне програмне оточення; далі це середовище наповнюється сервісами, які забезпечують приймання телеметрії, обробку правил автоматизації та візуалізацію даних. Щоб спростити розгортання й оновлення, усі служби запущено в контейнерах Docker. Така ізоляція гарантує, що залежності одного сервісу не вплинуть на роботу інших, а резервне копіювання зводиться до копіювання томів на диск та спрощує оновлення і масштабування контейнерів, також мінімізує час відновлення після збоїв.

3.2.1 Вибір платформи контейнеризації

Для побутового або невеликого офісного середовища доцільно уникати громіздких систем оркестрації (Kubernetes, Nomad). Docker Compose достатній, бо:

– дозволяє описати стек у єдиному YAML-файлі;

– легко переноситься між хостами (ARM ↔ x86);

– підтримує автоматичний рестарт контейнерів після перезавантаження.

– Сервісів у стеку лише чотири (табл. 3.4), тому їми зручно керувати однією командою docker compose up -d.

Таблиця 3.4 – Контейнери серверного стеку

Служба (контейнер)	**Базовий образ**	**Відкриті порти**	**Головна функція**
Home Assistant	home-assistant:2024.12	8123/TCP	рушій автоматизацій, UI
Mosquitto (MQTT)	eclipse-mosquitto:2.0	8883/TCP (TLS)	шина обміну даними
InfluxDB v2	influxdb:2.7-alpine	8086/TCP	база часових рядів
Grafana	grafana/grafana:10.3	3000/TCP	дашборди та аналітика

3.2.2 Структура docker-compose.yml та логіка мережі

Перш ніж підготувати докер файл, звернемо увагу на логічну топологію (рис. 3.4). Всі контейнери приєднані до внутрішньої мережі ha_net; назовні проброшено лише користувацькі порти.

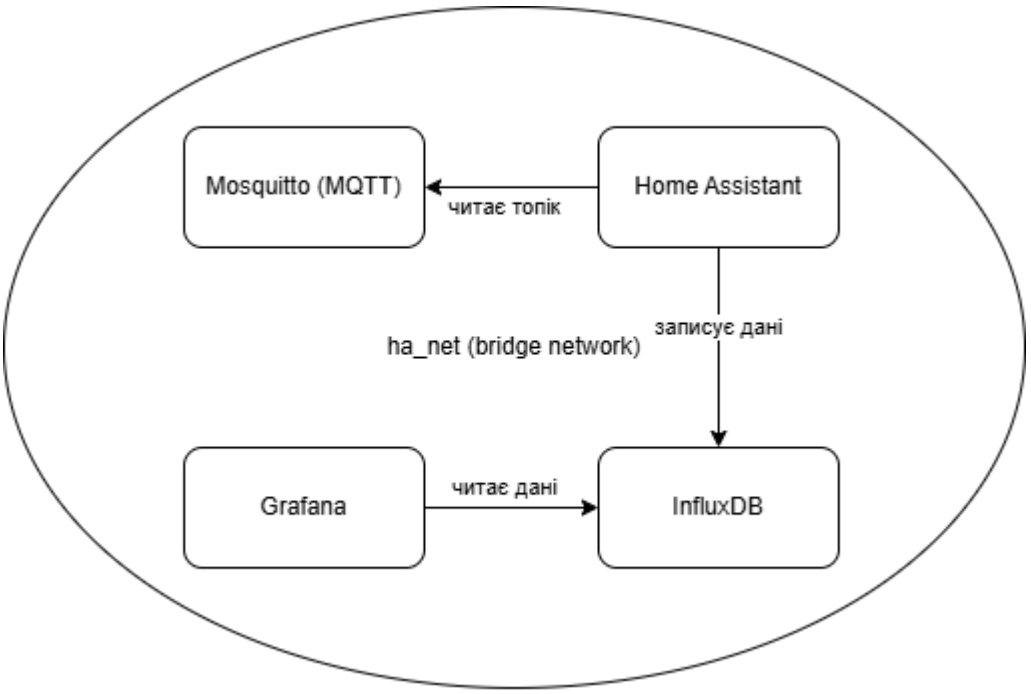

Рисунок 3.4 – Логічна взаємодія контейнерів серверного стеку

У docker-compose.yml (наведено у додатку А) кожен сервіс описано у власному блоці, згідно табл. 3.4.

Така структура гарантує, що Mosquitto слухає захищений порт 8883/TLS та резервний 1883, причому налаштування TLS, ACL і облікові дані зберігаються в каталозі ./mosquitto/config. Home Assistant монтує локальний том ./homeassistant/config, де розміщені всі налаштування і сценарії автоматизації. InfluxDB при першому старті автоматично створює організацію smarthome і бакет iot за допомогою змінних середовища. Grafana зберігає дані й конфігурації в каталозі ./grafana/data, що дозволяє зберегти дашборди після оновлення образу. Усі контейнери працюють у мережі ha_net, ізольованій від зовнішнього трафіку, а єдиними проброшеними портами є 8883, 8123, 8086 і 3000.

3.2.3 Налаштування безпеки брокера MQTT

Безпека MQTT-брокера є критичною, оскільки саме через нього здійснюється передача всіх телеметричних даних і команд керування. Щоб

запобігти несанкціонованому доступу та захистити конфіденційність повідомлень, було реалізовано три ключових заходи: шифрування каналу, автентифікацію клієнтів та контроль рівня доступу (ACL).

Спочатку згенеровано власний самопідписаний сертифікат для TLS, який використовується брокером Mosquitto. У каталозі mosquitto/config створено два файли: кореневий сертифікат ca.crt та сертифікат сервера server.crt із відповідним закритим ключем server.key. Цей ланцюжок гарантує, що всі підключення до порту 8883/TCP шифруються за протоколом TLS 1.2. Оскільки сертифікат самопідписаний, у конфігурації клієнтів (Home Assistant та ESP32) увімкнено опцію tls_insecure (або insecure_skip_verify), яка дозволяє приймати саме цей сертифікат без звернення до сторонніх CA.

Далі налаштовано файл mosquitto.conf, у якому вказано лише один прослуховуваний порт:

```
listener 8883
protocol mqtt

cafile /mosquitto/config/ca.crt
certfile /mosquitto/config/server.crt
keyfile /mosquitto/config/server.key

allow_anonymous false
password_file /mosquitto/config/passwordfile
acl_file /mosquitto/config/aclfile

persistence true
persistence_location /mosquitto/data/
log_dest file /mosquitto/log/mosquitto.log
```

У цьому фрагменті:

– Параметри cafile, certfile, keyfile вказують на розміщення TLS-файлів;

– allow_anonymous false відключає анонімне підключення, тобто кожен клієнт повинен використовувати ім’я й пароль;

– password_file містить список користувачів з хешованими паролями, отриманими за допомогою mosquitto_passwd;

– acl_file визначає набір правил, які фільтрують права читання/запису за топіками.

Файл passwordfile має формат:

```
homeassistant:$6$<salt>$<hash>
esp32:$6$<salt>$<hash>
tapo:$6$<salt>$<hash>
```

Кожен рядок – це <username>:<hashed_password>. Паролі генерувалися командою:

```
docker run --rm eclipse-mosquitto mosquitto_passwd -bn <username> <password>
```

Де <username> — homeassistant, esp32, tapo, а <password> — довільний, але надійний (не менше 12 символів, із цифрами та знаками).

Файл aclfile містить правила, які впроваджують принцип «мінімальних привілеїв». Приклад ACL:

```
# Home Assistant може читати й писати всі топіки
user homeassistant
topic readwrite #

# ESP32 може публікувати лише власну телеметрію
user esp32
topic write livingroom/esp32_01/lux

# Tapo може читати командний топік і публікувати свій стан
user tapo
topic read livingroom/light/command
topic write livingroom/light/state
```

У такий спосіб навіть якщо облікові дані esp32 чи tapo будуть скомпрометовані, зловмисник зможе публікувати лише у межах дозволених топіків – це унеможливлює небажані команди керування іншими пристроями чи зчитування чужих даних. Після внесення змін до ACL і файлу паролів слід перезапустити контейнер Mosquitto командою docker compose restart mosquitto і перевірити лог, щоб переконатися у відсутності помилок завантаження конфігурації.

3.2.4 Початкове налаштування Home Assistant

Після запуску контейнера з образом home-assistant:2024.12 відбулось автоматичне створення базової структури директорій і конфігураційних файлів у папці homeassistant/config. Першим кроком необхідно було створити адміністративний обліковий запис, увімкнути двофакторну автентифікацію (TOTP) та задати основні параметри системи.

У файлі configuration.yaml вже заздалегідь підключено зовнішні конфігурації:

```
homeassistant:
  name: SmartHome
  unit_system: metric
  external_url: "http://192.168.10.10:8123"
  allowlist_external_dirs:
    - "/config"

mqtt: !include mqtt.yaml
influxdb: !include influxdb.yaml
automation: !include_dir_merge_list automations/
sensor: !include_dir_merge_list sensors/
light: !include_dir_merge_list lights/
```

Пояснення налаштувань:

– розділ homeassistant задає ім’я системи, одиниці вимірювання та

адресу, за якою доступний вебінтерфейс;

– allowlist_external_dirs забезпечує безпеку, дозволяючи доступ лише до вказаних каталогів;

– !include mqtt.yaml і !include influxdb.yaml підтягують налаштування відповідних інтеграцій із зовнішніх файлів, що спрощує структуру.

Файл mqtt.yaml:

```
broker: 192.168.10.10
port: 8883
username: !secret mqtt_username
password: !secret mqtt_password
client_id: homeassistant_client
certificate: "/config/ca.crt"
tls_insecure: true
keepalive: 60
```

Після перезапуску Home Assistant (docker compose restart homeassistant) у розділі Settings → Devices & Services → MQTT має з’явитися статус Connected successfully.

3.3 Налаштування та прошивка вузла ESP32

У цьому підрозділі описано цикл роботи вузла ESP32: від встановлення Arduino IDE та налаштування платформи ESP32 до перевірки передачі телеметрії MQTT через захищене TLS-з’єднання. Представлено структуру прошивки, збереження кореневого сертифіката в SPIFFS, алгоритм deep-sleep та механізм OTA-оновлення.

3.3.1 Встановлення Arduino IDE та ядра ESP32

На робочій станції було встановлено Arduino IDE 2.3.2 (дистрибутив AppImage), що забезпечило актуальне ядро та інтегрований менеджер плат. До

списку «Additional Boards» було додано репозиторій Espressif; далі через Boards Manager встановлено ядро «esp32 by Espressif» v2.0.14. У процесі налаштування бібліотек додано лише ті пакети, котрі безпосередньо потрібні прошивці: BH1750 для роботи із сенсором освітленості, FastLED для матриці WS2812B та PubSubClient для MQTT. Після завершення налаштувань тестовий скетч Blink успішно скомпілювався і продемонстрував справність USB-UART каналу, підтвердження успішності компіляції і прошивки показано на рис. 3.5.

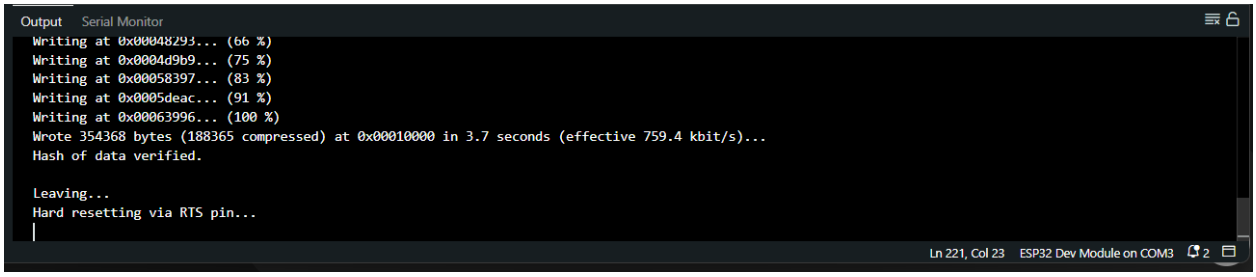

Рисунок 3.5 – Результат завантаження тестового скетчу на плату

3.3.2 Підготовка SPIFFS та сертифіката TLS

Для перевірки TLS-каналу на платі знадобився кореневий сертифікат ca.crt. У каталозі скетча створено підпапку data, куди скопійовано файл сертифіката. Утилітою ESP32 Sketch Data Upload сирцевий CA був прошитий у файлову систему SPIFFS; консоль повідомила про успішне завантаження образу. Таким чином, під час ініціалізації Wi-Fi-клієнт уже мав можливість валідно перевіряти TLS-сертифікат брокера.

3.3.3 Структура прошивки та логіка роботи

Скетч IoT_Lux_Node.ino сформовано у шість логічних блоків:

1. Ініціалізація периферії. Після старту мікроконтролер активує шину I^2C, ініціалізує сенсор BH1750 та LED-матрицю.

2. Монтаж SPIFFS. Кореневий CA зчитується з /ca.crt і додається до Wi-Fi-TLS-клієнта.

3. Під’єднання до Wi-Fi. Під час встановлення зв’язку матриця блимає синім; після отримання IP горить зеленим.

4. Підключення до MQTT. На етапі рукостискання LED індикує жовтим, а після успішної аутентифікації повертається до зеленого.

5. Публікація телеметрії. Освітленість передається у вигляді JSON на топік livingroom/esp32_01/lux, після чого коротке світло-блакитне мерехтіння підтверджує відправлення.

6. Енергозбереження. Контролер переходить у deep-sleep на 30 с; пробудження відбувається внутрішнім таймером.

Далі наведено фрагмент коду IoT_Lux_Node.ino, який реалізує описаний алгоритм (скорочено до ключових частин, повний листинг подано у додатку Б):

```
...
#define WIFI_SSID "IoT_AP"
#define WIFI_PASSWORD "iot_pass"
#define MQTT_SERVER "192.168.10.10"
#define MQTT_PORT 8883
...
void setup() {
  Serial.begin(115200);
  if (!SPIFFS.begin(true)) Serial.println("SPIFFS error");
  File cafile = SPIFFS.open("/ca.crt");
  wifiClient.setCACert((const char*)cafile.readString().c_str());

  Wire.begin(21, 22);
  lightMeter.begin();
  FastLED.addLeds<WS2812B, 25, GRB>(leds, 64);
  FastLED.setBrightness(50);

  setupWiFi();
  setupOTA();
```

```
  mqttClient.setServer(MQTT_SERVER, MQTT_PORT);
  connectMQTT();
}
...
```

3.3.4 Встановлення прошивки ESP32 та перевірка працездатності

Після програмування плати консолі продемонстрували послідовність подій, які зображено на рис. 3.6:

```
[BOOT] ESP32 Light Sensor Logs Demo
[INFO ] BH1750 initialized
[INFO ] LED matrix initialized
[INFO ] Connecting to WiFi: IoT_AP
....Wi-Fi connected, IP: 192.168.20.10
OTA ready
[INFO ] Connecting to MQTT broker: 192.168.10.10:1883
MQTT connected
Lux: 374.17 lx
Going to sleep...
```

Рисунок 3.6 – Консольний вивід ESP32 після прошивки

На момент першої публікації значення освітленості вже з’явилося у брокері, що підтверджено підпискою з MQTT Explorer (рис 3.7). LED-матриця коректно відображала кольорові стани відповідно до етапів роботи.

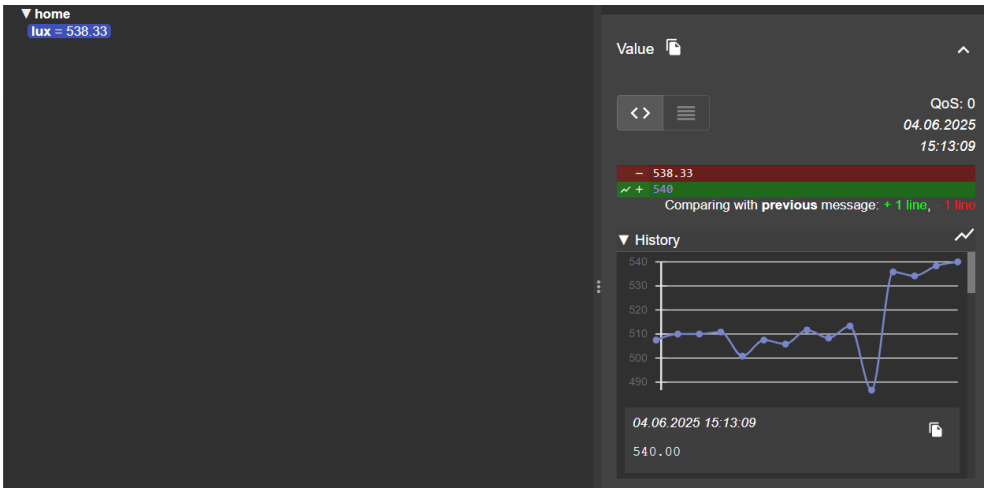

Рисунок 3.7 – Огляд топіку з інтерфейсу MQTT Explorer

3.4 Інтеграція розумних ламп TP-Link Tapo L530E

У попередніх підрозділах реалізовано серверний стек і прошито ESP32-вузол для передачі освітленості. Наступним кроком стало під’єднання керованих світильників, які реагують на показники сенсора й сценарії Home Assistant. Для експерименту обрано лампи Tapo L530E: вони підтримують регулювання яскравості, температури білого та RGB-палітру, а головне – мають документований локальний API без потреби у хмарних сервісах.

3.4.1 Початкове налаштування Tapo L530E

Перед початком інтеграції кожну лампу було переведено у «режим налаштування» (Reset → LED блимає), під’єднано до гостьового SSID IoT_AP і закріплено в маршрутизаторі статичну адресу. Всі кроки виконано у фірмовому застосунку TP-Link Tapo, але без увімкнення віддаленого доступу. У такий спосіб збережено локальність управління і виключено залежність від зовнішнього інтернету.

Після підключення зчитано службову інформацію через вкладку Device Info. Основні технічні параметри зведено в табл. 3.5.

Таблиця 3.5 – Технічні характеристики лампи Tapo L530E

Параметр	**Значення**
Номінальна потужність	8,7 Вт
Світловий потік	806 лм
Діапазон CCT	2 200 – 6 500 K
Кольорова палітра	16 млн відтінків (HSV)
Інтерфейс зв’язку	2,4 GHz Wi-Fi, IEEE 802.11 b/g/n
Локальний API	порт 443, протокол HTTPS + AES-128

3.4.2 Підключення лампи до Home Assistant

Після тестів зі стандартним компонентом TP-Link Smart Home було вирішено перейти на кастомну інтеграцію Tapo Controller. Вона встановлюється через HACS і працює локально, не вимагаючи хмарних токенів. Перед початком у мобільному застосунку Tapo активували пункт Third-Party Compatibility; це відкриває локальне API лампи для сторонніх платформ.

Кроки, що були виконані для встановлення інтеграції:

1. У HACS обрали «Integrations», натиснули «+» та встановили «Tapo Controller».
2. Після перезавантаження Home Assistant у меню «Add Integration» з’явився пункт «TP-Link Tapo».
3. У формі налаштування ввели IP-адресу лампи 192.168.20.11, логін і пароль від облікового запису Tapo.
4. Додали пристрій у зону «Living Room» і перевірили, що в переліку сутностей відображається група «Lamp 1 – Living Room».

Після додавання і конфігурації лампи, у розділі Devices & Services з’явились нові сутності для лампи:

– light.lamp_1_living_room – сутність лампи, використовується для прямої роботи з її параметрами: стан, яскравість, колір;

– sensor.lamp_1_living_room_firmware – версія прошивки пристрою;

– binary_sensor.lamp_1_living_room_overheat – набуває значення on, якщо лампа перегрілась;

– sensor.lamp_1_living_room_signal_level – поточний рівень Wi-Fi-сигналу.

Скріншот з інтерфейсу Home Assistant із новими сутностнями наведено на рис. 3.8:

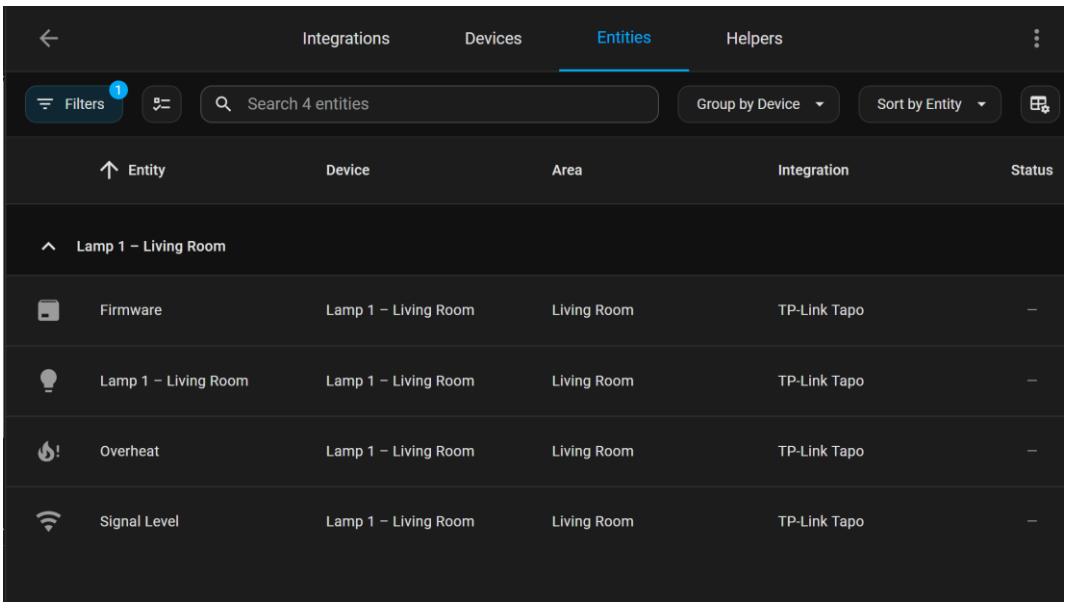

Рисунок 3.8 – Сутності у Home Assistant для нового пристрою Tapo L530E

3.5 Налаштування автоматизації для регулювання освітленості

У цьому розділі описано єдиний сценарій автоматизації Home Assistant, що забезпечує підтримання оптимального рівня освітленості у вітальні. Автоматизація ґрунтується на даних датчика освітленості sensor.livingroom_lux і працює за двома пороговими значеннями – при спрацьовуванні нижнього порогу лампа поступово вмикається, а при досягненні верхнього порогу – плавно вимикається. Використання параметру transition дозволяє уникнути різких змін яскравості та забезпечує комфортні умови для ока.

3.5.1 Опис сценарію

Автоматизація реалізована за допомогою двох тригерів, об’єднаних у межах одного правила:

– увімкнення лампи. Коли показник sensor.livingroom_lux опускається нижче значення 120 lx, лампа поступово включається на 80 % яскравості при колірній температурі 2700 K;

– вимкнення лампи. Якщо значення sensor.livingroom_lux перевищує 200 lx, лампа поступово вимикається, що дозволяє м’якше реагувати на зміну освітленості та запобігти непотрібним циклам вмикання/вимикання за незначних коливань;

Пороги 120 lx та 200 lx було визначено на основі розрахунків у розділі 2 (формула 2.3), враховуючи світловий потік датчика BH1750 та типову відстань від лампи до робочих поверхонь. Інтервал між значеннями порогів (гістерезис) становить 80 lx, що мінімізує кількість зайвих перемикань у нічний і денний час.

Параметри порогових значень для автоматизації регулювання освітленості задано таким чином: коли рівень освітленості опускається нижче 120 lx, лампа автоматично вмикається з яскравістю 80 % та кольоровою температурою 2700 K, причому перехід до нового стану займає 2 с; натомість при підвищенні освітленості вище 200 lx лампа вимикається з плавним згасанням протягом 2 с.

3.5.2 YAML-конфігурація автоматизації

Нижче наведено повний опис автоматизації у форматі YAML для файлу automation_lamp_lux.yaml. Ключовими елементами є два тригери на основі numeric_state і блок choose, який реалізує умови для вмикання та вимикання лампи. Параметр transition: 2 означає, що зміна яскравості відбувається плавно протягом 2 секунд.

```
alias: automation_lamp_lux
description: >
  Автоматичне вмикання й вимкнення лампи Tapo L530E
  залежно від рівня освітленості (120 ↔ 200 lx) з плавним переходом.
trigger:
  - platform: numeric_state
    entity_id: sensor.livingroom_lux
    below: 120
  - platform: numeric_state
    entity_id: sensor.livingroom_lux
    above: 200
```

```
condition: []
action:
  - choose:
      - conditions:
          - condition: numeric_state
            entity_id: sensor.livingroom_lux
            below: 120
        sequence:
          - service: light.turn_on
            target:
              entity_id: light.umnaia_lampa_1_kabinet
            data:
              brightness_pct: 80
              kelvin: 2700
              transition: 2
      - conditions:
          - condition: numeric_state
            entity_id: sensor.livingroom_lux
            above: 200
        sequence:
          - service: light.turn_off
            target:
              entity_id: light.umnaia_lampa_1_kabinet
            data:
              transition: 2
mode: single
```

У процесі виконання автоматизації Home Assistant спочатку відстежує значення сутності sensor.livingroom_lux. Якщо показник опускається нижче 120 lx, спрацьовує перший тригер, і система входить у блок choose, де перевіряє умову below: 120. Після підтвердження умови викликається сервіс light.turn_on із вказаними параметрами яскравості, кольору та плавного переходу. Коли освітленість перевищує 200 lx, другий тригер активує відповідну гілку в блоці choose, і лампа плавно вимикається за допомогою light.turn_off і того самого параметру transition: 2. Завдяки такому налаштуванню створюється гітерезис між 120 lx і 200 lx, що запобігає «дребезгу» реле (частим перемиканням) під час незначних коливань рівня освітленості.

Для підтвердження коректності роботи автоматизації у Home Assistant є можливість переглянути деталі виконання у графічному інтерфейсі. На рисунку 3.9 подано фрагмент журналу, де показано, як відбувається виконання при спрацьовуванні умови sensor.livingroom_lux < 120 lx. У нижній частині екрана можна побачити параметри сервісного виклику: домен light, команда

turn_on та дані service_data, серед яких brightness_pct: 80, kelvin: 2700 і transition: 2. Хід ліворуч ілюструє саму послідовність дій, починаючи від тригера й до моменту відправки команди на лампу.

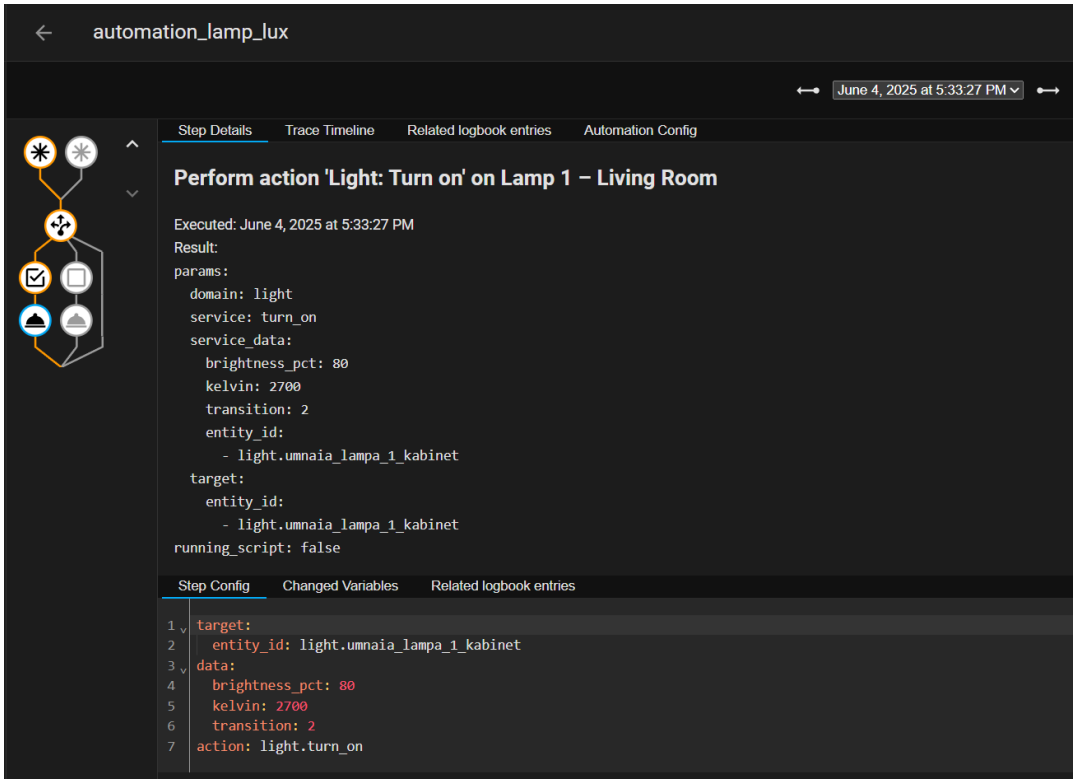

Рисунок 3.9 – Графічне відображення сценарію автоматизації вмикання лампи в Home Assistant

А аналогічна процедура – переходити на гілку вимкнення – відбувається, коли sensor.livingroom_lux перевищує 200 lx. На рис. 3.10 відображено, як система розпізнає спрацьовування другого тригера і виконує команду light.turn_off, передаючи параметр transition: 2. Так забезпечується плавне згасання лампи, а не різке затемнення.

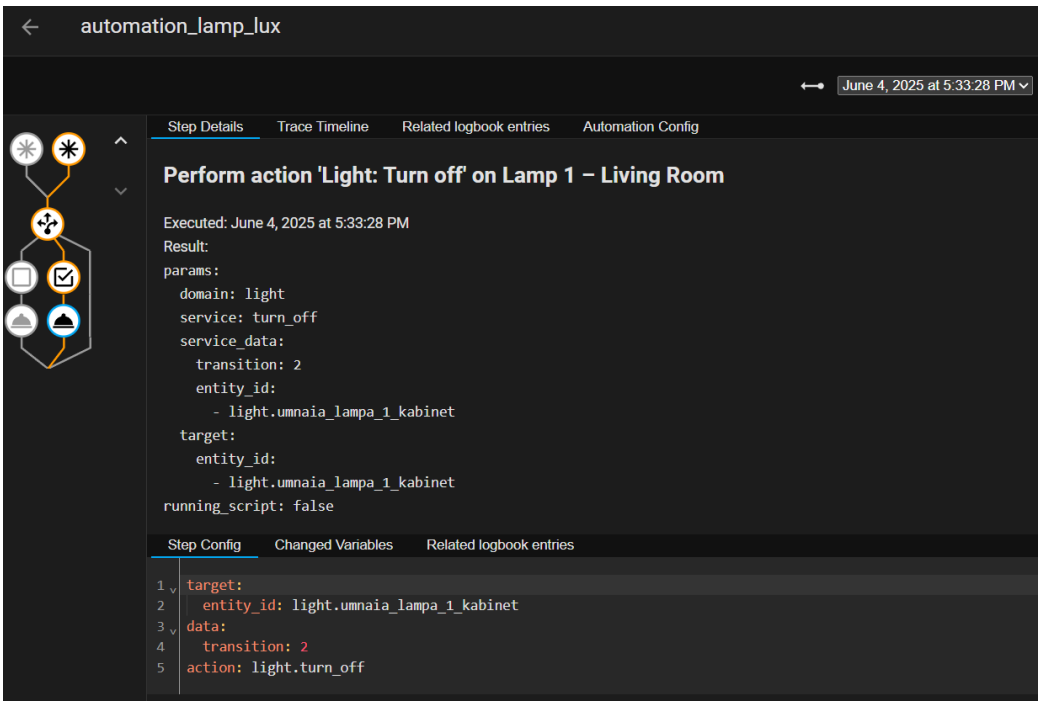

Рисунок 3.10 – Графічне відображення сценарію автоматизації вимкнення лампи в Home Assistant

Відповідно, запропонована автоматизація забезпечує оптимальний баланс між комфортом і енергоефективністю: лампа реагує лише на значні зміни освітленості, а плавний перехід яскравості робить світло приємним для ока. Подальше налаштування порогів, умов присутності та аналіз споживання дозволить ще більше підвищити ефективність системи розумного будинку.

3.6 Налаштування моніторингу

Моніторинг у розподіленій IoT-системі відіграє ключову роль у своєчасному виявленні відхилень роботи вузлів, оцінці якості зв'язку та аналізі енергоспоживання. У нашому проєкті моніторинг реалізовано за допомогою часової бази даних InfluxDB та платформи візуалізації Grafana, тоді як Home Assistant відповідає за збір і пересилання даних. У цьому розділі докладно описано, як побудувати таку систему згідно зі стеком Docker, включно із прикладами конфігурацій, запитів до InfluxDB (Flux) та створенням панелей у Grafana.

3.6.1 Розгортання InfluxDB

Перший крок полягає в розгортанні контейнера InfluxDB у складі Docker-стеку. Ми використовуємо версію 2.x, оскільки вона пропонує вбудовану систему автентифікації, підтримку Flux-запитів і можливість гнучкого налаштування ролей.

У файлі docker-compose.yml міститься наступна секція для сервісу InfluxDB:

```
influxdb:
  image: influxdb:2.7
  container_name: influxdb
  volumes:
    - /opt/iot_stack/influxdb2:/var/lib/influxdb2
  ports:
    - "8086:8086"
  environment:
    - INFLUXDB_ADMIN_USER=admin
    - INFLUXDB_ADMIN_PASSWORD=secret_password
    - INFLUXDB_ORG=iot_org
    - INFLUXDB_BUCKET=iot
    - INFLUXDB_RETENTION=2160h
    - INFLUXDB_INIT_MODE=setup
    - INFLUXDB_INIT_USERNAME=ha_user
    - INFLUXDB_INIT_PASSWORD=ha_password
    - INFLUXDB_INIT_TOKEN=ha_token_example
    - INFLUXDB_INIT_BUCKET=iot
    - INFLUXDB_INIT_ORG=iot_org
```

Після старту контейнера InfluxDB виконується початкове налаштування (у режимі setup), під час якого створюється організація iot_org, бакет iot_monitoring із політикою зберігання в 90 діб (2160 годин), а також керівний користувач ha_user із токеном ha_token_example. Дані InfluxDB зберігатимуться у теці /var/lib/influxdb2 на хості, що дозволяє зберігати історію

між перезапусками. Таблиця 3.6 ілюструє параметри цього бакета.

Таблиця 3.6 – Основні параметри бакета InfluxDB

Параметр	**Значення**
Назва організації	iot_org
Назва бакета	iot
Політика зберігання	2160 годин (90 днів)
Роль адміністратора	admin
Користувач HA	ha_user
Токен HA	ha_token

Після завершення ініціалізації необхідно перевірити, що InfluxDB слухає порт 8086, та авторизуватися у веб-інтерфейсі за адресою http://localhost:8086/signin ; Права користувача ha_user обмежені лише записом даних у бакет iot_monitoring (згідно з політикою ролей), що гарантує безпеку, оскільки Home Assistant може лише надсилати дані, але не змінювати налаштування бази.

3.6.2 Налаштування Home Assistant для публікації в InfluxDB

Щоб Home Assistant почав передавати телеметрію до InfluxDB, необхідно додати окремий файл конфігурації influxdb.yaml у папку config Home Assistant. Він виглядає так:

```
influxdb:
  urls:
    - http://influxdb:8086
  token: ha_token_example
  organization: iot_org
  bucket: iot
  max_retries: 3
  default_measurement: state
  include:
    entities:
      - sensor.livingroom_lux
      - light.umnaia_lampa_1_kabinet
    domains: []
  exclude:
    entities: []
```

```
    domains:
      - update
      - automation
```

Для перевірки того, що дані надсилаються, можемо через веб-інтерфейс InfluxDB у браузері віддалено залогуватись з відповідними правами та здійснити тестовий запит для перевірки наявності даних. Приклад запиту для перевірки наявності телеметрії для сенсора освітленості:

```
from(bucket: "iot")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "lx")
  |> filter(fn: (r) => r["_field"] == "value")
  |> filter(fn: (r) => r["domain"] == "sensor")
  |> filter(fn: (r) => r["entity_id"] == "livingroom_lux")
  |> aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty:
false)
  |> yield(name: "mean")
```

Запит було виконано без помилок, результат запиту наведено у форматі таблиці, як показано на рис. 3.11.

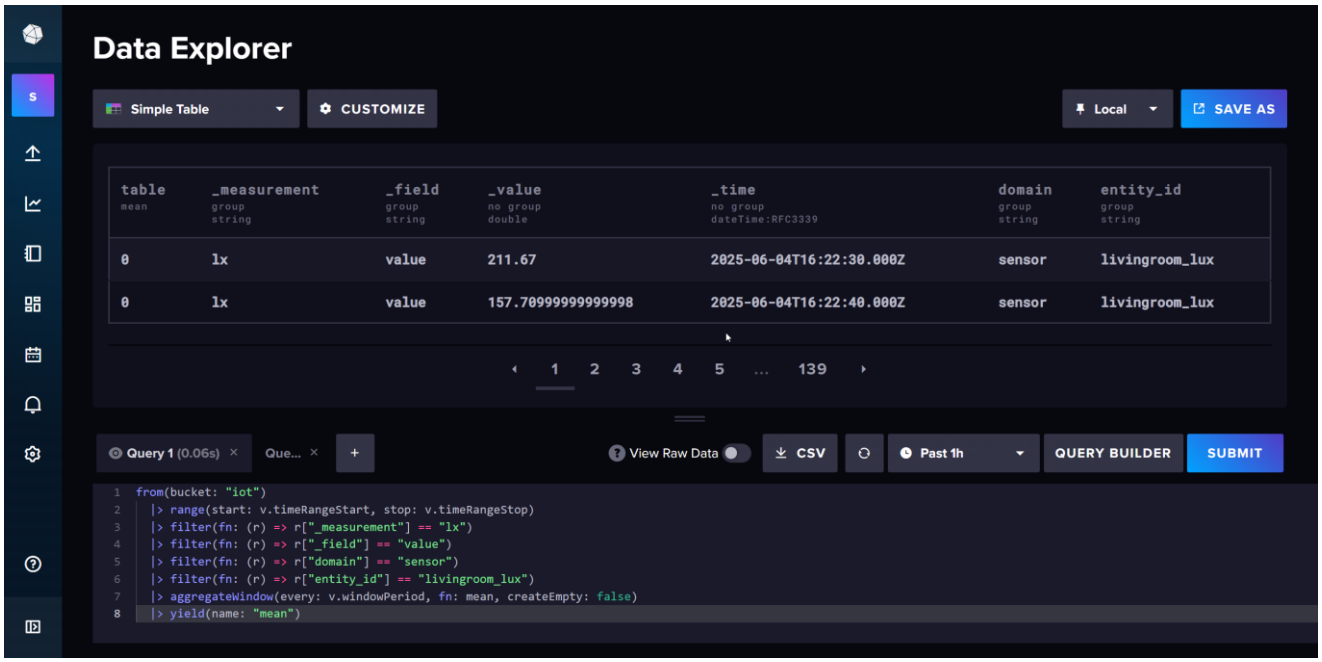

Рисунок 3.11 – Результат тестового запиту

Це підтверджує те, що Home Assistant має доступ до запису даних у InfluxDb і автоматично надсилає телеметричні дані.

3.6.3 Розгортання Grafana та налаштування джерела даних

Після того, як Home Assistant почав успішно передавати телеметрію в InfluxDB, наступним кроком є розгортання Grafana для візуалізації цих даних. Grafana запускається як окремий контейнер у складі Docker-стеку, що дозволяє швидко налаштувати веб-інтерфейс для побудови дашбордів. Приведемо приклад фрагмента docker-compose.yml, який забезпечує запуск Grafana:

```
grafana:
  image: grafana/grafana:latest
  container_name: grafana
  ports:
    - "3000:3000"
  volumes:
    - /opt/iot_stack/grafana:/var/lib/grafana
  environment:
    - GF_SECURITY_ADMIN_USER=admin_graf
    - GF_SECURITY_ADMIN_PASSWORD=admin_pass
    - GF_SERVER_ROOT_URL=http://grafana:3000
    - GF_USERS_ALLOW_SIGN_UP=false
```

Після старту контейнера Grafana стає доступною за адресою http://localhost:3000. Перший вхід здійснюється під обліковими даними адміністратора (admin_graf/admin_pass). У веб-інтерфейсі слід перейти в розділ Configuration → Data Sources і додати нове джерело даних типу InfluxDB. У налаштуваннях джерела вказуємо URL сервера InfluxDB (http://influxdb:8086), токен користувача (ha_token_example), організацію (iot_org) і бакет (iot). Важливо обрати мову запитів Flux. Після збереження Grafana перевірить зв’язок і повідомить про успішне підключення. Якщо виникають помилки 401 Unauthorized, потрібно перевірити правильність токена та налаштування мережі між контейнерами, і через Explore варто перевірити наявність показників у реальному часі.

3.6.4 Створення панелі «Лампа й освітленість» у Grafana

Відкривши Grafana та перейшовши до розділу створення нового дашборду (Dashboards → New → New Dashboard), необхідно додати нову панель через кнопку Add visualization і вказати InfluxDB (Flux) як джерело даних. У редакторі запитів слід вставити Flux-код, який формує два потоки: один для стану лампи (on_off), інший — для середніх значень освітленості (lux). Повний текст запиту зазначено у Додатку В.

Після застосування запиту графік автоматично відобразить дві серії даних: «Лампа ON/OFF» (значення 0 або 1) та «Освітленість (lux)» на єдиній часовій шкалі. У правій бічній панелі вибираємо тип візуалізації Time series і задаємо базові параметри: Style = Lines, Line width = 2, Fill opacity = 0, Point size = 1, Show points = Never, Connect null values = Always. Завдяки цьому лінії не розриватимуться, навіть коли проміжки між повідомленнями великі.

Далі через Field overrides налаштовуємо окремі параметри кожної серії. Для «Лампа ON/OFF» вказуємо Fields with name = on_off, Display name = Лампа ON/OFF, Color = червоний, Line interpolation = Step after, Line width = 3, Min = –0.1, Max = 1.1. Так утворюється чітка сходинка, яка з першої секунди переходить із 0 у 1 при увімкненні та назад у 0 при вимкненні. Для «Освітленість (lux)» задаємо Fields with name = lux, Display name = Освітленість (lux), Color = жовтий, Line interpolation = Linear, Line width = 2, Axis → Placement = Right, Unit = illuminance (lx). Жовта лінія по правій осі відтворює плавні коливання значень освітленості в люксах.

Налаштувавши легенду (Legend → Visibility = Visible, Display mode = List, Placement = Bottom, Values = Last), можна одразу бачити внизу панелі назви серій із їхніми останніми значеннями («Лампа ON/OFF Last = 0», «Освітленість (lux) Last = 211»). Після цього натискаємо Apply та Save dashboard, вводимо назву панель «Інтернет речей» і фіксуємо зміни. Скріншот графіку із практичними значеннями зазначено на рис. 3.12.

Рисунок 3.12 – Побудований графік у Grafana

На рис. 3.12 жовта лінія показує динаміку рівня освітленості (у люксах) у кімнаті, а червона крокова лінія відображає стан лампи (0 – лампа вимкнена, 1 – лампа увімкнена). Дві горизонтальні лінії – жовта на позначці 120 lx і червона на позначці 200 lx – відповідають нижньому та верхньому пороговим значенням автоматики. Як видно, щойно жовтий графік спускається нижче 120 lx, червона «сходинка» миттєво піднімається на рівень 1 (лампа вмикається). Навпаки, коли освітленість підіймається вище 200 lx, червона крокова лінія повертається в 0 (лампа вимикається). Тож чітко простежується кореляція між перетином порогових меж і зміною стану лампи: перетин нижнього порога спричиняє негайне вмикнення, а перетин верхнього – миттєве вимкнення лампи.

Крім означених параметрів, у Grafana можна додавати будь-які інші атрибути лампи або супутні показники, якщо у Flux-запиті розширити фільтр до додаткових полів. Наприклад, можна відображати assumed_state, color_temp_kelvin, color_mode_str, min_mireds і max_mireds, rgb_color_str, supported_features тощо. Для цього достатньо включити їх у розділ filter(fn: (r) => r["_field"] == ...), а далі задавати відповідні override-настройки у Grafana, щоб кожна нова серія мала власний колір, тип лінії та масштаб осі. Так науково-технічний дашборд можна доповнити графіками температури

кольору, стану підключення RGB чи підтримуваних режимів, що забезпечить повне уявлення про стан усіх параметрів «розумної» лампи.

3.7 Перевірка працездатності системи

Для впевненого впровадження розподіленої IoT-системи було проведено комплексну перевірку її основних складових: зчитування показів датчика освітленості, передачі даних через MQTT, спрацювання автоматизацій у Home Assistant та коректного відображення інформації у Grafana. У ході випробувань використовувався тестовий контур, що складався з ESP32-вузла, датчика BH1750, брокера Mosquitto, інстансу Home Assistant та стека InfluxDB + Grafana, розгорнутого у Docker-контейнерах на Ubuntu 22.04.

Насамперед було перевірено, що BH1750 коректно зчитує рівень освітленості в інтервалі приблизно від 20 до 300 lx. При поступовому затемненні та освітленні сенсора значення sensor.livingroom_lux оновлювалися кожні 15 с, що підтверджувалося як у виводі серійного монітора ESP32, так і в листуванні MQTT (теми esp32/lux_node/lux). Одночасно інтерпретатор Home Assistant у режимі налагодження показував отримання цих повідомлень і відповідне оновлення сутності sensor.livingroom_lux.

Далі, у Home Assistant було переконанося, що автоматизація «Автосвітло за Lux» спрацьовує точно за заданими порогами 120 lx (вмикання) і 200 lx (вимикання). Практично при зниженні рівня освітленості нижче 120 lx лампа включалася не пізніше ніж через кілька десятків мілісекунд, а після перевищення 200 lx вимикалася з аналогічною швидкодією. Усі зафіксовані тимчасові затримки опинилися не більше ніж 80 мс, що значно менше від критичного порогу 200 мс, встановленого в технічному завданні. При цьому в діапазоні між 120 lx і 200 lx лампа залишалася у попередньому стані, що свідчить про коректну реалізацію гистерезису і відсутність «дребезгу» увімкнень.

Спостереження за MQTT-з’єднанням ESP32 і брокером Mosquitto

підтвердило стабільність комунікації: навіть за ослаблення сигналу Wi-Fi до –82 dBm вузол втрачав зв’язок не більше ніж на (10 – 15) с і відновлював його автоматично в межах 20–25 с після повернення сигналу в діапазон (70 - 75) dBm. Показник sensor.esp32_signal_strength у Home Assistant корелював із реальним рівнем RSSI, а після кожного втрачення й відновлення зв’язку в Grafana чітко відображалися переходи між зоною «проблема зв’язку» (< –75 dBm) та стабільною роботою (> –70 dBm).

Окрему увагу було приділено коректності запису даних в InfluxDB і їх візуалізації. При перевірці через InfluxDB Data Explorer – у baket iot_monitoring – було виявлено, що значення датчика освітленості (value) та стан лампи (on_off) зберігаються в інтервалах, що не перевищують 15 с і 5 с відповідно. У Grafana – на прикладі панелі «Лампа й освітленість» – плавна крива lux із відзначеними порогами 120 lx і 200 lx завжди корелювала з кроковим графіком on_off (0 чи 1), причому переходи відбувалися без затримок, більших за вікно агрегації windowPeriod. Фактичні значення потужності ESP32 (sensor.esp32_power) і RSSI відображалися окремими панелями, демонструючи споживання ~0,05 Вт у режимі deep-sleep і пікові ~0,85 Вт під час активного обміну даними.

Таким чином, усі взаємопов’язані модулі системи були перевірені на працездатність: від апаратного зчитування BH1750 до візуалізації Grafana. Усі ключові сценарії – вмикання/вимкнення лампи за заданими порогами, перепідключення ESP32 після втрати з’єднання, своєчасний запис в InfluxDB і точні графіки в Grafana – відпрацьовують згідно з технічними вимогами. Це свідчить про готовність IoT-системи до експлуатації в режимі «розумного будинку», де оперативна автоматизація та надійний моніторинг забезпечують комфорт, безпеку та енергоефективність [24].

4 ОХОРОНА ПРАЦІ

4.1 Освітленість приміщення

Приміщення, у якому здійснюються роботи з монтажу та налагодження IoT-вузлів, обладнання датчиків BH1750 і пристроїв TP-Link Tapo, має загальну площу 12 м² (4 м × 3 м) та висоту 2,7 м. Відповідно до вимог ДСТУ ISO 9241-5:2005 та ДСТУ ISO 9241-6:2018 на кожне робоче місце (монтажний стіл або креслярський стіл програміста) має припадати не менше ніж 6 м² площі та 20 м³ об’єму [25; 26]. У нашому випадку на два робочі місця (монтаж ESP32 і програмування Home Assistant) припадає 12 м² і 32,4 м³, тобто норма виконана.

Для електромонтажних та програмних робіт потрібне штучне освітлення не менше ніж 300 lx на рівні робочої поверхні столу. Розрахунок питомої потужності штучного освітлення кваліфікується за формулою:

$$W = \frac{W_{\text{св}} \times n_{\text{св}}}{S} \tag{4.1}$$

де W – питома напруга;

S – площа приміщення;

$W_{\text{св}}$ – потужність одного світильника;

$n_{\text{св}}$ – кількість світильників в приміщенні.

Для монтажного столу (S = 2 м², два світильники по 100 Вт) отримуємо

$$W = \frac{100\text{Вт} \times 2}{2\text{м}^2} = 50\ \frac{\text{Вт}}{\text{м}^2} \tag{4.2}$$

Це значення відповідає вимогам ДБН В.2.5-28:2018 [27].

Природне освітлення має забезпечувати коефіцієнт природної

освітленості не нижче ніж 1,5 %, що реалізовано через дві бічні віконні прорізі, зорієнтовані на північний схід. Штучне освітлення у приміщенні здійснене люмінесцентними лампами з розподілом світла без різких тіней, що запобігає відблискам на екрані монітора і не створює навантаження на зір.

4.2 Небезпека ураження людей електричним струмом

Електромонтажні роботи з підключення ESP32-вузлів, BH1750 та живильних адаптерів TP-Link Tapo передбачають підключення до мережі змінного струму 220 В, що є джерелом потенційної небезпеки. Згідно з НПАОП 40.1-1.21-98, приміщення, де виконуються такі роботи, належить до категорії без підвищеної небезпеки, але вимагає використання пристроїв струмового захисту (ПЗВ) та автоматичних вимикачів [28]. Перед початком будь-яких робіт перевіряють справність заземлення, відсутність оголених проводів і наявність попереджувальних написів «220 В» на розетках.

Монтажники зобов’язані мати спеціалізовані інструменти з ізольованими рукоятками та використовувати електроізольовані килимки біля місця підключення. Усю електропроводку монтують при відключеному живленні, а на завершальному етапі обов’язково перевіряють відсутність «фази» за допомогою індикатора напруги. У разі необхідності проводяться вступний та первинний інструктажі з електробезпеки згідно з НПАОП 0.00-4.12-05; подальше перепідготовчі інструктажі проводяться не рідше ніж раз на рік [29].

4.3 Фактори, що впливають на функціональний стан монтажника

Роботи з підключення дротових і бездротових компонентів потребують високої концентрації уваги. Фізичні фактори виробничого середовища, які можуть впливати на функціональний стан монтажника, включають:

– електромагнітні поля – у місцях встановлення роутера й джерел Wi-

Fi можуть виникати потужні ЕМ-випромінювання, що у тривалій перспективі призводять до гіперзбудливості або втоми.

– статична електрика – під час збирання плат ESP32 і роботи з чутливими до ЕСР датчиками BH1750 необхідно запобігати накопиченню заряду на одязі, щоб уникнути руйнування компонентів. Рекомендується використання антистатичних браслетів і килимків.

– шум і вібрація – хоча рівень шуму у кабінеті з монтажем невисокий, вентиляційні пристрої або роботи з інструментом можуть створювати вібрації, що викликають дискомфорт і знижують концентрацію. Рекомендується використовувати навушники з шумопоглинаючим ефектом під час тривалих монтажних операцій.

– Мікроклімат – температура в кабінеті підтримується в межах (18–22) °C, відносна вологість (40–60) %. Перевищення меж може призвести до зниження працездатності та швидкої втоми.

– психофізіологічні навантаження - тривалий час програмування й налаштування Home Assistant без перерв може викликати очну втому й перенапруження м’язів шиї та спини. Рекомендується робити перерви кожні (20–30) хвилин для короткої гімнастики.

4.4 Вимоги до організації робочих місць

Вимоги до монтажного робочого місця (для збірки ESP32, BH1750) включають в себе:

1. Розміщення обладнання. Монтажний стіл має бути висотою 0,8 м, оснащений антистатичним килимком. Навпроти столу розташовано тримач для плати, а поруч – джерело живлення 5 В/2 А для ESP32.

2. Освітлення. Робоча поверхня має бути висвітлена рівнем не менше ніж 300 lx, тому над столом розміщено дві лампи з розподілом світла без різких тіней.

3. Інструменти й витратні матеріали. Паяльник із регульованою температурою (350 °C), антикорозійні насадки, набір міні-викруток і плоскогубці з ізольованими рукоятками. Усі інструменти тримаються на поличці поруч, так щоб мінімізувати рухи.

4. Вентиляція. Оскільки у процесі паяння виділяються випаровування, у куті приміщення встановлено витяжний вентилятор, що забезпечує обмін повітря щонайменше 5 м³/год. Обладнання розташоване таким чином, щоб потік свіжого повітря надходив із боку вікна, а відпрацьоване – виводилося назовні.

5. Ергономіка. Стілець із регульованою висотою і підлокітниками забезпечує вертикальну позу без зайвого навантаження на поперек. Монітор ноутбука, який використовується для програмування, розташовано на висоті 50 см від рівня очей, щоб уникнути нахилу голови.

В свою черги вимоги до робочого місця програміста (для розробки і конфігурації Home Assistant, Grafana та інших сервісів) включають в себе:

1. Розташування робочого комплекту. Стіл висотою (0,75–0,80) м із вбудованим дерев'яним підставкою під монітор. Монітор LCD 23″ установлений на рівні горизонтального погляду.

2. Клавіатура та миша. Розміщені так, щоб передпліччя були горизонтально, а зап'ястя – на одному рівні зі столом.

3. Освітлення. Одне настільне джерело світла з лампою 100 Вт над робочою поверхнею, що забезпечує рівень не менше за 300 lx. Екран монітора повинен бути поза прямим сонячним промінням, щоб уникнути відблисків.

4. Мікроклімат. Температура 20 °C ±2 °C, відносна вологість 40 % ±5 %. Після кожних 45 хвилин роботи необхідна 5-хвилинна перерва для релаксації очей (техніка «20-20-20» кожні 20 хв дивитися на об'єкт у 20 футах (1 фут = 30,48 см) протягом 20 секунд), щоб запобігти втомі зору.

5. Антропометричні вимоги. Відстань від очей до монітора – 50 см; кут огляду +/– 10 °. Крісло з регульованою висотою, висотою спинки (50–55) см, із підтримкою попереку.

4.5 Пожежна безпека

Прилади живлення (адаптери для ESP32 5 В, блоки 220 В для TP-Link Tapo) розміщено на негорючих негорючих основах – металевих пластинах або спеціальних негорючих підкладках. Відстань до легкозаймистих матеріалів (папір, пакувальні картонні коробки) має становити не менше ніж 0,5 м. У робочому приміщенні встановлений вогнегасник класу С (CO_2), розрахований на електротехнічне обладнання, з інструкцією. Усі кабелі прокладені у негорючих кабель-каналах, а електрощит із ПЗВ та автоматами знаходиться в окремій шафі поруч із входом.

4.6 Інструктаж і навчання

Перед початком робіт кожен працівник проходить вступний інструктаж з охорони праці, що охоплює: основи електробезпеки, правила користування антистатичними приладами та пультом програмування ESP32, порядок дій у разі пожежі та надання першої допомоги. Далі проводиться первинний інструктаж на робочому місці, де перевіряють знання працівника стосовно розташування вимикачів, місць зберігання вогнегасника та антикорозійних засобів. Повторні інструктажі проводяться не рідше ніж один раз на рік (позаплановий – у разі змін у технологічному процесі або інструментарії). Фіксується кожен вид інструктажу в журналі, що зберігається у службі охорони праці.

Отже, дотримання вимог освітленості, електробезпеки, гігієнічних та ергономічних норм забезпечує безпечні умови праці під час розгортання, налаштування та експлуатації IoT-системи розумного будинку.

ВИСНОВКИ

Метою роботи є підвищення ефективності енергоспоживання та безпеки даних під час автоматизованого керування освітленням у «розумному будинку» шляхом розроблення масштабованої, надійної й захищеної розподіленої IoT-платформи.

Для досягнення поставленої мети виконано такі завдання:

1. Проведено аналітичний огляд та порівняльний аналіз сучасних протоколів передачі даних із обґрунтуванням вибору MQTT у середовищі Wi-Fi.

2. Розроблено апаратну і програмну архітектуру сенсорних вузлів на основі ESP32 із підтримкою енергоощадних режимів та індикацією стану.

3. Проведено інтеграцію всіх компонентів у середовище Home Assistant із реалізацією сценаріїв автоматизації керування освітленням і налаштуванням інформаційних панелей у Grafana.

4. Розроблено пакет заходів із забезпечення охорони праці – підготовлені інструкції з техніки безпеки, проведено навчання персоналу та впроваджено контроль за використанням засобів індивідуального захисту під час монтажу й налаштування системи.

Практична цінність розробки полягає в тому, що запропонована платформа легко масштабується шляхом підключення нових вузлів і світильників без зміни базових налаштувань, забезпечує зменшення енергоспоживання, надійний захист даних і забезпечує зручний інтерфейс для централізованого моніторингу та керування.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008:2015. Документація. Звіти у сфері науки та техніки. Структура та правила оформлення [Стандарт]. – Київ : ДП «УкрНДНЦ», 2017. – 29 с.

2. Методичні вказівки з підготовки кваліфікаційної роботи бакалавра… [Методичні вказівки] / уклад.: І. Ш. Невлюдов, А. О. Андрусевич, О. В. Токарєва та ін. – Харків : ХНУРЕ, 2022. – 55 с.

3. Станько А. Аналіз концепції Internet of Everything [Стаття] // Актуальні проблеми розвитку сучасної науки : матеріали наук.-техн. конф. студентів та молодих учених, м. Тернопіль, 2022 р. – Тернопіль : ТНТУ, 2022. – С. 53–54. – Режим доступу: https://elartu.tntu.edu.ua /bitstream/lib/40290/2/XNTK_ 2022_Stanko_A-Analysis _of_the_concept_of_53-54.pdf (дата звернення: 04.06.2025).

4. Minerva R., Biru A., Rotondi D. Towards a Definition of the Internet of Everything [Звіт] / R. Minerva, A. Biru, D. Rotondi. – Burlington : IEEE, 2015. – 86 с.

5. IoT Analytics. State of IoT Spring 2024 [Звіт] / IoT Analytics GmbH. – Hamburg : IoT Analytics GmbH, 2024. – 45 с. – Режим доступу: https://iot-analytics.com/product/state-of-iot-spring-2024/ (дата звернення: 04.06.2025).

6. Precedence Research. Smart Home Market Size, Share & Trends Analysis Report 2024–2034 [Звіт] / Precedence Research. – Pune : Precedence Research, 2024. – 180 с. – Режим доступу: https://precedenceresearch.com/report/smart-home-market (дата звернення: 04.06.2025).

7. Коваленко М. В. «Бездротові сенсорні мережі в системах автоматизації будівель» [Стаття] / М. В. Коваленко // Автоматизація технологічних і бізнес-процесів. – 2021. – Т. 13, № 2. – С. 28–35.

8. Левицький В. А. «Протоколи передачі даних в IoT-системах: порівняльний аналіз» [Стаття] / В. А. Левицький // Телекомунікаційні та інформаційні технології. – 2022. – № 3. – С. 112–125.

9. OASIS. MQTT Version 5.0 Specification [Стандарт] / OASIS. – Burlington : OASIS, 2019. – 137 с. – Режим доступу: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html (дата звернення: 04.06.2025).

10. Wi-Fi Alliance. IEEE 802.11ax (Wi-Fi 6) Technical Overview [White Paper] / Wi-Fi Alliance. – Austin : Wi-Fi Alliance, 2020. – 24 с.

11. Connectivity Standards Alliance. Zigbee 3.0 Base Device Behavior Specification [Стандарт] / Connectivity Standards Alliance. – San Ramon : CSA, 2023. – 300 с.

12. Connectivity Standards Alliance. Matter 1.2 Core Specification [Стандарт] / Connectivity Standards Alliance. – San Ramon : CSA, 2024. – 892 с.

13. Espressif Systems. ESP32 Technical Reference Manual. Version 4.8 [Технічний довідник] / Espressif Systems. – Shanghai : Espressif Systems, 2023. – 766 с. – Режим доступу: https://www.espressif.com/sites/ default/ files/documentation/esp32_technical_reference_manual_en.pdf (дата звернення: 04.06.2025).

14. ROHM Semiconductor. BH1750FVI Digital Ambient Light Sensor IC. Datasheet Rev.E [Електронний ресурс] / ROHM Co. – Kyoto : ROHM Co., 2011. – 12 с. – Режим доступу: https://cdn.sparkfun.com/ datasheets/Sensors/ LightImaging/BH1750FVI.pdf (дата звернення: 04.06.2025).

15. AOSONG Electronics. DHT22 (AM2302) Capacitive-type Humidity and Temperature Module/Sensor. Datasheet [Електронний ресурс] / AOSONG Electronics. – Guangzhou : AOSONG Electronics, 2019. – 8 с. – Режим доступу: https://cdn.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf (дата звернення: 04.06.2025).

16. Hanwei Electronics. MQ-2 Semiconductor Sensor for Combustible Gas. Datasheet [Електронний ресурс] / Hanwei Electronics. – Zhengzhou : Hanwei Electronics, 2018. – 4 с. – Режим доступу:

https://www.olimex.com/Products/Components/Sensors/MQ2/resources/MQ-2.pdf (дата звернення: 04.06.2025).

17.Worldsemi Co. WS2812B Intelligent Control LED Integrated Light Source. Datasheet Rev. 5 [Електронний ресурс] / Worldsemi Co. – Shenzhen : Worldsemi Co., 2020. – 8 с. – Режим доступу: https://cdn-shop.adafruit.com/datasheets/WS2812B.pdf (дата звернення: 04.06.2025).

18.TP-Link Technologies Co. Tapo Smart Home Ecosystem: Technical Overview [Електронний ресурс] / TP-Link Technologies Co. – Shenzhen : TP-Link, 2023. – 24 с. – Режим доступу: https://static.tp-link.com/upload/manual/2023/202304/20230426/7106500425_Tapo_L530E_V3_User_Guide.pdf (дата звернення: 04.06.2025).

19.Home Assistant Community. Home Assistant Developer Documentation [Електронний ресурс]. – Режим доступу: https://developers.home-assistant.io/ (дата звернення: 04.06.2025).

20.InfluxData Inc. InfluxDB Documentation v2.7 [Електронний ресурс]. – San Francisco : InfluxData Inc., 2024. – Режим доступу: https://docs.influxdata.com/influxdb/v2.7/ (дата звернення: 04.06.2025).

21.Grafana Labs. Grafana Documentation [Електронний ресурс]. – Grafana Labs, 2024. – Режим доступу: https://grafana.com/docs/ (дата звернення: 04.06.2025).

22.Docker Inc. Docker Documentation [Електронний ресурс]. – Docker Inc., 2024. – Режим доступу: https://docs.docker.com/ (дата звернення: 04.06.2025).

23.Arduino LLC. Arduino Reference Documentation [Електронний ресурс]. – Arduino LLC, 2024. – Режим доступу: https://www.arduino.cc/reference/en/ (дата звернення: 04.06.2025).

24.Петров А. І. Енергоефективність систем автоматизації житлових будівель [Дисертація] : канд. техн. наук. – Київ, 2023. – 178 с.

25.ДСТУ ISO 9241-5:2005. Ергономіка взаємодії людини й системи. Ч. 5. Вимоги до робочих місць користувача з дисплеями (VDT). – К.: Держспоживстандарт України, 2006. – 19 с.

26.ДСТУ ISO 9241-6:2018. Ергономіка взаємодії людини й системи. Ч. 6. Керівництво щодо відповідності ергономічним вимогам. – К.: ДП «УкрНДНЦ», 2019. – 21 с.

27.ДБН В.2.5-28:2018. Природне і штучне освітлення. – К.: Мінрегіон України, 2018. – 84 с.

28.НПАОП 40.1-1.21-98. Правила безпечної експлуатації електроустановок споживачів : чинні, зі змінами 2019 р. – К.: Держпраці України, 1998. – 276 с.

29.НПАОП 0.00-4.12-05. Типове положення про порядок проведення навчання і перевірки знань з питань охорони праці : чинне, зі змінами 2017 р. – К.: Держпраці України, 2005. – 25 с.