Міністерство освіти і науки України

Луцький національний технічний університет

(повне найменування закладу вищої освіти)

Факультет комп'ютерних та інформаційних технологій

(повне найменування факультету)

Кафедра комп'ютерної інженерії та безпеки

(повне найменування кафедри)

КВАЛІФІКАЦІЙНА РОБОТА ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «МАГІСТР»

ДОСЛІДЖЕННЯ ТА РОЗРОБКА СИСТЕМИ БЕЗПЕКИ ДЛЯ ПРИВАТНОГО БУДИНКУ З МОБІЛЬНИМ ІНТЕРФЕЙСОМ ТА ІОТ-ФУНКЦІОНАЛОМ

RESEARCH AND DEVELOPMENT OF A PRIVATE HOME SECURITY SYSTEM WITH A MOBILE INTERFACE AND IOT FUNCTIONALITY

спеціальність 123 Комп'ютерна інженерія
(шифр і назва спеціальності)
освітня програма «Комп'ютерна інженерія»
(назва освітньої програми)

Виконав: здобувач вищої освіти
групи КІм-21
Якобчук Богдан Анатолійович

_____
(підпис)

Керівник: к.т.н., доцент
Христинець Наталія Анатоліївна

_____
(підпис)

Кваліфікаційну роботу
допущено до захисту
«___» грудня 2025 р.
Гарант освітньої програми:
к.т.н., доцент Гринюк Сергій Васильович

_____
(підпис)

Луцьк – 2025 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *комп'ютерної інженерії та безпеки*
Ступінь вищої освіти: *магістр*
Галузь знань: *12 Інформаційні технології*
Спеціальність: *123 Комп'ютерна інженерія*
Освітня програма: *«Комп'ютерна інженерія»*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ доц. Т.ТЕРЛЕЦЬКИЙ
«_____» _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Якобчуку Богдану Анатолійовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи *Дослідження та розробка системи безпеки для приватного будинку з мобільним інтерфейсом та IoT-функціоналом*

Керівник роботи *к.т.н., доцент Христинець Наталія Анатоліївна*

затверджені наказом закладу вищої освіти від «17» червня 2025 року № 290/01-02
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи 09.12.2025р.

3. Вихідні дані до роботи *Джерелом розробки є науково-технічна література та публікації в періодичних виданнях з даного питання, опубліковані зарубіжні та вітчизняні роботи в даній області, різні інтернет-ресурси технічного спрямування*

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити):
Вступ
Аналіз предметної області систем домашньої автоматизації та постановка завдань дослідження
Теоретичні основи побудови IoT-систем для «розумного дому»
Проєктування системи, реалізація та результати тестування прототипу
Висновки

5. Перелік графічного (ілюстративного) матеріалу:
Структурні схеми архітектури IoT-систем, рівнева модель протоколів взаємодії, узагальнена мережна схема Domovyk
Схеми підключення компонентів системи Domovyk
Скріншоти мобільного інтерфейсу програми
Графіки результатів експериментів
Порівняльні діаграми

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз проблеми за темою роботи та постановка завдань дослідження	*Христинець Н.А., доцент*		
Теоретичне дослідження та практична реалізація	*Христинець Н.А., доцент*		
Практична реалізація об'єкта проектування	*Христинець Н.А., доцент*		
Нормоконтроль	*Багнюк Н.В., доцент*		
Гарант ОП	*Гринюк С.В., доцент*		
Показник запозичень тексту	***___%***		
Академічна доброчесність	*Міскевич О.І., ст.викладач*		

7. Дата видачі завдання 18.06.2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	*Огляд літератури із досліджуваної проблеми*	До 01.08.2025 р.	
2.	*Аналітичний огляд систем безпеки з iot-функціоналом*	До 20.08.2025 р.	
3.	*Дослідження та проєктування системи Domovyk*	До 25.09.2025 р.	
4.	*Реалізація та аналіз роботи розробленої системи*	До 20.10.2025 р.	
5.	*Висновки та пропозиції*	До 25.10.2025 р	
6.	*Формування списку використаних джерел*	До 27.10.2025 р.	
7.	*Формування додатків*	До 30.10.2025 р.	
8.	*Оформлення ілюстративного матеріалу*	До 05.11.2025 р.	
9.	*Представлення остаточного варіанту кваліфікаційної роботи керівникові*	До 11.11.2025 р.	
10.	*Нормоконтроль*	До 29.11.2025 р.	
11.	*Інструментальна перевірка на академічний плагіат*	До 02.12.2025 р.	
12.	*Здача кваліфікаційної роботи та всіх супровідних документів на кафедру*	До 09.12.2025 р.	

Здобувач вищої освіти _____ Якобчук Б. А.
(підпис) (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ Христинець Н. А.
(підпис) (прізвище, ініціали)

АНОТАЦІЯ

Якобчук Б. А. Дослідження та розробка системи безпеки для приватного будинку з мобільним інтерфейсом та IoT-функціоналом. Рукопис.

Кваліфікаційна робота магістра ОП «Комп'ютерна інженерія» спеціальності 123 Комп'ютерна інженерія. Луцький національний технічний університет. Луцьк, 2025.

Кваліфікаційна робота складається з вступу, трьох розділів, висновків, списку використаних джерел, додатків.

Перший розділ присвячено огляду сучасних IoT-систем безпеки, проаналізовано моделі взаємодії пристроїв, протоколи зв'язку та комерційні рішення (Ajax, U-Prox, ATIS/Tuya, Dahua, Hikvision). Визначено вимоги до локальних, автономних і модифікованих систем домашньої автоматизації.

В другому розділі наведено архітектуру системи Domovyk, обґрунтовано вибір апаратних засобів (ESP32, ESP32-CAM, сенсори SHT45, MQ-9, HC-SR501, MC-38) та описано алгоритми роботи: експоненційне згладжування, антидребезг, гістерезис, асинхронний обмін через WebSocket. Подано структуру веб-інтерфейсу та механізми оброблення подій.

Третій розділ представлено реалізацію прототипу та результати тестування. Показано затримку реакції датчиків, точність вимірювань, зменшення хибних спрацювань після калібрування та стабільність роботи в автономному режимі. Виконано порівняння з Ajax Systems, що підтвердило переваги Domovyk у відкритості, можливості кастомізації, локальній автономності та зручності для навчальних і дослідницьких цілей.

Ключові слова: «розумний дім», IoT, система безпеки, ESP32, WebSocket, HTTP/REST, експоненційне згладжування, гістерезис, сенсори, локальна обробка подій.

ANNOTATION

Yakobchuk B. Research and development of a private home security system with a mobile interface and IoT functionality. Manuscript.

Qualifying work of a Master's of EP «Computer Engineering» specialty 123 Computer Engineering. Lutsk National Technical University. Lutsk, 2025.

Qualification work consists of an introduction, three sections, conclusions, a references, three appendices.

The first section is devoted to an overview of modern IoT security systems, analyzing device interaction models, communication protocols, and commercial solutions (Ajax, U-Prox, ATIS/Tuya, Dahua, Hikvision). Requirements for local, autonomous, and modified home automation systems are defined.

In the second section presents the architecture of the Domovyk system, justifies the choice of hardware (ESP32, ESP32-CAM, SHT45, MQ-9, HC-SR501, MC-38 sensors), and describes the operating algorithms: exponential smoothing, anti-jitter, hysteresis, asynchronous exchange via WebSocket. The structure of the web interface and event processing mechanisms are presented.

The third section presents the implementation of the prototype and the results of testing. The reaction delay of sensors, measurement accuracy, reduction of false alarms after calibration, and the stability of autonomous operation are demonstrated. A comparison with Ajax Systems was performed, confirming the advantages of Domovyk in openness, customizability, local autonomy, and suitability for educational and research purposes.

Keywords: smart home, IoT, security system, ESP32, WebSocket, HTTP/REST, exponential smoothing, hysteresis, sensors, local event processing.

ЗМІСТ

ВСТУП

Актуальність теми. Стрімкий розвиток технологій Інтернету речей (IoT) призвів до широкого впровадження розумних систем керування у побуті, безпеці та моніторингу навколишнього середовища. Сучасні рішення охоронної автоматизації здебільшого орієнтовані на комерційний сегмент, є закритими для модифікації та вимагають постійного доступу до хмарних сервісів. Це суттєво обмежує можливості досліджень, експериментів і навчання у сфері комп'ютерної інженерії. Потреба у відкритих, гнучких і локально автономних системах, які дозволяють досліджувати архітектуру IoT, протоколи комунікації, методи оброблення даних та алгоритми прийняття рішень, зумовила необхідність створення прототипу, що може бути використаний як у навчальних, так і у практичних застосуваннях.

Система Domovyk відповідає цим викликам, забезпечуючи можливість повної кастомізації, вивчення алгоритмів взаємодії сенсорів, тестування підсистем безпеки та аналізу ефективності методів фільтрації, згладжування і гістерезису.

Метою роботи є розроблення, реалізація та дослідження локально автономної IoT-системи моніторингу та керування Domovyk на основі мікроконтролерів ESP32, із застосуванням асинхронних методів оброблення даних, протоколів взаємодії і алгоритмів підвищення стабільності роботи.

Об'єкт дослідження – IoT-системи моніторингу та автоматизації для розумного дому.

Предмет дослідження – архітектура, алгоритми та методи взаємодії апаратних та програмних компонентів локально автономної системи Domovyk.

Завдання, які необхідно виконати:

– розглянути сучасні підходи до побудови IoT-систем, структуру комунікаційних протоколів, архітектуру сенсорних мереж та огляд аналогічних рішень у сфері домашньої автоматизації;

– обґрунтувати вибір апаратної та програмної платформи, принципів побудови архітектури Domovyk та методів взаємодії між її підсистемами;

– реалізувати апаратну частину системи на базі ESP32, ESP32-CAM та додаткових сенсорів і виконавчих модулів;

– розробити програмне забезпечення контролера з використанням асинхронних технологій, алгоритмів згладжування, гістерезису та REST/WebSocket-комунікації;

– візуалізувати роботу системи за допомогою вебінтерфейсу локального зберігання у файловій системі LittleFS;

– дослідити точність, затримку реакції, стабільність роботи та вплив калібрування на показники сенсорів;

– запропонувати напрями подальшого удосконалення системи та можливості інтеграції з хмарними сервісами та протоколами MQTT.

Апробація результатів роботи представлені на II Міжнародній науковій конференції «Scientific Horizons of the XXI Century: Multidisciplinary Researches», яка проводилася з 6 травня по 7 травня 2025 року у м. Ужгород [1] та наведені у додатку А.

РОЗДІЛ 1
АНАЛІТИЧНИЙ ОГЛЯД СИСТЕМ БЕЗПЕКИ З ІОТ-ФУНКЦІОНАЛОМ

1.1 Сучасний стан розвитку систем «розумний дім» та охоронних комплексів

1.1.1 Поняття та суть концепції «розумного дому»

У сучасних наукових публікаціях «розумний дім» (Smart Home) розглядається як різновид інтелектуального житлового середовища, у якому взаємодіють IoT-пристрої, сенсори, виконавчі механізми, комунікаційна інфраструктура та програмні сервіси, що забезпечують автоматизоване керування побутовими процесами й підсистемами безпеки житла [2, 3].

При цьому наголошується, що ключовою ознакою таких систем є не лише наявність підключених пристроїв, а саме інтегрована здатність збирати дані, аналізувати їх та ініціювати керуючі дії без постійного втручання користувача.

Огляди останніх років показують, що єдиного загальноприйнятого визначення поняття «розумний дім» не існує: у мета-дослідженні [2] виявлено сотні різних формулювань, які по-різному акцентують увагу на технічних і соціальних аспектах цієї концепції.

Частина авторів підкреслює саме автоматизацію та комфорт користувача такі, як керування освітленням, мікрокліматом, побутовими приладами, мультимедійними системами, інші – інтеграцію великої кількості сенсорів і виконавчих пристроїв у єдину IoT-інфраструктуру з підтримкою різних протоколів зв’язку [3].

Окремий напрям досліджень зосереджується на тому, що «розумний дім» виступає платформою для сервісів безпеки, енергоефективності та моніторингу стану мешканців, де провідну роль відіграють алгоритми обробки даних, штучний інтелект і аналітика подій у реальному часі [2, 4, 5].

У низці робіт система «розумного дому» трактується як IoT-орієнтоване рішення, що поєднує локальні пристрої, домашній шлюз або контролер та хмарну платформу, через яку реалізуються сервіси автоматизації, мобільний доступ, інтеграція з голосовими асистентами й аналітичними модулями [2, 4].

Водночас відзначаються проблеми гетерогенності: різні виробники застосовують відмінні апаратні платформи, протоколи та моделі безпеки, що ускладнює побудову єдиного керованого середовища [3, 4].

У контексті безпеки житла сучасні оглядові праці розглядають «розумний дім» як багатокомпонентну систему, де взаємодіють хмарні платформи, кінцеві пристрої та канали зв’язку, а вразливості на будь-якому з цих рівнів безпосередньо впливають на приватність, фізичну безпеку й цілісність майна мешканців [4].

Наголошується, що зростання ролі хмарних сервісів посилює вимоги до криптографічного захисту, контролю доступу та надійної автентифікації, а також стимулює дослідження гібридних архітектур із частковим перенесенням обробки даних на локальний рівень (edge/fog-обчислення) для зниження затримок і підвищення автономності систем безпеки [2, 4].

Окремий напрям сучасних досліджень пов’язаний із використанням методів машинного навчання у «розумному домі». Такі роботи демонструють можливість прогнозування відмов обладнання, виявлення аномальної активності, адаптивного налаштування режимів роботи побутових приладів та підсистем безпеки на основі накопичених даних, що переводить smart-home-середовище з рівня реактивної автоматизації до проактивного управління [2, 5].

Узагальнюючи наведені підходи, у межах даної роботи під «розумним домом» розумітимемо інтегровану IoT-орієнтовану систему керування приватним житлом, яка об’єднує мережу сенсорів, виконавчих пристроїв, локальні контролери та хмарні сервіси в єдине інтелектуальне середовище. Відмінною рисою такої системи є наявність програмно-апаратної логіки, здатної автоматично реагувати на зміни стану середовища й дії користувача, а також реалізовувати функції моніторингу, безпеки, комфорту й енергоефективності з можливістю подальшого розширення та кастомізації. Саме в цьому контексті інтегрована підсистема безпеки «розумного дому» розглядається як ключовий об’єкт подальших досліджень і розробки.

1.1.2 Структура сучасних систем «розумного дому»

Сучасні системи «розумного дому» будуються за модульним принципом, що забезпечує гнучкість і масштабованість. Типова архітектура включає контролери,

датчики, виконавчі пристрої, серверну частину та мобільний інтерфейс користувача. В основі роботи такої системи лежить взаємодія між фізичними компонентами та програмним середовищем, яке забезпечує керування процесами і моніторинг стану об’єктів. Кожен модуль виконує окрему функцію, але водночас взаємодіє з іншими елементами через єдину мережеву інфраструктуру. Така архітектура спрощує модернізацію системи, даючи змогу додавати нові пристрої або змінювати конфігурацію без потреби у повному перепроєктуванні. Завдяки цьому користувач отримує адаптивне середовище, яке може розвиватися відповідно до його потреб і технологічного прогресу.

Загальна структура взаємодії між основними модулями системи наведена на рисунку 1.1, де відображено принцип зв’язку між контролером, датчиками, виконавчими елементами, серверною частиною та мобільним інтерфейсом. Такий підхід дозволяє системі працювати як єдиний узгоджений комплекс, що забезпечує високий рівень гнучкості, надійності та масштабованості.

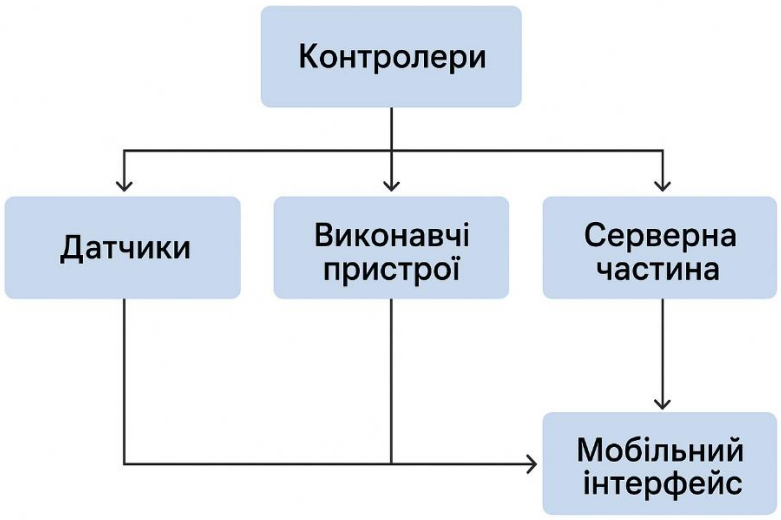

Рисунок 1.1 – Типова архітектура системи «розумного дому»

У сучасних дослідженнях контролер трактується не просто як «центральний модуль», а як розподілений інтелектуальний вузол, що може виконувати частину обробки локально (edge-processing). Підходи з edge-обробкою розглядаються як

критично важливі для систем безпеки, оскільки вони мінімізують затримку сигналів та забезпечують працездатність навіть при втраті доступу до хмари [3, 5].

У працях [2, 4] описані моделі, у яких контролери (ESP32, Raspberry Pi, ARM-платформи) виконують первинну фільтрацію даних, генерацію локальних подій та обробку критичних безпекових сценаріїв без участі хмари.

Саме цей підхід частково використано у даній магістерській роботі: локальний контролер виконує обробку сенсорних даних, приймає оперативні рішення (газ, рух, температура) та передає тільки агреговані події до веб-інтерфейсу.

Датчики виконують функцію збору інформації про навколишнє середовище. Вони можуть фіксувати температуру, вологість, рух, наявність диму, витік газу або відкриття дверей. Виконавчі пристрої реалізують керуючі дії. Це можуть бути реле, електричні замки, світильники або системи вентиляції. Взаємодія між усіма компонентами здійснюється через локальні або хмарні сервіси, які об’єднують пристрої в єдину мережу.

Серверна частина системи виконує роль обчислювального ядра, що зберігає конфігураційні дані, сценарії дій та історію подій. У хмарних рішеннях ця функція реалізується на віддалених серверах, що дає змогу отримувати доступ до системи з будь-якого місця через Інтернет. У локальних реалізаціях сервер може бути розгорнутий на домашньому комп’ютері або спеціалізованому контролері.

Мобільний інтерфейс забезпечує користувачу зручний доступ до всіх функцій системи. Через спеціальні додатки або вебінтерфейси власник може переглядати стан пристроїв, отримувати повідомлення про події і змінювати параметри роботи системи. Залежно від налаштувань система може працювати в автоматичному, ручному або комбінованому режимах. Автоматичні сценарії ґрунтуються на попередньо заданих умовах, ручний режим дозволяє користувачу безпосередньо керувати пристроями, а комбінований поєднує обидва підходи для досягнення максимальної гнучкості.

1.1.3 Місце систем безпеки у структурі «розумного дому»

У сучасних системах «розумного дому» підсистема безпеки посідає центральне місце та визначає архітектурні вимоги до всієї інфраструктури. У

дослідженнях останніх років підкреслюється, що саме компоненти безпеки формують вимоги до топології сенсорної мережі, моделі обробки даних та рівня автономності контролерів. Зокрема, у роботі [4] зазначено, що підсистема безпеки є ядром взаємодії між сенсорними вузлами, локальним контролером і сервісним рівнем, оскільки на відміну від побутових модулів автоматизації, вона потребує гарантовано низької затримки та стабільної реакції на події. Позицію про ключову роль безпеки підтверджує і загальний огляд сучасних смарт-екосистем, проведений авторами в роботі [2], де підкреслено, що саме модулі моніторингу та реагування задають фундаментальні вимоги до розподілу функцій між хмарною та локальною обробкою.

У «розумному домі» підсистема безпеки не обмежується виявленням небезпечних ситуацій, а інтегрується з іншими модулями для реалізації комплексних сценаріїв. У роботі [3] наведено приклади інтегрованих рішень, де сигнали від сенсорів руху, газу та температури обробляються спільно, а реакція системи формується з урахуванням контексту. Здатність підсистеми безпеки синхронізувати свою роботу з модулями клімат-контролю, енергоспоживання або освітлення забезпечує формування узгодженої поведінки, у якій реакція на небезпеку може включати блокування живлення, керування вентиляцією, автоматичне освітлення та інші механізми мінімізації ризиків. У контексті відеоспостереження такі інтеграційні можливості доповнюються аналітичними функціями, що дозволяє здійснювати верифікацію подій у реальному часі.

Суттєвою тенденцією є перехід від подієвих моделей сигналізації до сценарних і аналітико-прогностичних моделей. У межах подієвого підходу рішення приймаються безпосередньо після отримання сигналу від сенсора, що ускладнює відсів хибних тривог і обмежує функціональність системи. Роботи сучасників [3-4] демонструють, що сценарні моделі, побудовані на комбінації кількох джерел даних, значно підвищують точність і надійність. Найбільш перспективним напрямом сучасних досліджень є моделі, у яких система не лише реагує, але й прогнозує розвиток подій. Підхід до прогнозування відмов та небезпек у середовищі «розумного дому», запропонований авторами роботи [5], ілюструє

можливість імплементації алгоритмів машинного навчання для підвищення здатності системи передбачати аномалії на основі історичних даних.

Важливою характеристикою сучасних smart-home систем є тісний взаємозв'язок між безпекою та іншими підсистемами, що підтверджують як загальні огляди [2], так і практичні роботи з моделювання IoT-архітектур [3]. Взаємодія з кліматичними, енергетичними та мультимедійними модулями дозволяє формувати гнучкі сценарії реагування, а взаємозв'язок із серверною частиною забезпечує зберігання історії подій, аналітику та агрегування даних. Одночасно у роботах [4] і [5] підкреслюється, що хмарна обробка не може бути єдиною моделлю для безпекових систем через затримки й можливі ризики недоступності сервісу, тому локальні рішення є основою для реалізації високонадійних сценаріїв безпеки.

Аналіз сучасних підходів дозволяє зробити висновок, що підсистема безпеки у структурі «розумного дому» формується як інтегрований аналітичний модуль, який поєднує збір подій, локальну обробку, прогнозування та синхронізацію з усіма іншими компонентами системи. Саме така модель закладається в основу дослідження у цій роботі, де акцент зроблено на побудові локально орієнтованої підсистеми безпеки з можливістю прогнозування подій та мінімальною залежністю від зовнішніх сервісів.

1.1.4 Сучасні тенденції розвитку

Сучасний етап розвитку систем «розумного дому» характеризується переходом від простих сценаріїв автоматизації до інтелектуальних екосистем, у яких безпека, енергоменеджмент та конфіденційність користувача розглядаються як рівнозначні цілі. У новітніх оглядах відзначається зростання ролі аналітики даних, машинного навчання та локальної обробки подій, що дозволяє системам не лише реагувати на зміни середовища, а й прогнозувати потенційні загрози та відмови обладнання [4, 5, 6]. Такий зсув від «керування за подією» до «керування на основі прогнозу» є ключовою тенденцією останніх років.

Одним з головних векторів розвитку є посилення вимог до кібербезпеки та стійкості до зловмисного програмного забезпечення. Систематичні огляди сучасних загроз для розумних будинків показують, що зростає кількість атак, спрямованих на маршрутизатори, хаби та окремі IoT-пристрої, а класичні заходи

захисту, орієнтовані лише на периметр мережі, вже не є достатніми [4-6]. У роботі O. Alshamsi та співавторів проаналізовано спектр методів виявлення шкідливого ПЗ у розумних будинках та зроблено висновок, що перспективними є підходи, які поєднують сигнатурний аналіз із поведінковими моделями та профілюванням трафіку [6]. Це безпосередньо корелює з результатами досліджень, де пропонуються методики прогнозування відмов і аномалій на основі машинного навчання для компонентів «розумного дому» [5].

Важливим напрямом розвитку стають енергоефективність та інтеграція систем безпеки в загальну стратегію керування енергоспоживанням будинку. Огляд [7] демонструє, що IoT-рішення для енергоменеджменту все частіше використовують дані з охоронних сенсорів (присутність, відкриття дверей, режим «відсутній») для оптимізації роботи освітлення, опалення та вентиляції. Таким чином, підсистема безпеки перестає бути ізольованою і стає джерелом даних для енергозберігаючих сценаріїв, що особливо актуально в умовах підвищених тарифів і нестабільного енергопостачання.

Паралельно з питаннями безпеки посилюється увага до конфіденційності користувачів. У скопінг-огляді [8] показано, що користувачі все більше занепокоєні масштабом збирання персональних даних, профілюванням поведінки та непрозорістю політик їх використання. Автори підкреслюють, що сучасні системи безпеки в розумних будинках мають не лише захищати від фізичних загроз, а й мінімізувати цифрові ризики, починаючи від несанкціонованого доступу до відеопотоків до витоку історії подій охоронної системи [2, 4, 8]. Це зумовлює тренд на локальну обробку критичних даних (відео, журнали тривог, біометрія) з мінімальним виносом у хмару, а також на впровадження принципу privacy-by-design у архітектуру систем.

Ще однією визначальною тенденцією є поява універсальних стандартів сумісності, які покликані подолати фрагментацію ринку. Одним з найбільш показових прикладів є стандарт Matter, орієнтований на уніфікацію протоколів взаємодії між пристроями різних виробників і забезпечення локального керування як обов’язкової опції . У роботі [9] Matter розглядається як ключовий крок до побудови інтероперабельних смарт-домів: автори показують, що підтримка

IP-орієнтованих стеків, єдиного моделювання пристроїв та спрощених механізмів конфігурації суттєво знижує поріг входу для розробників і користувачів [9]. Для підсистем безпеки це означає можливість створювати гібридні рішення, у яких власні контролери, сенсори та актуатори можуть коректно взаємодіяти з комерційними хабами й сервісами без жорсткої прив'язки до однієї екосистеми.

Окремо виділяється тренд на використання периферійних обчислень (edge computing) та «edge AI» у розумних будинках. Згідно з оглядом [10], перенесення частини обробки з хмари на край мережі дозволяє суттєво зменшити затримки, підвищити доступність сервісів при нестабільному Інтернет-з'єднанні та локалізувати частину механізмів захисту. Це відкриває можливість реалізувати локальні моделі машинного навчання, які аналізують потоки даних з датчиків без постійної залежності від зовнішньої інфраструктури. Для систем безпеки приватного житла це особливо важливо, оскільки критичні рішення – від виявлення проникнення до активації сирени чи блокування замків – мають прийматися незалежно від доступності хмарного сервісу [4, 5, 10].

Таким чином, сучасні тенденції розвитку систем «розумного дому» поєднують кілька взаємопов'язаних векторів: посилення кіберзахисту і конфіденційності, перехід до інтероперабельних стандартів, інтеграцію енергоменеджменту та використання периферійних обчислень і методів машинного навчання. На цьому фоні особливої актуальності набувають рішення, які поєднують локальну обробку даних, інтелектуальні алгоритми аналізу подій і можливість гнучкої інтеграції з комерційними платформами. Саме до такого класу належить розроблювана в даній кваліфікаційній роботі система безпеки приватного будинку з веб-інтерфейсом та IoT-функціоналом, орієнтована на локальну аналітику, прогнозування подій і мінімізацію залежності від хмарних сервісів.

1.1.5 Проблеми та виклики сучасного етапу

Сучасні системи «розумного дому» стикаються з низкою проблем, які ускладнюють їх подальший розвиток і впровадження в приватному житлі. Однією з ключових є зростання кіберзагроз. У роботі [4] наголошується, що кількість атак на смарт-пристрої щороку збільшується, а зловмисники використовують складніші методи прихованого втручання у роботу сенсорів, маршрутизаторів та хабів.

Типові підходи до кіберзахисту, засновані на сигнатурних методах, виявилися недостатніми для сучасних атак, що здатні імітувати активність легітимних пристроїв та адаптуватися до середовища [6]. Це створює ризики для приватних користувачів, особливо з огляду на широке використання хмарних сервісів, де критично важлива інформація передається через зовнішні канали.

Суттєвою проблемою є питання конфіденційності. Як показано в роботі [8], сучасні системи «розумного дому» збирають значний обсяг персональних даних, таких як відеопотоки, журнали подій, поведінкові патерни та показники присутності. Централізація цієї інформації у хмарних сховищах створює ризики витоку, несанкціонованого доступу або аналізу даних третіми сторонами. Унаслідок цього виникає потреба у переході до архітектур, де обробка критичної інформації виконується локально, а зовнішні сервіси використовуються лише для другорядних функцій.

Викликом залишається й фрагментація протоколів взаємодії між пристроями. У роботі [9] відзначено, що навіть поява універсальних стандартів на кшталт Matter не повністю усуває проблему несумісності, оскільки значна кількість виробників продовжує застосовувати пропрієтарні рішення. Це ускладнює побудову цілісної інфраструктури, особливо коли йдеться про критичні підсистеми безпеки, що потребують стабільного та узгодженого функціонування всіх компонентів у режимі реального часу.

Ще одним суттєвим викликом є обмеження традиційних хмарних моделей. Як показано в дослідженні [10], затримки, нестабільність Інтернету або недоступність сервісу можуть призвести до несвоєчасного реагування системи на небезпечні події. У контексті приватної безпеки це є критичним, оскільки реакція має бути миттєвою навіть за умов повної відсутності підключення до мережі. Саме тому сучасні рішення дедалі частіше використовують парадигму edge computing, що дозволяє перенести обробку та аналіз подій безпосередньо на локальний контролер.

У роботах [2] та [5] також звертається увага на проблему хибних спрацювань, що особливо характерно для систем, які використовують подієві моделі реагування. Відсутність контекстного аналізу та недостатнє використання

історичних даних призводять до помилкових тривог, що знижує довіру користувача до системи та вимагає впровадження моделей, здатних адаптуватися до середовища і враховувати багатоджерельні сигнали. Прогностичні методи та аналіз поведінкових аномалій, запропоновані у роботі [5], демонструють можливість зниження кількості хибних спрацювань за рахунок локальних моделей машинного навчання.

Комплексність зазначених викликів формує потребу у нових підходах до побудови систем безпеки для приватного житла. В умовах зростаючих загроз, високого рівня фрагментації ринку та недосконалості хмарних технологій виникає практична необхідність у створенні рішень, здатних працювати автономно, адаптивно та з мінімальною залежністю від зовнішньої інфраструктури. Саме на цьому базується формаулювання проблеми дослідження в межах даної кваліфікаційної роботи.

Проблема полягає у розробленні моделі локальної підсистеми безпеки «розумного дому», яка поєднує багатоджерельний збір даних, локальну обробку подій та можливості прогнозування аномалій за відсутності стабільного доступу до хмарних сервісів. Наукова новизна такої моделі полягає у використанні принципів edge computing та адаптивної логіки для забезпечення гарантованої швидкості реакції, надійності та стійкості до зовнішніх загроз, що не може бути досягнуто за допомогою традиційних централізованих архітектур.

1.2 Принципи побудови IoT-систем та їх архітектури

Інтернет речей або IoT (Internet of Things) у сучасних оглядових роботах визначається як сукупність взаємопов’язаних фізичних об’єктів, оснащених сенсорами, засобами обробки даних та комунікаційними модулями, які здатні обмінюватися інформацією між собою та з користувачем через мережу Інтернет без постійного втручання людини [11, 12]. До таких об’єктів належать датчики, побутові прилади, контролери, транспортні засоби, виробниче обладнання та елементи міської інфраструктури. Основна ідея IoT полягає у створенні системи пристроїв, здатних автоматично збирати, передавати й аналізувати дані, що

забезпечує швидку реакцію на зміни середовища, підтримку прийняття рішень у реальному часі та виконання заданих дій [11, 13]. Завдяки цьому пристрої перестають бути пасивними елементами й перетворюються на частину єдиного інформаційного простору, де кожен вузол може взаємодіяти з іншими пристроями або центральним сервером. Прикладом є сценарій, у якому датчик температури передає дані до контролера опалення, що автоматично регулює рівень тепла у приміщенні та забезпечує комфорт і раціональне використання енергії [7, 12].

Технологія IoT активно застосовується у розумних будинках, промислових системах, медицині, транспорті, енергетиці та сільському господарстві [11, 13]. У цих галузях IoT використовується для моніторингу стану об’єктів, оптимізації ресурсів, впровадження предиктивного обслуговування та підвищення безпеки. У роботах останніх років окремо наголошується, що саме інтеграція сенсорних мереж, аналітики даних та хмарних або периферійних сервісів є ключовою передумовою для побудови масштабованих та керованих IoT-рішень [11-13]. При цьому для розумних будинків особливе значення має можливість локальної обробки подій і підтримка режимів роботи в умовах обмеженого або нестабільного доступу до Інтернету [3, 10].

Будь-яка IoT-система ґрунтується на низці базових принципів, які визначають її ефективність і стабільність роботи. У більшості сучасних оглядових досліджень до ключових належать підключення, масштабованість, безпека та енергоефективність [11, 12]. Підключення визначає, яким чином пристрої обмінюються даними між собою та з центральними сервісами. На практиці використовуються як дротові інтерфейси (Ethernet та інші промислові шини), так і бездротові технології, зокрема Wi-Fi, Bluetooth Low Energy, Zigbee, LoRaWAN та інші протоколи ближнього й дальнього радіозв’язку [12-14]. Огляд існуючих платформ показує, що вибір каналу зв’язку завжди є компромісом між швидкістю передавання даних, дальністю дії, енергоспоживанням і вартістю інфраструктури [15]. У системах типу «розумний дім» це особливо помітно: високошвидкісний Wi-Fi потрібен для відеопотоків і вебінтерфейсів, тоді як малопотужні датчики можуть працювати на низькошвидкісних протоколах з мінімальним енергоспоживанням [3, 7].

Масштабованість у контексті IoT означає здатність системи стабільно працювати при зростанні кількості вузлів, обсягу трафіку та складності сценаріїв обробки даних. У сучасних оглядових роботах підкреслюється, що класичні трирівневі архітектури не завжди достатні для масових інсталяцій, тому широко застосовуються розширені багаторівневі моделі з підтримкою периферійних (edge) та проміжних (fog) рівнів обробки [11-13, 16]. Це дає змогу частину аналітики переносити ближче до джерела даних, зменшуючи затримки та навантаження на хмарну інфраструктуру, а також забезпечуючи роботу системи в напів автономному режимі у разі втрати зв'язку з Інтернетом [10, 13].

Безпека розглядається як одна з найкритичніших властивостей IoT-систем. У сучасних оглядах і систематичних дослідженнях показано, що вразливості можуть виникати на всіх рівнях – від фізичних пристроїв і протоколів зв'язку до хмарних сервісів та користувацьких застосунків [4, 6, 8, 11]. Для систем «розумного дому», де обробляються чутливі дані про присутність людей і стан приміщень, ці аспекти безпеки мають безпосередній вплив на рівень довіри користувачів [2, 8].

Архітектура IoT-систем зазвичай має багаторівневу структуру (рисунок 1.2), у якій виділяють рівень пристроїв, комунікаційний рівень, рівень обробки даних і рівень користувацьких застосунків [11, 12, 17, 18].

Рисунок 1.2 – Схема багаторівневої архітектури IoT

Такий підхід забезпечує послідовний збір, фільтрацію, зберігання та використання даних у реальному часі, а також дозволяє ізолювати функції між рівнями та спрощує модернізацію окремих компонентів без повного перепроєктування системи [11-13]. На рівні пристроїв розміщуються сенсори, виконавчі механізми та контролери. Сенсори збирають дані про навколишнє середовище, зокрема температуру, вологість, освітленість, рух, вібрацію або стан дверей і вікон. Виконавчі пристрої реагують на отриману інформацію, вмикаючи освітлення, регулюючи вентиляцію, керуючи електромеханічними замками чи активуючи сигналізацію. Контролери координують роботу всіх елементів, агрегують дані з сенсорів і передають їх на вищі рівні архітектури [11, 13].

Комунікаційний рівень відповідає за обмін даними між пристроями та центральними сервісами й реалізується за допомогою набору спеціалізованих протоколів і стандартів. Огляд існуючих платформ показує, що в цій ролі найчастіше виступають MQTT, HTTP(S), CoAP та інші протоколи прикладного рівня поверх TCP/IP або UDP [12-14, 19]. У реальних системах часто поєднується кілька типів з’єднань, що підвищує надійність і стабільність передавання даних та дозволяє адаптувати мережу до обмежень різних бездротових технологій [11-13]. Рівень обробки даних виконує попередню фільтрацію, агрегування, аналіз та зберігання інформації. Залежно від вимог до затримок, обсягу трафіку й обчислювальних ресурсів обробка може здійснюватися на локальному сервері, периферійному вузлі або в хмарному середовищі [10-13, 20]. Хмарні рішення забезпечують глобальний доступ до даних, гнучке масштабування і інтеграцію з аналітичними сервісами, тоді як edge-обробка дає змогу реалізувати швидку реакцію в системах безпеки та зменшити залежність від зовнішньої мережевої інфраструктури.

Рівень користувацьких застосунків забезпечує взаємодію людини з системою через мобільні додатки, вебпанелі або інтеграцію з голосовими асистентами. У сучасних роботах показано, що вимоги до цього рівня включають не лише зручність інтерфейсу, а й підтримку конфігурації сценаріїв автоматизації, гнучкі механізми сповіщення та інструменти візуальної аналітики [2, 11, 13]. Для систем «розумного дому» цей рівень фактично визначає суб’єктивне сприйняття безпеки

та контролю користувачем, оскільки саме через нього здійснюється моніторинг стану будинку й керування режимами роботи окремих підсистем.

У системах Інтернету речей комунікація між пристроями може відбуватися за різними моделями. У сучасних роботах поширеним є поділ на взаємодію «пристрій-пристрій», «пристрій-шлюз», «пристрій-сервер» та моделі безпосередньої взаємодії з користувачем [16, 17].

Модель пристрій-пристрій (Device-to-Device, D2D) (рисунок 1.3) передбачає прямий обмін даними між двома або кількома пристроями без участі центрального сервера й використовується, зокрема, у локальних мережах на базі Bluetooth або Zigbee, де важлива мінімальна затримка реакції та низьке енергоспоживання [16, 17].

Модель пристрій-сервер (Device-to-Server, D2S) (рисунок 1.4) реалізує передавання даних безпосередньо на центральний сервер або хмарну платформу, де інформація зберігається, аналізується й використовується для формування керуючих впливів; саме ця модель домінує в більшості комерційних IoT-рішень [11-13, 16].

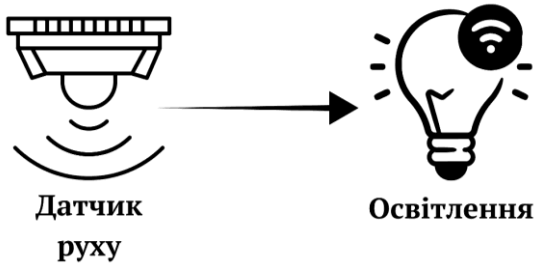

Рисунок 1.3 – Модель пристрій-пристрій (D2D)

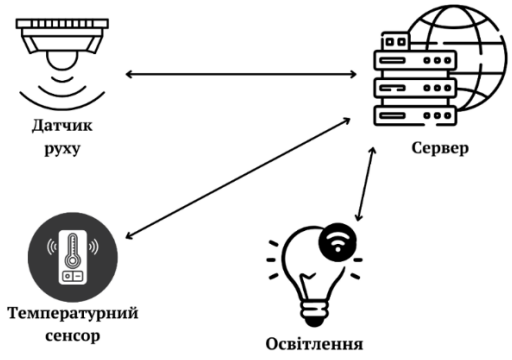

Рисунок 1.4 – Модель пристрій-сервер (D2S)

Модель сервер-сервер (Server-to-Server, S2S) (рисунок 1.5) використовується в інтегрованих рішеннях, коли необхідний обмін даними між кількома платформами або доменами, наприклад між системою відеоспостереження та зовнішньою аналітичною службою [11, 12].

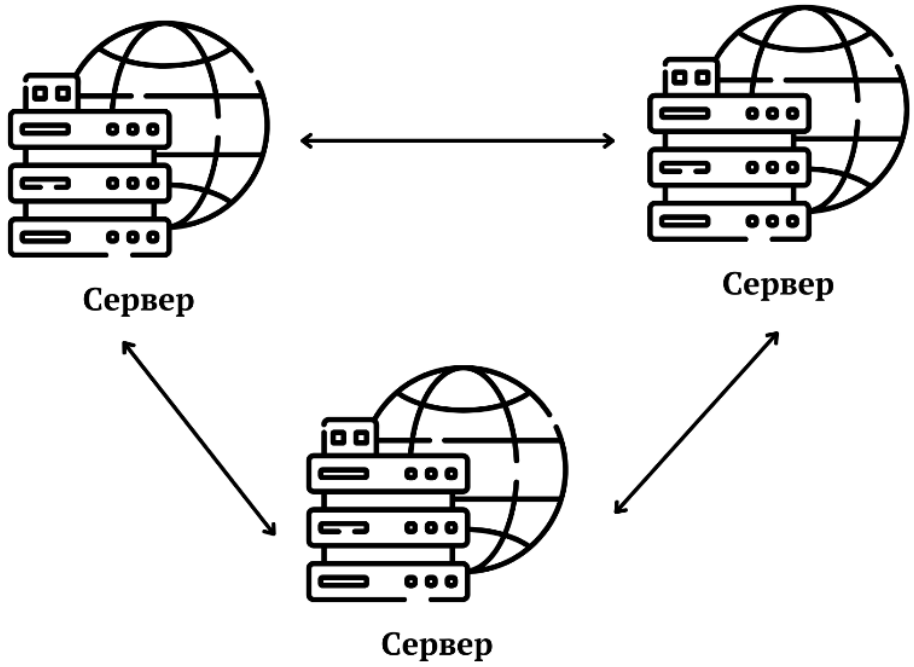

Рисунок 1.5 – Модель сервер-сервер (S2S)

Модель пристрій-користувач (Device-to-User, D2U) (рисунок 1.6) забезпечує прямий зв’язок між окремим пристроєм та інтерфейсом користувача через мобільний застосунок або вебпанель і є ключовою для сценаріїв моніторингу та оперативного керування у розумних будинках [2, 3, 13].

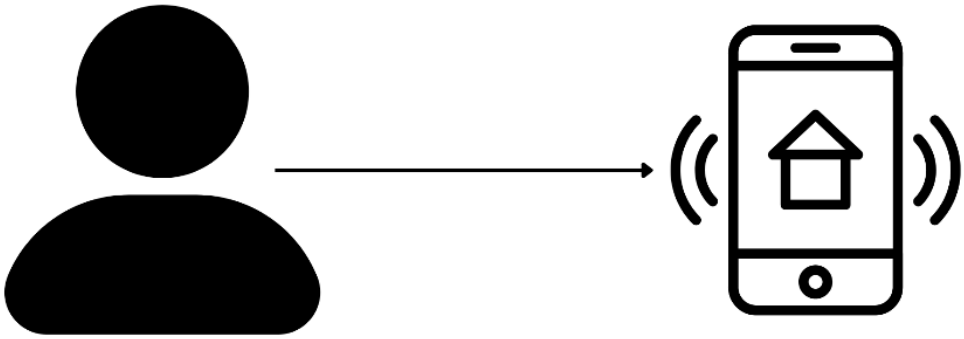

Рисунок 1.6 – Модель пристрій-користувач (D2U)

На практиці сучасні IoT-системи часто поєднують кілька моделей одночасно, формуючи гібридну архітектуру, у якій частина функцій реалізується локально, а інші – у хмарі. Таке поєднання дозволяє збалансувати швидкість реакції, автономність вузлів і можливості централізованого аналізу даних [11-13, 16]. Для підсистем безпеки «розумного дому» це означає, що критично важливі події, пов'язані з тривогами, мають оброблятися локально із мінімальною затримкою, тоді як менш чутливі функції (зберігання історії, довгострокова аналітика, віддалений доступ) можуть покладатися на хмарні сервіси. У контексті цієї кваліфікаційної роботи саме така комбінація принципів підключення, масштабованості, безпеки та енергоефективності, а також вибір відповідної архітектури та моделей комунікації використовуються як база для побудови локально орієнтованої системи безпеки приватного будинку з IoT-функціоналом.

1.3 Протоколи взаємодії у системах IoT

Однією з ключових умов коректного функціонування систем Інтернету речей є раціональний вибір протоколів взаємодії між вузлами. Саме вони визначають затримку доставлення повідомлень, надійність зв'язку, енергоспоживання пристроїв та можливості інтеграції з хмарними сервісами й користувацькими інтерфейсами. У сучасних оглядових роботах протоколи для IoT пропонується розглядати щонайменше у трьох вимірах: фізичний/канальний рівень (технологія бездротового чи дротового доступу), мережевий рівень та прикладний рівень (протоколи обміну даними між застосунками, сервісами та датчиками/акторами) [18, 19].

На фізичному рівні використовують як традиційні технології Wi-Fi та Bluetooth, так і спеціалізовані низькопотужні стандарти (Zigbee, Z-Wave, Thread, Wi-Fi HaLow (IEEE 802.11ah) тощо). Вони відрізняються робочою частотою, пропускною здатністю, радіусом дії, топологією мережі та енергоспоживанням. У роботах останніх років підкреслюється, що для побутових «розумних» систем зазвичай комбінують високошвидкісний канал (Wi-Fi) для передавання даних

великого обсягу та малопотужні mesh-мережі (Zigbee, Z-Wave, Thread) для численних сенсорів і виконавчих пристроїв [18, 21, 22, 23].

Узагальнено рівневу структуру комунікацій в IoT можна подати у вигляді моделі, де нижній рівень утворюють технології доступу (Wi-Fi, Ethernet, Zigbee, Z-Wave, мобільні мережі тощо), середній – транспортні протоколи (TCP, UDP, іноді QUIC), а верхній – прикладні протоколи обміну (HTTP/REST, MQTT, CoAP, WebSocket та ін.) [18-19]. Взаємодія цих рівнів забезпечує маршрутизацію повідомлень, керування сесіями, контроль якості обслуговування (QoS), механізми шифрування та авторизації. Узагальнену схему рівнів протоколів взаємодії у системах Інтернету речей наведено на рисунку 1.7.

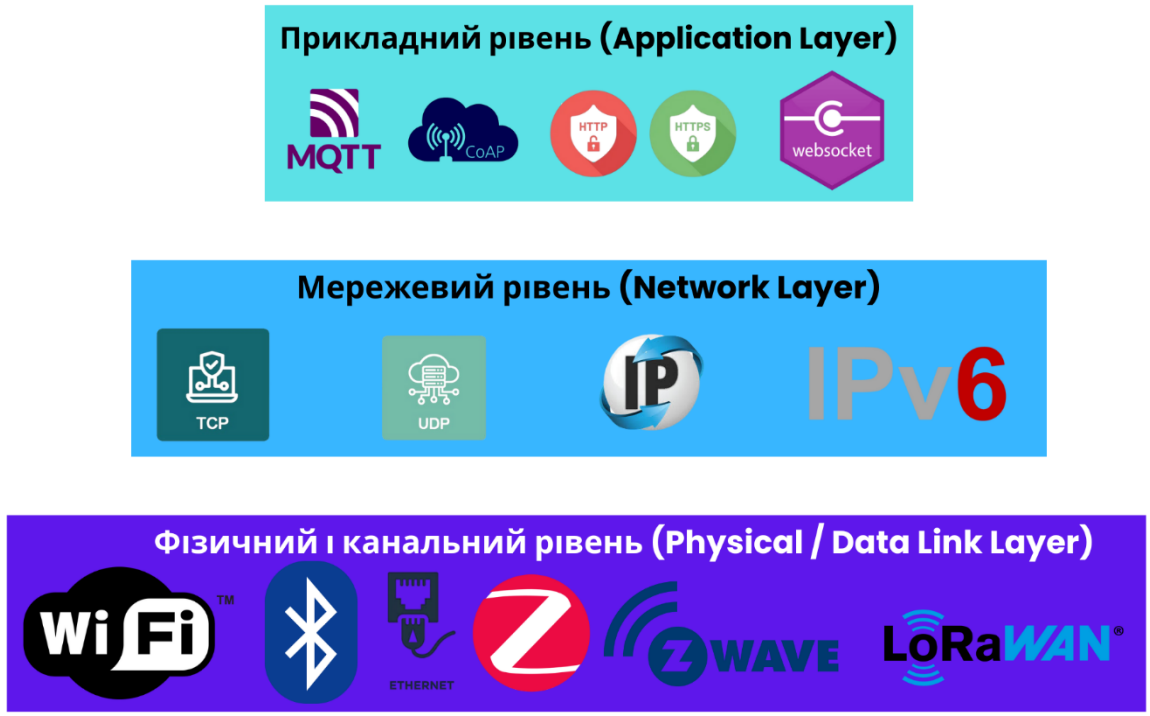

Рисунок 1.7 – Рівнева структура протоколів взаємодії в системах IoT

Для систем домашньої автоматизації особливо поширеним є стандарт Wi-Fi (IEEE 802.11n/ac/ax) завдяки високій пропускній здатності та простій інтеграції з існуючою інфраструктурою локальних мереж. Він дозволяє передавати відеопотоки, журнали подій та інші дані значного обсягу, що є критично важливим для систем відеоспостереження та аналітики. Недоліком Wi-Fi є підвищене

енергоспоживання, тому його важче застосовувати для автономних датчиків на батарейному живленні.

Для таких вузлів у літературі рекомендують Bluetooth Low Energy (BLE), Zigbee, Z-Wave, Thread, що підтримують режим сну, невеликий інформаційний обмін і топологію типу mesh [18, 22]. Zigbee (на основі IEEE 802.15.4) орієнтований на малопотужні мережі короткого радіусу дії та дозволяє будувати кластеризовані чи сітчасті мережі для автоматизації будівель [23]. Z-Wave працює в діапазоні нижче 1 ГГц, що зменшує перешкоди від Wi-Fi та інших пристроїв у діапазоні 2,4 ГГц, забезпечуючи більший радіус дії й кращу проникність сигналу крізь перешкоди, але з меншою швидкістю передавання [22].

Перспективним напрямом розвитку Wi-Fi для IoT є стандарт IEEE 802.11ah (Wi-Fi HaLow), який працює в субгігагерцовому діапазоні, підтримує більший радіус дії і нижче енергоспоживання порівняно з класичним Wi-Fi, зберігаючи при цьому сумісність із TCP/IP-стеком та орієнтацію на масові IoT-вузли [24].

У системах «розумного дому» на практиці часто поєднують декілька технологій: Wi-Fi використовується для контролерів, шлюзів і пристроїв з великим обсягом передавання даних, тоді як Zigbee, Z-Wave або Thread забезпечують малопотужний збір даних від датчиків [21-23].

На прикладному рівні IoT системи базуються на наборах протоколів, які забезпечують логіку обміну, структуру повідомлень та семантику команд. Огляд [18] показує, що серед найуживаніших протоколів верхнього рівня для IoT – HTTP/REST, MQTT, CoAP, AMQP, DDS, OPC UA, LwM2M, причому вибір конкретного рішення пов’язаний із вимогами до енергоспоживання, надійності, масштабованості та інтеграції з веб-технологіями. У роботі [19] виконується детальне порівняння функціональності цих протоколів і даються рекомендації щодо їх застосування для різних класів пристроїв і шаблонів взаємодії (збір телеметрії, команди керування, подієві сповіщення, оновлення прошивки тощо).

MQTT (Message Queuing Telemetry Transport) є одним із найпоширеніших протоколів для передачі телеметрії в IoT. Він реалізує парадигму «видавець-підписник», де обмін ведеться через брокер повідомлень, а теми (topics) визначають логічну структуру даних. Основні переваги MQTT – малий розмір

заголовка, підтримка рівнів QoS, можливість зберігання останнього повідомлення (retained messages) та робота поверх TCP/IP [18, 19]. У сучасних роботах виконано детальну оцінку продуктивності та безпеки MQTT у поєднанні з шифруванням TLS на ресурсно обмеженому обладнанні. Зокрема, у [21] показано, що додавання криптографічного захисту збільшує затримку й споживання смуги пропускання, однак MQTT із TLS залишається придатним для широкого спектра IoT-застосунків, якщо оптимізувати вибір шифросуїт і параметрів QoS.

Порівняльні експерименти свідчать, що за умов обмеженої пропускної здатності MQTT демонструє нижчу затримку та менше коливання часу доставлення, ніж HTTP/REST, особливо для періодичних коротких повідомлень сенсорів [18, 22, 24]. Це робить протокол доцільним для систем із великою кількістю вузлів моніторингу та обмеженим каналом зв’язку, наприклад, розподіленого контролю мікроклімату чи промислового моніторингу.

CoAP (Constrained Application Protocol) був спеціально розроблений робочою групою IETF для ресурсно обмежених пристроїв у втратних середовищах і реалізує легковаговий варіант веб-парадигми REST поверх UDP. Протокол використовує методи GET, POST, PUT, DELETE, має фіксований невеликий заголовок, підтримує надійну та ненадійну доставку, а також механізми спостереження за ресурсами (observe), що дозволяє вузлам отримувати оновлення лише за наявності змін [20]. Для захищеного обміну CoAP застосовує DTLS поверх UDP, що мінімізує накладні витрати порівняно з TLS поверх TCP [20].

У в попередньому огляді показано, що CoAP завдяки малій сигнальній надлишковості та підтримці багатокасту добре підходить для бездротових сенсорних мереж і застосунків домотики, але ускладнює роботу за NAT-обладнанням і брандмауерами, де HTTP/HTTPS має природну перевагу. Порівняльні дослідження з MQTT демонструють, що CoAP виграє в сценаріях, де важлива подієва модель взаємодії на базі REST і пряме використання URI-ресурсів, тоді як MQTT краще масштабуються для багатьох публікаторів і підписників із брокером [18, 22].

HTTP/REST залишається універсальним прикладним протоколом, особливо коли потрібна пряма доступність пристроїв із веб-браузерів і простих клієнтів. У

[19] підкреслюється, що IoT-системи малої та середньої складності можуть взагалі обходитися без окремого «меседжингового» протоколу, використовуючи лише HTTP та WebSocket як транспорт для прикладних протоколів. Це спрощує розробку й дебаг, але збільшує накладні витрати та ускладнює оптимізацію енергоспоживання у великих розгортаннях.

WebSocket називають двонаправлений протокол поверх TCP, який починається як HTTP-запит (handshake), а далі забезпечує постійне full-duplex-з'єднання між клієнтом і сервером. У підсумках роботи [19] зазначається, що багато повідомлєвих протоколів (зокрема MQTT та інші) мають профілі для роботи поверх WebSocket, а сам WebSocket активно використовується для інтерактивних веб-інтерфейсів IoT. Експериментальні дослідження, виконані для реальних IoT-сценаріїв на ESP32, показують, що MQTT зазвичай забезпечує нижчу затримку для телеметрії, а WebSocket має більшу гнучкість для двосторонньої взаємодії з користувацьким інтерфейсом, при цьому затримки залишаються прийнятними для систем реального часу домашньої автоматизації [23].

Окремим напрямом розвитку є створення надпротокольних стандартів взаємодії «розумного дому», які абстрагуються від конкретної технології зв'язку. До таких належить стандарт Matter, що розвивається під егідою Connectivity Standards Alliance та підтримує роботу поверх IPv6, Thread, Wi-Fi та інших технологій. Matter орієнтований на взаємну сумісність пристроїв різних виробників, єдину модель безпеки та спрощене включення нових пристроїв до екосистеми [24]. У поєднанні з протоколом Thread це дозволяє створювати автономні mesh-мережі сенсорів, тоді як Wi-Fi забезпечує зв'язок із маршрутизатором та Інтернетом. У статтях, присвячених розвитку Z-Wave та інших протоколів, підкреслюється, що для збереження актуальності вони інтегруються з Matter через програмні мости, тобто також рухаються в напрямі уніфікації екосистем [22, 24].

На основі аналізу наукових праць та оглядових статей можна сформулювати критерії вибору протоколів для системи «розумного дому»: необхідність підтримки відеопотоку та періодичної телеметрії (температура, вологість, газ, рух), вимога до

інтерактивного веб-інтерфейсу (панель керування, мобільний браузер), відносно невелика кількість вузлів, більшість з яких живляться від мережі, потреба у локальній роботі без зовнішнього Інтернету та простому розгортанні у домашніх умовах.

З огляду на це, для розроблюваної системи доцільним є вибір на фізичному/канальному рівні Wi-Fi 2,4 ГГц, оскільки більшість контролерів мають вбудований Wi-Fi-модуль, що забезпечує достатню пропускну здатність для відеопотоку від камери, одночасної передачі телеметрії та оновлення веб-інтерфейсу без потреби у додаткових шлюзах Zigbee/Z-Wave. На прикладному рівні для обміну між головним контролером, веб-сервером та клієнтами застосовується комбінація HTTP/REST у поєднанні з WebSocket.

1.4 Огляд апаратних засобів для реалізації IoT-систем безпеки

У системах «розумного дому» з функціями безпеки апаратна складова відіграє визначальну роль. Саме від вибору компонентів залежить стабільність, точність роботи та масштабованість усієї інфраструктури. Апаратні засоби забезпечують зв’язок між фізичним середовищем і програмною логікою, виконують збір, обробку та передавання інформації, а також реалізують реакцію на події. У системах безпеки вони об’єднані в єдину мережу, де кожен елемент виконує чітко визначену функцію.

Базовою ланкою апаратної архітектури є контролер. Це головний обчислювальний елемент, що координує роботу всіх інших модулів. У сучасних IoT-рішеннях найпопулярнішими є мікроконтролери серій ESP, Arduino та Raspberry Pi (рисунок 1.8), які поєднують доступність, гнучкість і підтримку бездротових інтерфейсів. Мікроконтролери ESP32 особливо цінуються завдяки вбудованим модулям Wi-Fi і Bluetooth, високій обчислювальній потужності та енергоефективності. Вони можуть одночасно обробляти сигнали з кількох сенсорів, керувати виконавчими пристроями й передавати дані до хмарного сервера або мобільного застосунку. Raspberry Pi, у свою чергу, використовується

як міні-комп’ютер для складніших обчислень, зберігання баз даних або обробки відеопотоків.

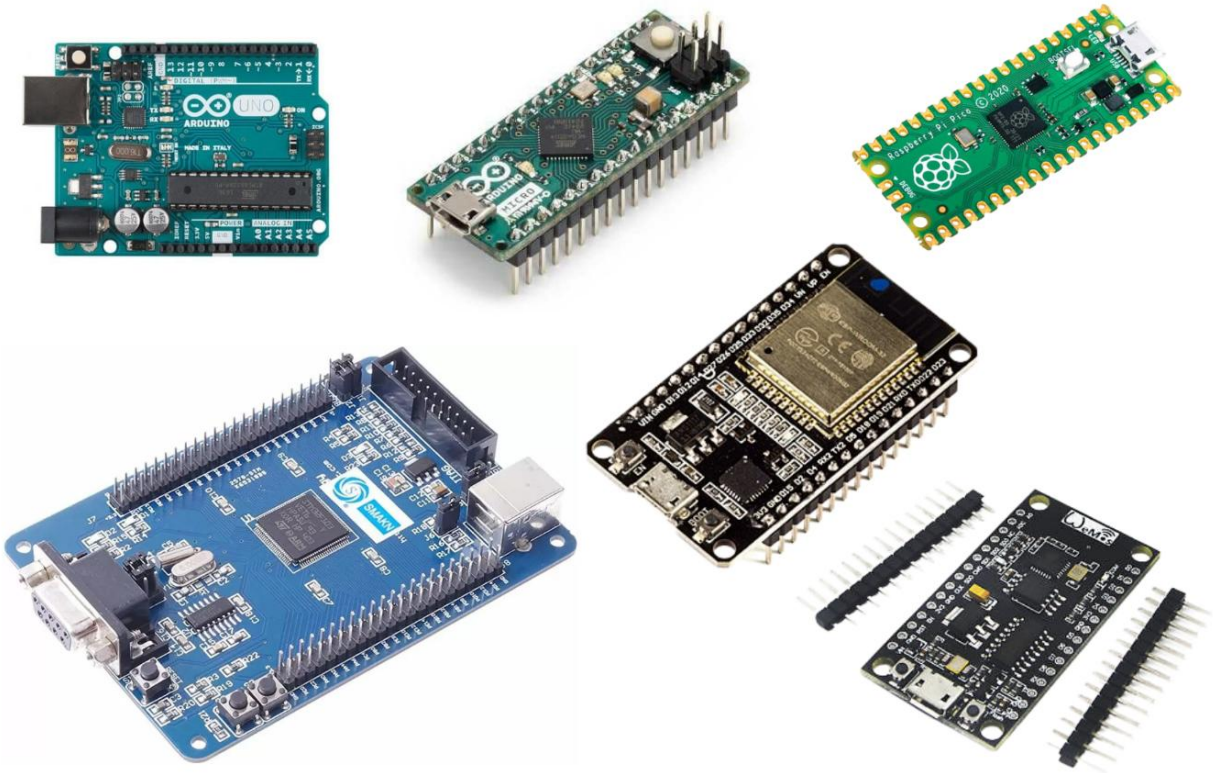

Рисунок 1.8 – Найпопулярніші моделі мікроконтролерів

Другим ключовим елементом є сенсорна система, що включає датчики для виявлення змін у навколишньому середовищі (рисунок 1.9). У контексті безпеки використовуються датчики руху, відкриття дверей, диму, витоку газу, температури й вологості. Наприклад, популярний модуль HC-SR501 реагує на рух у межах приміщення, а магнітний датчик MC-38 фіксує факт відкриття вікон або дверей. Датчик газу MQ-9 дозволяє виявляти небезпечні концентрації чадного газу або метану, запобігаючи аварійним ситуаціям. Дані з усіх сенсорів передаються до контролера для подальшої обробки й формування рішень. Для підвищення надійності зчитування деякі сенсори використовують алгоритми згладжування та фільтрації, що зменшує вплив випадкових шумів. Крім того, сенсорна підсистема підтримує модульну розширюваність, що дає змогу підключати додаткові датчики без змін у базовій архітектурі системи.

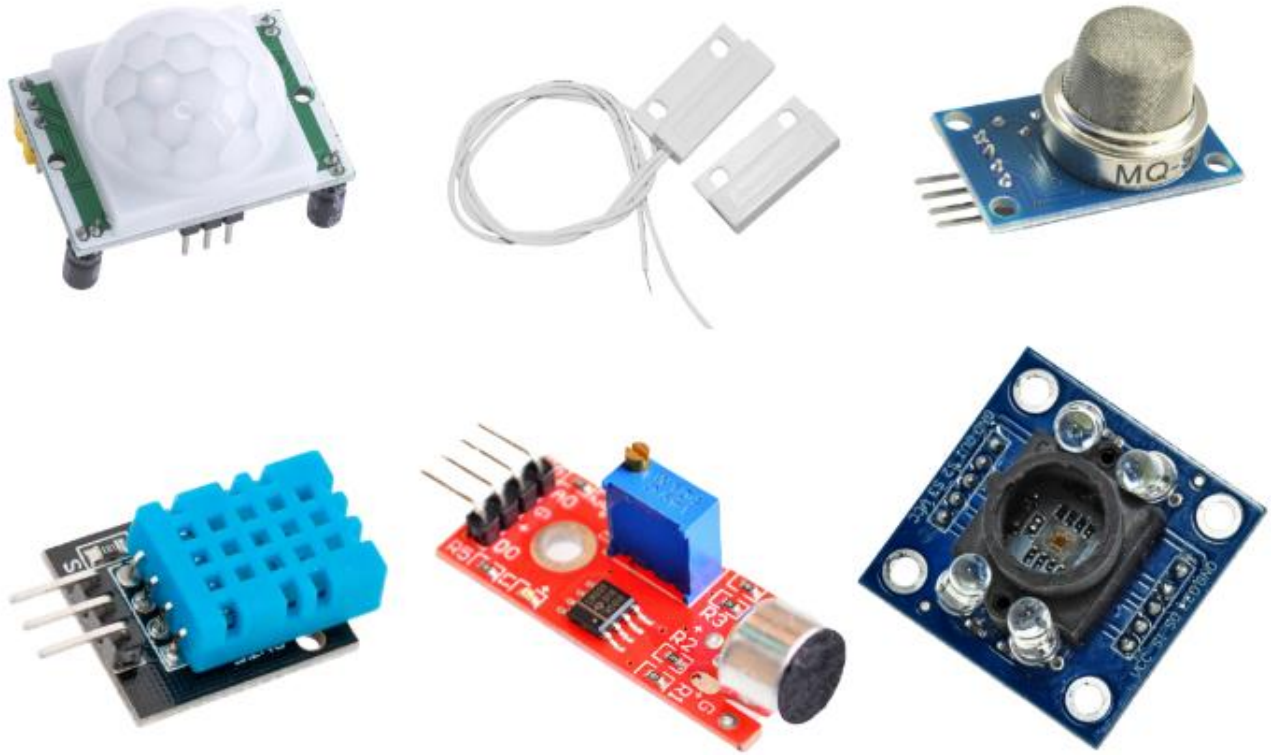

Рисунок 1.9 – Приклади датчиків для фіксації змін у навколишньому середовищі

Наступним компонентом виступають виконавчі пристрої, що відповідають за реалізацію дій у системі. До них належать реле, електромеханічні замки, серводвигуни, сигнальні пристрої та освітлювальні модулі. За допомогою цих елементів система може блокувати двері, вмикати сирену, активувати вентиляцію або змінювати режим освітлення.

Наприклад, серводвигун MG90S 360 використовується для керування воротами або дверима, а звуковий сигналізатор (бузер) подає звукове сповіщення у разі спрацювання сигналізації. Приклади виконавчих пристроїв наведено на рисунку 1.10. Усі виконавчі модулі інтегровані через уніфіковані інтерфейси керування, що дозволяє швидко змінювати їхню конфігурацію та додавати нові елементи без зміни основної логіки. Крім того, система підтримує сценарну взаємодію, за якої кілька виконавчих пристроїв можуть активуватися послідовно або паралельно залежно від типу події.

Рисунок 1.10 – Приклади виконавчих пристроїв в IoT-системах

Для візуального контролю часто застосовуються камери з передаванням відео через Wi-Fi, такі як ESP32-CAM, яка підтримує зйомку у реальному часі та передавання зображення безпосередньо до мобільного додатку або вебінтерфейсу. Камера забезпечує додатковий рівень захисту, дозволяючи не лише фіксувати події, але й вести моніторинг території з будь-якої точки.

Окрему роль відіграють модулі ідентифікації користувачів, серед яких RFID-зчитувачі MFRC522 і сканери відбитків пальців R307. Вони дозволяють реалізувати функції контролю доступу та автентифікації. Користувач може відкрити двері або вимкнути сигналізацію за допомогою персонального ключа, картки або відбитка пальця, що підвищує рівень безпеки системи.

Для взаємодії користувача з системою використовуються дисплеї та клавіатури. Екран LCD 1602 дозволяє відображати повідомлення, коди доступу чи поточний стан системи. Сенсорний екран ESP32-2432S028R використовується як універсальний інтерфейс для керування режимами, перегляду даних з датчиків і налаштування параметрів. Матрична клавіатура 3x4 забезпечує ручний ввід

паролів або команд. Такі елементи дозволяють користувачу взаємодіяти із системою навіть без мобільного додатку.

Важливою частиною апаратної структури є енергетична система, яка забезпечує стабільне живлення всіх компонентів. Для цього застосовуються блоки живлення постійного струму, перетворювачі напруги та модулі резервного живлення. У системах, що використовують пристрої з різними робочими напругами (5 В і 3,3 В), застосовуються макетні плати з регуляторами та адаптерами живлення. Для керування високострумовими навантаженнями, такими як вентилятори або світлодіодні стрічки, використовуються MOSFET-модулі, які дозволяють ефективно комутувати енергоспоживання.

Загальна архітектура апаратних засобів IoT-системи безпеки побудована за принципом багаторівневої інтеграції. На нижньому рівні розташовані сенсори, які здійснюють моніторинг середовища. Середній рівень становлять контролери, що обробляють дані та передають їх до серверної частини або мобільного інтерфейсу. На верхньому рівні розміщені пристрої взаємодії з користувачем, які забезпечують відображення, керування і реагування. Такий підхід дозволяє створити стабільну, масштабовану та адаптивну систему, яка відповідає сучасним вимогам безпеки й комфорту.

1.5 Аналіз популярних рішень на ринку

Сучасний ринок систем «розумного дому» в Україні активно розвивається завдяки впровадженню IoT-технологій і зростанню попиту на рішення, що поєднують безпеку, комфорт та енергоефективність. Українські виробники успішно конкурують із міжнародними брендами, пропонуючи локалізовані системи, адаптовані до потреб вітчизняних користувачів. Основний акцент робиться на інтеграції бездротових протоколів зв’язку, зручних мобільних інтерфейсах і підтримці хмарних сервісів.

У межах аналізу було розглянуто кілька найпоширеніших рішень, доступних на українському ринку. До вибірки увійшли такі системи, як Ajax Systems, U-Prox Security, ATIS Smart Security, Dahua Smart Security та Hikvision AX PRO. Ці рішення

поєднують апаратну та програмну частини, підтримують роботу через Wi-Fi, GSM або радіопротоколи, мають мобільні додатки для дистанційного керування та інтегруються в екосистему «розумного дому».

Для зручності оцінювання основних характеристик систем було створено таблицю 1.1, де наведено їх типи, функціональні можливості, підтримувані технології та особливості експлуатації. Це дозволяє наочно порівняти різні підходи до побудови IoT-систем безпеки та визначити, які з них найкраще підходять для впровадження в рамках приватного житлового об’єкта.

Таблиця 1.1 – Порівняльний аналіз популярних IoT-рішень для систем безпеки

№	Назва системи / рішення	Тип системи	Основні функції	Підтримувані протоколи та технології	Особливості
1	2	3	4	5	6
1	Ajax Systems (Україна)	Комплексна система безпеки	Відеоспостереження, контроль доступу, сигналізація, автоматизація	Jeweller, Wi-Fi, Ethernet, GSM	Власний радіопротокол мобільний застосунок Ajax Security System, хмарна інтеграція
2	U-Prox Security (Україна)	Охоронна система з контролем доступу	Сигналізація, контроль доступу, аналітика, моніторинг	868 МГц, AES-256, Wi-Fi, GSM	Високий рівень захисту даних, мобільний додаток U-Prox Home
3	ATIS Smart Security (Україна / Китай)	IoT-система на базі Tuya	Сигналізація, датчики руху, диму, газу, протікання	Wi-Fi, GSM, Tuya Smart	Інтеграція з екосистемою Tuya, дистанційне керування, підтримка голосових асистентів та голосового керування
4	Dahua Smart Security	Система відеоспостереження та автоматизації	Камери, сигналізація, керування доступом, мобільний моніторинг	Wi-Fi, Ethernet, IoT-платформи	Мобільні застосунки DMSS, Tuya, висока якість відео, хороше програмне забезпечення

Продовження таблиці 1.1

1	2	3	4	5	6
5	Hikvision AX PRO	Інтелектуальна охоронна система	Сигналізація, відеоспостереження, контроль доступу	Wi-Fi, Ethernet, RF 868 МГц	Мобільний застосунок Hik-Connect, підтримка хмарних сервісів, розширювана архітектура

Одним із найпоширеніших рішень на українському ринку є Ajax Systems (рисунок 1.11), яка поєднує функції охоронної сигналізації, відеоспостереження та контролю доступу. Система складається з центрального хабу, датчиків руху, відкриття дверей і вікон, димових сенсорів, сирен та ретрансляторів сигналу. Всі елементи працюють на основі фірмового радіопротоколу Jeweller, що забезпечує стабільне з’єднання навіть на великій відстані. Керування здійснюється через мобільний застосунок Ajax Security System, який підтримує хмарну синхронізацію та сповіщення в реальному часі.

Рисунок 1.11 – Приклад базових компонентів системи Ajax Systems [25]

Ще одним українським прикладом є U-Prox Security, яка орієнтована на побутові та корпоративні рішення. Система включає бездротові датчики руху, клавіатури, брелоки, модулі контролю доступу та ретранслятори. Для захисту даних використовується шифрування AES-256, що гарантує безпечну передачу сигналів. Завдяки мобільному додатку U-Prox Home користувач може змінювати режими охорони, отримувати сповіщення про події та контролювати стан об’єктів у будь-який час. На рисунку 1.12 наведено зовнішній вигляд типових елементів системи U-Prox Security.

Рисунок 1.12 – Приклад елементів системи U-Prox Security [26]

Високий попит серед користувачів має також ATIS Smart Security, побудована на базі платформи Tuya Smart. Згідно джерела [27], ця система поєднує простоту встановлення з широкими можливостями інтеграції, адже підтримує датчики руху, температури, диму, витоку газу, IP-камери та модулі домофонії. Всі пристрої можуть об’єднуватися в єдину екосистему через мобільний застосунок, що дозволяє користувачеві віддалено керувати об’єктом і переглядати відеопотік у реальному часі (рисунок 1.13).

Рисунок 1.13 – Приклад елементів системи безпеки ATIS Smart Security

Серед комплексних рішень для відеонагляду та охорони важливе місце посідає Dahua Smart Security. Система об’єднує відеокамери, реєстратори, сенсори руху, контролери доступу та інші модулі в єдину мережу. Її особливістю є можливість роботи з хмарним сервісом Imou Cloud, який забезпечує дистанційне зберігання відео та віддалене керування. Для користувачів розроблено застосунок DMSS, що дозволяє переглядати потік із камер, керувати сигналізацією та створювати сценарії автоматизації. Зовнішній вигляд елементів системи наведено на рисунку 1.14.

Рисунок 1.14 – Приклад елементів системи Dahua Smart Security [28]

Окрему увагу заслуговує Hikvision AX PRO, яка поєднує переваги бездротової сигналізації та системи відеоспостереження. До її складу входять різні типи датчиків, бездротові клавіатури, сирени, IP-камери та мережеві модулі. Всі пристрої взаємодіють через фірмовий протокол RF 868 МГц, що гарантує стабільність роботи навіть у складних умовах. Користувач має можливість повного контролю системи через мобільний застосунок Hik-Connect, який підтримує функції віддаленого моніторингу, повідомлень та інтеграцію з іншими IoT-пристроями. Зразки компонентів Hikvision AX PRO подано на рисунку 1.15.

Рисунок 1.15 – Приклад елементів системи Hikvision AX PRO [29]

Проведений аналіз свідчить, що сучасні українські системи безпеки активно впроваджують технології Інтернету речей, поєднуючи локальні рішення з хмарними сервісами. Вони вирізняються високою стабільністю зв’язку, можливістю розширення та підтримкою мобільного керування. Такі рішення роблять системи безпеки більш доступними для користувачів і дозволяють гнучко адаптувати їх під конкретні завдання, забезпечуючи комплексний захист житлових і комерційних об’єктів.

РОЗДІЛ 2
ДОСЛІДЖЕННЯ ТА ПРОЄКТУВАННЯ СИСТЕМИ DOMOVYK

Розвиток концепції розумного будинку спричинив появу численних систем безпеки з використанням технологій Інтернету речей. Проте аналіз існуючих рішень, проведений у попередньому розділі, показує, що більшість з них мають низку обмежень, а саме: жорстку залежність від закритих платформ, відсутність можливості глибокого налаштування, високу вартість та складність інтеграції нових компонентів. Для користувачів і розробників, які прагнуть створювати системи з індивідуальними функціями та власною логікою роботи, такі обмеження є суттєвим бар'єром.

Саме тому актуальним напрямом дослідження є проєктування відкритої, гнучкої та модульної системи безпеки, яку можна легко адаптувати до потреб конкретного будинку або користувача. Основна ідея полягає у створенні архітектури, що дозволяє вільно поєднувати різні типи сенсорів, контролерів, виконавчих пристроїв і засобів керування, не змінюючи базову структуру комплексу. Такий підхід забезпечує масштабованість, простоту обслуговування та можливість поступового розширення системи без повної перебудови її логіки.

Метою цього розділу є дослідження принципів побудови і розроблення архітектури системи Domovyk – апаратно-програмного комплексу безпеки приватного будинку з IoT-функціоналом, який поєднує можливості моніторингу, реагування, керування й інтерактивної взаємодії з користувачем. У межах проєктування буде розглянуто структурну модель системи, визначено взаємозв'язки між її апаратними й програмними складовими, обґрунтовано вибір основних компонентів та описано загальні алгоритми роботи.

Особливу увагу приділено гнучкості та універсальності рішень, що дозволяють реалізувати не лише базові функції охорони, а й додаткові можливості автоматизації, такі як: контроль мікроклімату, відеоспостереження, керування освітленням, енергоспоживанням і доступом до приміщень. Таким чином, система Domovyk розглядається як відкрита платформа для створення індивідуальних сценаріїв безпеки, здатна адаптуватися до різних умов і вимог користувача.

2.1 Архітектура та структурна модель системи Domovyk

Проєктування архітектури є одним із ключових етапів створення будь-якої системи Інтернету речей, оскільки саме на цьому рівні формується загальна логіка взаємодії між усіма компонентами, визначається структура зв’язків і розподіл функцій між окремими вузлами. Від якості архітектурних рішень залежить не лише працездатність системи, але і її масштабованість, стабільність, енергоефективність та можливість подальшого розширення. Архітектура виступає своєрідним «каркасом» системи, який об’єднує апаратну й програмну частини в єдину функціональну структуру, що здатна гнучко реагувати на зміни навколишнього середовища та вимоги користувача.

У системах IoT архітектурне проєктування набуває особливого значення, адже такі системи поєднують безліч різнорідних пристроїв, сенсорів, модулів зв’язку та сервісів, які мають працювати як цілісний механізм. Успішна побудова архітектури передбачає визначення ролей кожного елемента, способів обміну інформацією між ними, логіки керування потоками даних і механізмів забезпечення синхронної роботи. З огляду на це, процес архітектурного проєктування є не лише технічним завданням, а й дослідницьким етапом, під час якого вивчаються можливі моделі взаємодії пристроїв, проводиться аналіз оптимальних схем комунікації та обґрунтовується вибір загальної структури системи.

Створення архітектури системи Domovyk є результатом попереднього аналізу вимог до сучасних систем безпеки, описаних у першому розділі, та спостереження за тенденціями розвитку IoT-технологій. На основі виявлених закономірностей визначено необхідність розроблення гнучкої, модульної та масштабованої архітектури, яка дозволяє легко адаптувати систему під конкретні умови використання. Архітектура Domovyk повинна забезпечувати ефективну взаємодію між сенсорними вузлами, контролерами, виконавчими пристроями та користувацьким інтерфейсом, при цьому залишаючи простір для подальшої модернізації й додавання нових компонентів без повної перебудови системи.

Архітектура системи Domovyk (рисунок 2.1) базується на багаторівневій структурі, що відображає логічний розподіл функцій між апаратною, мережевою та програмною частинами. Такий підхід забезпечує гнучкість, простоту оновлення та можливість масштабування системи без втручання у базову логіку її роботи. На верхньому рівні розташовується програмна логіка, веб-інтерфейс користувача та інші засоби візуалізації даних. Нижче знаходиться програмний рівень, що відповідає за обробку даних, синхронізацію модулів та управління виконавчими пристроями. На найнижчому рівні розміщені апаратні компоненти (сенсори, контролери, модулі зворотного зв'язку та пристрої фізичного впливу).

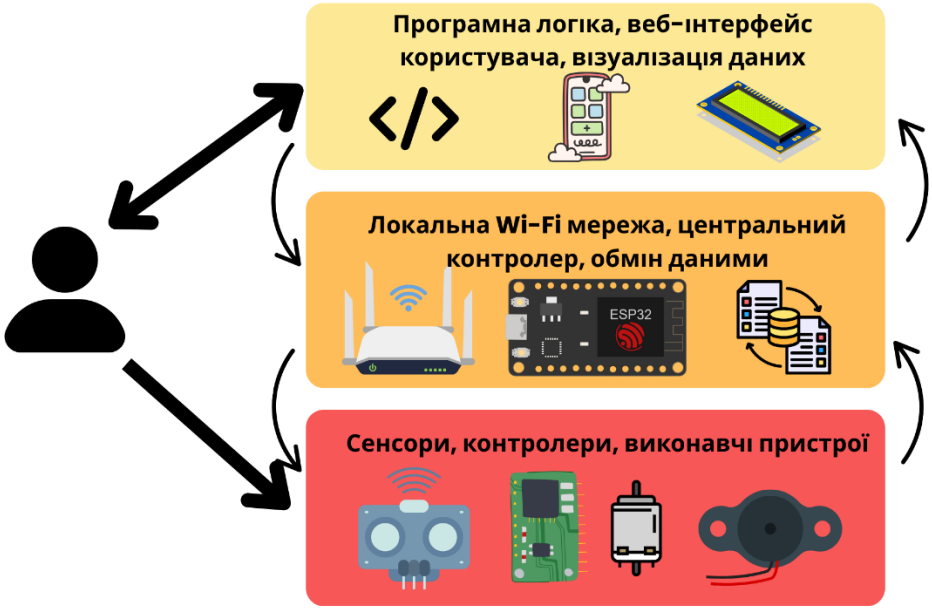

Рисунок 2.1 – Архітектура системи Domovyk

У процесі дослідження архітектури особливу увагу приділено питанню зв'язності компонентів. У системах Інтернету речей структура взаємодії може бути централізованою, децентралізованою або гібридною. Для Domovyk обрано комбінований підхід, у якому центральний контролер виконує функції керування та координації, тоді як допоміжні вузли мають певний рівень автономії. Це дозволяє зменшити навантаження на головний модуль і підвищити стійкість системи до можливих збоїв.

Архітектура Domovyk побудована таким чином, щоб забезпечити повну сумісність між рівнями, незалежність окремих модулів і можливість доповнення новими функціями. Наприклад, у подальшому можна інтегрувати додаткові сенсори або підсистеми автоматизації без суттєвих змін у програмному забезпеченні чи логіці обміну даними. Такий принцип модульності відповідає сучасним тенденціям проєктування IoT-рішень, де головна увага приділяється не лише функціональності, а й потенціалу розвитку системи.

Структурна модель системи Domovyk демонструє взаємодію між основними функціональними модулями комплексу. До її складу входять центральний контролер, сенсорні вузли, виконавчі пристрої, користувацький веб-інтерфейс і відеосервер. Кожен елемент виконує власну роль, але водночас інтегрований у єдине середовище обміну даними. Центральний контролер є ядром системи, що отримує, опрацьовує та передає інформацію між модулями. Сенсорні вузли формують дані про стан середовища, які передаються контролеру через локальні цифрові інтерфейси. Виконавчі пристрої реалізують реакцію системи – умикання сигналізації, керування освітленням або іншими механізмами. Веб-інтерфейс забезпечує двосторонній обмін: користувач надсилає команди, а система повертає актуальні показники сенсорів і повідомлення про події. Відеосервер функціонує як окремий вузол, що передає відеопотік у реальному часі безпосередньо користувачу. Загальна структурна схема представлена на рисунку 2.2, де відображено дві паралельні гілки взаємодії – обчислювальну (через головний контролер) і візуальну (через відеосервер).

Рисунок 2.2 – Структурна модель системи Domovyk

Передавання даних у системі здійснюється у двох напрямах. Контролер є приймачем інформації від сенсорів і джерелом команд для виконавчих вузлів. Обмін із користувачем відбувається через локальну Wi-Fi мережу з використанням легких текстових структур даних, що дає змогу забезпечити оперативну реакцію системи та мінімізувати затримки. Завдяки уніфікованим форматам даних кожен вузол може бути модернізований або замінений без зміни загальної логіки функціонування. Схему обміну між компонентами наведено на рисунку 2.3, де показано напрями потоків даних між сенсорними, виконавчими та керуючими модулями, а також канали зв'язку між користувачем, контролером і відеосервером.

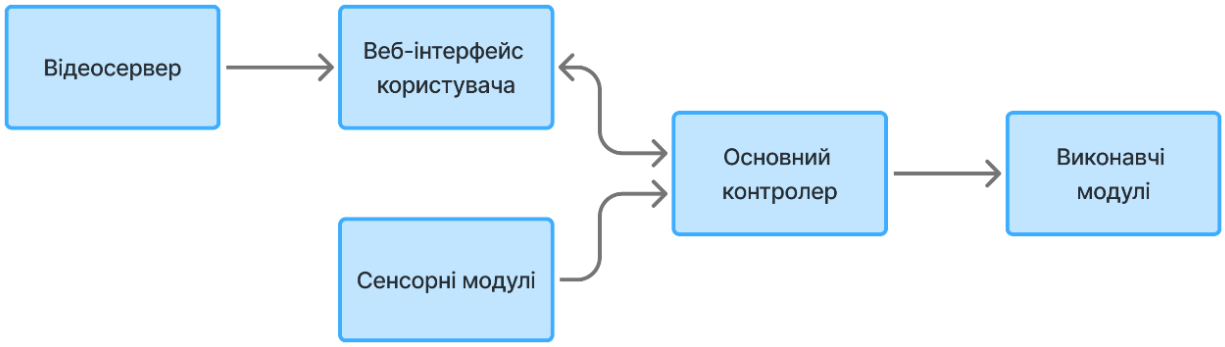

Рисунок 2.3 – Схема взаємодії між основними вузлами системи Domovyk

Програмна частина системи має модульну логічну структуру (рисунок 2.4). Вона складається з підсистем збирання даних, їх обробки, управління подіями, а також серверної та користувацької частини. Модуль збору даних опитує сенсорні вузли та передає результати у менеджер подій. Менеджер подій визначає пріоритетність сигналів, активує відповідні виконавчі дії та формує повідомлення для відображення у веб-інтерфейсі. Веб-сервер підтримує двосторонню комунікацію між користувачем і системою, а інтерфейс користувача забезпечує зручний перегляд параметрів, відео та керування функціями.

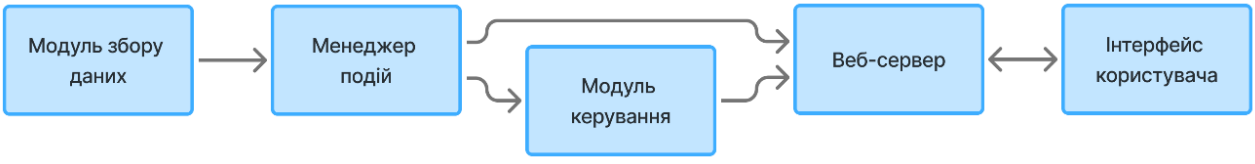

Рисунок 2.4 – Логічна структура програмної частини системи Domovyk

Вибір саме розподіленої архітектури з двома серверами обґрунтовується необхідністю розділити обчислювальні навантаження та підвищити гнучкість системи. Основний контролер відповідає за обмін даними й керування, тоді як відеосервер функціонує автономно, забезпечуючи стабільну передачу зображення.

У результаті дослідження встановлено, що запропонована архітектура відповідає сучасним принципам побудови IoT-систем, забезпечуючи гнучкість, масштабованість і простоту інтеграції нових модулів. Вона формує основу для подальшого вибору апаратних засобів і розроблення програмного забезпечення, яке реалізує зазначену модель у повному обсязі.

2.2 Вибір і обґрунтування апаратних засобів

Апаратна частина є основою системи Domovyk, оскільки саме вона забезпечує збирання, передавання, обробку та реалізацію інформаційних процесів, які визначають ефективність роботи всього комплексу. Вибір компонентів здійснюється з урахуванням специфіки поставленої задачі, що полягає у побудові розподіленої системи безпеки з підтримкою технологій Інтернету речей. Це вимагає повної узгодженості між сенсорними вузлами, виконавчими пристроями, контролерами та модулями комунікації. Від якості апаратних засобів залежить не лише стабільність функціонування системи, а й точність збирання даних, швидкість реагування та загальна енергоефективність.

На етапі апаратного проєктування важливо досягнути балансу між функціональністю та вартістю компонентів. Система має бути технічно збалансованою, доступною для реалізації та подальшого розширення без повної зміни своєї структури. Апаратна архітектура Domovyk побудована на принципах модульності та сумісності, що дозволяє використовувати вузли з різними інтерфейсами зв’язку і при цьому забезпечує можливість їхньої заміни або оновлення у майбутньому. Такий підхід відповідає сучасним тенденціям розроблення IoT-рішень, у яких гнучкість системи є визначальним чинником її життєздатності.

Особливу увагу під час вибору апаратної платформи приділено інтеграції контролерів, сенсорів і виконавчих пристроїв у єдину комунікаційну мережу, що працює в реальному часі. У системі Domovyk передбачено незалежну роботу кількох вузлів, які взаємодіють через бездротові канали зв’язку, зберігаючи при цьому узгодженість дій і синхронізацію подій. Такий підхід забезпечує високу надійність і гнучкість системи, а також створює можливість поступового розширення її функціональності без зміни базової апаратної конфігурації.

Під час формування апаратної частини системи Domovyk визначальним завданням є вибір таких компонентів, які забезпечують стабільну роботу комплексу в умовах постійного обміну даними, взаємодії з користувачем та автономного реагування на події. До основних критеріїв, які застосовуються під час вибору апаратних засобів, належать продуктивність, сумісність, енергоефективність, надійність і вартісна доцільність. Ці параметри мають взаємозалежний характер, тому оптимізація одного з них не повинна негативно впливати на інші показники.

Продуктивність є ключовим фактором для мікроконтролерних систем, оскільки саме вона визначає здатність обробляти велику кількість подій у реальному часі. Контролер повинен забезпечувати достатню швидкодію для опитування сенсорів, передавання даних у мережі та формування відповідних команд виконавчим модулям. Високий рівень інтеграції дозволяє зменшити кількість додаткових периферійних компонентів, що сприяє компактності конструкції та зниженню енергоспоживання.

Сумісність компонентів має особливе значення для системи з модульною архітектурою. Кожен вузол повинен підтримувати стандартні інтерфейси обміну даними, що спрощує інтеграцію окремих частин у єдину мережу. Використання уніфікованих протоколів комунікації забезпечує коректну взаємодію між сенсорними, керуючими та виконавчими елементами. Завдяки цьому можлива заміна або оновлення окремих модулів без зміни логіки роботи системи.

Енергоефективність є важливою характеристикою для пристроїв, що працюють у цілодобовому режимі. Система повинна споживати мінімальну кількість енергії, особливо у стані очікування або при низькому навантаженні. Для

цього застосовуються контролери з енергозберігальними режимами роботи, а також оптимізовані алгоритми керування, які мінімізують непотрібну активність вузлів. Такий підхід знижує теплові втрати та продовжує строк служби пристроїв.

Надійність апаратних засобів безпосередньо впливає на безпеку користувача. Компоненти повинні функціонувати стабільно навіть за наявності коливань живлення або короткочасних збоїв зв’язку. Висока надійність досягається шляхом використання перевірених елементів, передбачення захисту від перевантажень та помилок передавання даних. Окрім цього, апаратна конфігурація має бути достатньо простою для обслуговування, що дозволяє швидко виявляти і усувати несправності.

Вартісна доцільність є заключним критерієм вибору апаратних засобів. Оскільки система призначена для практичного застосування у побуті, важливо, щоб її вартість залишалася доступною для споживача без втрати якості та функціональності. Вибір компонентів здійснюється з урахуванням співвідношення ціни до технічних характеристик, а також наявності цих елементів на ринку. Таким чином, формується оптимальний баланс між ефективністю, економічністю та технологічною сумісністю.

Апаратна частина системи Domovyk побудована на основі поєднання двох мікроконтролерів, які виконують різні, але взаємопов’язані завдання. Основним контролером обрано модуль ESP32 з платою розширення, який має достатню кількість цифрових і аналогових входів, підтримує бездротові протоколи Wi-Fi та Bluetooth, а також характеризується високою обчислювальною потужністю і низьким енергоспоживанням. Саме цей модуль виконує роль центрального процесора системи, забезпечує збирання даних із сенсорів, керування виконавчими пристроями і формування локального веб-сервера, який працює автономно без залучення хмарних сервісів. Використання ESP32 дозволяє об’єднати всі вузли системи в єдину локальну мережу, забезпечуючи повний контроль і синхронізацію подій у реальному часі. Загальний вигляд основного контролера з платою розширення подано на рисунку 2.5.

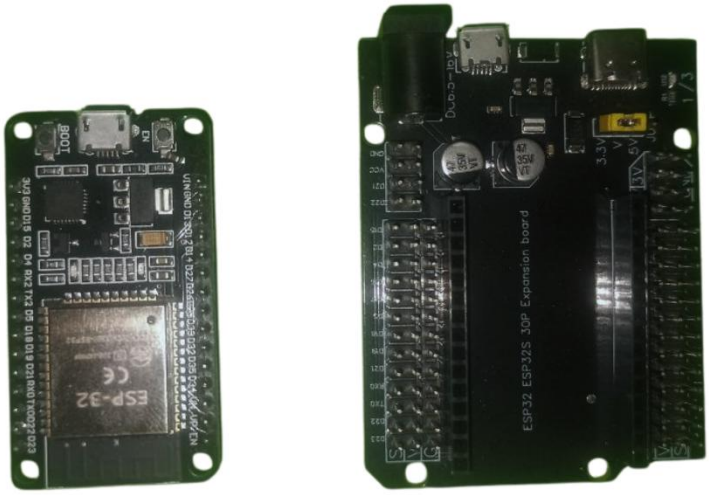

Рисунок 2.5 – Головний мікроконтролер системи Domovyk та карта доповнення до нього

Другим функціональним елементом є ESP32-CAM-MB (рисунок 2.6), який виконує роль окремого відеосервера. Цей модуль оснащено камерою OV2640 і має власний веб-інтерфейс для передачі відеопотоку безпосередньо у браузер користувача. Його застосування дає змогу розвантажити головний контролер, підвищити стабільність і швидкодію системи під час роботи з візуальними даними. ESP32-CAM функціонує у межах тієї ж Wi-Fi мережі, що й основний контролер, що спрощує інтеграцію і виключає потребу у додаткових комунікаційних адаптерах. Завдяки невеликим розмірам, низькому енергоспоживанню та доступній вартості цей модуль є оптимальним рішенням для реалізації базового відеоспостереження.

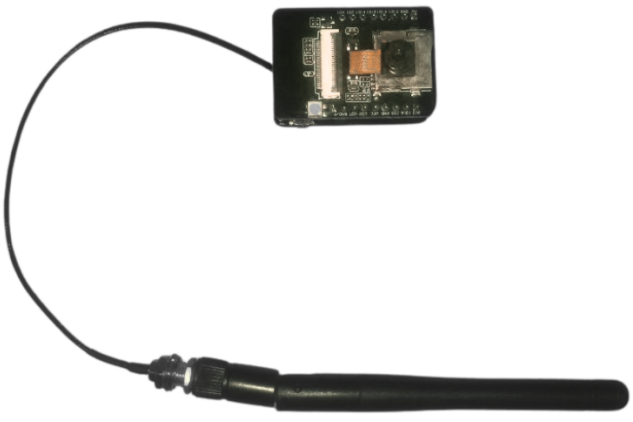

Рисунок 2.6 – Модуль з камерою ESP32-CAM-MB

Для зчитування параметрів середовища використано набір сенсорних вузлів, які забезпечують моніторинг ключових фізичних параметрів. Датчик SHT45 застосовується для вимірювання температури і вологості повітря завдяки високій точності, стабільності показів і цифровому інтерфейсу I2C, який спрощує інтеграцію з контролером. Для визначення концентрації газів використовується MQ-9, який реагує на чадний газ і метан, що дає змогу вчасно виявляти потенційно небезпечні умови в приміщенні. Виявлення руху реалізовано через HC-SR501, який працює на основі інфрачервоного принципу та має регульовану чутливість. Контроль стану дверей і вікон здійснюється за допомогою магнітних датчиків MC-38 з нормально розімкненим контактом, які фіксують відкривання або спроби несанкціонованого доступу. Усі вищезазначені датчики зображені на рисунку 2.7.

Рисунок 2.7 – Датчики SHT45, MQ-9, MC-38 та HC-SR501

До складу системи також входить сканер відбитків пальців R307, який забезпечує біометричну ідентифікацію користувачів, та RFID-модуль MFRC522, який дозволяє відкривати двері або ворота за допомогою безконтактних карток. Ці два елементи доповнюють функції безпеки, створюючи багаторівневу систему доступу. Для додаткового керування передбачено матричну клавіатуру 3 на 4 та дисплей LCD1602 з адаптером I2C, що використовуються для введення пароля, відображення коротких повідомлень і стану системи. Такий набір елементів забезпечує зручність користування навіть у разі відсутності підключення до мобільного веб-інтерфейсу. Візуально, сканер відбитку пальців R307, RFID-модуль MFRC522, матрична клавіатура та дисплей LCD1602 показано на рисунку 2.8.

Рисунок 2.8 – R307, MFRC522, матрична клавіатура та дисплей LCD1602

У ролі виконавчих елементів (рисунок 2.9) застосовуються серводвигуни MG90S 360°, які реалізують механізми відкривання і закривання воріт або дверей, вентилятор постійного струму номінальним струмом 0,1 А для вентиляції приміщення, бузер для сигналізації та світлодіодна стрічка RGBCW довжиною один метр, яка може змінювати колір відповідно до поточного режиму роботи системи. Для керування цими пристроями використовується MOSFET-модуль високого струму, що дозволяє безпечно комутувати навантаження великої потужності, зокрема вентилятор або стрічку, не перевантажуючи виходи мікроконтролера. Така схема забезпечує стабільну роботу системи і підвищує надійність усіх підключених елементів.

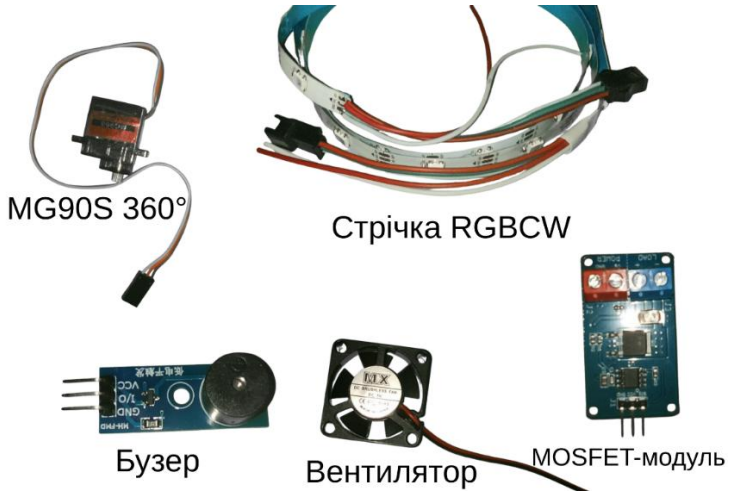

Рисунок 2.9 – Виконавчі елементи, які обрано для реалізації системи Domovyk

Важливою частиною комплексу є сенсорний дисплей ESP32-2432S028R (рисунок 2.10), який виконує роль локальної панелі керування. Він дозволяє відображати інформацію про стан системи, активувати чи деактивувати окремі функції та взаємодіяти з користувачем без необхідності відкривати веб-інтерфейс. Наявність власного мікроконтролера і Wi-Fi-з’єднання забезпечує двосторонній обмін даними між дисплеєм та основним контролером, що підвищує зручність і надійність експлуатації.

Рисунок 2.10 – Сенсорний дисплей ESP32-2432S028R

Для забезпечення стабільного живлення всієї системи використовується імпульсний блок живлення DC 5V 8A (рисунок 2.11), який гарантує стабільну подачу напруги та захист від перевантажень.

Рисунок 2.11 – Блок живлення DC 5V 8A

Загальна апаратна схема передбачає використання модульних підключень, що дає можливість змінювати конфігурацію системи відповідно до потреб користувача. Кожен обраний елемент відповідає принципам сумісності, надійності та енергоефективності, а також може бути використаний у подальшому для розширення функціональності комплексу. Такий підхід створює гнучку платформу, яка здатна працювати як демонстраційна модель, так і як основа для реальної системи безпеки житлового приміщення.

У результаті проведеного дослідження та аналізу технічних рішень сформовано оптимальний склад апаратних засобів, що забезпечують повну функціональність системи Domovyk. Вибрані компоненти дозволяють реалізувати розподілену архітектуру з двома незалежними контролерами, сенсорними вузлами, виконавчими пристроями та елементами інтерфейсу користувача. Така конфігурація гарантує стабільність роботи системи, гнучкість налаштування та можливість масштабування без суттєвих апаратних змін.

Проведене обґрунтування показало, що використання мікроконтролерів сімейства ESP32 забезпечує необхідний рівень обчислювальної потужності, енергетичну ефективність та підтримку сучасних мережевих технологій. Вибрані сенсори й виконавчі модулі повністю відповідають вимогам до побудови побутових IoT-систем: вони є точними, надійними та взаємно сумісними. Комплекс додаткових елементів, таких як RFID-зчитувач, сканер відбитків пальців, матрична клавіатура, дисплей LCD1602 та сенсорний екран ESP32-2432S028R, підвищує зручність користування і рівень безпеки, роблячи систему інтерактивною та адаптивною до потреб користувача.

Таким чином, апаратна платформа Domovyk поєднує високу функціональність, енергоефективність і технологічну гнучкість. Вона створює надійне підґрунтя для подальшої реалізації програмної архітектури та алгоритмів роботи системи. Розроблена конфігурація забезпечує можливість подальшого вдосконалення та розширення комплексу без порушення основних принципів побудови, що робить Domovyk перспективною основою для сучасних IoT-рішень у сфері розумних систем безпеки.

2.3 Розроблення програмної архітектури системи

Програмна архітектура є основою логічної взаємодії всіх модулів системи Domovyk і визначає спосіб реалізації закладених функцій. Її побудова спрямована на забезпечення стабільного обміну даними між апаратними вузлами, контролерами та інтерфейсом користувача, а також на підтримку синхронної роботи сенсорів, виконавчих пристроїв і серверних процесів. Структура проєкту спроєктована так, щоб дозволяти розширення системи, доповнення новими функціями і модернізацію без зміни основної логіки роботи.

У розробленні програмної частини системи Domovyk використано дві основні мови програмування: C++ (Arduino core для ESP32) для прошивок головного контролера та відеовузла, а також JavaScript для клієнтського вебінтерфейсу. C++ забезпечує контроль над пам’яттю, роботу з апаратними таймерами, перериваннями, мережевим стеком і дає змогу точно прогнозувати час виконання критичних ділянок коду. Її недоліком є складніша діагностика помилок із пам’яттю та вищий поріг входу, однак для систем реального часу з обмеженими ресурсами це виправданий компроміс. JavaScript обрано для фронтенду, оскільки він нативно підтримується браузерами, має подієву модель виконання і дозволяє реалізувати логіку WebSocket та динамічне оновлення елементів інтерфейсу без додаткових плагінів. Альтернативи на кшталт MicroPython для ESP32 або повністю безскриптових інтерфейсів були відхилені через більші затримки, вищі накладні витрати пам’яті та обмежену гнучкість при роботі з асинхронними подіями.

Метою розроблення програмної архітектури є створення модульної структури, яка об’єднує різнорівневі компоненти в єдине інтегроване середовище. Така архітектура має забезпечувати ефективну комунікацію між усіма елементами системи, а також реалізовувати механізми керування, моніторингу та взаємодії з користувачем у режимі реального часу. Програмне забезпечення Domovyk побудоване на основі асинхронної подієвої логіки, що дозволяє одночасно обробляти події від декількох вузлів, зменшуючи затримки і підвищуючи швидкодію системи.

Програмна архітектура системи Domovyk побудована як розподілена подієва структура, у якій кожен компонент виконує власну логічну функцію, зберігаючи при цьому взаємозв’язок із рештою модулів. Основний контролер відповідає за опитування сенсорів, керування виконавчими пристроями, оброблення даних і підтримку мережевої взаємодії. Відеосервер функціонує як незалежний вузол, який генерує відеопотік і передає його користувачеві без проміжної обробки на головному контролері. Така побудова мінімізує затримки під час передавання відео та дозволяє досягти більшої швидкодії під час реагування на події. Головний контролер і відеосервер працюють у спільній Wi-Fi мережі, підтримуючи синхронізацію даних у реальному часі.

Основна логіка головного контролера реалізована за допомогою асинхронного вебсервера, який підтримує два механізми комунікації: WebSocket для подій у реальному часі та REST API для отримання історичних або довідкових даних. Це дозволяє системі одночасно реагувати на миттєві події та обслуговувати запити користувача без конфліктів у роботі. Передавання даних відбувається у форматі JSON, що спрощує оброблення інформації на стороні клієнта. Оновлення показників датчиків, таких як температура, вологість, рівень газу або стан вентилятора, здійснюється через WebSocket, тоді як доступ до архіву вимірювань реалізовано через HTTP-запити. Таким чином розділяються короткі подійні процеси та тривалі операції читання історії, що зменшує затримки й підвищує стабільність системи.

Відеосервер ESP32-CAM працює окремо від головного контролера і формує власний HTTP-сервер для передачі зображення. Камера створює окремий потік даних, який транслюється у браузер користувача у вигляді JPEG-кадрів або відеопотоку в режимі реального часу. Така реалізація знижує навантаження на головний контролер, оскільки відеопередавання не конкурує за ресурси з іншими процесами. Для керування параметрами камери використовуються допоміжні маршрути, що дозволяють змінювати роздільну здатність, яскравість або активувати підсвічування. Завдяки ізоляції цього вузла досягається стійкість системи навіть у разі короткочасних перевантажень у роботі основного контролера.

Передавання даних у системі стандартизовано за допомогою структур JSON, де кожне повідомлення має визначений тип і набір параметрів. Це забезпечує зручність інтеграції нових компонентів і дозволяє виконувати тестування без зміни основного коду. Файлова система LittleFS використовується для зберігання статичних вебсторінок, що робить систему повністю автономною та незалежною від зовнішніх серверів. Обидва контролери працюють зі статичними IP-адресами, що полегшує їхнє виявлення та спрощує налагодження. Такий підхід забезпечує передбачувану поведінку системи та полегшує повторення експериментів під час дослідження ефективності.

Асинхронність є головним принципом архітектури Domovyk. Головний контролер опитує сенсори з певною періодичністю, накопичує дані у буферах і обробляє їх паралельно з іншими процесами. У той час відеосервер має власний цикл оброблення кадрів і не блокує роботу решти компонентів. Це дозволяє підтримувати стабільну реакцію системи навіть під час високого навантаження. Архітектура, створена на таких засадах, забезпечує масштабованість, швидкість і надійність роботи комплексу, а також відкриває можливість подальшого розширення без зміни його базової структури.

Архітектура головного контролера базується на модульній організації коду з розподілом відповідальностей за мережеві сервіси, сенсорні підсистеми, керування виконавчими елементами та обмін даними з інтерфейсом користувача. Програмний проєкт структуровано у середовищі PlatformIO. Основні файли розміщені у каталозі src, зокрема головний файл програми, допоміжні заголовки та модулі для роботи із сенсорами й виконавчими пристроями. Статичні ресурси вебінтерфейсу розміщені у файловій системі LittleFS у каталозі data. Для мережевої взаємодії використано бібліотеку асинхронного вебсервера, а для кодування повідомлень вузлів застосовано ArduinoJson. Комунікацію з периферією забезпечують стандартні бібліотеки WiFi, Wire для I2C та SPI або GPIO для локальних підключень. Для керування світлодіодною підсвіткою може застосовуватися спеціалізована бібліотека керування світлодіодними стрічками, а для генерації сигналів виконавців використовуються апаратні таймери контролера.

Життєвий цикл застосунку організовано через функції setup та loop. На етапі setup ініціалізуються послідовний інтерфейс для налагодження, мережевий стек Wi-Fi у режимі точки доступу або клієнта локальної мережі, файлову систему LittleFS з подальшою публікацією статичних ресурсів, асинхронний вебсервер із маршрутизаторами REST та канал WebSocket для подій реального часу. Тут же реєструються обробники команд від клієнта, налаштовуються інтервали опитування сенсорів, створюються циклічні буфери для збереження короткої історії вимірювань, ініціалізуються піни виконавчих пристроїв та встановлюються безпечні початкові стани. Функція loop працює неблокуюче: обслуговує таймери опитування, перевіряє наявність нових даних, виконує фільтрацію та перетворення вимірювань, формує повідомлення у форматі JSON для надсилання через WebSocket, а також обробляє внутрішні події менеджера станів системи охорони.

Логіка роботи з сенсорами передбачає періодичне опитування цифрових та аналогових каналів із урахуванням допустимого навантаження на шину і часу стабілізації перетворювачів. Для температури та вологості застосовується інтерфейс I2C, для газового сенсора використовується аналоговий вхід із подальшим програмним згладжуванням та приведенням до уніфікованої шкали, для датчиків руху і магнітних контактів аналізуються фронти зміни стану з антидребезгом у програмі. Оброблення вимірювань побудовано на простих математичних моделях, які дозволяють кількісно оцінити поведінку системи.

Для усунення випадкових стрибків значення застосовується або ковзне середнє, або експоненційне згладжування. Для ковзного середнього використовується вікно розміром m вимірювань, і розраховується за допомогою формули (2.1):

$$\bar{x}_t = \frac{1}{m}\sum_{i=0}^{m-1} x_{t-i}, \tag{2.1}$$

де $\bar{x}_t$ – згладжене значення;

m – розмір вимірювань вікна;

x_t – поточне «сире» значення сенсора.

Для експоненційного згладжування використовується рекурентне рівняння за формулою (2.2):

$$y_t = \alpha x_t + (1 - \alpha) y_{t-1}, \tag{2.2}$$

де y_t – згладжене значення за допомогою експоненційного рівняння;

α – коефіцієнт зглажування.

Коефіцієнт α підбирається експериментально і лежить в межах від нуля до одиниці. Менше α дає більш плавний графік, але збільшує час реакції системи; більше α зменшує затримку, але робить систему чутливішою до шуму. У ході експериментів у системі Domovyk значення α обиралося так, щоб час реакції на стрибок вологості не перевищував 1-2 періодів опитування, але графік залишався візуально стабільним (рисунок 2.12).

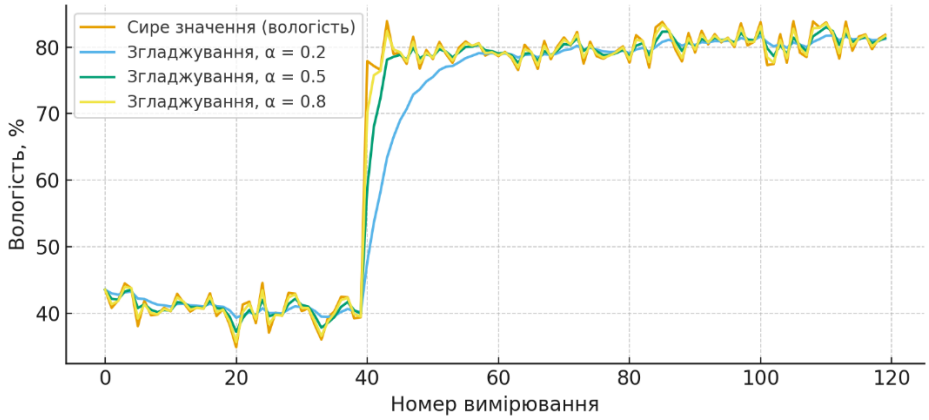

Рисунок 2.12 – Вплив коефіцієнта α на експоненційне зглажування

Для газового сенсора використовується модель порогового спрацювання з гістерезисом. Спершу виміряний сигнал x_t нормується до діапазону [0;1] за формулою (2.3):

$$x_t^{norm} = \frac{x_t - x_{min}}{x_{max} - x_{min}}, \tag{2.3}$$

де x_{min} та x_{max} – межі робочого діапазону, визначені на етапі калібрування.

Кожен вимір після фільтрації потрапляє у кільцевий буфер обмеженої довжини, що дозволяє без додаткової пам’яті зберігати коротку історію для побудови графіків у вебінтерфейсі. Вебклієнт отримує цю історію одним запитом через REST-маршрут або частинами через WebSocket. На основі цих буферів у третьому розділі роботи аналізуються затримки оновлення, частота оновлення графіків і вплив параметрів згладжування на швидкодію.

Керування виконавчими пристроями реалізовано через єдиний модуль, який приймає події від менеджера логіки і перетворює їх у конкретні дії. Для сервоприводів застосовується генерація імпульсів з урахуванням обмежень по куту та плавності ходу. Для вентилятора і світлодіодної стрічки використовується широтно-імпульсна модуляція або керування через силові ключі на окремих пинах, що дозволяє дозувати яскравість та швидкість. Звукова сигналізація активується короткими шаблонами, які не блокують основний цикл. Уся логіка підпорядковується поточному режиму охорони. Події від сенсорів, які мають критичний пріоритет, викликають негайну реакцію і паралельне інформування інтерфейсу користувача.

Обмін даними з вебінтерфейсом поєднує два підходи. Канал WebSocket використовується для миттєвих оновлень стану та телеметрії, включно з push-повідомленнями про спрацювання датчиків, зміну режиму охорони, положення виконавців і службові повідомлення. REST-маршрути застосовуються для доступу до агрегованих даних і історії, для завантаження сторінок інтерфейсу з LittleFS, а також для конфігурування параметрів. Повідомлення мають компактну JSON-структуру з явним типом події, часовою міткою та набором полів значень, що спрощує оброблення у клієнтському скрипті та полегшує налагодження під час дослідження затримок і пропускної здатності.

Для забезпечення надійності передбачено кілька механізмів. Стан вузлів зберігається у внутрішній таблиці, яка синхронізується з інтерфейсом при кожному підключенні клієнта. Критичні дії дублюються локально індикацією на дисплеї, що зменшує залежність від браузера. Обробники помилок у мережевих маршрутах повертають зрозумілі коди і короткі пояснення. Параметри опитування сенсорів та

інтенсивність оновлень можна коригувати під час експериментів, що зручно для вимірювань у розділі з тестуванням ефективності.

Схему програмної архітектури головного контролера показано на рисунку 2.13. На ній зображено взаємодію модулів збору даних, менеджера подій, підсистеми керування виконавцями, асинхронного вебсервера з каналом WebSocket та файлової системи LittleFS, а також напрямки потоків даних від сенсорів до буферів і далі до інтерфейсу користувача. Окремо виділено контури оброблення критичних подій охорони, які обходять довгі операції і одразу активують необхідні виконавці та повідомлення клієнту. Така архітектура забезпечує послідовність і передбачуваність роботи, а також створює резерв для подальшого масштабування програмного коду без зміни базової логіки контролера.

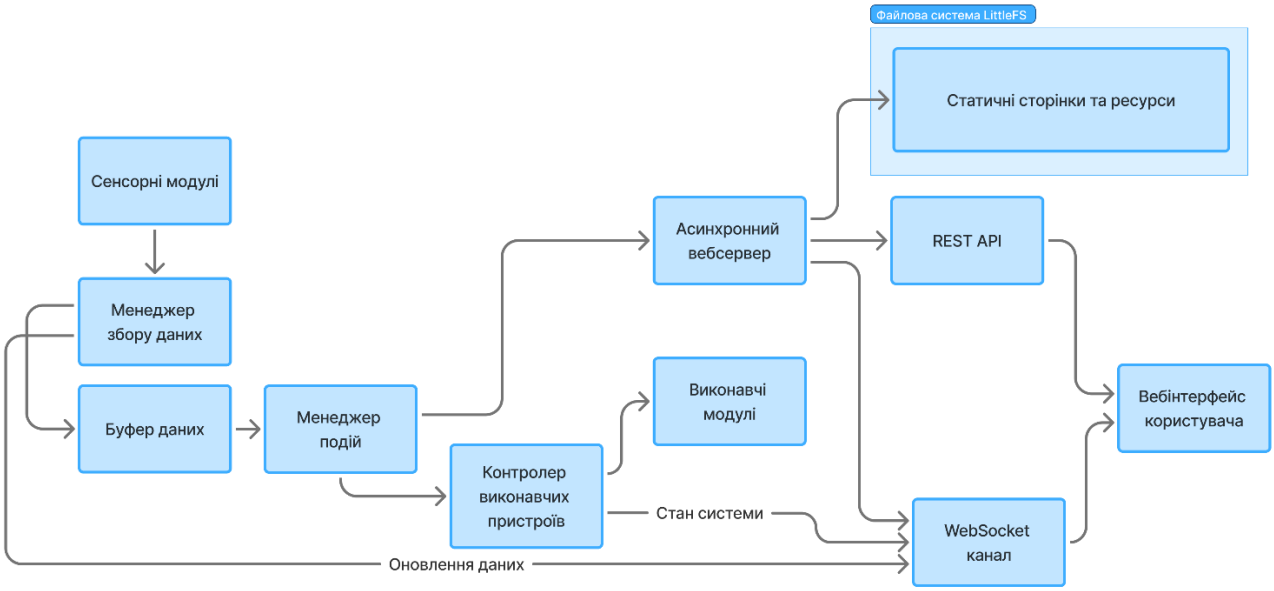

Рисунок 2.13 – Програмна архітектура головного контролера Domovyk

Відеосервер працює як незалежний вузол, що формує зображення з камери та обслуговує клієнтські запити через власний вебсервер. Ініціалізація сенсора виконується окремим блоком, після чого запускається оброблення кадрів і формування відповіді у вигляді послідовності JPEG або безперервного потоку з маркуванням меж кадрів. Логіка побудована так, щоб передавання відео не залежало від завантаження головного контролера. Керування параметрами зйомки виконується через службові маршрути з можливістю змінювати роздільну

здатність, експозицію або підсвічування. Така ізоляція забезпечує сталу частоту кадрів і зменшує ймовірність виникнення затримок під час паралельної роботи сенсорних підсистем.

Комунікація між вузлом камери та клієнтом відбувається без посередництва головного контролера. Браузер під'єднується до вебсервера камери за статичною адресою та отримує потік кадрів без додаткових перетворень. Для керування параметрами зйомки клієнт надсилає HTTP-запити на службові маршрути, відповіді яких мають компактний текстовий або JSON-формат. Така організація дає змогу розділити канали даних: телеметрія і керування передаються через головний контролер, а відео надходить до інтерфейсу безпосередньо. Це підвищує стійкість системи, оскільки короткочасні навантаження на один з вузлів не впливають на роботу іншого.

Клієнтська частина системи Domovyk реалізована у вигляді багатосторінкового вебдодатку з єдиним головним меню, яке виконує функцію навігаційного центру між усіма розділами системи. Основна сторінка побудована за принципом візуального панелювання, де користувачу подано набір інтерактивних плиток із піктограмами, що ведуть до відповідних функцій: камера, вологість, температура, ворота, двері, вентиляція, освітлення, газ і журнал подій. Кожна плитка є активним елементом керування, який відкриває окрему сторінку або модуль у межах системи. Такий підхід забезпечує інтуїтивне керування навіть на мобільних пристроях завдяки великій розмітці елементів та адаптивному дизайну.

Вебінтерфейс системи Domovyk виконано з використанням HTML, CSS і JavaScript, при цьому особлива увага приділена адаптивності та швидкодії. Дані між сторінками і головним контролером передаються у реальному часі через WebSocket-канал. Після завантаження сторінки відбувається автоматичне встановлення з'єднання з головним контролером, який надсилає оновлення стану пристроїв і сенсорів у форматі JSON. Статичні файли вебінтерфейсу зберігаються у файловій системі LittleFS, що забезпечує автономність роботи без необхідності зовнішнього підключення. Логіка роботи вебінтерфейсу (рисунок 2.14) передбачає

чітку послідовність обміну даними між клієнтською частиною, головним контролером та відеосервером.

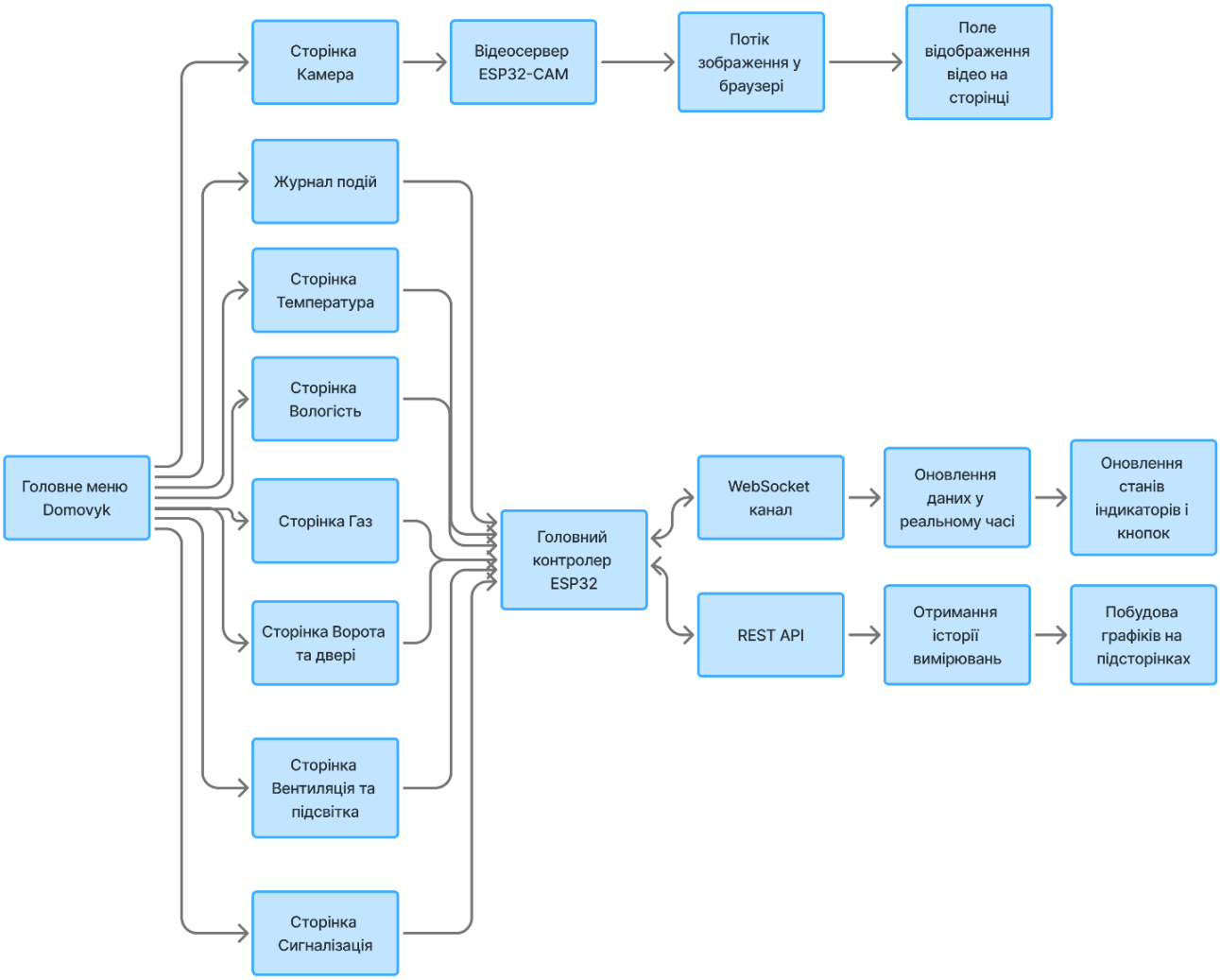

Рисунок 2.14 – Логічна схема вебінтерфейсу системи Domovyk

На головній сторінці розміщено окремий блок керування сигналізацією, який дозволяє вмикати або вимикати охоронний режим безпосередньо з вебінтерфейсу. Поточний стан сигналізації відображається кольором кнопки: зелений відповідає активному режиму, а червоний – вимкненому. Такий підхід забезпечує миттєву візуальну ідентифікацію стану системи без потреби переходу на додаткові сторінки. Візуальне оформлення головного меню витримане у спокійних тонах із використанням м’яких кольорів і тіней, що робить інтерфейс зручним для тривалої взаємодії.

Перехід між сторінками відбувається миттєво, без затримок або перезавантаження головної сторінки, що створює враження єдиного інтерактивного середовища. На підсторінках користувач може переглядати графіки, натискати кнопки для керування пристроями та отримувати сповіщення про стан системи у режимі реального часу. Кожна сторінка реалізує власну логіку обміну з сервером, використовуючи WebSocket для оперативних подій та HTTP для запитів до архівів і конфігурацій. Завдяки цьому вебінтерфейс забезпечує безперервну взаємодію з контролером і формує цілісне користувацьке середовище.

У результаті розроблення програмної архітектури системи Domovyk сформовано модульну, розподілену структуру, що забезпечує узгоджену взаємодію між усіма компонентами комплексу. Основний контролер і відеосервер функціонують як незалежні, але синхронізовані вузли, які обмінюються даними у межах локальної Wi-Fi мережі. Такий підхід дозволяє розділити обчислювальні навантаження, підвищити стабільність і швидкодію системи. Використання асинхронного вебсервера з підтримкою WebSocket і REST API забезпечує безперервне оновлення даних у реальному часі та гнучкість інтеграції нових функцій. Вебінтерфейс користувача реалізує зручний механізм керування і моніторингу, що робить систему інтуїтивною, швидкою у реагуванні й масштабованою. Таким чином, розроблена програмна архітектура є логічним продовженням апаратної платформи, формуючи комплексне, автономне та ефективне IoT-рішення для системи безпеки приватного будинку.

2.4 Моделювання алгоритмів роботи системи

Моделювання алгоритмів є ключовим етапом розроблення системи Domovyk, оскільки саме на цьому рівні визначається логіка взаємодії компонентів і послідовність виконання дій у різних режимах. Побудова алгоритмів дозволяє не лише перевірити коректність функціональних зв’язків між модулями, а й оцінити час реакції системи, стійкість до збоїв і ефективність оброблення подій. У ході дослідження застосовується підхід послідовного моделювання процесів, який дає змогу візуалізувати всі етапи роботи починаючи від отримання сигналу сенсором

до виконання команди виконавчим пристроєм і відображення результату у вебінтерфейсі.

У системі Domovyk передбачено кілька основних сценаріїв функціонування, які відображено у вигляді блок-схем. Перший сценарій – алгоритм роботи охоронного режиму (рисунок 2.15). Після активації сигналізації головний контролер переходить у режим постійного моніторингу датчику руху, магнітних контактів та біометричних модулів. При фіксації події контролер формує тривожний сигнал, активує звукове сповіщення та надсилає повідомлення у вебінтерфейс користувача. Якщо система перебуває в режимі «Охорона активна», подія обробляється негайно, і відповідний стан дублюється у локальній панелі управління. При цьому користувач має змогу вимкнути сигналізацію або переглянути журнал подій.

Рисунок 2.15 – Алгоритм роботи сигналізації

Другий сценарій – алгоритм керування доступом (рисунок 2.16). Цей процес починається із зчитування коду RFID або сканування відбитка пальця. Якщо автентифікація успішна, головний контролер формує команду на відкриття дверей чи воріт, активуючи відповідний серводвигун. Після завершення операції система автоматично повертає виконавчий пристрій у вихідне положення, а у вебінтерфейсі оновлюється повідомлення про статус дії. У разі невдалої перевірки контролер подає попереджувальний звуковий сигнал і записує подію до журналу спроб доступу.

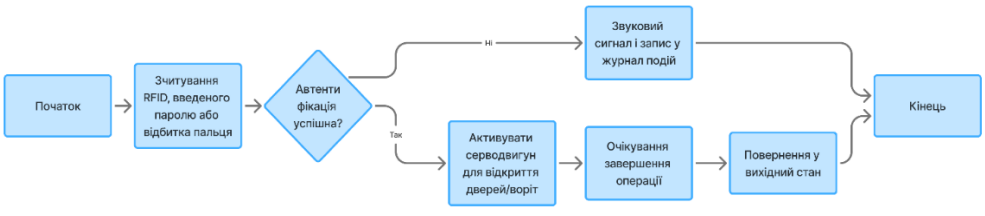

Рисунок 2.16 – Алгоритм керування доступом

Третій сценарій – алгоритм роботи середовищного моніторингу (рисунок 2.17). У цьому режимі контролер періодично опитує сенсори температури, вологості, газу та руху. Зібрані дані фільтруються і зберігаються у буфері для подальшої візуалізації. Якщо параметри перевищують допустимі межі, система автоматично активує вентиляцію або подає сигнал користувачеві. Цей механізм працює незалежно від охоронного режиму, забезпечуючи безперервний моніторинг навколишнього середовища.

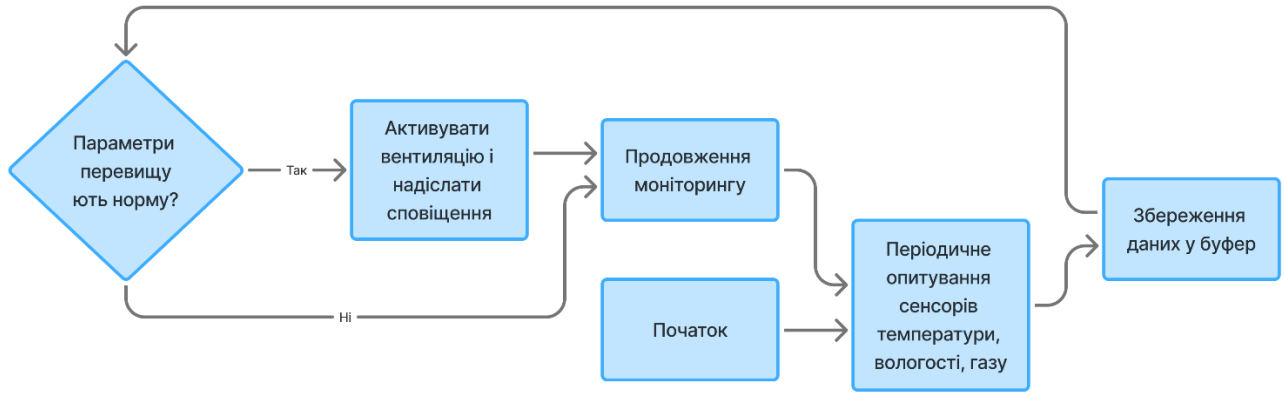

Рисунок 2.17 – Алгоритм моніторингу середовища

Моделювання підтвердило ефективність обраної архітектури, де головний контролер виступає центром оброблення подій, а інші вузли діють автономно в межах визначених сценаріїв. Розроблені алгоритми забезпечують узгоджену послідовність дій системи Domovyk, її стабільну роботу та можливість подальшої оптимізації за рахунок адаптації логіки під реальні умови експлуатації.

2.5 Забезпечення інформаційної безпеки та надійності системи

Інформаційна безпека є однією з важливих складових під час розроблення інтелектуальних IoT-систем, оскільки їхня структура передбачає постійний обмін даними між великою кількістю взаємопов’язаних вузлів. У системі Domovyk питання безпеки реалізовано на рівнях апаратного, програмного та комунікаційного захисту. Основна мета полягає у запобіганні несанкціонованому

доступу до системи, збереженні цілісності даних, захисті від збоїв та забезпеченні стабільності функціонування під час безперервної роботи.

На апаратному рівні безпека досягається шляхом ізоляції вузлів у межах локальної Wi-Fi мережі, створеної головним контролером. Кожен пристрій має унікальну MAC-адресу і підключається лише після аутентифікації за заздалегідь заданим SSID та паролем точки доступу. Це унеможливлює стороннє підключення до системи ззовні. Модулі ESP32 і ESP32-CAM працюють у межах приватного діапазону IP-адрес, а сам контролер не має виходу в Інтернет, що додатково підвищує захищеність. Для зниження ризику помилкових спрацювань у ланцюгах керування використано апаратні фільтри перешкод і стабілізоване живлення, яке гарантує коректну роботу навіть за короткочасних перепадів напруги.

На програмному рівні безпека забезпечується через обмеження доступу до критичних функцій системи. Усі команди, які змінюють стан охорони, відкривають двері або активують виконавчі пристрої, мають перевірку прав доступу. Для входу у вебінтерфейс передбачено базову авторизацію, а команди користувача передаються у форматі, який виключає виконання небажаних скриптів чи ін’єкцій. У разі повторної спроби підключення або збоїв система автоматично ініціює відновлення сесії, що запобігає втраті даних і блокуванню вузлів. Вебсервер фільтрує всі запити, розподіляючи їх за рівнем пріоритету, щоб уникнути перевантаження мікроконтролера.

Комунікаційна безпека базується на використанні асинхронного WebSocket протоколу, який дозволяє обмінюватися даними лише з перевіреними клієнтами. Повідомлення, що передаються, мають унікальні службові теги, за якими контролер ідентифікує джерело команди. Передбачено механізм перевірки коректності даних, який включає контроль довжини пакета і перевірку контрольної суми. Це дозволяє виявляти пошкоджені або підроблені повідомлення ще до їх виконання. Усі канали зв’язку працюють у межах внутрішньої Wi-Fi мережі, що зменшує ризик перехоплення даних із зовнішнього середовища.

Надійність роботи системи Domovyk забезпечується за рахунок резервування критичних процесів і використання механізмів самодіагностики. Головний контролер періодично перевіряє працездатність сенсорних вузлів та стан

мережевого з’єднання. Якщо будь-який пристрій не відповідає протягом певного часу, система фіксує подію як помилку і надсилає повідомлення у вебінтерфейс. Завдяки циклічному буферу даних останні результати вимірювань залишаються доступними навіть у разі короткочасних збоїв зв’язку. На рівні прошивки реалізовано механізм автоматичного перезапуску контролера після критичних збоїв, що дозволяє швидко відновлювати роботу без втручання користувача.

Додатково підвищено захищеність на фізичному рівні, а саме у моделі будинку передбачено окремі канали живлення для контролерів і виконавчих модулів, що мінімізує ризик виходу з ладу через коротке замикання або стрибки напруги. Система працює у цілодобовому режимі без необхідності постійного підключення до зовнішніх серверів, тому залежність від мережевої інфраструктури зведено до мінімуму.

Загалом комплекс Domovyk демонструє багаторівневий підхід до забезпечення інформаційної безпеки й надійності. Поєднання апаратних, програмних і комунікаційних засобів захисту створює стійку систему, здатну функціонувати автономно, оперативно реагувати на події та запобігати несанкціонованим втручанням. Така архітектура відповідає сучасним вимогам до безпечних IoT-рішень і може бути розширена новими методами захисту без зміни базової логіки роботи комплексу.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ ТА АНАЛІЗ РОБОТИ РОЗРОБЛЕНОЇ СИСТЕМИ

3.1 Реалізація апаратної частини системи Domovyk

Реалізація апаратної частини системи Domovyk є практичним етапом, у якому розроблена архітектура набуває фізичної форми. Її мета полягає у створенні діючого прототипу, який підтверджує працездатність запропонованих технічних і програмних рішень. Система побудована на основі модульного підходу, що дозволяє легко масштабувати й модифікувати її склад без зміни основної структури. Усі апаратні вузли інтегровані у тривимірну модель будинку, роздруковану на 3D-принтері, що дозволяє наочно продемонструвати роботу комплексу безпеки та керування в реальному середовищі.

Основою системи є головний контролер ESP32, який виконує функції збору даних із сенсорів, керування виконавчими пристроями, обробки подій та комунікації з вебінтерфейсом. Другим важливим елементом є ESP32-CAM, що працює як окремий вузол відеоспостереження та забезпечує передачу потоку зображення у вебдодаток користувача. До складу комплексу входять також сенсор температури та вологості SHT45, газовий сенсор MQ-9, датчик руху HC-SR501, магнітні контакти MC-38, рідкокристалічний дисплей LCD1602 з адаптером I2C, RFID-модуль MFRC522, матрична клавіатура 3 на 4, сканер відбитків пальців R307, серводвигун MG90S 360°, світлодіодна стрічка RGB, звуковий бузер, вентилятор постійного струму та сенсорний дисплей ESP32-2432S028R, який використовується як локальна панель керування.

Під час побудови апаратної конфігурації особливу увагу приділено маршрутизації живлення та розділенню сигнальних і силових ліній, що дозволило мінімізувати електромагнітні завади та підвищити стабільність роботи сенсорів. Для кожного модуля було визначено оптимальне розташування всередині моделі будинку.

Загальна схема підключення компонентів подана на рисунку 3.1, де показано взаємозв’язки між контролерами, сенсорами, виконавчими модулями та допоміжними пристроями.

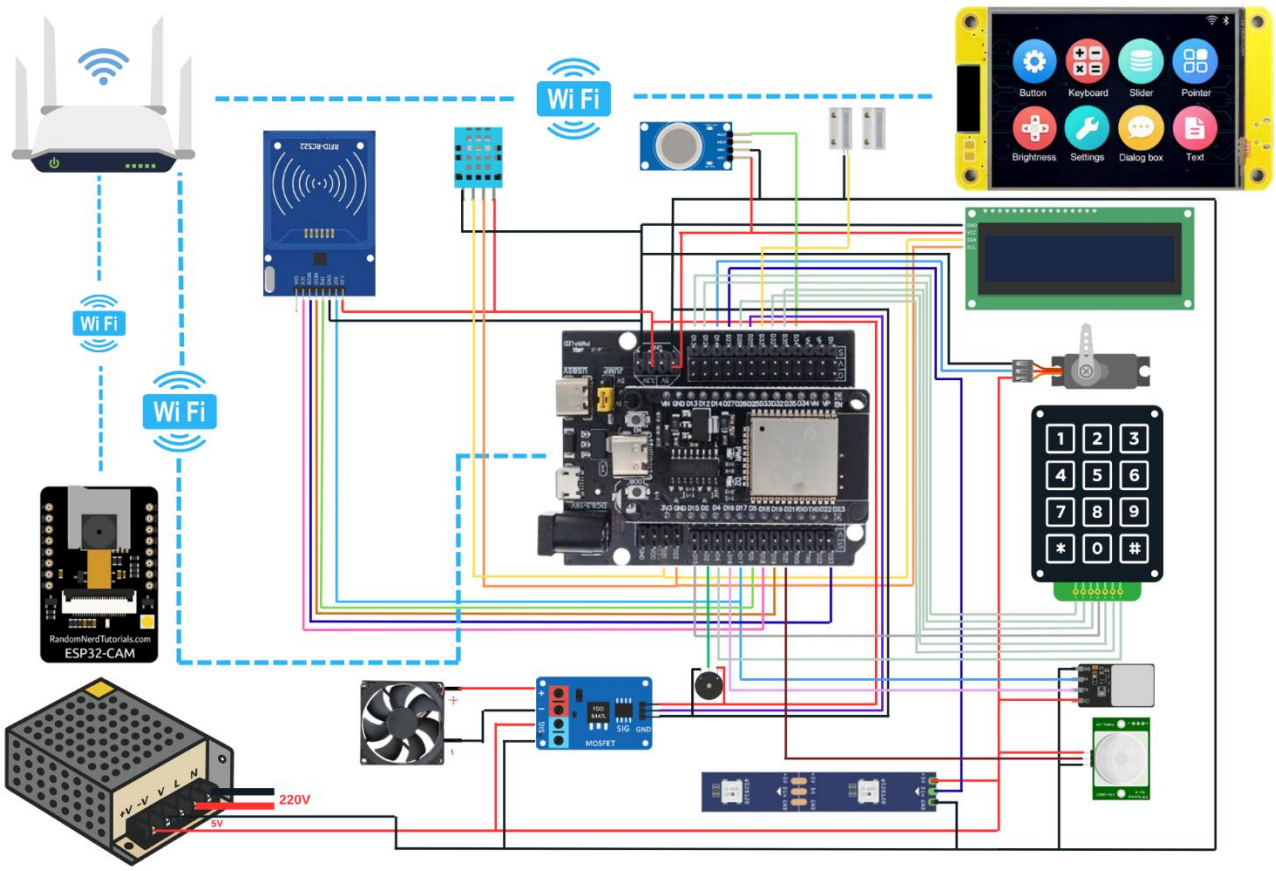

Рисунок 3.1 – Схема підключення компонентів у системі Domovyk

Взаємодія між вузлами організована через цифрові інтерфейси ESP32. Сенсори температури та вологості SHT45 і дисплей LCD1602 підключені до спільної шини I²C, де контакт SDA під'єднано до GPIO21, а SCL до GPIO22. Це забезпечує одночасну передачу даних без конфліктів між пристроями. Модуль RFID MFRC522 працює через інтерфейс SPI, у якому контакт SCK підключений до GPIO18, MOSI до GPIO23, MISO до GPIO19, CS до GPIO5, а RST до GPIO17. Газовий сенсор MQ-9 з'єднано з аналоговим входом GPIO34 через резистор, який стабілізує рівень сигналу та зменшує шум. Датчик руху HC-SR501 підключений до GPIO25, а магнітні датчики MC-38 приєднані до окремих цифрових входів, що дозволяє фіксувати момент відкривання дверей або вікон.

Сервоприводи MG90S отримують сигнал керування з GPIO14, при цьому живлення подається безпосередньо від блока живлення 5 В із загальною землею. Клавіатура 3×4 підключена до портів GPIO4, GPIO12, GPIO13, GPIO15, GPIO26, GPIO32 та GPIO33. Світлодіодна стрічка керується з GPIO27, а звуковий бузер активується через окремий цифровий вихід. Сенсорний дисплей ESP32-2432S028R

функціонує як клієнт у локальній мережі, забезпечуючи двосторонній обмін даними з головним контролером. У таблиці 3.1 наведено відповідність між компонентами та контактами мікроконтролера.

Таблиця 3.1 – Особливості підключення компонентів у системі Domovyk

№	Назва модуля або сенсора	Призначення	Контакти ESP32	Тип з'єднання / інтерфейс	Напруга живлення
1	2	3	4	5	6
1	SHT45	Вимірювання температури та вологості	SDA – GPIO21, SCL – GPIO22	I2C	3,3 В
2	LCD1602 I²C	Відображення повідомлень і станів системи	SDA – GPIO21, SCL – GPIO22	I2C (спільна шина із SHT45)	5 В
3	RFID MFRC522	Зчитування безконтактних карт доступу	SCK – GPIO18, MOSI – GPIO23, MISO – GPIO19, CS – GPIO5, RST – GPIO17	SPI	3,3 В
4	MQ-9	Контроль рівня газу (CO, CH_4)	AOUT – GPIO34	Аналоговий вхід	5 В
5	HC-SR501	Виявлення руху	SIG – GPIO25	Цифровий вхід	5 В
6	MC-38	Контроль відкривання дверей/вікон	SIG – окремі цифрові входи	Цифровий вхід	3,3 В
7	MG90S 360°	Керування дверима та воротами	PWM – GPIO14	PWM / керуючий сигнал	5 В
8	Клавіатура 3×4	Введення коду доступу	GPIO4, GPIO12, GPIO13, GPIO15, GPIO26, GPIO32, GPIO33	Цифровий вхід/вихід	5 В
9	RGB-стрічка	Індикація режимів роботи	GPIO27	PWM / керуючий сигнал	5 В
10	Бузер	Звуковий сигнал тривоги	GPIO визначений у прошивці	Цифровий вихід	5 В
11	Вентилятор постійного струму	Вентиляція приміщення	GPIO через MOSFET модуль	Цифровий вихід / силова комутація	5 В
12	ESP32-CAM	Передача відео у вебінтерфейс	Wi-Fi мережа (окремий вузол)	Бездротовий зв'язок	5 В
13	ESP32-2432S028R	Сенсорна панель керування	Wi-Fi (клієнт у локальній мережі)	Бездротовий зв'язок	5 В
14	Блок живлення	Живлення системи	Вихід +5V, GND (спільна земля)	Енергоживлення	240 В / 5 В

Живлення всієї системи забезпечується стабілізованим імпульсним блоком постійного струму на 5 В із вихідним струмом до 8 А. На вхід блока подається

напруга 240 В, а вихід використовується для живлення всіх вузлів системи. До блока підключено головний контролер, сенсори, виконавчі пристрої та допоміжні модулі. Спільна земля з’єднана для всіх компонентів, що гарантує узгодженість рівнів сигналів і стабільність роботи навіть за підвищеного навантаження. Для уникнення перешкод у колах керування силові елементи живляться напряму від блока, а сигнальні схеми ізольовані через транзисторні ключі MOSFET.

Для демонстрації функціональності системи створено тривимірну модель будинку, що складається з трьох частин. Перша частина (рисунок 3.2) – це корпус-коробка, у якій розміщено головний контролер, блок живлення, усі розгалуження проводів і серводвигуни, орієнтовані вертикально для відкривання дверей та воріт. Друга частина (рисунок 3.3) – це кришка з отворами для виходу проводів до сенсорів і виконавчих елементів. Третя частина (рисунок 3.4) – це конструкція стін будинку, у якій передбачено посадкові місця для розміщення сенсорів газу, руху, магнітних датчиків та камери ESP32-CAM.

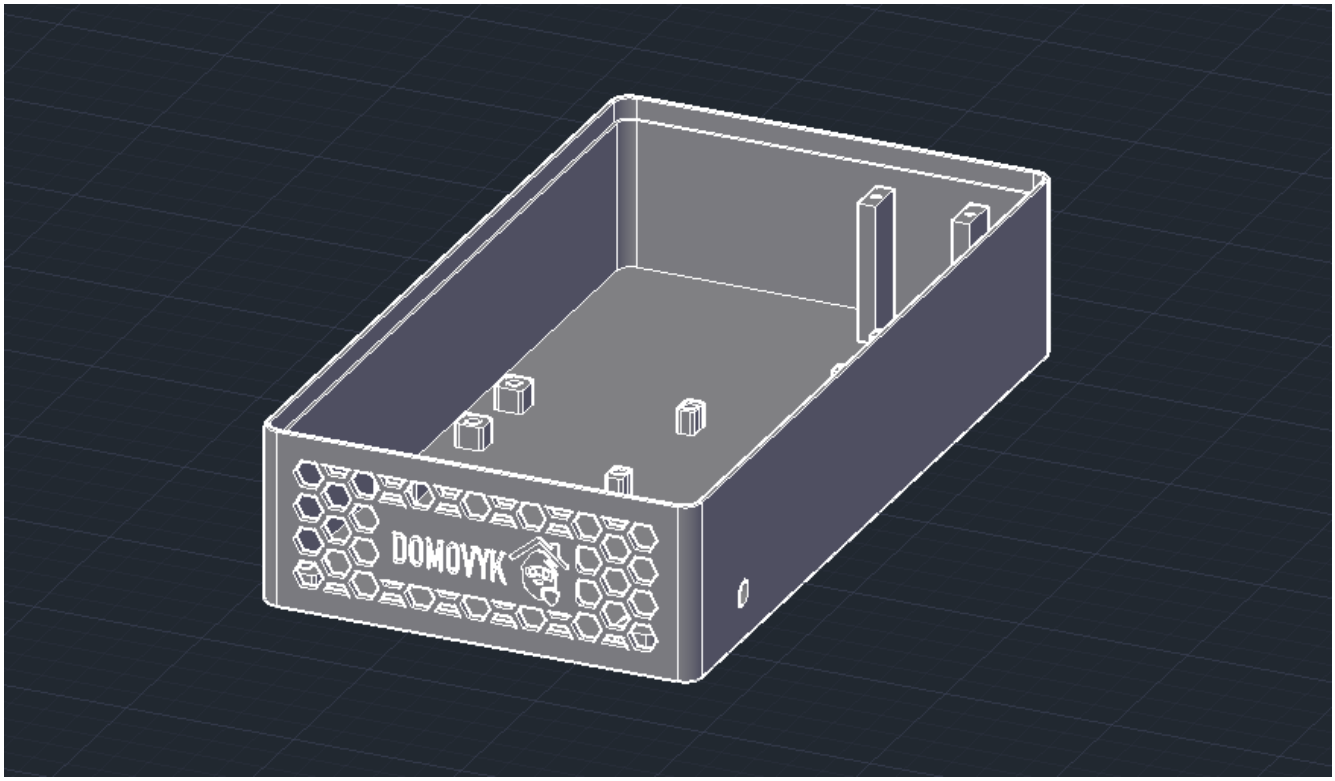

Рисунок 3.2 – Корпус-коробка з головним контролером, блоком живлення та сервоприводами системи Domovyk

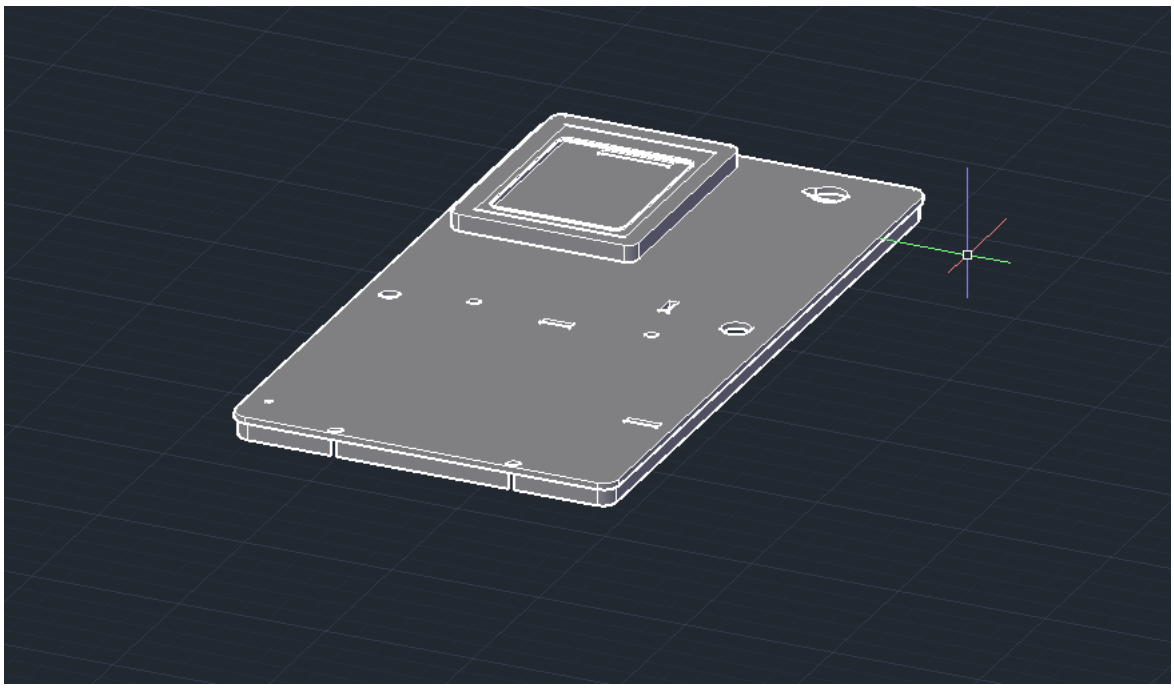

Рисунок 3.3 – Кришка корпусу з технологічними отворами для підключення сенсорів і виконавчих пристроїв

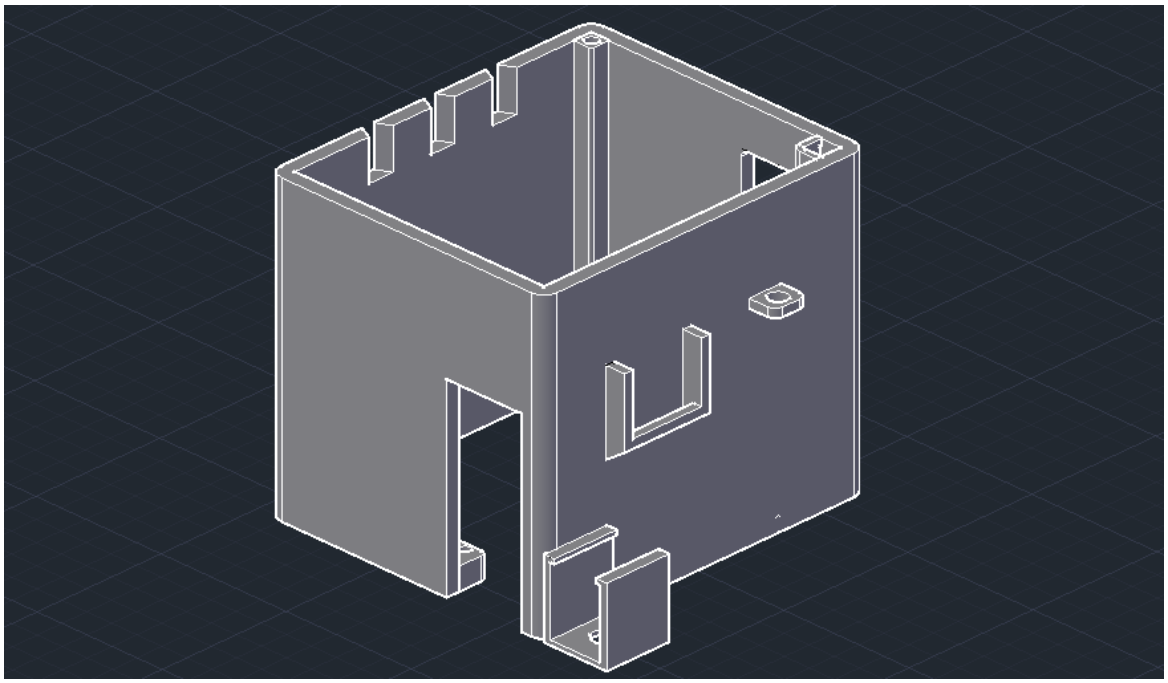

Рисунок 3.4 – Конструкція стін будинку з передбаченими посадковими місцями для сенсорів і камери ESP32-CAM

Для локального відображення станів системи використовується дисплей LCD1602, який показує повідомлення про введення пароля та підтвердження доступу. Сенсорний дисплей ESP32-2432S028R, інтегрований у систему, дозволяє

дублювати інформацію та виконувати керування режимами без використання вебінтерфейсу. Його екран забезпечує зручну взаємодію з користувачем і швидкий доступ до основних функцій.

Монтаж компонентів у модель проводився поетапно, з урахуванням зручності обслуговування, прокладання проводів і забезпечення доступу до основних вузлів. Спочатку в корпус-коробку було встановлено головний контролер ESP32, блок живлення та сервоприводи. Кожен модуль розташовано так, щоб мінімізувати довжину провідників і зменшити рівень можливих перешкод у сигнальних лініях. Проводи живлення і даних зібрано у спільні джгути, зафіксовані всередині корпусу пластиковими стяжками, що забезпечує акуратність і безпечність монтажу. На рисунку 3.5 показано внутрішнє розташування елементів у коробці керування.

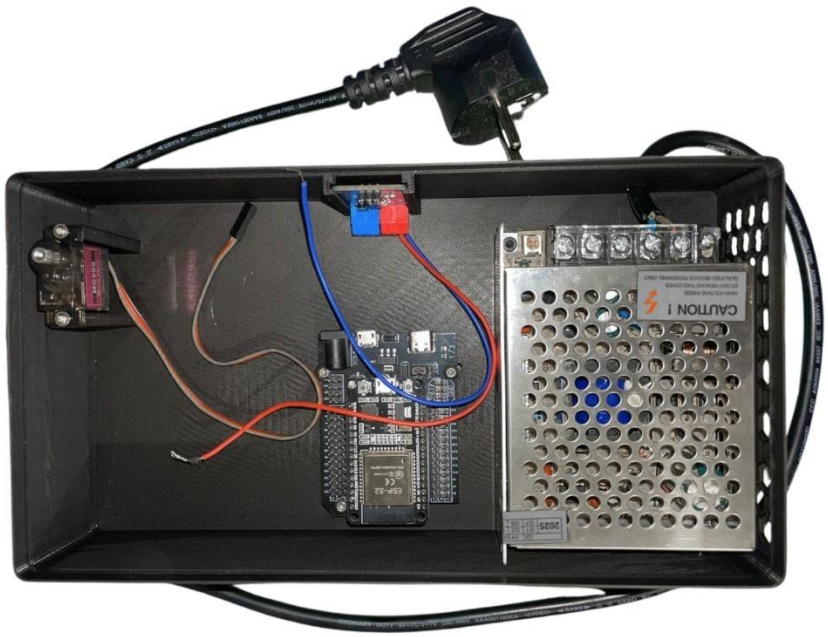

Рисунок 3.5 – Внутрішнє розташування основних модулів у корпусі системи Domovyk

Після цього було виконано прокладання проводів до сенсорів і виконавчих пристроїв через отвори в кришці корпусу. Кожен кабель позначено кольоровими мітками для спрощення підключення та подальшого обслуговування. Провідники,

що з’єднують контролер із сервоприводами, виведено окремо для зменшення електромагнітного впливу на сигнальні лінії. Роз’ємне підключення до сенсорів забезпечує швидку заміну або модернізацію окремих елементів без необхідності демонтажу всієї конструкції.

Монтаж сенсорів у стіни моделі проводився після завершення внутрішніх з’єднань (рисунок 3.6). Усі сенсорні модулі було встановлено у заздалегідь підготовлені посадкові місця, що забезпечує їх фіксацію і правильну орієнтацію у просторі. Камеру ESP32-CAM закріплено у верхній частині моделі, що дозволяє охоплювати найбільшу частину умовного приміщення. Газовий сенсор розташовано ближче до основи, оскільки важкі гази накопичуються у нижніх шарах повітря, а датчик руху встановлено на рівні середини стіни для оптимальної чутливості. Магнітні датчики MC-38 розміщені біля макетного вікна для демонстрації охоронних функцій.

Рисунок 3.6 – Монтаж сенсорів і виконавчих елементів у конструкцію стін моделі будинку

Фінальним етапом стало підключення допоміжних елементів, таких як світлодіодна стрічка, вентилятор і бузер. Світлодіодну стрічку розміщено у верхній

частині макету для рівномірного розсіювання світла, а вентилятор встановлено у нижній частині для імітації вентиляційного каналу. Усі елементи підключено через транзисторні ключі MOSFET, що забезпечують безпечну комутацію навантаження. Після завершення монтажу система була протестована на стабільність живлення, цілісність з’єднань та відсутність коротких замикань. На рисунку 3.7 подано зібрану модель із повністю змонтованими елементами.

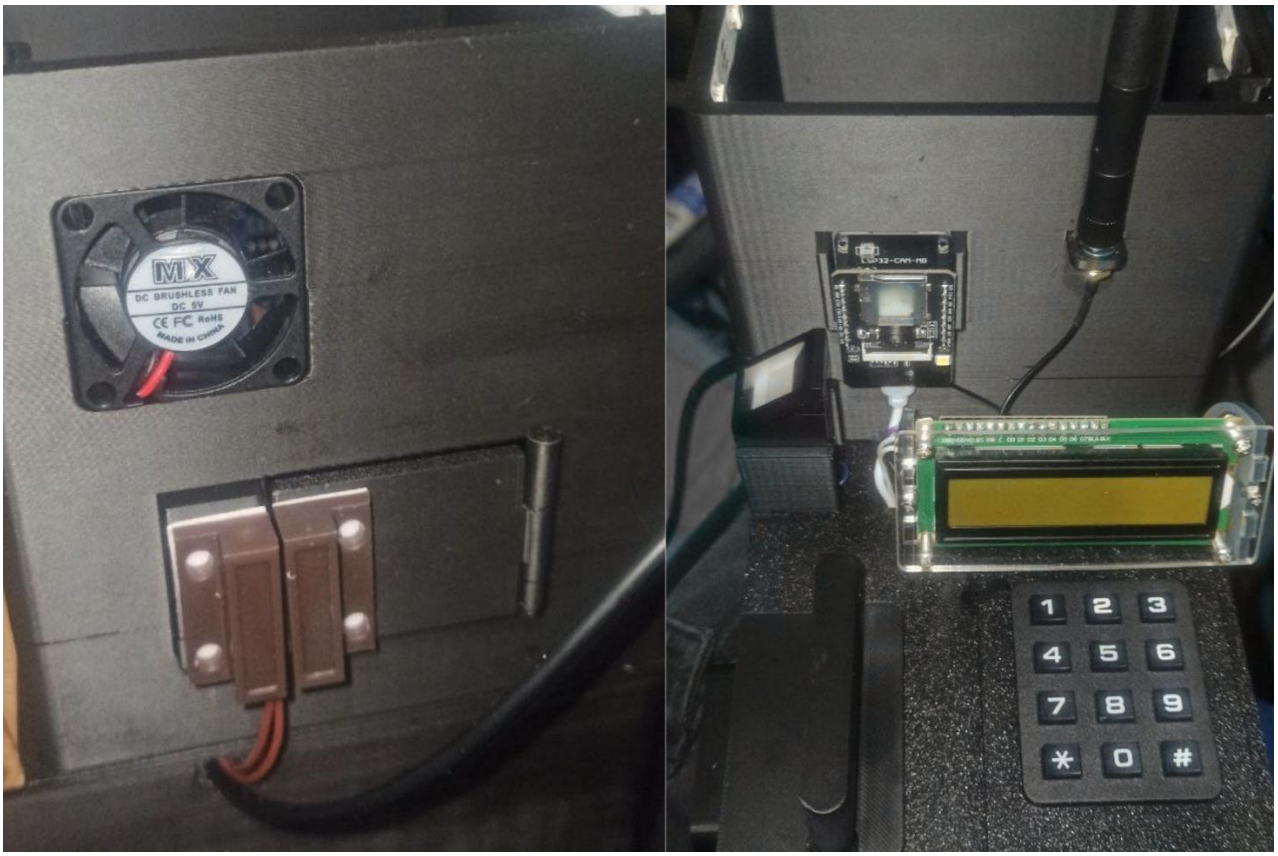

Рисунок 3.7 – Зібрана модель системи Domovyk із повністю змонтованими компонентами

Під час складання моделі проводилися перевірки стабільності живлення та рівнів сигналів, що дозволило визначити оптимальне розміщення сервоприводів і сенсорів. У процесі експериментів з реальним прототипом уточнено довжини провідників і схему розведення для мінімізації перешкод

У процесі реалізації апаратної частини системи Domovyk створено повноцінний прототип, який підтверджує практичну можливість реалізації спроєктованої архітектури. Усі компоненти зібрано в єдиний комплекс, що

забезпечує стабільну роботу сенсорів, виконавчих пристроїв і контролерів. Конструкція 3D-моделі дозволяє легко проводити обслуговування та розширення системи, а організація живлення і зв’язків забезпечує безпеку та надійність функціонування. Зібраний прототип повністю відповідає вимогам до сучасних IoT-рішень і є основою для подальшого аналізу програмної частини та тестування ефективності роботи комплексу.

3.2 Реалізація програмного забезпечення контролера

Програмне забезпечення головного контролера є центральним елементом системи Domovyk. Воно забезпечує обмін даними між усіма апаратними модулями, виконує оброблення сигналів сенсорів, керує виконавчими пристроями та організовує зв’язок із вебінтерфейсом користувача. Розроблення проводилося у середовищі PlatformIO з використанням фреймворку Arduino для мікроконтролера ESP32.

Основна увага приділялася стабільності роботи в умовах багатозадачності, мінімізації затримок під час обміну повідомленнями та забезпеченню модульності коду. Логіка реалізована у вигляді набору незалежних функцій, які взаємодіють через асинхронні виклики.

Додатково реалізовано механізми контролю стану вузлів і періодичного самообслуговування, що включають перевірку доступності сенсорів, відновлення мережевих з’єднань та автоматичне перезапускання окремих підсистем у разі помилок. Такий підхід дозволив створити стійку до збоїв програмну інфраструктуру, здатну підтримувати безперервну роботу навіть за умов пікового навантаження або короткочасних втрат зв’язку.

На початковому етапі відбувається ініціалізація файлової системи LittleFS і підняття локальної точки доступу Wi-Fi. Контролер виступає у ролі вебсервера, який самостійно обслуговує сторінки користувацького інтерфейсу.

Лістинг 3.1 показує запуск Wi-Fi, монтування файлової системи та створення асинхронного сервера для обміну даними з клієнтом. Це дозволяє системі функціонувати без зовнішньої інфраструктури і бути повністю автономною.

Лістинг 3.1 – Ініціалізація Wi-Fi, LittleFS і підняття асинхронного вебсервера

```
#include <WiFi.h>
#include <ESPAsyncWebServer.h>
#include <LittleFS.h>
#include <ArduinoJson.h>

AsyncWebServer server(80);
AsyncWebSocket ws("/ws");

const char* ssid = "Domovyk_AP";
const char* pass = "********";

void setup() {
  Serial.begin(115200);

  if (!LittleFS.begin()) {
    Serial.println("LittleFS error");
    while (true) delay(100);
  }

  WiFi.mode(WIFI_AP);
  WiFi.softAP(ssid, pass);

  server.serveStatic("/", LittleFS, "/").setDefaultFile("index.html");
  server.addHandler(&ws);
  server.begin();
}

void loop() {}
```

кінець лістингу 3.1

Після запуску система починає опитування сенсорів. Дані з SHT45 зчитуються через шину I2C і проходять згладжування для усунення випадкових коливань. Як показано в лістингу 3.2, використовується метод експоненційного середнього значення, який робить покази стабільнішими.

Сенсор MQ-9 вимірює концентрацію газів, і його аналоговий сигнал перетворюється у цифрове значення. У лістингу 3.3 наведено приклад простої функції порівняння із заданим порогом, що дозволяє визначати момент перевищення допустимого рівня газу та передавати відповідне повідомлення у менеджер подій. Додатково реалізовано механізм фільтрації шумів за допомогою ковзного середнього, що підвищує стабільність показників та зменшує хибні спрацювання. У разі тривалого перевищення порогу модуль формує розширене попередження, яке виводиться у мобільному інтерфейсі. Такий підхід забезпечує надійну роботу підсистеми контролю газу та підвищує загальну безпеку IoT-комплексу.

Лістинг 3.2 – Опитування SHT45 по I2C з базовим згладжуванням

```
#include <Wire.h>
float t_avg = 0.0f, h_avg = 0.0f;
const float alpha = 0.2f;   // коефіцієнт EMA

bool readSHT45(float& t, float& h); // реалізація у додатку

void pollEnv() {
  float t, h;
  if (readSHT45(t, h)) {
    t_avg = alpha * t + (1 - alpha) * t_avg;
    h_avg = alpha * h + (1 - alpha) * h_avg;
  }
}
```

кінець лістингу 3.2

Лістинг 3.3 – Зчитування MQ-9 з приведенням до шкали та пороговою сигналізацією

```
const int MQ9_PIN = 34;
const int GAS_THRESHOLD = 620;

int gasRaw() { return analogRead(MQ9_PIN); }

bool gasAlarm() {
  int v = gasRaw();
  return v >= GAS_THRESHOLD;
}
```

кінець лістингу 3.3

Для фіксації спрацьовування магнітних контактів МС-38 використано антидребезговий алгоритм, який дозволяє уникнути помилкових спрацьовувань. У лістингу 3.4 представлено логіку зчитування сигналів та зміну станів охоронного режиму.

Коли система перебуває у режимі охорони, будь-яке спрацювання датчика руху або відкривання дверей переводить її у стан тривоги. Така схема забезпечує чітке відокремлення подій та дозволяє швидко реагувати на загрози. Додатково реалізовано часову валідацію станів, що унеможливлює випадкові переходи режимів у разі короткочасних перешкод або нестабільних зчитувань. Після підтвердженого спрацювання система генерує подію тривоги, яка передається у центральний модуль для подальшої обробки та активації сирени, яка повідомляє користувача про порушення безпеки. Такий підхід підвищує надійність охоронної

підсистеми та забезпечує коректне функціонування навіть у умовах зашумленого середовища.

Лістинг 3.4 – Оброблення магнітного датчика та стан машини охорони

```
enum class GuardState { Off, Armed, Alarm };
GuardState gstate = GuardState::Off;

const int REED_PIN = 33;
const int PIR_PIN  = 25;
const unsigned long debounceMs = 30;

bool readReed() {
  static int last = HIGH;
  static unsigned long ts = 0;
  int now = digitalRead(REED_PIN);
  if (now != last && millis() - ts > debounceMs) {
    last = now; ts = millis();
    return now == LOW;
  }
  return false;
}

void guardTick() {
  bool door = readReed();
  bool motion = digitalRead(PIR_PIN) == HIGH;

  if (gstate == GuardState::Armed && (door || motion)) {
    gstate = GuardState::Alarm;
  }
}
```

кінець лістингу 3.4

Керування сервоприводами, вентилятором та світлодіодною стрічкою реалізовано через апаратні канали ШІМ.У лістингу 3.5 показано приклад керування сервоприводом, який відкриває або закриває двері залежно від отриманої команди.

Лістинг 3.5 – Керування сервоприводом для воріт

```
#include <ESP32Servo.h>
Servo gate;
const int SERVO_PIN = 14;

void actuatorsInit() {
  gate.attach(SERVO_PIN);
  gate.write(90);
}

void gateOpen()  { gate.write(10); }
void gateClose() { gate.write(170); }
```

кінець лістингу 3.5

Лістинг 3.6 демонструє регулювання швидкості вентилятора та яскравості підсвічування. Таке рішення дозволяє економно використовувати енергію та адаптувати режим роботи системи до поточних умов.

Лістинг 3.6 – Керування вентилятором і стрічкою через MOSFET

```
const int FAN_PIN  = 16;
const int LED_PIN  = 27;

void pwmInit() {
  ledcSetup(0, 1000, 8);
  ledcAttachPin(FAN_PIN, 0);
  ledcSetup(1, 1000, 8);
  ledcAttachPin(LED_PIN, 1);
}

void setFan(uint8_t duty) { ledcWrite(0, duty); }
void setLed(uint8_t duty) { ledcWrite(1, duty); }
```

кінець лістингу 3.6

Основою комунікації між контролером і браузером користувача є двосторонній канал WebSocket, який забезпечує передачу даних у реальному часі. У лістингу 3.7 наведено приклад формування JSON повідомлень, які містять температуру, вологість, стан охорони та показники газу.

Лістинг 3.7 – Відправлення телеметрії через WebSocket у форматі JSON

```
void wsBroadcastEnv() {
  StaticJsonDocument<256> doc;
  doc["t"] = t_avg;
  doc["h"] = h_avg;
  doc["gas"] = gasRaw();
  doc["guard"] = (int)gstate;

  String out;
  serializeJson(doc, out);
  ws.textAll(out);
}
```

Кінець лістингу 3.7

Прийом команд керування від користувача показано у лістингу 3.8. Тут обробляються текстові команди з полем «cmd», що дозволяє з легкістю додавати нові типи дій, наприклад «open», «close», «arm» чи «disarm». Архітектура командної обробки побудована таким чином, що кожна команда маршрутизується у відповідний обробник, де виконується перевірка коректності параметрів,

авторизація та безпосереднє виконання дії. Такий підхід забезпечує масштабованість системи та спрощує інтеграцію нових функцій у майбутньому.

Для запитів історичних даних використовується простий REST-маршрут, поданий у лістингу 3.9. Збереження даних здійснюється у кільцевій структурі в оперативній пам’яті, що дає змогу швидко формувати графіки на вебсторінці без перевантаження файлової системи. Кільцева буферизація також дозволяє контролювати використання ресурсів, оскільки обсяг пам’яті залишається сталим незалежно від часу роботи системи. Дані вибірки можуть бути запитані клієнтським інтерфейсом у будь-який момент, що забезпечує оперативний моніторинг стану датчиків та зручну візуалізацію тенденцій у режимі реального часу. Для доступу до приміщення реалізовано два незалежні методи ідентифікації: RFID і клавіатура.

Лістинг 3.8 – Прийом команд керування від клієнта

```
void onWsEvent(AsyncWebSocket*, AsyncWebSocketClient* c,
               AwsEventType type, void* arg, uint8_t* data, size_t len) {
  if (type != WS_EVT_DATA) return;
  AwsFrameInfo* info = (AwsFrameInfo*)arg;
  if (!info->final || info->index || info->len != len) return;

  StaticJsonDocument<256> doc;
  if (deserializeJson(doc, data, len) != DeserializationError::Ok) return;

  const char* cmd = doc["cmd"] | "";
  if (strcmp(cmd, "arm") == 0)   gstate = GuardState::Armed;
  if (strcmp(cmd, "disarm") == 0) gstate = GuardState::Off;
  if (strcmp(cmd, "open") == 0)   gateOpen();
  if (strcmp(cmd, "close") == 0)  gateClose();
}
```

кінець лістингу 3.8

Лістинг 3.9 – REST маршрут для отримання короткої історії вимірювань

```
#include <deque>
std::deque<float> histT, histH;

void pushHist(float t, float h) {
  if (histT.size() > 120) histT.pop_front();
  if (histH.size() > 120) histH.pop_front();
  histT.push_back(t);
  histH.push_back(h);
}

void serveHistory(AsyncWebServerRequest* req) {
  StaticJsonDocument<2048> doc;
  JsonArray ta = doc.createNestedArray("t");
  JsonArray ha = doc.createNestedArray("h");
  for (auto v : histT) ta.add(v);
  for (auto v : histH) ha.add(v);
  String out; serializeJson(doc, out);
  req->send(200, "application/json", out);
}
```

кінець лістингу 3.9

У лістингу 3.10 показано перевірку зчитаної карти та відкривання дверей у разі успішної авторизації. Такий підхід забезпечує швидке безконтактне розпізнавання користувачів.

У лістингу 3.11 наведено приклад оброблення введення коду через клавіатуру 3 на 4. Користувач вводить послідовність цифр, а підтвердження пароля відбувається після натискання кнопки «#». Обидва методи можна використовувати паралельно або як резервні один для одного.

Лістинг 3.10 – Перевірка RFID і відкривання

```
bool rfidAuthorized(const byte* uid, byte len);

void rfidTick() {
  byte uid[10]; byte n = 0;
  if (readRfid(uid, n)) {
    if (rfidAuthorized(uid, n)) gateOpen();
  }
}
```

кінець лістингу 3.10

Лістинг 3.11 – Читання коду з клавіатури 3 на 4

```
void onKey(char k) {
  if (k == '#') {
    if (code == "******") gateOpen();
    code = "";
  } else if (k == '*') {
    code = "";
  } else {
    if (code.length() < 8) code += k;
  }
}
```

Кінець лістингу 3.11

Програмне забезпечення контролера реалізує повний цикл оброблення подій у системі Domovyk, починаючи від опитування сенсорів і аналізу даних до керування виконавчими пристроями та двостороннього зв’язку з користувачем. Використання асинхронної архітектури забезпечує високу швидкодію, а модульна структура полегшує оновлення й розширення функціоналу.

3.3 Реалізація веб-інтерфейсу користувача

Вебінтерфейс системи Domovyk створено у вигляді багатосторінкового вебдодатка, який зберігається у файловій системі LittleFS головного контролера ESP32. Його структура охоплює головне меню (index.html) та тематичні сторінки для окремих підсистем: camera.html, humidity.html, temperature.html, gas.html, light.html, ventilation.html, door.html і gate.html.

Усі сторінки реалізовані з використанням стандартних вебтехнологій HTML, CSS і JavaScript. Дані оновлюються у реальному часі через WebSocket канал, що відкривається між браузером і контролером після завантаження сторінки. Історичні графіки та журнальні записи отримуються через звичайні HTTP запити до REST API.

Головне меню (рисунок 3.8) виконує функцію навігаційного центру системи. З нього користувач може перейти на будь-яку з функціональних сторінок. Інтерфейс адаптований під мобільні пристрої та планшети, має великі сенсорні елементи керування і мінімалістичний дизайн. Верхня частина кожної сторінки містить спільну панель із логотипом і кнопкою «Назад», що повертає користувача до головного меню.

Рисунок 3.8 – Головне меню вебінтерфейсу системи Domovyk

Сторінка камери (camera.html) (рисунок 3.9) відображає відеопотік із модуля ESP32-CAM. Вона забезпечує базові функції відтворення, паузи, перезапуску та переходу в повноекранний режим. Користувач може регулювати масштаб зображення та орієнтацію камери, що полегшує спостереження за об’єктом.

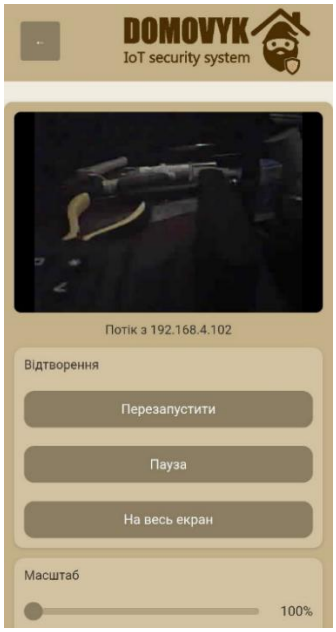

Рисунок 3.9 – Сторінка камери системи Domovyk

Сторінки вимірювання параметрів середовища (рисунок 3.10) (humidity.html, temperature.html, gas.html) побудовані за єдиним візуальним шаблоном. У центральній частині відображається поточне значення вимірюваної величини, а нижче – графік зміни параметра в реальному часі.

Наприклад, на сторінці вологості реалізовано відображення живого графіка за останні 5 хвилин, а також індикатори стану сенсора і тенденції зміни значень. Додатково передбачено систему кольорового маркування, яка автоматично змінює відтінок інтерфейсу залежно від наближення параметра до критичних меж. Це забезпечує швидке візуальне сприйняття ситуації без необхідності аналізувати числові дані.

Кожна сторінка також використовує оптимізовані WebSocket-канали, що мінімізують затримку між оновленням показників та їхнім відображенням на клієнтському пристрої.

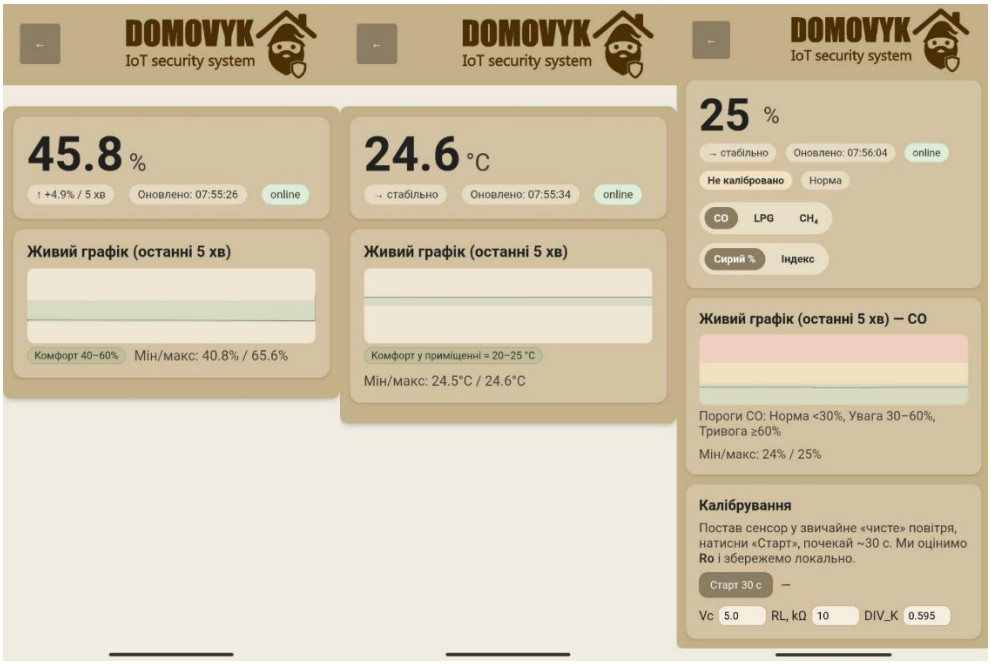

Рисунок 3.10 – Сторінки відображення параметрів середовища

На сторінках керування освітленням, вентиляцією, дверима та воротами (рисунок 3.11) (light.html, ventilation.html, door.html, gate.html) розміщено інтерактивні кнопки, повзунки та індикатори, що дозволяють користувачеві безпосередньо керувати виконавчими пристроями. Кожна дія одразу надсилається контролеру через WebSocket, і стан системи оновлюється на екрані без перезавантаження сторінки.

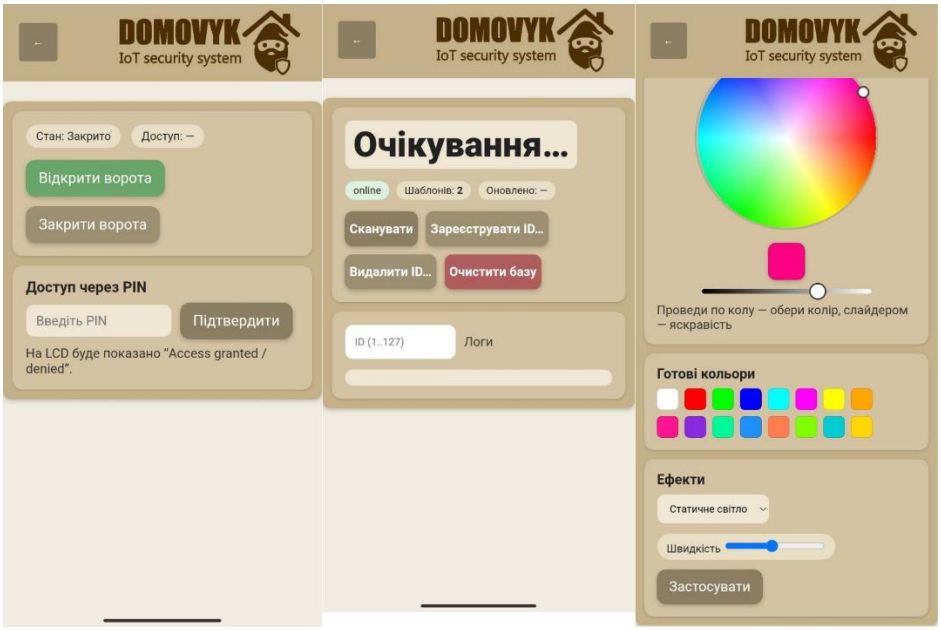

Рисунок 3.11 – Приклади сторінок керування виконавчими пристроями

Загальний стиль оформлення всіх сторінок витримано в єдиній кольоровій палітрі, що поєднує теплі відтінки та просту геометрію елементів. Такий підхід робить інтерфейс приємним для сприйняття, а навігацію – інтуїтивно зрозумілою.

Для стабільності роботи на мобільних пристроях передбачено обмеження орієнтації екрана у вертикальному режимі, що зберігає пропорції макета й запобігає спотворенню елементів.

Реалізований вебінтерфейс забезпечує повноцінну взаємодію користувача з системою Domovyk. Він дозволяє спостерігати за показниками сенсорів, переглядати відеопотік, отримувати сповіщення та керувати пристроями в реальному часі. Завдяки модульній структурі й використанню асинхронного обміну даними інтерфейс зберігає швидкодію навіть за активного оновлення кількох сторінок одночасно.

3.4 Тестування, експериментальні дослідження та порівняльний аналіз ефективності системи Domovyk

Метою експериментальної частини дослідження є кількісна оцінка роботи системи Domovyk та перевірка ефективності запропонованих алгоритмів оброблення даних і програмно-апаратної архітектури. Тестування проводилося на зібраному прототипі в умовах лабораторної моделі будинку за участі всіх основних підсистем: сенсорів температури, вологості й газу, датчиків руху та відкривання, виконавчих пристроїв, відеомодуля ESP32-CAM та вебінтерфейсу користувача.

У ході дослідження аналізувалися такі ключові показники: затримка реакції системи на події, точність вимірювань відносно еталонних приладів, вплив параметрів згладжування на співвідношення «швидкість реакції/стійкість до шуму», стабільність роботи при тривалому навантаженні, а також порівняльні характеристики відносно типової промислової системи безпеки Ajax Systems.

Першу серію експериментів було спрямовано на оцінку затримки між фізичною подією та реакцією системи. Для цього фіксувалися моменти спрацьовування датчика руху HC-SR501, магнітного контакту MC-38 або перевищення порогового значення газового сенсора MQ-9 та час відображення

відповідної події у вебінтерфейсі й активації виконавчих пристроїв (сирена, світлодіодна індикація, сервоприводи). Вимірювання виконувалися шляхом аналізу відеозапису екрана та журналу подій із часовими мітками. Для кожного типу події було виконано не менше 50 повторів, після чого обчислювалися середні, мінімальні та максимальні значення затримки.

У результаті середня затримка реакції на спрацювання датчика руху становила приблизно 0,28 с, для магнітних контактів близько 0,31 с, для перевищення порогу газу – близько 0,35 с. Межі розкиду значень не виходили за інтервал 0,21-0,44 с. Такі значення відповідають суб'єктивно «миттєвій» реакції з точки зору користувача і демонструють (рисунок 3.12), що застосована асинхронна модель оброблення подій та використання WebSocket-каналу дозволяють уникнути суттєвих затримок у типових сценаріях.

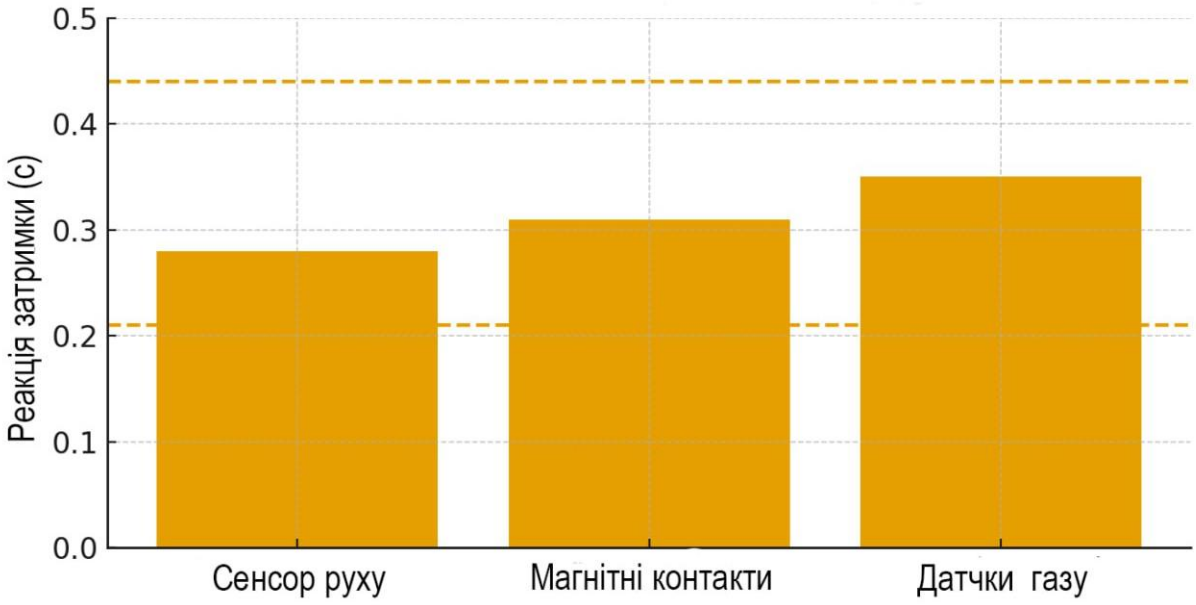

Рисунок 3.12 – Затримка спрацювань датчиків у системі Domovyk

Друга серія експериментів була присвячена оцінці точності вимірювання температури та відносної вологості сенсором SHT45. Для цього покази Domovyk порівнювалися з даними еталонної побутової метеостанції з паспортною похибкою не більше ±1° C і ±3 %RH. Спочатку було виміряно дані до зглажування (рисунок 3.13).

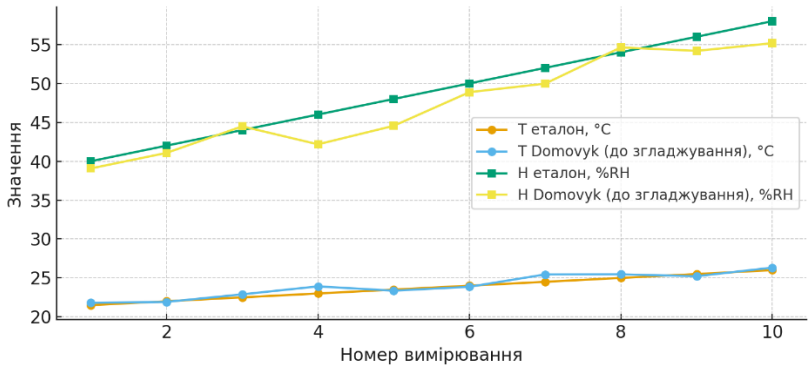

Рисунок 3.13 – Похибка вимірювання температури та вологості до застосування зглажування

Вимірювання виконувалися у кількох точках діапазону, характерного для житлового приміщення, з усередненням результатів за 10 повторів. Отримані значення показали, що розбіжність температури не перевищувала приблизно 0,5° С, а похибка вологості – 1-2 %RH. З урахуванням використання експоненційного згладжування, описаного в розділі 2, дані суттєво уточнилися і стабілізувалися (рисунок 3.14).

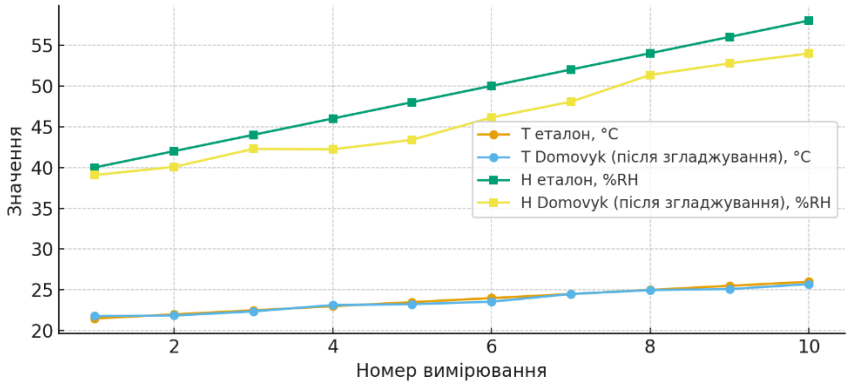

Рисунок 3.14 – Похибка вимірювання температури та вологості до застосування зглажування

Подібним чином досліджувався вплив порогів в алгоритмі гістерезису для газового сенсора MQ-9. У базовій схемі порівняння використовувався одинарний поріг, при якому будь-яке коливання сигналу навколо граничного значення призводило до частих перемикань стану тривоги. Експериментальні спостереження показали, що за рахунок гістерезису кількість хибних переходів у стан тривоги в

умовах слабкого шуму зменшується з кількох десятків до одиничних за годину тесту, при цьому час реагування на реальне перевищення рівня газу практично не змінюється.

Третій блок досліджень стосувався стабільності роботи вебінтерфейсу та відеопідсистеми. Оцінювалися час завантаження головної сторінки, час встановлення WebSocket-з'єднання, частота оновлення графіків та стабільність відеопотоку ESP32-CAM при одночасній роботі інших модулів. Експерименти показали, що типове підключення до локальної точки доступу Wi-Fi та завантаження інтерфейсу займає не більше двох секунд, частота оновлення даних на сторінках моніторингу становить близько 5-7 разів за секунду, а відеопотік з роздільною здатністю 800×600 пікселів стабільно підтримує 10-15 кадрів за секунду без помітних провалів при звичайному навантаженні. Додаткові добові випробування безперервної роботи не виявили зависань, втрат з'єднання або пошкодження файлової системи, а після короткочасного вимкнення живлення контролер коректно відновлював попередній стан.

Для того щоб оцінити запропонований підхід у контексті сучасних промислових рішень, було виконано порівняльний аналіз системи Domovyk із типовою комерційною системою безпеки Ajax Systems, яка широко використовується для захисту квартир, будинків та офісів і поєднує охоронні датчики, відеоспостереження, пожежну сигналізацію та автоматизацію. Система Ajax використовує власні двосторонні радіопротоколи Jeweller та Wings з шифруванням і захистом від глушіння, підтримує значну дальність зв'язку між хабом та бездротовими пристроями, багатоканальні канали передавання даних (Ethernet, мобільний зв'язок 2G/3G/LTE) та хмарну інфраструктуру з мобільними застосунками для керування й моніторингу в реальному часі.

Оскільки прототип Domovyk орієнтований насамперед на дослідницькі та навчальні сценарії, а система Ajax Systems – на сертифікований комерційний ринок, порівняння виконувалося не за абсолютними показниками надійності чи відповідності професійним стандартам охорони, а за узагальненими функціональними критеріями. До них належать можливість повністю автономної роботи без доступу до Інтернету, відкритість програмної платформи до

модифікацій, гнучкість сценаріїв, прозорість алгоритмів оброблення даних та орієнтація на кінцевого користувача й розробника.

Умовне зіставлення показало, що Ajax демонструє значно більшу масштабованість, глибоку інтеграцію з хмарними сервісами, широкий асортимент сертифікованих датчиків і суттєвий запас по дальності та завадостійкості радіоканалу, що є критично важливим для професійних систем охорони житла та бізнес-об’єктів. Водночас система Ajax є закритою з точки зору внутрішніх алгоритмів, не передбачає повного доступу до програмної логіки та апаратних інтерфейсів, а користувач обмежений типовими сценаріями, реалізованими виробником.

Прототип Domovyk, навпаки, має низку переваг саме в контексті кастомізації та дослідження. Уся логіка виконання, зберігання інтерфейсу та оброблення даних зосереджена на мікроконтролері ESP32, що забезпечує повну локальну автономність без залежності від зовнішніх серверів і хмарних платформ. Структура прошивки і вебінтерфейсу є відкритою: розробник може змінювати алгоритми згладжування, порогові значення, періоди опитування, формати повідомлень, структуру сторінок та елементи інтерфейсу, спостерігаючи вплив цих змін у реальному часі. Калібрувальні можливості, реалізовані безпосередньо у вебзастосунку (налаштування порогів газового сенсора, параметрів гістерезису, швидкості оновлення графіків тощо), дозволяють адаптувати систему під конкретне приміщення та експериментальні умови без перепрошивання контролера.

Окремою перевагою Domovyk є підтримка кількох незалежних каналів взаємодії з користувачем: вебінтерфейс із живими графіками й керуванням виконавцями, локальна сенсорна панель ESP32-2432S028R, текстовий LCD-дисплей, а також мультифакторні способи доступу (RFID-карти, клавіатура 3 на 4, модуль відбитків пальців). Така комбінація функцій рідко зустрічається в готових комерційних системах і дає змогу досліджувати різні моделі авторизації та сценарії роботи без зміни апаратної платформи. Крім того, архітектура Domovyk легко розширюється за рахунок додавання нових сторінок, сенсорів або

виконавчих модулів, що робить її зручною базою як для навчальних лабораторних робіт, так і для подальших наукових експериментів у сфері IoT.

Таким чином, результати тестування показують, що Domovyk забезпечує рівень швидкодії та точності, достатній для побутового сценарію «розумного дому», а запропоновані математичні та архітектурні рішення (експоненційне згладжування, інтерактивне калібрування порогів, гістерезис, асинхронна комунікація через WebSocket) дають змогу досягти стабільної роботи без помітних затримок і без надмірної чутливості до шуму. Порівняння з промисловою системою Ajax Systems засвідчує, що розроблений прототип не конкурує з нею за масштабованістю та сертифікацією, проте заповнює нішу відкритих, гнучко модифікованих і відносно недорогих рішень для дослідження, навчання та індивідуальної кастомізації у сфері домашньої автоматизації. Це підтверджує доцільність обраного напряму роботи та створює підґрунтя для подальших удосконалень, зокрема інтеграції з хмарними сервісами, впровадження протоколу MQTT, розширеної аналітики даних та використання алгоритмів машинного навчання для прогнозування подій і адаптивного налаштування режимів системи.

ВИСНОВКИ

Система «Domovyk» є локально автономним IoT-комплексом для моніторингу мікроклімату, охоронних подій та керування виконавчими пристроями, розробленим із навчально-дослідницькою та практичною метою.

В результаті виконання кваліфікаційної роботи було:

– проведено аналітичний огляд наукових публікацій, технічної документації та сучасних рішень у сфері IoT, зокрема систем Ajax Systems, U-Prox, Dahua, Hikvision та Tuya, та визначено вимоги до побудови відкритої, гнучкої та локально автономної системи домашньої автоматизації;

– обґрунтовано вибір апаратної і програмної платформи, доведено доцільність використання мікроконтролерів ESP32 та ESP32-CAM, асинхронного вебсервера, WebSocket-каналу, REST-API, файлової системи LittleFS та алгоритмів фільтрації даних (експоненційне згладжування, гістерезис, антидребезг);

– реалізовано апаратну частину системи, що включає комплекс сенсорів (SHT45, MQ-9, HC-SR501, MC-38), виконавчих пристроїв (MG90S, вентилятор, RGB-стрічка, бузер), два контролери ESP32 та модель будинку, виготовлену методом 3D-друку, з повним розведенням живлення та сигнальних ліній;

– розроблено програмне забезпечення головного контролера, у якому реалізовано асинхронну обробку подій, механізми опитування сенсорів, фільтрацію та нормалізацію даних, керування виконавчими елементами, менеджер охоронного режиму та двосторонню комунікацію з клієнтом у режимі реального часу;

– створено багатосторінковий вебінтерфейс локального зберігання, який забезпечує візуалізацію параметрів мікроклімату, керування дверима, воротами, освітленням і вентиляцією, а також перегляд відеопотоку ESP32-CAM без залучення зовнішніх серверів;

– проведено експериментальні дослідження точності та швидкодії системи, встановлено, що середня затримка реакції становить 0,28-0,35 с, а похибка вимірювання температури й вологості – до 0,5° С та 1-2 %RH відповідно; доведено

ефективність алгоритмів згладжування та гістерезису, які зменшили хибні спрацювання газового сенсора з десятків до одиничних;

– запропоновано перспективні напрями вдосконалення системи, серед яких інтеграція MQTT-протоколу, взаємодія з хмарними сервісами, розширення функцій сенсорної панелі ESP32-2432S028R, мобільний застосунок, енергозберігаючі режими та застосування алгоритмів машинного навчання.

Результати кваліфікаційної роботи підтверджують актуальність обраної теми. Розроблення системи «Domovyk» дозволило комплексно дослідити архітектуру IoT-рішень, методи взаємодії сенсорних мереж, математичні моделі оброблення даних та алгоритми підвищення стабільності роботи локально автономних систем. Запропонований прототип продемонстрував можливість поєднання апаратної та програмної частин у єдину подієву інфраструктуру, що забезпечує низьку затримку реакції, достатню точність вимірювань та стійкість до шумів.

Отримані результати свідчать, що система здатна виконувати повний цикл моніторингу й керування у реальному часі без залучення хмарних сервісів, що є суттєвою перевагою для навчальних, дослідницьких і лабораторних цілей. Проведені експерименти підтвердили ефективність використаних алгоритмів згладжування, гістерезису та асинхронної комунікації, а також показали потенціал Domovyk для подальшого розширення функціоналу.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Христинець Н., Якобчук Б. Функціональні особливості структури адміністративних панелей засобами PHP та JS. *Наукові горизонти XXI століття: мультидисциплінарні дослідження*: зб. тез II міжнар. наук. конф., м. Ужгород, 6-7 трав. 2025 р. Ужгород, 2025. С. 391-393.

2. Torres-Hernandez C. M., Tapia-Fonllem C., Figueroa-Flores F. Smart Homes: A Meta-Study on Sense of Security and Home Automation. *Technologies.* 2025. Vol. 13, No. 8. P. 320-336.

3. Wulandarini N. Integration of Multiple Sensors and Actuators in Smart Home Automation System Based on IoT Concept. *Proceedings of ICoSTAS-EAS 2024, Advances in Engineering Research.* 2024. Vol. 249, No. 02. P. 682-690.

4. Yuan B., Li L., Li J., Yan Q. On the Security of Smart Home Systems: A Survey. *Journal of Computer Science and Technology.* 2023. Vol. 38, No. 2. P. 228-247.

5. Zhebka V., Skladannyi P., Zhebka S., Shlianchak S., Bondarchuk A. Methodology for Predicting Failures in a Smart Home based on Machine Learning Methods. *Cybersecurity Providing in Information and Telecommunication Systems 2024 (CPITS 3654).* 2024. P. 322-332.

6. Alshamsi O., Shaalan K., Butt U. Towards Securing Smart Homes: A Systematic Literature Review of Malware Detection Techniques and Recommended Prevention Approach. *Information.* 2024. Vol. 15, No. 20. P. 157-159.

7. Poyyamozhi M., Murugesan B., Rajamanickam N., Shorfuzzaman M., Aboelmagd Y. IoT – A Promising Solution to Energy Management in Smart Buildings. *Buildings.* 2024. Vol. 14, No. 11. P. 1-26.

8. Ahmed A., Ungureanu V., Gaber T., Watterson C., Masmoudi F. Smart Home Privacy: A Scoping Review. *10th Int. Conf. on Information Systems Security and Privacy (ICISSP 2024).* 2024. P. 1420-1637.

9. Madadi-Barough S., Ruiz-Blanco P., Lin J., Vidal R., Gomez C. Matter. IoT Interoperability for Smart Homes. *IEEE Communications Magazine.* 2024. Vol. 63, No. 4. P. 1711-1820.

10. Sheikh A. M., Islam M. R., Habaebi M. H., Zabidi S. A., Bin Najeeb A. R., Kabbani A. A Survey on Edge Computing (EC) Security Challenges: Classification, Threats and Mitigation Strategies. *Future Internet.* 2025. Vol. 17, No. 4. P. 175-194.

11. Choudhary A. Internet of Things: a comprehensive overview, architectures, applications, simulation tools, challenges and future directions. *Discover Internet of Things.* 2024. Vol. 4, No 31. P. 15-33.

12. Mansour M., Gamal A., Ahmed A. I. Internet of Things: A Comprehensive Overview on Protocols, Architectures, Technologies, Simulation Tools, and Future Directions. *Energies.* 2023. Vol. 16, No. 8. P. 77-91.

13. Dauda A., Flauzac O., Nolot F. A Survey on IoT Application Architectures. *Sensors.* 2024. Vol. 24, No. 16. P. 237-250.

14. Alotaibi N. S., Sayed Ahmed H. I., Kamel S. O. M., ElKabbany G. F. Secure Enhancement for MQTT Protocol Using Distributed Machine Learning Framework. *Sensors.* 2024. Vol. 24, No. 5. P. 82-99.

15. Almudayni Z., Soh B., Samra H., Li A. Energy Inefficiency in IoT Networks: Causes, Impact and a Strategic Framework for Sustainable Optimisation. *Electronics.* 2025. Vol. 14, No. 1. P. 61-72.

16. Колісник М. Гарантоздатність компонентів і систем Інтернету речей. *Радіоелектронні і комп'ютерні системи.* 2021, № 1. С. 133-141.

17. Chang Q., Ma T., Yang W. Low power IoT device communication through hybrid AES-RSA encryption in MRA mode. *Scientific Reports.* 2025. Vol. 15, No. 6. P. 18-27.

18. Hamid H. G., Alisa Z. T. A survey on IoT application layer protocols. *Indonesian Journal of Electrical Engineering and Computer Science.* 2021. Vol. 21, No. 3. P. 1663-1672.

19. Wytrębowicz J., Cabaj K., Krawiec J. Messaging Protocols for IoT Systems A Pragmatic Comparison. *Sensors.* 2021. Vol. 21, No. 20. P. 64-81.

20. Tariq M. A., Khan M., Raza Khan M. T., Kim D. Enhancements and Challenges in CoAP A Survey. *Sensors.* 2020. Vol. 20, No. 21. P. 119-133.

21. Gentile A. A Network Performance Analysis of MQTT Security Protocols with Constrained Hardware in the Dark Net for DMS. *Applied Sciences.* 2024. Vol. 14, No. 18. P. 141-164.

22. Bayilmis C., Durmus M., Karaarslan E. A survey on communication protocols and performance evaluation for Internet of Things. *Array.* 2022. Vol. 16, No. 21. P. 245-271.

23. Zigbee vs Z-Wave: Which is Better for Home Automation. *Dusun IoT.* 2023. URL: https://www.dusuniot.com/blog/zigbee-vs-z-wave-which-is-better-for-home-automation/ (дата звернення: 14.07.2025)

24. Evaluating HTTP, MQTT over TCP and MQTT over WebSocket for Real-Time IoT Applications Using ESP32 Microcontrollers and DHT22 Sensors. *International Journal of Innovative Research in Science & Society (IJIRSS).* 2025. Vol. 9, No. 1. P. 1-10.

25. Зберіть свою систему Ajax для охорони та автоматизації. *Ajax Systems.* URL: https://ajax.systems/ua/tools/configurator/ (дата звернення: 23.07.2025).

26. Рішення для Вашого будинку або дачі – jхоронна система U-Prox. *Системи сигналізацій U-PROX.* URL: https://security.u-prox.systems/ua/kalkuliator/calc-dla_domu/ (дата звернення: 23.07.2025).

27. ATIS: гіпермаркет охоронних систем і засобів безпеки. *Bezpeka-Shop.com* URL: https://www.bezpeka-shop.com/ua/manufacturers/atis/ (дата звернення: 25.07.2025).

28. Комплект IP-відеоспостереження Dahua на 8 купольних 2 Мп IP-камер DH-IP1118OW-2MP: повний огляд, характеристики, ціни. *Smart-security.com.ua*. URL: https://surl.li/anfslp (дата звернення: 27.07.2025).

29. Комплект бездротової охоронної сигналізації Hikvision AX PRO DS-PWA96-Kit-WE. *VIATEC.UA.* URL: https://surl.li/jmianj (дата звернення: 30.07.2025).