ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «ІНТЕГРОВАНА АРХІТЕКТУРА КЕРУВАННЯ "РОЗУМНИМ БУДИНКОМ" ІЗ ВИКОРИСТАННЯМ СЕРЕДОВИЩА AJAX IOT»

на здобуття освітнього ступеня бакалавра
зі спеціальності 126 Інформаційні системи та технології
(код, найменування спеціальності)
освітньо-професійної програми Інформаційні системи та технології
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ *Євгеній ЛАЩЕВСЬКИЙ*
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконав: здобувач вищої освіти гр. ІСД-42
Євгеній ЛАЩЕВСЬКИЙ
Ім'я, ПРІЗВИЩЕ

Керівник: Каміла СТОРЧАК
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ

Рецензент:
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ

Київ 2026

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
Навчально-науковий інститут Інформаційних технологій

Кафедра 126 Інформаційних систем та технологій
Ступінь вищої освіти Бакалавр
Спеціальність 126 Інформаційних систем та технологій
Освітньо-професійна програма Комп'ютерна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедрою Каміла СТОРЧАК
Ім'я, ПРІЗВИЩЕ
«_____» _____ 2026р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Лащевський Євгеній Сергійович
(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: Інтегрована архітектура керування "Розумним будинком" із використанням середовища Ajax IoT
керівник кваліфікаційної роботи Каміла СТОРЧАК,
(Ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «20» 02.2026р. № 51
2. Строк подання кваліфікаційної роботи «08» 05 2026р.
3. Вихідні дані до кваліфікаційної роботи: Науково-технічна література, методи системного аналізу, штучний інтелект, математичний апарат, емпіричні методи
Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Дослідження сучасних рішень та застосування технології ajax у веб-архітектурах систем «розумний дім»
2. Розробка комплексної архітектури системи «розумний дім» на базі технологій ajax
3. Створення iot-системи для автоматизації житлового простору на базі технологій ajax
4. Перелік ілюстративного матеріалу: *презентація*
5. Дата видачі завдання «09» 02 2026р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Збір даних	25.02-24.03.2026	Виконано
2	Обробка матеріалу	23.03-05.04.2026	Виконано
3	Розробка теоретичної частини	06.04-10.04.2026	Виконано
4	Розробка практичної частини	11.04-20.04.2026	Виконано
5	Висновки за результатами аналізу	21.04-06.05.2026	Виконано
6	Оформлення пояснювальної записки	07.05-08.05.2026	Виконано
7	Розробка презентації	09.05-11.05.2026	Виконано
8	Передзахист	27.05-29.05.2026	Виконано
9	Здача в деканат	02.06-05.06.2026	Виконано

Здобувач вищої освіти _____ Євгеній ЛАЩЕВСЬКИЙ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник кваліфікаційної роботи _____ Каміла СТОРЧАК
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 76 стор., 35 рис., 1 табл., 29 джерела.

Мета дослідження – розробка інтегрованої архітектури керування системою «Розумний будинок» із використанням захищеного середовища Ajax та технологій IoT.

Об'єкт дослідження – процес автоматизації, моніторингу та управління апаратними і програмними компонентами системи «Розумний будинок».

Предмет дослідження – інтегрована архітектура керування «Розумним будинком» на базі професійного середовища Ajax Systems та кастомних IoT-вузлів.

Короткий зміст роботи: Існуючі системи автоматизації житла («розумні будинки») відрізняються значною розрізненістю стандартів та протоколів. Користувачі часто змушені використовувати різні непов'язані додатки для контролю освітлення, клімату та систем безпеки. Окремою проблемою є закритість професійних охоронних систем, таких як Ajax, що ускладнює їх повноцінну взаємодію з іншими побутовими IoT-пристроями. Тож постає питання, як об'єднати високонадійну систему безпеки із гнучкими засобами побутової автоматизації в єдине інформаційне середовище. Це досягається шляхом розробки комплексної гібридної архітектури.

Запропонована архітектура забезпечує централізоване керування та стабільний обмін даними між пристроями Ajax та кастомними IoT-модулями (на базі мікроконтролерів ESP32/ESP8266) через єдиний сервер (MQTT-брокер) та керуючий програмний модуль на мові Python. Це дозволяє налаштовувати складні крос-системні сценарії автоматизації.

На основі проведеного дослідження було обґрунтовано вибір апаратного забезпечення, розроблено алгоритми реєстрації пристроїв і перевірки сценаріїв, а також створено аналітичну підсистему на базі нейронних мереж (RNN/LSTM) для пошуку прихованих закономірностей у роботі обладнання.

КЛЮЧОВІ СЛОВА: РОЗУМНИЙ БУДИНОК, AJAX SYSTEMS, ІНТЕРНЕТ РЕЧЕЙ (IOT), ІНТЕГРОВАНА АРХІТЕКТУРА, СИСТЕМА БЕЗПЕКИ, СЕНСОРИ, АВТОМАТИЗАЦІЯ, MQTT, МІКРОКОНТРОЛЕРИ.

ABSTRACT

Textual part of the qualification work for the bachelor's degree: 76 pages, 35 figures, 1 tables, 29 sources.

The purpose of the work – Development of an integrated "Smart Home" management architecture using the secure Ajax environment and IoT technologies.

Object of research – The process of automation, monitoring, and management of hardware and software components of the "Smart Home" system.

Subject of research – Integrated "Smart Home" management architecture based on the professional Ajax Systems environment and custom IoT nodes.

Summary of the work: Existing home automation systems ("smart homes") are characterized by a significant fragmentation of standards and protocols. Users are often forced to use various disconnected applications to control lighting, climate, and security systems. A separate problem is the closed nature of professional security systems, such as Ajax, which complicates their full interaction with other household IoT devices. Therefore, the question arises of how to combine a highly reliable security system with flexible home automation tools into a single information environment. This is achieved by developing a complex hybrid architecture.

The proposed architecture provides centralized management and stable data exchange between Ajax devices and custom IoT modules (based on ESP32/ESP8266 microcontrollers) through a single server (MQTT broker) and a control software module in Python. This allows configuring complex cross-system automation scenarios.

Based on the research, the choice of hardware was justified, algorithms for device registration and scenario verification were developed, and an analytical subsystem based on neural networks (RNN/LSTM) was created to find hidden patterns in the operation of the equipment.

KEYWORDS: SMART HOME, AJAX SYSTEMS, INTERNET OF THINGS (IOT), INTEGRATED ARCHITECTURE, SECURITY SYSTEM, SENSORS, AUTOMATION, MQTT, MICROCONTROLLERS.

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

ПОДАННЯ
ГОЛОВІ ДЕРЖАВНОЇ ЕКЗАМЕНАЦІЙНОЇ КОМІСІЇ
ЩОДО ЗАХИСТУ КВАЛІФІКАЦІЙНОЇ РОБОТИ
на здобуття освітнього ступеня бакалавра

Направляється здобувач Лащевський Є. С. до захисту кваліфікаційної роботи
(прізвище та ініціали)
за спеціальністю 126 Інформаційні системи та технології
(код, найменування спеціальності)
освітньо-професійної програми Інформаційні системи та технології
(назва)
на тему: «Інтегрована архітектура керування "Розумним будинком" із використанням середовища Ajax IoT».
Кваліфікаційна робота і рецензія додаються.

Директор ННІ _____ Катерина НЕСТЕРЕНКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Висновок керівника кваліфікаційної роботи

Здобувач Лащевський Є. С показав відмінну теоретичну підготовку, уміння володіти та розуміння новими комп'ютерними технологіями, користуватися навчальною і науково-технічною літературою. Працюючи над завданнями, які були поставлені керівником, проявив ініціативність, сумлінність та хист до інженерної роботи.
Все це дозволяє оцінити кваліфікаційну роботу студента Лащевського Євгенія Сергійовича на оцінку «відмінно», та присвоїти йому кваліфікацію «бакалавр з інформаційних систем та технологій».

Керівник кваліфікаційної роботи _____ Каміла СТОРЧАК
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

«_» _____ 2026 року

Висновок кафедри про кваліфікаційну роботу

Кваліфікаційна робота розглянута. Здобувач *Лащевський Є. С.* допускається до захисту даної роботи в Екзаменаційній комісії.

Завідувач кафедрою Інформаційнх систем та технологій _____ Каміла СТОРЧАК
(назва) *(підпис)* *(Ім'я, ПРІЗВИЩЕ)*

ВІДГУК РЕЦЕНЗЕНТА
на кваліфікаційну бакалаврську роботу

здобувача вищої освіти Лащевський Євгеній Сергійович
(прізвище, ім'я, по батькові)
на тему: «Інтегрована архітектура керування "Розумним будинком" із використанням середовища Ajax IoT»

Актуальність:

Бакалаврська кваліфікаційна робота присвячена одній із найактуальніших проблем сучасного ринку Інтернету речей (IoT) — створенню єдиного, інтегрованого та безпечного середовища для систем «Розумного будинку». В умовах, коли ринок перенасичений закритими екосистемами від різних виробників, тема дослідження, спрямована на поєднання професійної охоронної системи Ajax із кастомними засобами домашньої автоматизації, має високу практичну та наукову цінність. Автор пропонує дієвий перехід від розрізненого керування до централізованої гібридної архітектури, що відповідає сучасним вимогам до безпеки та комфорту житла.

Позитивні сторони:

1. У роботі наведено глибоке технічне обґрунтування вибору апаратної бази та комунікаційних протоколів для побудови надійної локальної мережі.
2. Автор продемонстрував високу компетентність у галузі програмування, розробивши повноцінний керуючий модуль, здатний реєструвати пристрої, обробляти інформаційні потоки та виконувати крос-системні сценарії автоматизації.
3. Практична цінність роботи суттєво підсилюється розробкою інноваційної аналітичної підсистеми на базі рекурентних нейронних мереж. Автор не лише спроєктував інфраструктуру, але й навчив систему самостійно виявляти логічні закономірності в роботі пристроїв, що є ознакою високого інженерного рівня.

Недоліки:

1. У роботі недостатньо уваги приділено питанню енергоефективності кастомних бездротових модулів. Важливо було б навести розрахунки тривалості їх автономної роботи у режимі глибокого сну.
2. Бажано було б розширити експериментальну базу та протестувати стійкість MQTT-брокера до пікових навантажень при одночасному спрацюванні великої кількості сенсорів, що дозволило б підтвердити масштабованість системи для великих об'єктів.

Висновок: *кваліфікаційна бакалаврська робота заслуговує оцінку " відмінно", а здобувач Лащевський Є. С. заслуговує присвоєння кваліфікації: бакалавра з комп'ютерної інженерії.*

Рецензент: _____ _____
науковий ступінь, вчене звання *підпис* *Ім'я, ПРІЗВИЩЕ*

ЗМІСТ

ВСТУП.....9
1 ДОСЛІДЖЕННЯ СУЧАСНИХ РІШЕНЬ ТА ЗАСТОСУВАННЯ ТЕХНОЛОГІЇ AJAX У ВЕБ-АРХІТЕКТУРАХ СИСТЕМ «РОЗУМНИЙ ДІМ».....10
1.1 Сутність та базові принципи роботи інтелектуальних систем управління оселею.....10
1.2 Аналіз передових технологій Інтернету речей для автоматизації житлового простору.....14
1.3 Архітектурні моделі та підходи до проєктування систем «Розумний дім»25
2 РОЗРОБКА КОМПЛЕКСНОЇ АРХІТЕКТУРИ СИСТЕМИ «РОЗУМНИЙ ДІМ» НА БАЗІ ТЕХНОЛОГІЙ AJAX.....32
2.1 Визначення цілей та формулювання завдання щодо комплексної модернізації архітектурних рішень.....32
2.2 Підбір оптимального комплексу апаратного забезпечення для формування IoT-інфраструктури об'єкта на базі типових компонентів.....35
2.3 Схемотехнічні рішення для підключення обладнання та логіка взаємодії системних вузлів.....49
3 СТВОРЕННЯ IOT-СИСТЕМИ ДЛЯ АВТОМАТИЗАЦІЇ ЖИТЛОВОГО ПРОСТОРУ НА БАЗІ ТЕХНОЛОГІЙ AJAX.....57
3.1 Розробка та програмна реалізація модуля для управління апаратними компонентами системи.....57
3.2 Практична реалізація підсистеми для безперервного моніторингу та поглибленої аналітики функціонування IoT-обладнання.....64
ВИСНОВКИ.....71
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....73
Додаток А.....76

ВСТУП

В наш час концепція «Розумного будинку» та Інтернет речей (IoT) стали невід’ємною складовою сучасного комфортного життя. Вони присутні в житлових комплексах, офісах та на підприємствах, забезпечуючи автоматизацію рутинних процесів, енергоефективність та високий рівень безпеки. Сучасний ринок пропонує величезну кількість розумних пристроїв від різних виробників: від інтелектуальних ламп та термостатів до складних систем відеоспостереження. Проте, при спробі побудувати комплексну систему користувачі стикаються із серйозною проблемою — фрагментацією. Пристрої часто працюють на різних протоколах та керуються через окремі, не пов'язані між собою додатки.

Найбільш гостро ця проблема постає при спробі інтегрувати професійні системи безпеки, такі як Ajax Systems, із загальною побутовою автоматизацією. Охоронні комплекси проєктуються як максимально закриті та автономні системи для забезпечення найвищого рівня надійності. Проте у випадку ізольованого використання системи втрачається можливість створення корисних сценаріїв: наприклад, автоматичного перекриття води при спрацюванні датчика протікання Ajax, або увімкнення всього освітлення в будинку при спрацюванні охоронної тривоги.

Для того, щоб користувач мав змогу керувати всією оселею як єдиним організмом, потрібно правильно налаштувати взаємодію між закритими протоколами безпеки та відкритими IoT-стандартами. Саме таку можливість надає розробка інтегрованої архітектури керування. Вона дозволяє об'єднати високонадійне обладнання Ajax та доступні мікроконтролери для специфічних завдань під управлінням єдиного центрального сервера на базі протоколу MQTT. Це, в свою чергу, усуває бар'єри між пристроями, забезпечує високу швидкість реакції системи та відкриває необмежені можливості для автоматизації та інтелектуального моніторингу життєвого простору.

1 ДОСЛІДЖЕННЯ СУЧАСНИХ РІШЕНЬ ТА ЗАСТОСУВАННЯ ТЕХНОЛОГІЇ AJAX У ВЕБ-АРХІТЕКТУРАХ СИСТЕМ «РОЗУМНИЙ ДІМ»

1.1 Сутність та базові принципи роботи інтелектуальних систем управління оселею

Поняття "розумного будинку" охоплює створення інтегрованого комплексу, що об'єднує автоматизоване керування всіма інженерними системами та побутовими пристроями в житловому просторі. Це дозволяє сформувати єдиний технологічний ансамбль, за допомогою якого власник може здійснювати всебічний контроль над освітленням, опаленням, кондиціонуванням повітря, системами безпеки, мультимедійними пристроями та іншими складовими. Керування цим складним організмом може відбуватися як у ручному режимі за допомогою смартфона, планшета чи навіть голосових команд, так і в автоматичному, згідно з попередньо запрограмованими сценаріями життєдіяльності. Візуалізація цього принципу взаємодії наведена на схематичному зображенні розумного будинку на рис.1.1.

Рисунок 1.1 – Схематичне зображення концепції та взаємозв'язку компонентів розумного будинку

Фундаментальними характеристиками, які визначають ефективність та комфорт такої системи, є, перш за все, глибока автоматизація процесів. Це означає, що система здатна самостійно регулювати рівень освітленості, температуру та навіть роботу складної побутової техніки, динамічно адаптуючись до часу доби, погодних умов чи простої присутності мешканців. Важливим аспектом є також централізоване керування, де всі розрізнені пристрої під'єднуються до єдиного інтелектуального центру, яким може бути локальний контролер або ж віддалений хмарний сервіс. Ця архітектура забезпечує синхронність і злагодженість їхньої роботи. Не менш вагомим є прагнення до енергоефективності, що полягає в оптимальному та раціональному використанні ресурсів, таких як електроенергія, вода та тепло, з метою мінімізації втрат та витрат. І нарешті, система безпеки є невід'ємною частиною, яка інтегрує відеоспостереження, різноманітні сигналізації, а також спеціалізовані датчики, які реагують на дим, рух, витік газу чи води, забезпечуючи комплексний захист оселі [1].

Концептуальною основою, яка уможливлює функціонування таких складних систем, є Internet of Things (IoT), або Інтернет речей. Ця концепція передбачає, що найрізноманітніші фізичні об'єкти (речі) отримують можливість підключатися до глобальної мережі Інтернет та обмінюватися даними між собою, а також із центральними системами керування, практично без безпосередньої участі людини. Уяву про масштаби застосування Internet of Things можна отримати з прикладу на рис.1.2.

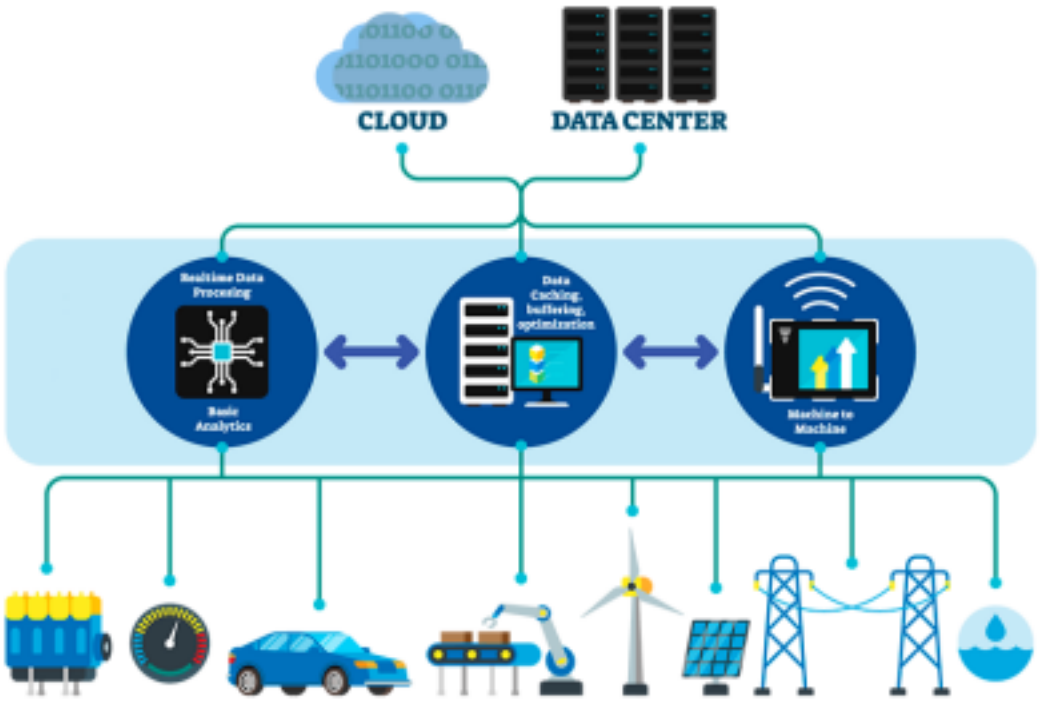

Рисунок 1.2 – Приклад впровадження та застосування технології Internet of Things (IoT)

Ці пристрої охоплюють широченний спектр: від звичайних побутових приладів, таких як холодильники, лампи, термостати та телевізори, до складного промислового обладнання, автомобілів, медичних сенсорів, що моніторять стан здоров'я, та інтелектуальних годинників. Усі вони оснащуються вбудованими датчиками, мікроконтролерами (які виконують роль "мозку") та різноманітними комунікаційними модулями зв'язку, що

дозволяють їм взаємодіяти через Wi-Fi, Bluetooth, GSM чи інші мережі. Ця технологічна база дає можливість збирати інформацію з навколишнього середовища, передавати її, аналізувати та автоматично реагувати на зміни.

Весь процес функціонування IoT-екосистеми можна описати як безперервний цикл, який розпочинається зі збору даних. Кожен окремий пристрій, оснащений вбудованими датчиками, постійно фіксує різноманітні фізичні параметри, такі як температура, вологість, рівень освітлення, рух або поточне споживання енергії. Ці зібрані дані стають фундаментальною основою для подальшої роботи системи. На наступному етапі відбувається передача даних. Уся накопичена інформація миттєво надсилається через доступні мережеві технології, які можуть включати Wi-Fi, Bluetooth, ZigBee, 4G/5G або LoRaWAN, до інших пристроїв або безпосередньо на центральний сервер, роль якого часто виконує потужна хмарна платформа. Після цього розпочинається етап обробки даних. Сервер або хмарна система аналізує отриману інформацію за допомогою складних алгоритмів та приймає виважені рішення. Наприклад, якщо датчики зафіксують, що температура в приміщенні піднялася вище встановленої межі, система автоматично прийме рішення про необхідність зменшення потужності кондиціонера. За цим слідує безпосереднє виконання дій. На основі результатів попереднього аналізу система надсилає точні команди відповідним пристроям. Так, якщо датчик руху виявить присутність людини, система автоматично увімкне освітлення в цій кімнаті. Завершальним, але не менш важливим етапом є зворотний зв’язок та керування. Користувач зберігає за собою можливість віддалено контролювати всі підключені пристрої за допомогою мобільного додатка або голосового помічника, а також у будь-який момент часу отримувати вичерпну інформацію про стан всієї системи в режимі реального часу, забезпечуючи повний контроль над своїм розумним домом.

1.2 Аналіз передових технологій Інтернету речей для автоматизації житлового простору

Сучасний ринок технологій пропонує надзвичайно широкий спектр різноманітних платформ, які слугують надійним фундаментом для впровадження та повноцінної реалізації систем Інтернету речей (IoT). Ці рішення знаходять своє застосування як у повсякденному побуті звичайних користувачів, так і у вузькоспеціалізованих професійних сферах. Серед усього розмаїття найбільш помітними та популярними гравцями на сьогодні є такі гіганти, як Google Home, Apple HomeKit, універсальна хмарна платформа Tuya Smart, а також професійна система безпеки Ajax Systems. Кожна із зазначених платформ формує свою унікальну екосистему, пропонуючи користувачам специфічний набір функціональних можливостей, протоколів взаємодії та різний рівень сумісності з обладнанням від сторонніх виробників. Детальніший огляд особливостей цих архітектур подано далі [2].

Екосистема Google Home являє собою комплексне рішення для розумного дому, розроблене всесвітньо відомою корпорацією Google. Головна ідея цієї платформи полягає в об'єднанні найрізноманітніших розумних пристроїв у єдину інтелектуальну мережу за допомогою потужної хмарної платформи та віртуального помічника Google Assistant. На рис. 1.3 можна побачити зовнішній вигляд лінійки пристроїв, призначених для комфортного керування простором, які безшовно інтегруються через платформу Google Home завдяки використанню сучасного протоколу Matter. До цього переліку входять різноманітні акустичні системи, такі як розумні колонки серій Nest Audio, Home та компактні Nest Mini. Окрему нішу займають інтелектуальні дисплеї Nest Hub різних поколінь, деякі з яких оснащені додатковим корисним функціоналом, як-от моніторинг якості сну або розширений температурний інтерфейс. Усе це доповнюється зручним мобільним додатком, що слугує пультом управління системою.

Рисунок 1.3 – Взаємодія екосистеми Google Home та протоколу Matter для контролю обладнання розумної оселі

Використовуючи цю систему, користувачі отримують можливість зручно й централізовано керувати цілим спектром хатнього обладнання: від базового освітлення та налаштування клімату до складних систем сигналізації та мультимедійних центрів. Управління здійснюється максимально природно — за допомогою голосових команд або інтуїтивно зрозумілого мобільного застосунку. Серед ключових особливостей Google Home варто виділити безпрецедентну підтримку тисяч сумісних пристроїв від найрізноманітніших світових виробників. Центральним елементом взаємодії є передове голосове керування, яке дозволяє не лише віддавати прямі команди, а й створювати складні автоматизовані сценарії. Наприклад, активація тригера «добрий ранок» може одночасно розсунути штори, увімкнути комфортне освітлення та запустити обрану музику. Крім того, платформа відзначається глибокою та органічною інтеграцією з усіма пристроями на базі операційної системи Android та іншими фірмовими сервісами екосистеми Google.

Альтернативним підходом до автоматизації житлового простору є Apple HomeKit – фірмова розробка від компанії Apple. Головною філософією цієї системи є забезпечення безкомпромісного рівня безпеки та суворої приватності персональних даних користувача. На рис. 1.4 візуалізовано

екосистему Apple HomeKit разом із наочними прикладами сумісного периферійного обладнання. Тут представлені спеціалізовані датчики безпеки, інтелектуальні вимикачі (наприклад, iDevices SWITCH), а також інтерфейс фірмового додатка «Дім» на різних мобільних пристроях компанії (iPad, iPhone та Apple Watch), що ілюструє процес контролю над підключеним хатнім обладнанням.

Рисунок 1.4 – Складові екосистеми Apple HomeKit та зразки підтримуваного обладнання для автоматизації

Керування всім розумним простором у цьому випадку відбувається виключно через централізований застосунок Apple Home або за допомогою фірмового голосового помічника Siri. Ключовою особливістю цієї платформи є надзвичайно високий рівень безпеки, який досягається завдяки алгоритмам наскрізного шифрування даних та жорсткому контролю доступу до системи. Це рішення ідеально підійде тим, хто вже є активним користувачем техніки Apple, оскільки воно пропонує максимально тісну та безшовну інтеграцію з iPhone, iPad, Apple Watch та розумними акустичними системами HomePod. Важливим кроком у розвитку платформи стала імплементація універсального протоколу Matter, що суттєво розширило можливості взаємодії з пристроями

від інших виробників. Користувачам також доступний функціонал створення складних автоматичних сценаріїв та гнучкого управління домом на основі даних геолокації. Це дозволяє системі, наприклад, автоматично вимикати все світло та переходити в режим енергоощадження, щойно останній член родини залишає межі будинку [3].

Своєрідним універсальним рішенням на ринку виступає Tuya Smart – масштабна міжнародна хмарна IoT-платформа. Вона пропонує компаніям та розробникам вже готові програмно-апаратні рішення для швидкого підключення та організації керування найрізноманітнішими «розумними» пристроями. Саме завдяки такій універсальності та гнучкості цю платформу як основу для своїх продуктів обрали сотні брендів по всьому світу. На рис. 1.5 продемонстровано багатство асортименту пристроїв, що здатні функціонувати в екосистемі Tuya Smart. Цей перелік охоплює практично все необхідне для побудови повноцінного розумного дому: розумні сирени безпеки, датчики витоку води, інфрачервоні сенсори руху, датчики контролю температури та вологості. Також до системи входять універсальні комунікаційні шлюзи, що працюють на базі протоколів Zigbee або Wi-Fi, розумні кнопки, розетки, спеціальні перехідники та навіть вилки з функцією розумної підсвітки. Усі ці апаратні засоби легко об'єднуються та контролюються через спеціалізований мобільний додаток.

Рисунок 1.5 – Варіанти розумних пристроїв, що функціонують на базі хмарної платформи Tuya Smart

Важливою технологічною перевагою платформи є одночасна підтримка кількох найбільш розповсюджених протоколів зв'язку, зокрема Wi-Fi, ZigBee та Bluetooth. Це надає змогу стороннім розробникам без перешкод інтегрувати власні апаратні новинки у єдину, глобальну екосистему. Серед головних переваг Tuya Smart експерти виділяють її вражаючу гнучкість та можливість практично безмежного масштабування архітектури. Платформа здатна безперебійно підтримувати колосальну кількість пристроїв найрізноманітнішого побутового призначення. Користувачі отримують зручний інструмент для дистанційного керування всією домашньою технікою через фірмові мобільні застосунки Tuya Smart або Smart Life. Крім того, система має відкритий характер і легко інтегрується з популярними голосовими асистентами, такими як Google Assistant та Amazon Alexa, що робить щоденне використання ще більш комфортним.

Окремої уваги в контексті розвитку IoT-технологій заслуговує вітчизняна розробка – компанія Ajax Systems. Це інноваційне українське підприємство, яке спеціалізується на створенні професійних систем безпеки та комплексних рішень для автоматизації, що базуються на передових принципах Інтернету речей. На рис.1.6 наведено зразки високотехнологічних пристроїв, що входять до складу екосистеми Ajax Systems і призначені для забезпечення надійної охорони та інтелектуального управління будь-якими приміщеннями. Серед них можна побачити магнітоконтактні сенсори відкриття вікон та дверей, інфрачервоні датчики руху (зокрема просунуті моделі з вбудованою фотокамерою для візуальної верифікації причин тривог серії MotionCam), а також потужні централі (хаби) для обробки даних, ретранслятори сигналу ReX для суттєвого розширення зони радіопокриття на великих об'єктах та гучні бездротові сирени.

Рисунок 1.6 – Елементи системи безпеки та автоматизації від виробника Ajax Systems

Професійне обладнання від Ajax Systems знаходить своє широке застосування не лише в приватних житлових будинках, а й в сучасних офісних

центрах та на об'єктах промислового призначення. Світовий успіх компанії ґрунтується на вдалому поєднанні абсолютної надійності роботи, сучасного стриманого дизайну та використання передових технологічних розробок. Фундаментальною особливістю побудови цієї системи є використання власного, закритого та глибоко зашифрованого радіопротоколу Jeweller, який гарантує стабільний та стійкий до перешкод зв’язок між усіма компонентами навіть на дуже великих відстанях. Екосистема включає цілий арсенал високочутливих інтелектуальних сенсорів, здатних миттєво реагувати на несанкціонований рух, появу диму, перші ознаки затоплення чи спроби силового проникнення. Для зручної взаємодії розроблено потужний та інтуїтивний мобільний застосунок, який забезпечує постійний моніторинг ситуації на об'єкті в режимі реального часу. Попри свою яскраво виражену професійну орієнтацію на безпеку, система є відкритою до взаємодії та підтримує інтеграцію з глобальними платформами розумного дому, такими як Google Home або Alexa.

Як можна побачити, сучасний технологічний ландшафт рясніє різноманітними платформами, кожна з яких покликана забезпечити стабільне функціонування систем Інтернету речей (IoT) як у домашньому побуті, так і в корпоративному середовирі. Оскільки кожне з цих рішень — Google Home, Apple HomeKit, Tuya Smart чи Ajax Systems [4] — володіє унікальною архітектурною будовою, специфічним набором функцій, використовує різні протоколи передачі даних та сповідує власні підходи до забезпечення кібербезпеки, вибір оптимального варіанту потребує системного аналізу. Ці комплекси кардинально відрізняються за філософією своєї побудови, методами взаємодії з периферійним обладнанням та глибиною можливої інтеграції в зовнішні інформаційні середовища. Для того, щоб більш наочно продемонструвати ці відмінності, деталізовану порівняльну характеристику вищезгаданих популярних IoT-систем було систематизовано та наведено у табл. 1.1.

Таблиця 1.1 – Порівняльна характеристика найпопулярніших платформ Інтернету речей

Параметр	Google Home	Apple HomeKit	Tuya Smart	Ajax Systems
Компанія-розробник	Google LLC	Apple Inc.	Tuya Inc.	Ajax Systems (Україна)
Основне призначення	Комплексна автоматизація розумного дому та управління мультимедійними системами.	Керування розумним простором із яскраво вираженим акцентом на безпеку та приватність.	Універсальна хмарна платформа для створення та управління IoT-пристроями різних брендів.	Професійна система безпеки з широкими можливостями комплексної автоматизації приміщень.
Тип архітектури системи	Централізована архітектура з опорою на хмарні обчислення.	Централізована архітектура з можливістю локального дублювання процесів.	Гібридна архітектура, що поєднує локальні та хмарні можливості.	Гібридна архітектура (робота через власний захищений сервер у поєднанні з хмарою).
Підтримувані протоколи зв’язку	Wi-Fi, Bluetooth, Thread, Matter.	Wi-Fi, Bluetooth, Thread, Matter.	Wi-Fi, ZigBee, Bluetooth, NB-IoT.	Пропрієтарні протоколи Jeweller та Wings, а також Ethernet, Wi-Fi, GSM.

Продовження таблиці 1.1.

Способи керування системою	Голосові команди через помічника Google Assistant, а також спеціалізований мобільний застосунок.	Голосові команди за допомогою асистента Siri та фірмовий застосунок Apple Home.	Мобільні застосунки Tuya Smart та Smart Life, інтеграція з голосовим керуванням через Alexa та Google Assistant.	Фірмовий мобільний застосунок Ajax, налаштування та виконання локальних сценаріїв автоматизації.
Основні функціональні можливості	Автоматизація систем освітлення, гнучкий клімат-контроль, керування мультимедіа та базові функції безпеки.	Автоматизація домашніх процесів, керування пристроями, створення інтелектуальних сценаріїв на основі геолокації користувача.	Масштабоване керування великою кількістю пристроїв одночасно, безшовна інтеграція рішень від різних виробників.	Цілодобовий моніторинг безпеки, миттєве оповіщення про тривоги, робота сенсорів руху, диму та затоплення.
Рівень безпеки даних	Високий, забезпечується надійною авторизацією через єдиний Google Account.	Дуже високий, що досягається завдяки глибокому шифруванню даних та жорсткій автентифікації пристроїв.	Середній, рівень захисту багато в чому залежить від конкретного виробника підключених пристроїв.	Надзвичайно високий, гарантується застосуванням власних, глибоко захищених протоколів зв’язку.

Продовження таблиці 1.1.

Можливості інтеграції з іншими системами	Висока сумісність із переважною більшістю пристроїв від сторонніх виробників IoT-обладнання.	Підтримка універсального протоколу Matter забезпечує широкі можливості взаємодії з екосистемами інших брендів.	Легка інтеграція з глобальними платформами Google Home, Amazon Alexa та Samsung SmartThings.	Підтримується інтеграція з Google Home, Alexa, а також можливість підключення до професійних пультів охоронних компаній.
Місце та спосіб зберігання даних	Хмарна інфраструктура Google.	Захищена хмара Apple (сервіс iCloud).	Хмарна платформа Tuya Cloud.	Дані зберігаються на фірмових серверах Ajax та локально у мобільному застосунку користувача.
Ключові особливості	Максимально глибока інтеграція з операційною системою Android та всіма супутніми сервісами Google.	Орієнтація на максимальну конфіденційність даних користувача та інтуїтивну простоту у використанні.	Абсолютно відкрита екосистема, пристосована для потреб сторонніх виробників обладнання.	Створення власної, незалежної екосистеми безпеки, що здатна працювати автономно навіть при втраті зовнішнього зв’язку.

Продовження таблиці 1.1.

Цільова аудиторія (тип користувача)	Побутовий сегмент, використання для потреб сім'ї.	Побутовий сегмент з орієнтацією на преміум-користувачів.	Широкий спектр використання: від побутового до комерційного сектору.	Універсальне застосування: побутовий, комерційний та великий корпоративний сектори.

Підсумовуючи всю інформацію та спираючись на детальний аналіз переваг і недоліків, висвітлених у порівняльній таблиці 1.1, можна зробити обґрунтований висновок щодо вибору оптимальної платформи. Для практичної реалізації надійної системи безпеки та комплексної автоматизації приміщення було прийнято рішення обрати екосистему Ajax Systems. Такий вибір є цілком закономірним і зумовлений цілою низкою вагомих факторів. Насамперед, це безпрецедентно високий рівень безпеки та захисту даних, який є критично важливим для охоронних систем. Крім того, система пропонує надзвичайно багатий функціональний інструментарій, гнучкі можливості для налаштування локальних сценаріїв автоматизації, виконання яких не залежить від стабільності зовнішнього інтернет-з'єднання. Наявність власного потужного хмарного сервера гарантує миттєву швидкість реакції системи, а сумісність з такими глобальними платформами, як Google Home чи Alexa, залишає широкий простір для подальшого масштабування можливостей розумного дому. Окремим, але не менш значущим бонусом є українське походження компанії-розробника, що гарантує користувачам бездоганну локалізацію інтерфейсів та оперативну, висококваліфіковану технічну підтримку безпосередньо від виробника.

1.3 Архітектурні моделі та підходи до проєктування систем «Розумний дім»

Архітектура систем Інтернету речей (IoT) виступає фундаментальним каркасом, що визначає принципи та правила організації взаємодії між усіма її складовими елементами. До цих елементів належать різноманітні сенсори, інтелектуальні контролери, комунікаційні шлюзи, потужні сервери обробки даних та користувацькі інтерфейси. Вибір конкретного архітектурного підходу під час проєктування є критично важливим кроком, адже він безпосередньо і суттєво впливає на кінцеву ефективність функціонування всієї системи, швидкість та затримки в обробленні масивів даних, загальну надійність та здатність до подальшого масштабування. Залежно від того, яким чином у системі відбувається розподіл обчислювальних ресурсів та як маршрутизуються потоки інформації, в інженерній практиці розрізняють три фундаментальні типи архітектур IoT: централізовану, розподілену та гібридну.

Централізована архітектура будується на принципі максимальної концентрації обчислень. Вона передбачає, що абсолютно всі дані, зібрані сенсорами та первинними контролерами, безперервно передаються по мережі до єдиного, потужного центрального вузла – яким може виступати локальний сервер або ж віддалена хмарна платформа. Саме на цьому магістральному рівні здійснюється весь цикл роботи з даними: їх надійне зберігання, глибоке оброблення, складний аналіз, а також формування логіки та керівних команд для всіх пристроїв системи. Таким чином, уся повнота прийняття рішень акумулюється централізовано, тоді як периферійні пристрої (сенсори та актуатори) виконують здебільшого пасивні функції первинного збору метрик та їх трансляції "нагору", а також беззаперечного виконання отриманих команд.

Безперечною перевагою такої архітектури є відносна простота адміністрування та технічної підтримки. Вона забезпечує виняткову зручність

глобального контролю та відкриває широкі можливості для централізованого аналізу колосальних обсягів інформації (Big Data). Проте цей підхід має і суттєві недоліки. Головними з них є надзвичайно високе навантаження на обчислювальні потужності центрального сервера та пропускну здатність каналів зв'язку. Це може призводити до помітних затримок у передачі даних та критичної залежності всієї екосистеми від стабільності мережевого з’єднання. У найгіршому сценарії, в разі виходу з ладу або тимчасової недоступності центрального вузла, система може частково або навіть повністю втратити свою працездатність, перетворивши "розумні" пристрої на звичайні [5].

Типовим і найпоширенішим прикладом використання централізованої архітектури є популярні масові системи «розумного будинку», які повністю покладаються на роботу через хмарні сервіси (наприклад, екосистеми Google Home чи Amazon Alexa). У таких системах усі домашні пристрої підключені до єдиного глобального інтернет-сервера компанії-провайдера.

Розподілена архітектура, навпаки, ґрунтується на концепції глибокої децентралізації процесів обробки інформації. У цій парадигмі значна частина обчислень та логічних операцій делегується і виконується безпосередньо на нижньому рівні – "на краю" мережі, тобто самими пристроями або локальними вузлами. Кожен такий елемент системи наділений власною, хоч і обмеженою, обчислювальною потужністю та здатний самостійно аналізувати дані й приймати оперативні рішення без необхідності постійного звернення до центрального сервера. Такий інноваційний підхід найчастіше реалізується за допомогою технології периферійних обчислень (edge computing).

Головними перевагами розподіленої архітектури є радикальне зменшення затримок у передачі даних (latency), оскільки інформація обробляється там же, де і генерується. Це призводить до значного підвищення загальної швидкодії системи та кардинального збільшення її автономності. Окремі локальні вузли або сегменти мережі можуть повноцінно функціонувати та підтримувати процеси навіть за повної відсутності доступу до глобальної мережі або центрального сервера. Разом із тим, проєктування та впровадження

такої архітектури є значно складнішим завданням. Вона вимагає розробки складних механізмів синхронізації розподілених баз даних, потребує використання значно потужніших (і відповідно дорожчих) периферійних пристроїв та висуває значно жорсткіші вимоги до забезпечення безпеки та цілісності локальних інформаційних потоків.

Цей розподілений підхід знайшов своє найширше застосування у сфері промислових IoT-систем (Industrial IoT або IIoT). На сучасному виробництві кожен розумний контролер або сенсор промислового обладнання здатен самостійно, в режимі реального часу, аналізувати критичні метрики та миттєво регулювати роботу верстата чи конвеєра, не чекаючи команди з диспетчерського центру.

Гібридна архітектура являє собою компромісний, але надзвичайно ефективний варіант, який гармонійно поєднує в собі елементи та переваги як централізованої, так і розподіленої моделей. Цей підхід дозволяє архітекторам оптимально збалансувати обчислювальне навантаження, раціонально розподіливши його між локальною обробкою "на краю" та централізованим глобальним керуванням. У гібридній системі первинна, найбільш ресурсомістка або критична до часу обробка даних (наприклад, фільтрація, агрегація або миттєва реакція на аварійні показники) виконується безпосередньо на рівні кінцевих пристроїв або локальних шлюзів (edge level). Тим часом, глибокий поглиблений аналіз, довгострокове зберігання масивів історичних даних (Big Data analytics), навчання моделей штучного інтелекту та формування стратегічних глобальних рішень здійснюється на потужних потужностях сервера чи хмарної платформи (cloud level).

Такий підхід забезпечує найбільш оптимальне співвідношення ключових метрик: швидкодії, системної гнучкості та загальної надійності. Він дозволяє суттєво скоротити марний обсяг сирих даних, що постійно передаються до хмари (економія трафіку), радикально знизити затримки у керуванні критичними пристроями та, водночас, зберегти всі переваги єдиного центру для глобального моніторингу та управління екосистемою. Водночас

варто визнати, що гібридна архітектура є найбільш складною в інженерній реалізації. Вона потребує бездоганної та ретельної координації процесів між абсолютно різними рівнями системи та глибокого узгодження різноманітних протоколів безперервного обміну даними.

Яскравим і масштабним прикладом успішного застосування гібридної архітектури є розгортання складних систем «розумного міста» (Smart City). У таких мегапроєктах частина оперативних даних (наприклад, оптимізація фаз світлофорів для регулювання дорожнього руху або локальне керування вуличним освітленням) обробляється локально, безпосередньо на рівні вуличних сенсорних вузлів чи контролерів кварталу. Тоді як глибока аналітична обробка великомасштабних даних (прогнозування заторів, оптимізація маршрутів громадського транспорту чи аналіз екологічного стану міста) акумулюється та здійснюється в глобальній хмарній інфраструктурі мегаполіса.

Варто також зазначити, що практична побудова будь-яких систем Інтернету речей (IoT) неможлива без спирання на стандартизовані архітектурні принципи та спеціалізовані протоколи обміну даними. Саме вони забезпечують надійну, безпечну та ефективну комунікаційну взаємодію між тисячами пристроїв, серверами обробки та кінцевими користувачами. Вибір конкретної моделі зв'язку чи мережевого протоколу завжди продиктований специфікою сфери застосування системи, очікуваним обсягом передаваних даних, жорсткістю вимог до швидкодії, надійності доставки пакетів та критичністю параметрів енергоспоживання (особливо для автономних датчиків).

Одним із наріжних аспектів проєктування IoT-систем є правильний вибір базової моделі комунікації. Ця модель фундаментально визначає, як саме, за якими правилами та в якій послідовності здійснюється передача інформації між усіма компонентами системи. У сучасній практиці найпоширенішими є традиційна клієнт-серверна модель комунікації та цілий спектр різноманітних

мережевих протоколів обміну даними, спеціально адаптованих для потреб IoT, таких як MQTT, HTTP, WebSocket, CoAP та AMQP [6].

Класична клієнт-серверна модель гарантує централізований та упорядкований обмін інформацією. У цій парадигмі клієнт (наприклад, датчик або мобільний додаток) завжди ініціює запити, а сервер слухає, обробляє їх та відповідає. Натомість, більш сучасні протоколи, такі як легковаговий MQTT чи потужний WebSocket, архітектурно орієнтовані на постійну, двосторонню та асинхронну взаємодію в режимі реального часу. Це робить їх незамінними та критично важливими для побудови ефективних систем безперервного моніторингу та оперативного керування.

У типовій, середньостатистичній системі Інтернету речей (IoT) складна взаємодія між парком периферійних пристроїв та центральною серверною частиною підпорядкована чітко визначеній та послідовній логіці. Ця логіка гарантує безперервний потік обміну інформацією, надійне керування пристроями та миттєвий зворотний зв’язок із кінцевим користувачем.

Увесь процес життєвого циклу функціонування такої системи можна детально описати у вигляді наступної послідовності взаємопов'язаних етапів:

Спочатку, на найнижчому фізичному рівні, розмаїття сенсорів та датчиків безперервно фіксує найменші зміни у навколишньому фізичному середовищі – це можуть бути коливання температури, зміни вологості, рівня освітленості, виявлення руху об'єктів чи перепади атмосферного тиску. Усі ці зібрані сирі дані негайно передаються до локальних мікроконтролерів або спеціалізованих модулів збору даних. Ці модулі виконують критично важливу функцію попередньої обробки "на льоту": вони здійснюють апаратну фільтрацію шумів та перешкод, виконують аналогово-цифрове перетворення (АЦП) сигналів та здійснюють базове структурування отриманих пакетів інформації.

Далі, вже підготовлена інформація маршрутизується та надсилається через спеціальний комунікаційний вузол – шлюз (gateway) безпосередньо до серверної частини або у хмарну платформу. У цій архітектурі шлюз відіграє

роль своєрідного "мосту" та інтелектуального посередника між закритою локальною мережею IoT-пристроїв (яка часто працює на специфічних протоколах типу ZigBee чи Bluetooth) та глобальним інтернетом (IP-мережею). Шлюз не просто транслює дані; він додатково агрегує їх у більші пакети для оптимізації трафіку, забезпечує надійне криптографічне шифрування для безпечної передачі та перетворює дані у стандартизований формат (наприклад, JSON), що є придатним для подальшого програмного аналізу на серверах.

На вищому, магістральному рівні – безпосередньо на потужних серверах або в хмарних кластерах – усі ці безперервні потоки даних надійно зберігаються у базах, проходять глибокий аналіз та складну інтерпретацію. Для цього активно застосовуються різноманітні аналітичні алгоритми, машинне навчання (ML) та системи штучного інтелекту (AI). Базуючись на результатах цього глибокого аналізу та заданих сценаріях, серверний мозок формує чіткі керівні команди для актуаторів системи. Наприклад, якщо датчики фіксують перегрів, сервер може прийняти рішення та відправити команду на зниження температури кондиціонера, автоматичне увімкнення фасадного освітлення при настанні сутінків або негайну активацію сирени сигналізації при виявленні руху в забороненій зоні.

Сформовані команди миттєво відправляються у зворотному напрямку – знову через мережу до локального шлюзу, а від нього безпосередньо до відповідних виконавчих пристроїв (актуаторів), які беззаперечно виконують необхідні механічні чи електричні дії. Протягом усього цього процесу власник системи (користувач) має повний контроль та можливість постійно спостерігати за роботою екосистеми. Він може в будь-який момент втручатися в автоматичні процеси, використовуючи зручний мобільний застосунок або веб-інтерфейс. Це забезпечує надійний зворотний зв’язок та дає користувачеві абсолютну владу дистанційного керування своїм розумним простором.

Візуальне відображення всіх цих складних потоків даних та логіки взаємодії компонентів у типовій системі IoT детально показано на рис.1.7.

Рисунок 1.7 – Структурна схема логіки взаємодії компонентів та потоків даних у системі IoT

2 РОЗРОБКА КОМПЛЕКСНОЇ АРХІТЕКТУРИ СИСТЕМИ «РОЗУМНИЙ ДІМ» НА БАЗІ ТЕХНОЛОГІЙ AJAX

2.1 Визначення цілей та формулювання завдання щодо комплексної модернізації архітектурних рішень

Сучасні комплекси «розумного дому» невід'ємно пов'язані з концепцією Інтернету речей (IoT), об'єднуючи під одним дахом величезний масив різноманітного обладнання від десятків різних виробників. Сюди входять датчики, керуючі контролери, елементи розумного освітлення, модулі клімат-контролю та камери відеоспостереження. Попри таку технологічну різноманітність, ключовою перепоною стає те, що ці пристрої часто не здатні повноцінно "спілкуватися" між собою. Їхня взаємодія виявляється суттєво обмеженою, а подекуди й повністю неможливою через кардинальні відмінності у використовуваних протоколах зв’язку, стандартах шифрування та форматах передачі даних, а також через несумісність програмних інтерфейсів (API).

Ця проблема набуває особливої гостроти, коли мова йде про спроби глибокої інтеграції вузькоспеціалізованих охоронних систем, зокрема рішень від Ajax Systems, у загальну архітектурну канву «розумного дому». Історично склалося так, що більшість професійних комплексів безпеки проєктуються як закриті, повністю автономні екосистеми. Вони не мають стандартизованих шлюзів чи єдиного каналу для безперешкодного обміну даними з іншим побутовим IoT-обладнанням і, як правило, не підтримують можливості централізованого керування через сторонні, загальні веб-інтерфейси чи застосунки. Як наслідок, кінцевий користувач стикається з необхідністю постійно перемикатися між кількома абсолютно різними додатками чи програмними платформами, щоб контролювати окремі підсистеми своєї оселі.

Такий фрагментований підхід не лише нівелює саму ідею комфорту та зручності, але й парадоксальним чином знижує загальний рівень безпеки та операційної ефективності управління житлом.

Додатковим викликом є критичний брак взаємодії на рівні подій: тривожні сповіщення та дані про статус безпеки (наприклад, сигнали від датчиків Ajax) майже неможливо напряму пов'язати з логікою роботи систем домашньої автоматизації. Через це стає складно налаштувати елементарні, але життєво важливі сценарії, як-от миттєве ввімкнення всього освітлення в будинку чи автоматичне блокування електронних замків на дверях у момент спрацювання охоронної сигналізації.

Отже, фундаментальна проблема зводиться до гострої нестачі єдиного, продуманого інтеграційного рішення. Ринок потребує архітектури, яка б змогла органічно та безшовно об'єднати специфічну охоронну систему Ajax із широким спектром інших IoT-компонентів «розумного будинку», створюючи єдиний, цілісний механізм. Такий механізм має гарантувати не лише зручне централізоване керування та цілодобовий моніторинг, але й абсолютно безпечний, зашифрований обмін даними між усіма без винятку підключеними пристроями.

Виходячи з описаної проблематики, головною метою даного проєктування є розробка інноваційної, комплексної інтегрованої архітектури системи «Розумний будинок». Вона покликана гармонійно поєднати передові технології Інтернету речей (IoT) із потужним функціоналом професійної охоронної системи Ajax, створюючи єдине, нерозривне інформаційне середовище для всебічного керування, безперервного моніторингу та гарантування максимальної безпеки житлових об’єктів [7]. Концепція запропонованої архітектури полягає у налагодженні безперебійної та стандартизованої взаємодії між абсолютно всіма складовими системи. Сюди відносяться різноманітні сенсори, обчислювальні контролери, центральний сервер, клієнтські мобільні застосунки та безпосередньо сам охоронний

комплекс. Ця взаємодія має будуватися виключно на основі надійних та уніфікованих протоколів обміну даними.

Практична реалізація цієї мети передбачає створення потужної централізованої платформи, яка відкриє перед користувачем принципово нові можливості. Насамперед, це здатність здійснювати повноцінний віддалений контроль та інтуїтивно зрозуміле керування всім парком пристроїв через єдиний зручний веб-портал або мобільний інтерфейс. Крім того, така глибока інтеграція охоронних модулів Ajax у загальну IoT-інфраструктуру дозволить якісно підвищити загальний рівень безпеки об'єкта. Окремим надважливим завданням є реалізація підтримки складних автоматизованих сценаріїв. Система повинна вміти самостійно та миттєво реагувати на критичні події, наприклад, автоматично вмикати прожектори чи превентивно блокувати всі входи у разі найменшого спрацювання охоронної сигналізації. При цьому базовою та непорушною вимогою до всієї цієї екосистеми залишається забезпечення максимально стабільного, швидкого та криптографічно захищеного обміну даними між усіма її підсистемами [8].

Підсумовуючи, можна стверджувати, що стратегічне завдання проєктування зводиться до формування єдиної, максимально гнучкої та стійкої до кіберзагроз архітектури. Її впровадження має не лише розширити функціональні горизонти звичного «розумного будинку», а й вивести на новий рівень показники надійності та повсякденного комфорту для користувача, ефективно синтезуючи весь потенціал сучасних технологій IoT та безкомпромісної безпеки від Ajax.

2.2 Підбір оптимального комплексу апаратного забезпечення для формування IoT-інфраструктури об'єкта на базі типових компонентів

Під час розробки апаратної частини системи «Розумний дім» ключовим етапом є підбір надійних та сумісних електронних компонентів. Нижче наведено детальний огляд обраного обладнання, що формує інфраструктуру Інтернету речей (IoT) на об'єкті.

Для реалізації обчислювального ядра системи були обрані мікроконтролери, що відповідають специфіці конкретних завдань. Для керування основними сенсорними модулями моніторингу (такими як вузли фіксації відкриття дверей IoTOpen, детектори газу IoTGas та протікання IoTLeak, а також датчики мікроклімату IoTTemp і приймачі інфрачервоних сигналів IoTIR) як базову платформу використано плати NodeMCU, побудовані на потужному чипі ESP-32. Таке рішення обґрунтоване тим, що слабші мікроконтролери не здатні забезпечити належний рівень швидкодії та стабільності при обробці багатьох потоків інформації. З іншого боку, для керування виконавчими пристроями, наприклад розумною розеткою (SmartSocket), де обчислювальне навантаження є мінімальним, застосовано економічну плату Wemos D1 mini на базі архітектури ESP-8266. Це дозволяє суттєво оптимізувати загальне енергоспоживання системи та знизити собівартість кінцевого пристрою [9].

Серверна частина, яка відповідає за маршрутизацію та обмін повідомленнями між усіма компонентами IoT-мережі, реалізована на базі MQTT-брокера. Для виконання цієї ролі було обрано мікрокомп’ютер Raspberry Pi 3. Цей одноплатний комп'ютер є ідеальним кандидатом для розгортання локального сервера, оскільки він поєднує в собі компактні розміри, низьке споживання електроенергії та цілком достатню обчислювальну потужність для безперебійного управління мережевим трафіком.

Автономне функціонування бездротових сенсорних модулів забезпечується надійним джерелом живлення, в якості якого виступає літій-іонна акумуляторна батарея стандарту 18650 від виробника Samsung. Цей елемент, схематично зображений на рис. 2.1, характеризується значною ємністю та високим ступенем надійності порівняно з бюджетними аналогами на ринку. Акумулятор гарантує тривалу роботу завдяки широкому діапазону робочої напруги: від максимальних 4.2 вольта при повному заряді до мінімально допустимих 2.8 вольта.

Рисунок 2.1 – Акумуляторна батарея формату 18650 від компанії Samsung

Для того щоб продовжити термін служби акумулятора та уникнути його виходу з ладу через глибокий розряд, у схему інтегровано спеціалізований контролер живлення TP4056. Цей модуль, наведений на рис. 2.2, відіграє роль запобіжника: він автоматично розриває ланцюг живлення, як тільки напруга на елементі опускається нижче критичної позначки у 2.8 В. Крім того, мікросхема здатна стабільно пропускати через себе струм силою до 1 ампера, що формує надійний запас потужності для безперебійної роботи підключеної периферії.

Рисунок 2.2 – Модуль TP4056 для безпечного керування процесами заряду та розряду

Паралельно з контролером заряду, для точного моніторингу енергоспоживання використовується вимірювальний модуль на базі високоточної мікросхеми INA219. Як показано на рис. 2.3, цей сенсор відповідає за зчитування ключових параметрів постійного струму, таких як робоча напруга, сила струму та загальна споживана потужність. Завдяки відсутності температурного дрейфу показників, компактності та малій вазі, цей чип ідеально вписується в архітектуру вбудованих IoT-рішень. Особливістю мікросхеми є її здатність двонаправленого вимірювання струму — вона автоматично адаптується до зміни полярності, що гарантує бездоганну точність збору даних навіть за умов динамічної зміни навантаження в мережі [10].

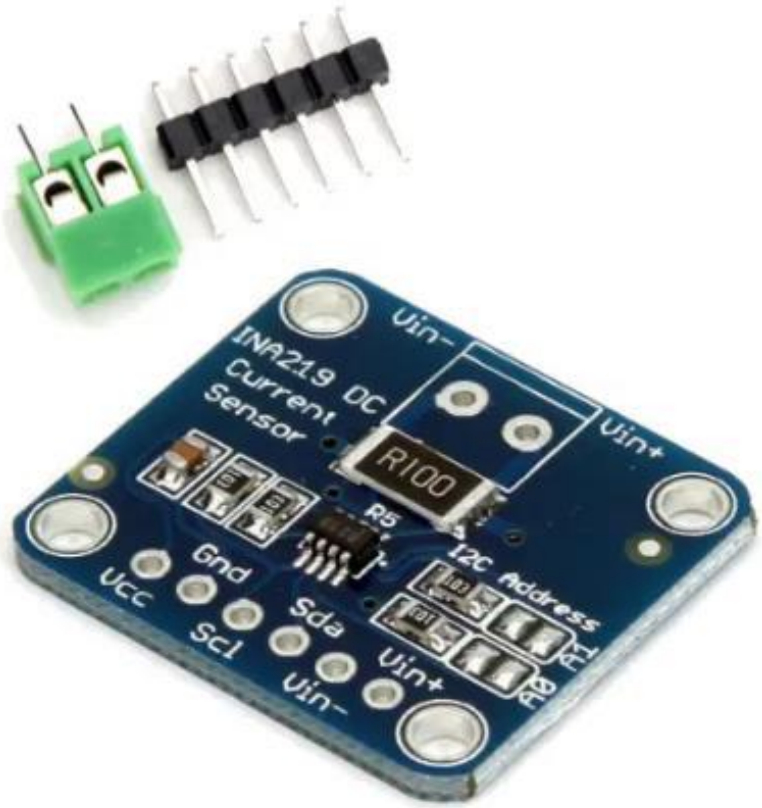

Рисунок 2.3 – Цифровий сенсор INA219 для комплексного моніторингу струму та напруги

Оскільки мікроконтролери та датчики вимагають стабільної напруги живлення на рівні 5 вольт, а напруга на акумуляторі постійно коливається в межах 2.8–4.2 В, виникає потреба у застосуванні підвищувального DC-DC перетворювача струму. Цей компактний електронний модуль, зовнішній вигляд якого наведено на рис. 2.4, приймає змінну напругу від батареї та конвертує її у стабільні 5 вольт на виході, забезпечуючи тим самим коректну роботу всієї обчислювальної логіки.

Рисунок 2.4 – Електронний DC-DC перетворювач для підвищення та стабілізації напруги

Для стаціонарних пристроїв, що живляться безпосередньо від побутової електромережі 220 В, необхідне перетворення змінного струму (AC) у постійний (DC). Для цього завдання обрано надійний імпульсний модуль HLK-PM1, зображений на рис. 2.5. Цей блок живлення відзначається високим коефіцієнтом корисної дії та відмінною стабілізацією вихідного сигналу. Модуль здатний працювати з вхідною напругою в діапазоні від 100 до 240 вольт змінного струму, видаючи на виході чіткі 5 вольт постійного струму при максимальному навантаженні до 600 міліампер. Пристрій адаптований до суворих умов експлуатації та може працювати при температурах від мінус 20 до плюс 60 градусів за Цельсієм. Маючи вагу близько 18 грамів та мініатюрні розміри 34 на 20 на 15 міліметрів, він легко розміщується у прихованих монтажних коробках.

Рисунок 2.5 – Мініатюрний імпульсний перетворювач змінного струму в постійний HLK-PM1

За збір даних про мікроклімат у приміщеннях відповідає цифровий датчик температури та вологості DHT-22, зовнішній вигляд якого представлено на рис. 2.6. Цей сенсор зарекомендував себе як надійне рішення завдяки високій точності вимірювань та легкості програмного поєднання з мікроконтролерами [12]. Елемент функціонує при напрузі живлення від 3 до 6 вольт і здатен вимірювати температуру навколишнього середовища від екстремальних мінус 40 до плюс 80 градусів за Цельсієм. При цьому похибка показників становить не більше ніж 0,5 градуса, а фіксація найменших температурних змін відбувається з кроком в 0,1 градуса.

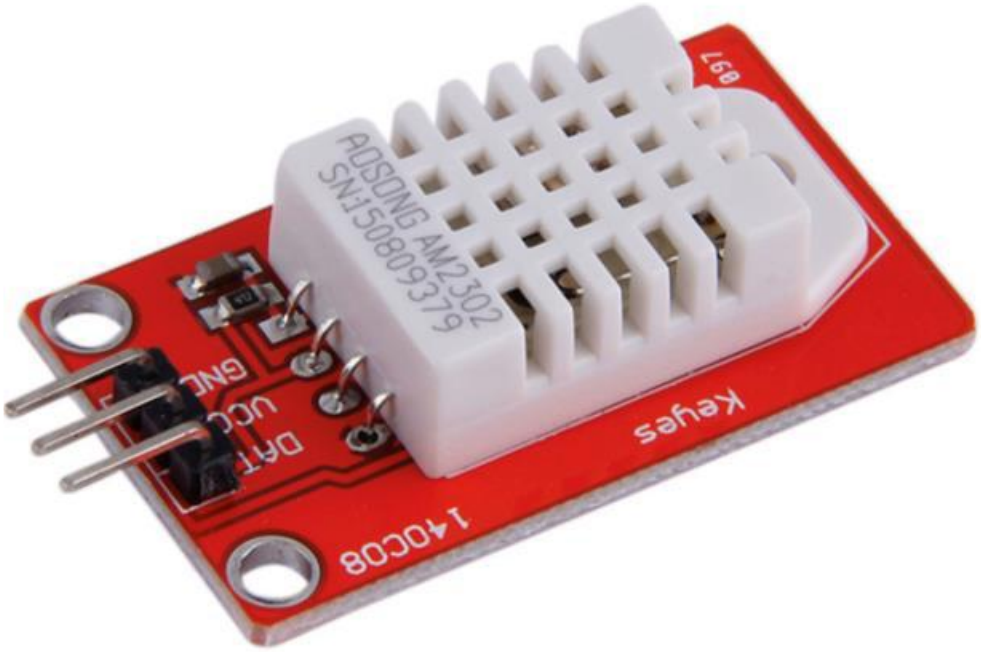

Рисунок 2.6 – Цифровий сенсор мікроклімату та температури простору DHT-22

Для забезпечення взаємодії системи зі звичними побутовими пультами дистанційного керування до архітектури включено модуль інфрачервоного приймача на базі елемента VS1838B. Як видно з рисунка 2.7, це невеликий компонент, що виконує роль "очей" системи для реєстрації та подальшого декодування ІЧ-імпульсів. Завдяки продуманій архітектурі, приймач успішно фільтрує сторонні електромагнітні завади. Модуль відзначається високою енергоефективністю, споживаючи менш як 1.5 міліампера від джерела з напругою від 2.7 до 5.5 вольт. Його оптичні характеристики дозволяють впевнено приймати команди на дистанції до 20 метрів у межах широкого кута чутливості близько 90 градусів.

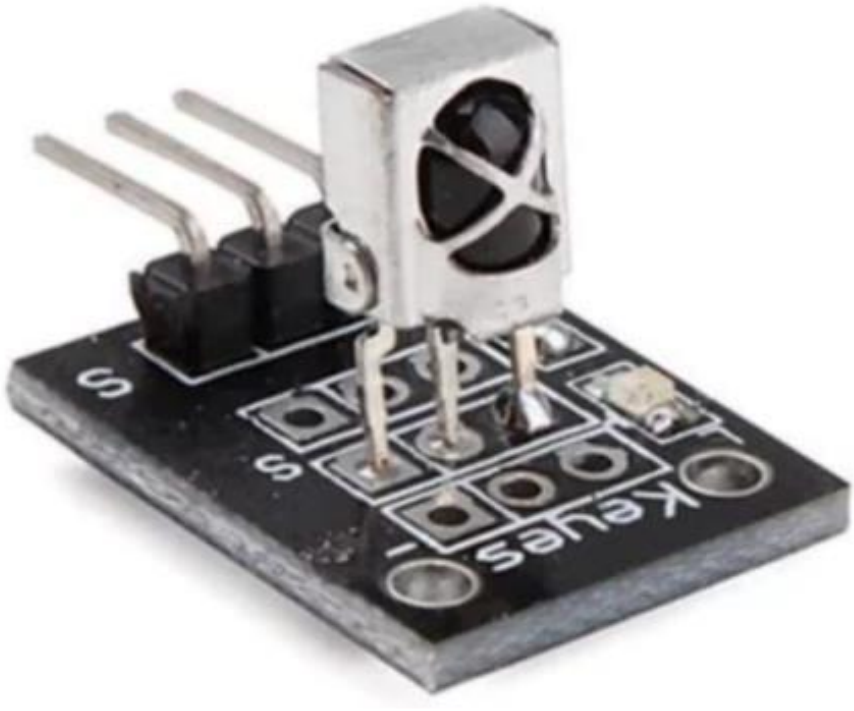

Рисунок 2.7 – Модуль оптичного прийому інфрачервоних команд типу VS1838B

Зворотний зв'язок та можливість інтелектуальної системи самостійно керувати старою побутовою технікою реалізовано через модуль інфрачервоного передавача KY-005, зображений на рис. 2.8. Завдяки інфрачервоному каналу трансляції даних, що є абсолютно нечутливим до радіоперешкод та не засмічує ефір домашньої Wi-Fi мережі, цей модуль стає ідеальним рішенням для імітації пультів керування. Застосування ІЧ-світлодіодів дозволяє створити надійний оптичний міст, що робить модуль KY-005 незамінним компонентом для прототипування алгоритмів домашньої автоматизації.

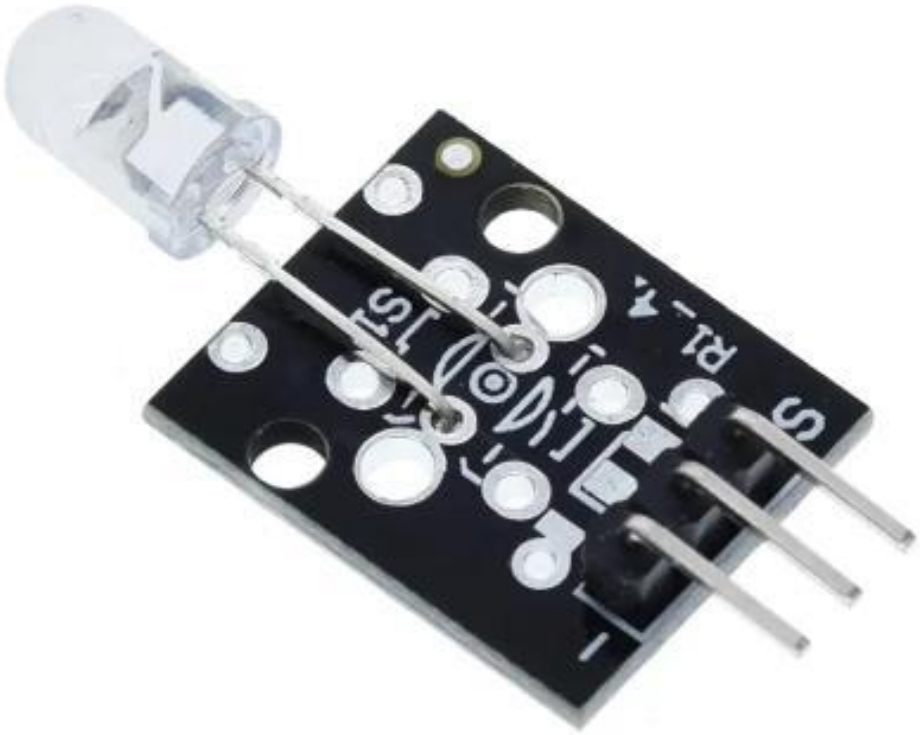

Рисунок 2.8 – Інфрачервоний передавальний модуль керування технікою KY-005

Критично важливим елементом безпеки є контроль якості повітря, для чого використовується газоаналізатор MQ-5, представлений на рис. 2.9. Його головне завдання — оперативне виявлення витоків небезпечних газів, таких як зріджені вуглеводні, метан, пропан, ізобутан та токсичний коксовий газ. Висока чутливість сенсора дозволяє системі миттєво підняти тривогу ще до досягнення критичної концентрації небезпечної речовини [13]. Датчик живиться від 5 вольт, працює в межах температур від мінус 10 до плюс 50 градусів і має високу точність фіксації в діапазоні 200–10000 частин на мільйон (ppm) для основних видів побутового газу. Його чутливість налаштована таким чином, щоб реагувати на концентрацію, яка перевищує безпечний фоновий рівень у п'ять разів. Модуль розміром 32 на 22 на 30 міліметрів видає як аналоговий, так і цифровий логічний сигнал, що робить його універсальним для підключення.

Рисунок 2.9 – Аналого-цифровий газовий сенсор безпеки MQ-5

Для фізичного включення та вимкнення потужних електроприладів у системі передбачено використання електромеханічного реле SLA-12VDC-SL-C, зовнішній вигляд якого наведено на рис. 2.10. Цей модуль слугує надійним бар'єром між високовольтною мережею та слабкострумовою керуючою електронікою. Завдяки потужним контактам, реле здатне безпечно комутувати струми до 30 ампер. При цьому для його активації мікроконтролеру достатньо надіслати імпульс струмом лише близько 5 міліампер. Плата також оснащена спеціальним резистором для захисту від перевантажень на вході, що гарантує тривалий термін експлуатації комутатора.

Рисунок 2.10 – Силове електромеханічне реле SLA-12VDC-SL-C для керування потужним навантаженням

Ще одним рубежем захисту житла є датчик наявності води T1592, який запобігає масштабним затопленням. Його зображено на рис. 2.11. Цей аналоговий сенсор контактного типу працює за простим фізичним принципом: при потраплянні води на відкриті металеві доріжки активної зони (розміром 40 на 16 міліметрів) різко змінюється електричний опір, і мікроконтролер миттєво фіксує аварію [14]. Датчик є вкрай економним, споживаючи менш ніж 20 міліампер від напруги 3–5 вольт, і стабільно функціонує в типових побутових умовах при температурах від +10 до +30 градусів та відносній вологості повітря до 90 відсотків.

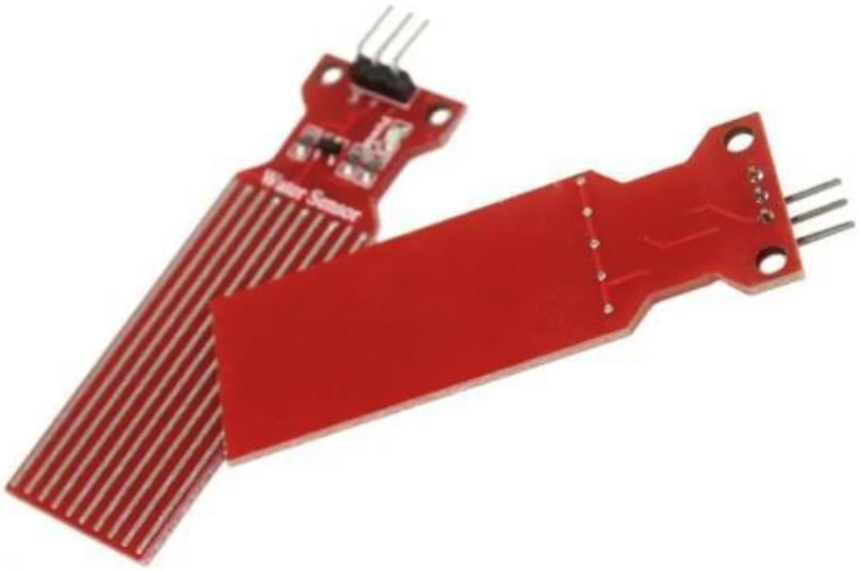

Рисунок 2.11 – Контактна плата датчика виявлення протікання рідини T1592

Для охорони периметра та контролю стану дверей, вікон або воріт використовується магнітоконтактний сенсор SP-01, зображений на рис. 2.12. Його конструкція складається з двох елементів, і як тільки магнітна частина віддаляється від основної, контакти розмикаються, генеруючи сигнал тривоги. Величезною перевагою цього датчика є його абсолютна автономність і пасивність: він підключається безпосередньо до керівної плати по двох проводах і не потребує жодного додаткового живлення [15].

Рисунок 2.12 – Магнітоконтактний геркон SP-01 для контролю відкриття

Фундаментом для побудови бездротової мережі та об'єднання всіх IoT-компонентів виступає високопродуктивний маршрутизатор TP-Link Archer C80, зовнішній вигляд якого наведено на рис. 2.13. Вибір цього пристрою зумовлений його здатністю забезпечувати стабільний зв'язок без затримок навіть за наявності великої кількості підключених смарт-гаджетів [10]. Роутер підтримує одночасну передачу даних у двох частотних діапазонах (2.4 ГГц і 5 ГГц), досягаючи загальної бездротової пропускної здатності до 1900 мегабіт за секунду. Для дротового підключення обладнання доступні гігабітні порти: один вхідний WAN та чотири локальні LAN-порти, що гарантує максимальну швидкість обміну інформацією.

Рисунок 2.13 – Потужний дводіапазонний Wi-Fi маршрутизатор TP-Link Archer C80

За візуальний контроль ситуації всередині будинку відповідає професійна інтелектуальна IP-камера Ajax IndoorCam White. Цей пристрій,

представлений на рис. 2.14, органічно поєднує можливості відеоспостереження з функціями активної безпеки [16]. Ядром камери є сучасна 4-мегапіксельна CMOS-матриця, яка видає деталізоване зображення з роздільною здатністю 2560 на 1440 пікселів та широким кутом огляду до 110 градусів. Унікальною особливістю моделі є інтегрований апаратний PIR-сенсор, який точно розпізнає рух теплових об'єктів (наприклад, людей), автоматично починаючи запис події. Камера легко підключається до локальної мережі Wi-Fi на частоті 2.4 ГГц та інтегрується з захищеним хмарним середовищем AJAX Cloud, забезпечуючи власникові цілодобовий доступ до трансляції через зручний мобільний додаток. Пристрій живиться від стандартних 5 вольт постійного струму та оснащений потужним інфрачервоним підсвічуванням, що дозволяє отримувати чітке зображення в повній темряві на дистанції до 10 метрів.

Рисунок 2.14 – Розумна IP-камера Ajax IndoorCam White з роздільною здатністю 4 Мп та PIR-сенсором

2.3 Схемотехнічні рішення для підключення обладнання та логіка взаємодії системних вузлів

На етапі практичної реалізації системи вкрай важливо чітко розуміти логіку взаємодії та правильність підключення всіх електронних компонентів. Детальний аналіз цих схем дозволяє уникнути помилок під час монтажу та гарантує стабільну роботу кожного вузла.

Почнемо з розгляду схеми електричних з'єднань сенсорного модуля моніторингу температури та вологості, який класифікується як «IoTTemp». Цю схему детально подано на рис. 2.15.

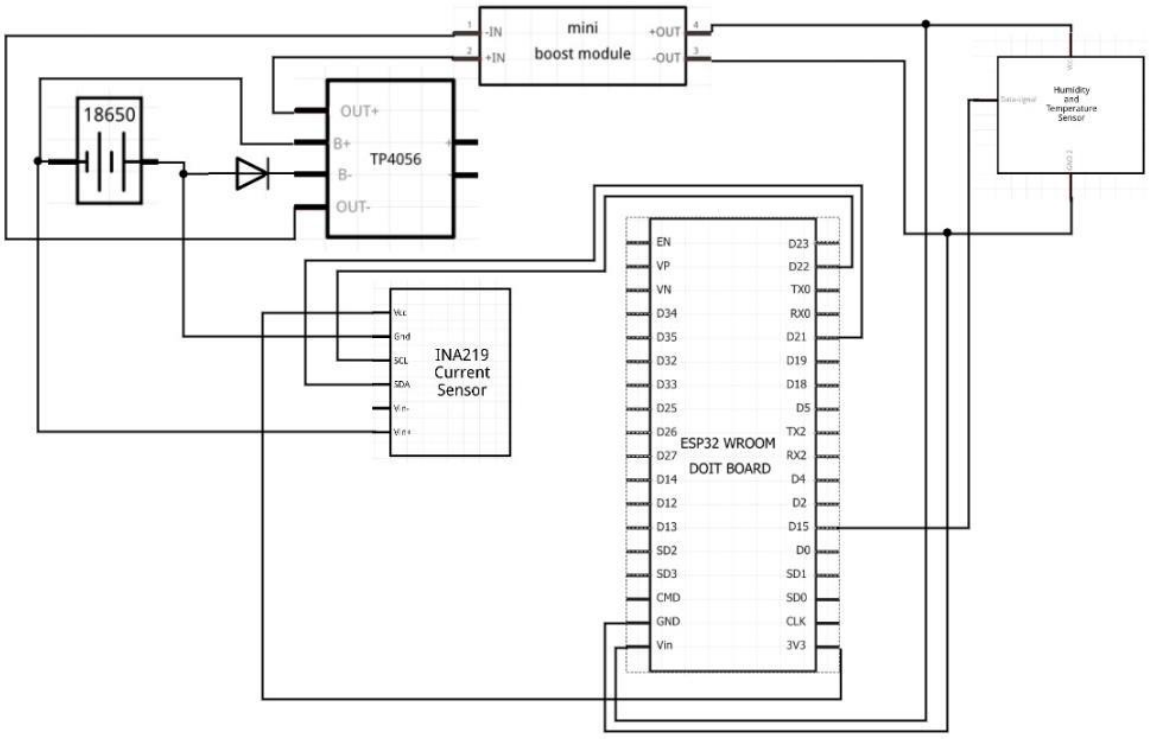

Рисунок 2.15 – Схемотехнічна модель взаємодії компонентів температурного вузла «IoTTemp»

Аналізуючи представлену схему підключення «IoTTemp», варто акцентувати увагу на кількох критично важливих інженерних рішеннях. Перш

за все, у ланцюгу живлення, безпосередньо перед входом модуля контролю заряду TP4056, встановлено напівпровідниковий діод. Його головна місія — виконувати роль своєрідного "запобіжника", що захищає всю чутливу електроніку від фатальних наслідків у разі випадкової зміни полярності (переплутування плюса і мінуса) під час підключення батареї. Проте використання діода має і зворотний ефект: під час проходження через нього електричного струму відбувається неминуче і невелике падіння напруги. Цей, здавалося б, незначний фактор може негативно вплинути на прецизійність показників вимірювального модуля INA219.

Для того щоб елегантно обійти цю фізичну перепону та гарантувати ідеальну точність зчитування енергоспоживання, застосовано специфічне схемотехнічне рішення: негативний полюс акумулятора (мінус) з'єднується з контактом заземлення («GND») мікросхеми INA219 не після, а безпосередньо до захисного діода. Таке підключення дозволяє повністю нівелювати вплив падіння напруги на вимірювання. Після збору даних про поточний стан батареї, модуль INA219 передає цю інформацію до головного мозку пристрою — мікроконтролера NodeMCU (ESP32) — використовуючи надійну та швидкісну цифрову шину передачі даних стандарту I²C. Окремо, окрім ліній живлення, до центральної плати мікроконтролера приєднано безпосередньо сам сенсор мікроклімату DHT22. Його завдання полягає у безперервному вимірюванні температури та вологості навколишнього середовища [17]. Враховуючи специфіку цього датчика, який формує вихідний сигнал спеціального формату, його лінія даних підключається до визначеного цифрового порту мікроконтролера (на схемі це порт D15) для подальшої програмної обробки.

Наступним кроком розглянемо архітектуру підключення інфрачервоного комунікаційного модуля, що має назву «IoTIR». Відповідна схема наведена на рис. 2.16.

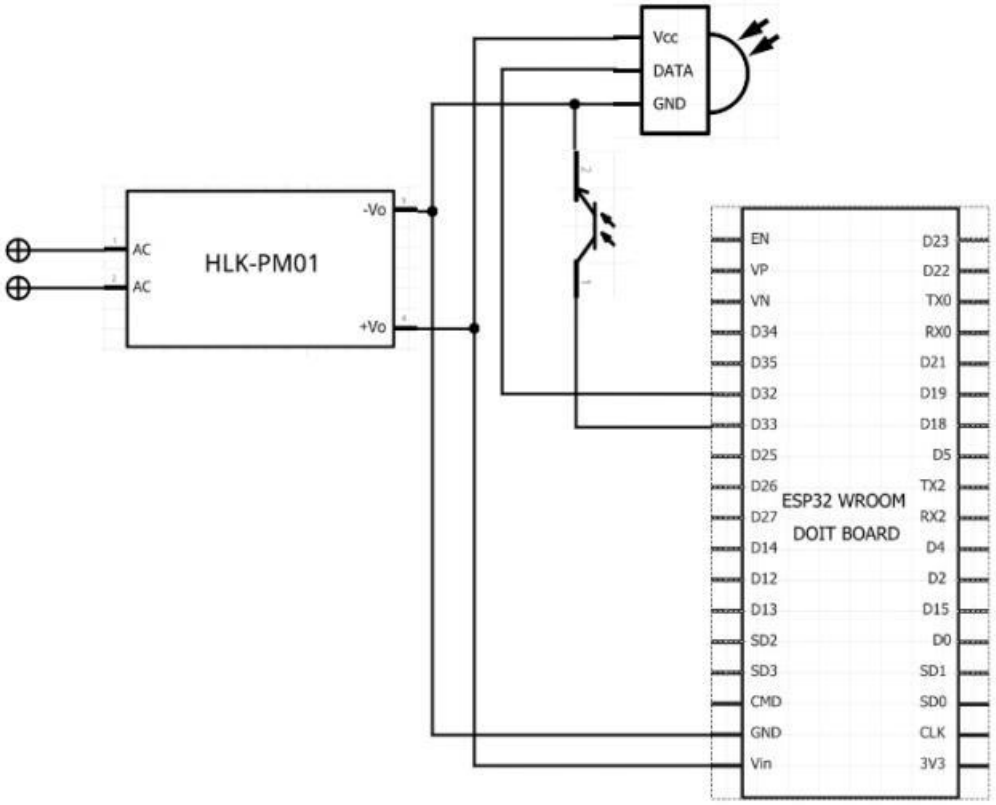

Рисунок 2.16 – Принципова електрична схема комутації елементів інфрачервоного модуля «IoTIR»

Ця схема наочно демонструє організацію підсистеми живлення та логіку обміну сигналами. Оскільки пристрій розрахований на стаціонарну роботу, його живлення здійснюється безпосередньо від побутової електромережі 220 В. Для цього використовується імпульсний перетворювач AC–DC типу HLK-PM1, який знижує та стабілізує високу змінну напругу до безпечних і постійних 5 вольт, необхідних для роботи електроніки. Ця стабільна напруга живить як саму плату мікроконтролера NodeMCU, так і чутливий модуль інфрачервоного приймача (IR Receiver).

Взаємодія з оптичними компонентами розподілена за різними каналами мікроконтролера. Так, інфрачервоний світлодіод (IR LED), який виконує роль передавача команд (наприклад, для керування телевізором чи кондиціонером), отримує керівні імпульси та живлення від цифрового піну D33. Натомість,

приймач (IR Receiver), який "слухає" ефір і реєструє сигнали від звичайних пультів дистанційного керування, транслює зчитані та попередньо декодовані дані через порт D32. Така конфігурація забезпечує повноцінний двосторонній оптичний зв'язок між системою розумного будинку та сторонньою побутовою технікою.

Далі звернемо увагу на схему підключення виконавчого пристрою — розумної розетки «IoTSocket», яка представлена на рис. 2.17.

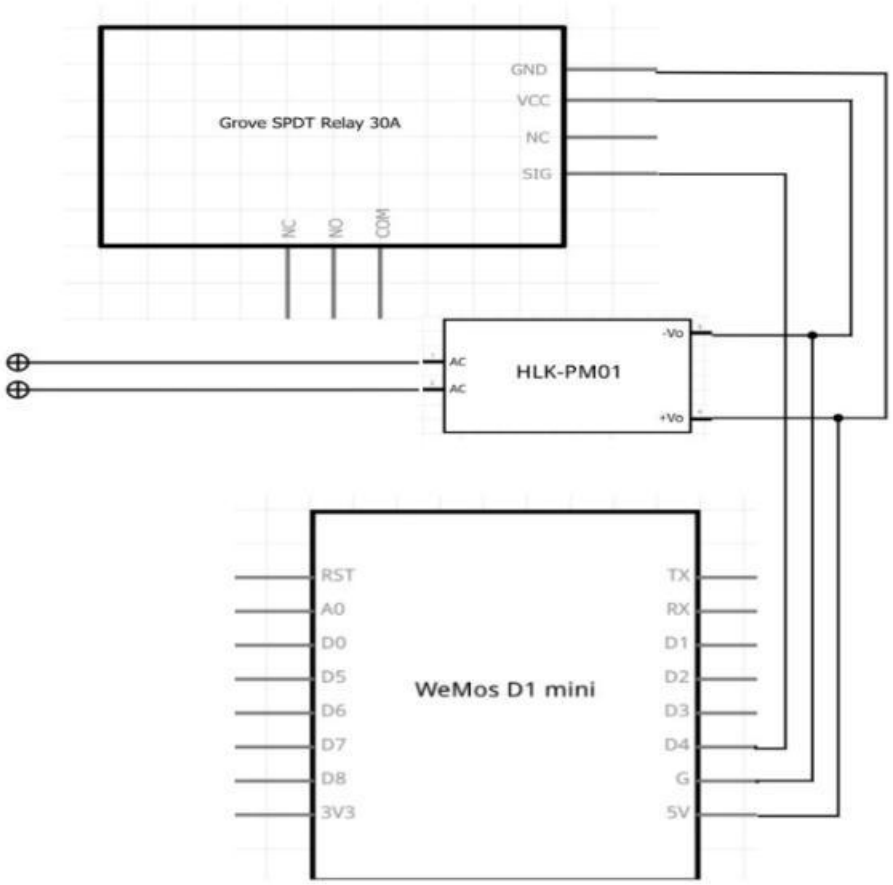

Рисунок 2.17 – Схема електричних з’єднань для реалізації керування навантаженням розумної розетки «IoTSocket»

На цій схемі ключовим виконавчим елементом виступає потужне силове реле. Керування його станом (замкнуто/розімкнуто) повністю делеговано цифровому виходу D4 мікроконтролера Wemos D1 mini. Відправляючи логічний сигнал на цей порт, система отримує здатність дистанційно подавати

або припиняти подачу електроенергії до будь-якого підключеного до розетки навантаження (світильника, обігрівача тощо).

Переходячи до підсистем безпеки, розглянемо рис. 2.18, де детально проілюстровано схему підключення модуля «IoTGas». Цей вузол виконує критично важливу функцію безперервного моніторингу концентрації вибухонебезпечних газів у повітрі приміщення.

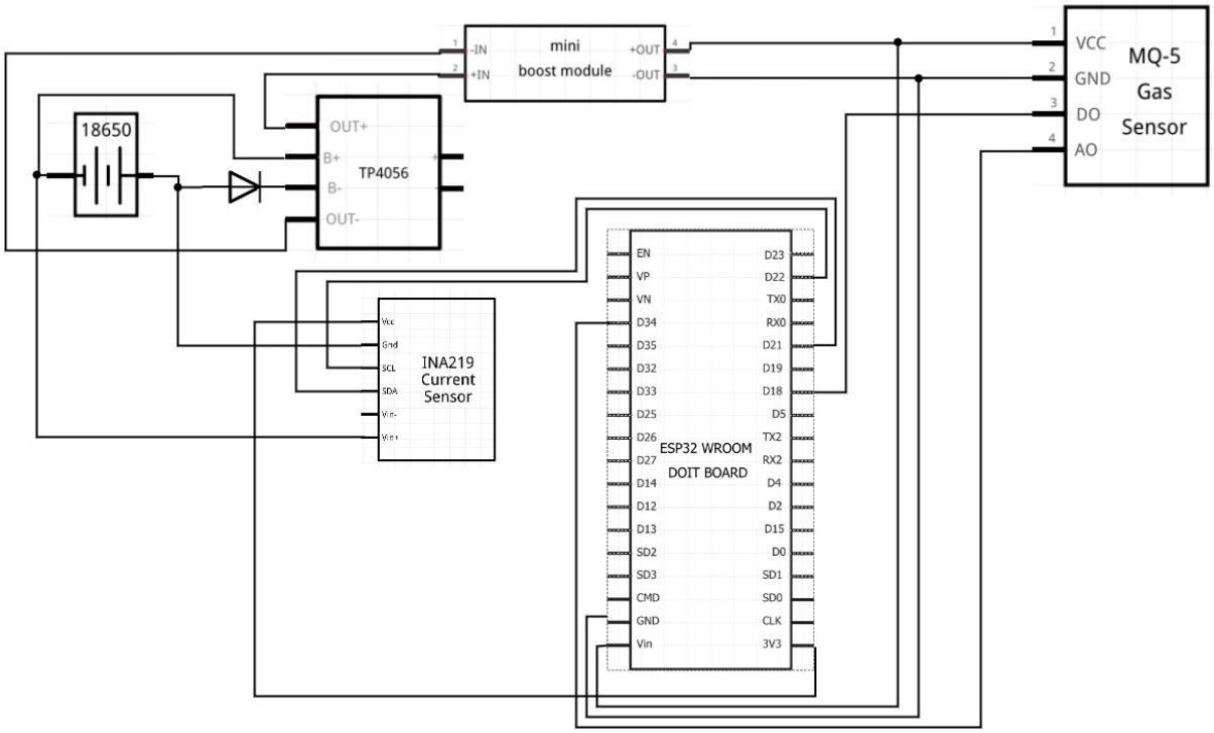

Рисунок 2.18 – Структурна організація та схема підключення газового сенсора «IoTGas»

Загальна архітектура живлення та керування цього модуля багато в чому перегукується зі схемою температурного сенсора «IoTTemp» (що розглядалася на рис. 2.15). Проте він має суттєві відмінності, зумовлені специфікою роботи газоаналізатора MQ-5 [18]. Найбільшої уваги тут заслуговує система передачі даних від сенсора до мікроконтролера, яка реалізована одразу через дві незалежні інформаційні лінії.

Перша лінія передачі даних підключена до цифрового порту D18. Цей канал працює за принципом простого логічного тригера: він миттєво

активується (подає сигнал високого рівня) лише у тому випадку, коли концентрація газу в приміщенні різко перевищує безпечний пороговий рівень, формуючи швидкий сигнал тривоги. Друга лінія є аналоговою і підключена до порту D34. Завдання цього каналу — передавати мікроконтролеру безперервний сигнал, напруга якого пропорційно змінюється залежно від поточної концентрації газу. Такий дворівневий підхід дозволяє системі не просто фіксувати факт витоку, а й вести точний, динамічний моніторинг стану атмосфери в режимі реального часу.

Ще одним компонентом безпеки периметра є модуль контролю доступу «IoTOpen», схема якого наведена на рис. 2.19. Цей пристрій призначений для відстеження стану стулок (відчинено або зачинено) на дверях, вікнах чи навіть гаражних воротах.

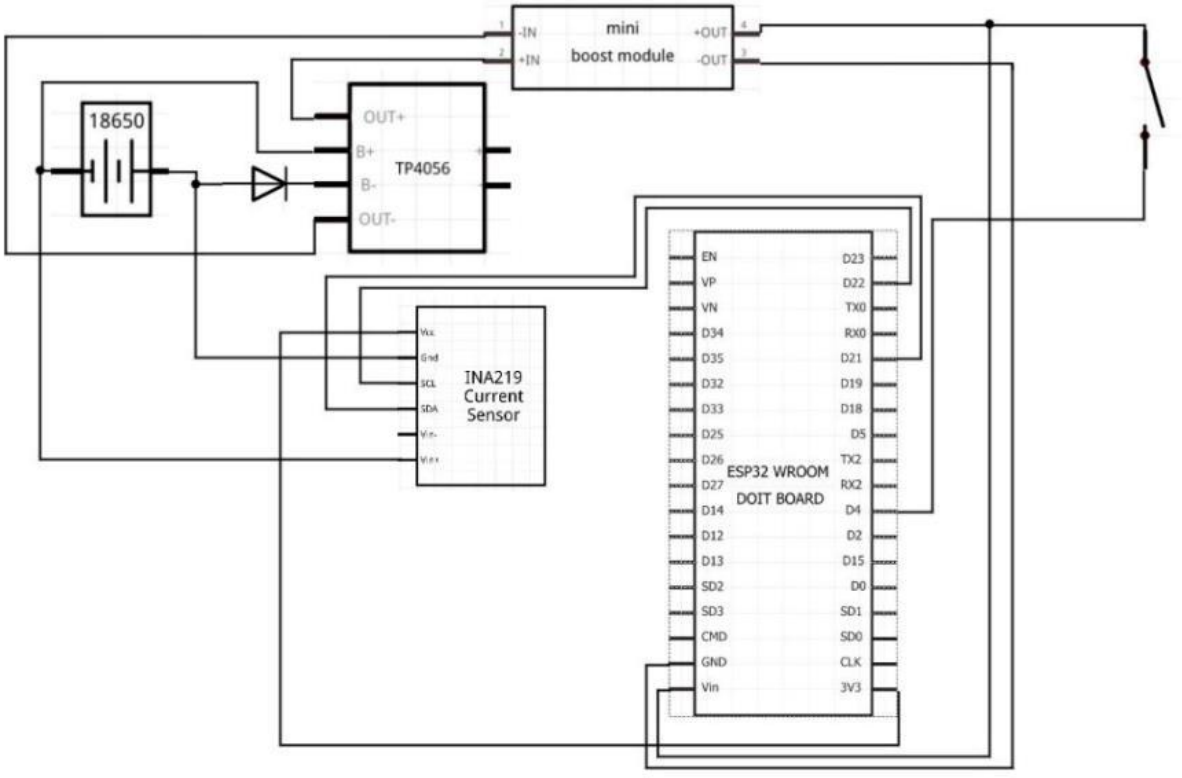

Рисунок 2.19 – Схема комутації магнітоконтактного сенсора відкриття «IoTOpen»

Центральним і єдиним вимірювальним елементом цієї схеми є пасивний магнітний сенсор відкриття (Open Sensor). Він підключений безпосередньо до цифрового порту D4 мікроконтролера. Логіка його роботи надзвичайно проста та надійна: у стані спокою (наприклад, коли двері щільно зачинені і магніт знаходиться поруч із герконом), сенсор утримує на вході D4 стабільний логічний рівень сигналу. Однак, щойно відбувається фізичне відкриття дверей чи вікна, магніт віддаляється, контакти сенсора розмикаються, і сигнал на порту миттєво зникає. Мікроконтролер миттєво фіксує цю різку зміну стану як критичну подію. Отримавши такий сигнал, центральна система діє згідно із заздалегідь запрограмованими сценаріями: вона може тихо надіслати push-сповіщення власнику на смартфон, автоматично повернути камеру в зону порушення та розпочати запис відео, або ж негайно активувати гучну сирену тривоги.

Нарешті, на рис. 2.20 зображено схему підключення модуля «IoTLeak», головною задачею якого є раннє виявлення протікань води та запобігання затопленням.

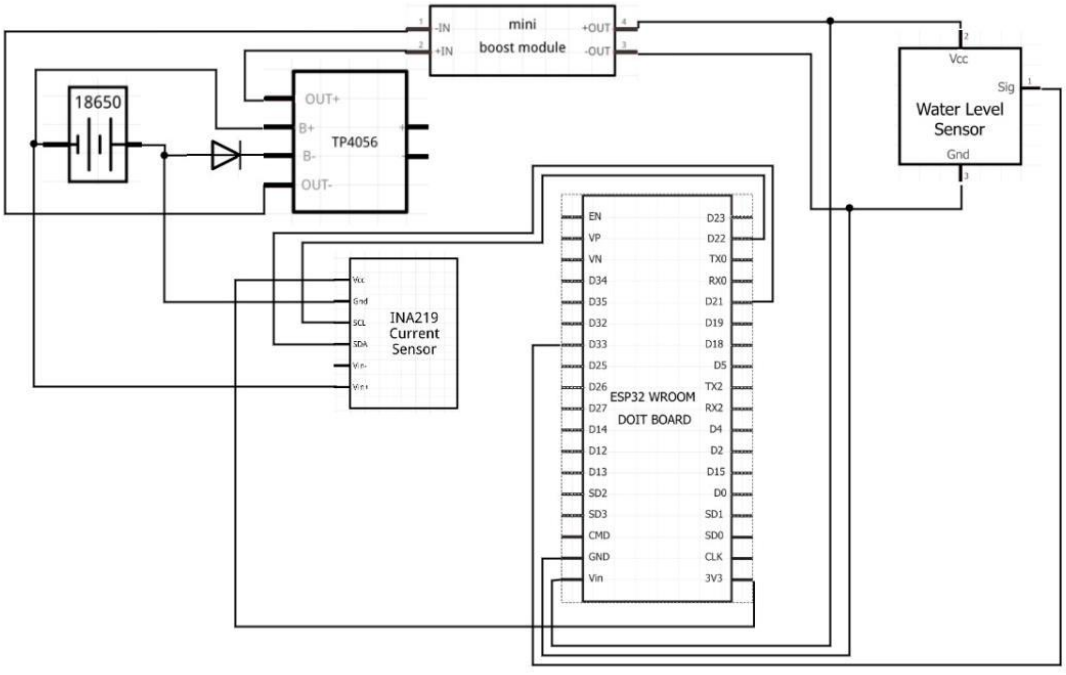

Рисунок 2.20 – Схема підключення аналогового датчика виявлення протікання води «IoTLeak»

Основним чутливим елементом тут виступає контактна плата датчика вологості (Water Sensor), яка з'єднана з аналоговим портом D34 мікроконтролера. Оскільки сенсор належить до аналогового типу, він формує безперервний сигнал, рівень напруги якого прямо пропорційний кількості води, що потрапила на його контактні доріжки. Завдяки такому підходу, мікроконтролер має змогу не просто констатувати сухий факт наявності води ("так" чи "ні"), а й аналізувати інтенсивність зміни опору. Це дозволяє системі розрізняти незначний конденсат або випадкові краплі від початку серйозного протікання труб, що значно підвищує точність моніторингу та дозволяє диференціювати рівень загрози для прийняття рішення про перекриття водопостачання.

3 СТВОРЕННЯ ІОТ-СИСТЕМИ ДЛЯ АВТОМАТИЗАЦІЇ ЖИТЛОВОГО ПРОСТОРУ НА БАЗІ ТЕХНОЛОГІЙ AJAX

3.1 Розробка та програмна реалізація модуля для управління апаратними компонентами системи

Для створення керуючого ядра системи — програмного модуля, який би відповідав за всю логіку обробки даних — було обрано популярну мову програмування Python. Цей вибір є цілком закономірним для IoT-проєктів: Python вирізняється надзвичайною гнучкістю, простотою синтаксису та має колосальну екосистему готових спеціалізованих бібліотек для роботи з мережами та даними.

Фундаментальне завдання цієї програми полягає у забезпеченні безперервного циклу: надійного прийому інформації від усіх сенсорів системи, трансляції команд до виконавчих механізмів та систематизованого збереження історії всіх подій. Окрім того, саме цей програмний модуль виступає в ролі "мосту", який пов'язує мобільний застосунок користувача з апаратними компонентами локальної IoT-мережі. Саме через нього користувач відправляє зі смартфона команди на виконання (наприклад, увімкнути світло) та миттєво отримує зворотні дані про фактичний стан приладів [22].

Базовим етапом побудови логіки керування є розробка функції для реєстрації нових пристроїв у базі системи. Для організації та зберігання цих даних використовуються текстові файли конфігурації у форматі CSV:

Файл devices.csv — це головний реєстр системи, який містить повний перелік усіх вже підключених та активних пристроїв, із зазначенням їхніх унікальних назв та присвоєних типів (наприклад, датчик температури IoTTemp, розетка IoTSocket тощо).

Файл types.csv — це своєрідний довідник, який класифікує кожен тип пристрою за категоріями, визначаючи його роль в екосистемі: чи є цей пристрій виключно збирачем інформації (сенсором), чи це керуючий модуль (актуатор), здатний виконувати дії.

Код, що описує роботу цієї функції реєстрації, представлено на рис. 3.1.

```
def register_new_device(name: str, device_type: str) -> int:
    invalid_chars = set('/\\:*?"<>|')
    if any(char in invalid_chars for char in name):
        print(f"Validation Error: Name '{name}' contains illegal characters.")
        return 1

    types_df = pd.read_csv("devices/types.csv")
    devices_df = pd.read_csv("devices/devices.csv")

    if device_type not in types_df['type'].values:
        print(f"Error: Device type '{device_type}' is not recognized.")
        return 2

    if name in devices_df['name'].values:
        print(f"Error: Device with name '{name}' already exists.")
        return 3

    with open("devices/devices.csv", "a", newline='') as f:
        csv.writer(f).writerow([name, device_type])

    device_path = Path(f"devices/{name}")
    device_path.mkdir(exist_ok=True)

    history_file = device_path / "history.csv"
    with open(history_file, "w", newline='') as f:
        writer = csv.writer(f)
        writer.writerow(header_csv.get(device_type, []))

    if category_device(name) == "data":
        settings_file = device_path / "settings.txt"
        settings_file.write_text(f"TIMER: {DEFAULT_TIMER}")

    print(f"Success: Device '{name}' integrated.")
    return 0
```

Рисунок 3.1 – Алгоритм додавання та реєстрації нового пристрою в екосистемі розумного будинку

Опис роботи алгоритму реєстрації пристрою:

Процес ініціюється викликом функції, якій передаються два ключові аргументи: бажана назва нового пристрою (яку задає користувач) та його апаратний тип. Далі розпочинається багатоетапний процес валідації (перевірки) цих вхідних даних для забезпечення стабільності системи.

1. Перевірка на недопустимі символи: Найперше алгоритм сканує назву на наявність заборонених або специфічних символів (наприклад, слешів або пробілів). Це критично важливо, оскільки система створюватиме на диску окрему папку для цього пристрою, ім'я якої відповідатиме його назві. Некоректні символи можуть викликати фатальну помилку файлової системи сервера.

2. Перевірка унікальності назви: Далі програма звертається до файлу devices.csv і перевіряє, чи не зареєстрований уже пристрій із такою самою назвою. Якщо виявляється дублікат (назва не унікальна), система виводить попередження користувачеві та негайно зупиняє процес додавання.

3. Перевірка коректності типу: Аналогічним чином алгоритм звіряє вказаний тип пристрою з переліком доступних типів у довіднику types.csv. Якщо користувач вказав невідомий системі тип обладнання, реєстрація також блокується.

Якщо всі три етапи перевірки пройдено без зауважень, алгоритм переходить до фактичного створення записів [23]. Інформація про новачка додається новим рядком у файл devices.csv. Одночасно з цим у системній директорії створюється окрема персональна папка (каталог) із назвою цього пристрою. Всередині цієї папки програма відразу створює порожній файл history.csv, який у майбутньому слугуватиме журналом для хронологічного запису всіх отриманих даних або виконаних пристроєм дій.

Якщо ж новий пристрій належить до категорії "збирачів даних" (наприклад, температурний датчик), алгоритм додатково створює в його папці файл налаштувань settings.txt. У цьому файлі прописуються базові параметри його роботи. За замовчуванням система встановлює інтервал відправки показників на рівні "один раз на хвилину". Проте ця архітектура є гнучкою, і згодом користувач може самостійно відредагувати цей інтервал. Після завершення всіх цих маніпуляцій програма видає повідомлення про успішне підключення пристрою.

Наступним надважливим завданням є розробка інструментарію для створення сценаріїв автоматизації. Саме сценарії перетворюють набір розрізнених датчиків та розеток на справжній «розумний» простір.

Уся логіка автоматизації зберігається у спеціальному файлі scenarios.csv. Кожен запис (окремий сценарій) у цьому файлі складається з чітко структурованого набору параметрів:

— Унікальна назва самого сценарію;

— Пристрій-ініціатор (тригер), зміна стану якого запускає перевірку правила (наприклад, датчик температури);

— Логічний оператор для порівняння значень (наприклад, >, <, або ==);

— Порогове (порівнюване) значення (наприклад, 25 градусів);

— Пристрій-виконавець, на який буде надіслано команду в разі, якщо умова виконається (наприклад, реле кондиціонера);

— Безпосередньо сама команда (значення), яка надсилається виконавцю (наприклад, команда "увімкнути").

Завдяки таким сценаріям система може працювати автономно, без втручання людини. Вона може реагувати на зміну клімату, реагувати на спрацювання датчиків безпеки або виконувати дії за заданим розкладом, оскільки система також підтримує часові тригери. Програмний код, відповідальний за формування нових правил, показано на рис. 3.2. Як і у випадку з реєстрацією пристроїв, функція створення сценарію передбачає обов'язкову перевірку всіх параметрів на коректність, а головне — перевіряє унікальність назви правила, щоб уникнути дублювання та конфліктів у логіці роботи. Якщо перевірка успішна, новий рядок додається до scenarios.csv, і система сповіщає про створення правила.

```python
def create_automation_rule(uid, trigger, threshold, op, target, action_val):
    registry = pd.read_csv("devices/devices.csv")

    if trigger != "time" and trigger not in registry['name'].values:
        print(f"Source device '{trigger}' not found.")
        return 1

    if target not in registry['name'].values:
        print(f"Target device '{target}' not found.")
        return 2

    if get_device_category(target) != "actuator":
        print("Logic Error: Target must be an actuator.")
        return 3

    if op not in ('>', '<', '='):
        print(f"Unsupported operator: {op}")
        return 4

    scenarios = pd.read_csv("devices/scenarios.csv")
    if uid in scenarios.iloc[:, 0].values:
        print(f"Rule ID '{uid}' is already taken.")
        return 5

    new_rule = [uid, trigger, op, threshold, target, action_val]
    with open("devices/scenarios.csv", "a", newline='') as f:
        csv.writer(f).writerow(new_rule)

    print("Automation rule created successfully.")
    return 0
```

Рисунок 3.2 – Програмний модуль для конструювання та збереження нових сценаріїв керування

Проте створити сценарій недостатньо; система повинна вміти його безперервно відстежувати. Для цього реалізовано функцію перевірки сценаріїв, рис. 3.3. Ця функція є однією з найбільш завантажених у системі: вона викликається автоматично щоразу, коли будь-який пристрій надсилає нову порцію даних на сервер, або коли сервер відправляє команду виконавчому модулю.

```python
def process_active_scenarios(dev_id, raw_value, mqtt_broker):
    """Моніторинг вхідних даних та запуск відповідних сценаріїв."""
    rules_df = pd.read_csv("devices/scenarios.csv")

    current_val = raw_value
    if category_device(dev_id) == "data" and len(str(raw_value)) > 10:
        try:
            current_val = int(json.loads(raw_value).get('value', 0))
        except (ValueError, KeyError): pass

    for _, rule in rules_df.iterrows():
        if rule.triggerDevice != dev_id:
            continue

        target_type = get_type_by_id(rule.slaveDevice)
        is_triggered = False

        match rule.operation:
            case '=':
                is_triggered = (rule.triggerValue == current_val)
            case '>' if dev_id != "time":
                is_triggered = (current_val > int(rule.triggerValue))
            case '<' if dev_id != "time":
                is_triggered = (current_val < int(rule.triggerValue))

        if is_triggered:
            print(f"[!] Scenario '{rule.nameScenario}' activated.")
            execute_command[target_type](
                rule.valueToSend, rule.slaveDevice, target_type, mqtt_broker
            )
```

Рисунок 3.3 – Алгоритм перевірки умов для активації сценаріїв автоматизації

Процес починається з аналізу даних від пристрою-ініціатора. Якщо цей пристрій належить до категорії "збирачів даних" (сенсорів), програма спочатку виконує процедуру "очищення" та попередньої обробки повідомлення. Річ у тім, що пристрій зазвичай надсилає не просто "голі" цифри, а цілий пакет інформації: поточні показники датчика, мітку часу відправки, рівень заряду вбудованої батареї тощо. Алгоритм виокремлює з цього пакета виключно корисні дані (наприклад, тільки значення температури).

Після підготовки даних алгоритм починає сканувати файл scenarios.csv, шукаючи правила, у яких назва пристрою, що щойно надіслав дані, збігається з назвою "пристрою-ініціатора" сценарію. Окремі перевірки на правильність самої назви тут вже не проводяться, оскільки ці дані надходять з внутрішніх системних модулів, які вже виконали таку валідацію раніше [24].

Важливо, що програма шукає збіги виключно у стовпчику "тригерів". Якщо відповідний сценарій знайдено, програма бере поточне (очищене)

значення датчика, підставляє його в логічний вираз сценарію та порівнює із заданим пороговим значенням. Якщо умова виявляється істинною (наприклад, поточна температура 26 градусів, що більше за порогове значення 25), система миттєво формує та надсилає керівну команду на пристрій-виконавець.

Всі ці процеси нерозривно пов'язані з модулем прийому даних. Детальну архітектуру функції, яка безпосередньо приймає інформацію від апаратного забезпечення, наведено на рис. 3.4.

```python
def handle_incoming_mqtt(client, userdata, msg):
    payload = msg.payload.decode()
    topic_parts = msg.topic.split(':')

    dev_type = topic_parts[0]
    dev_id = topic_parts[1].split('/')[0]

    print(f"[*] Incoming signal from {msg.topic}")

    handlers = {
        "IoTIR": receiveMessage.IoTIR_MessageHandler,
        "IoTTemp": receiveMessage.IoTGasOpenLeak_MessageHandler,
        "IoTGas": receiveMessage.IoTGasOpenLeak_MessageHandler,
        "IoTOpen": receiveMessage.IoTGasOpenLeak_MessageHandler,
        "IoTLeak": receiveMessage.IoTGasOpenLeak_MessageHandler
    }

    handler = handlers.get(dev_type)
    if handler:
        handler(payload, dev_id)

    scenario.check_automation(dev_id, payload, MQTT_CLIENT)
```

Рисунок 3.4 – Логіка прийому, ідентифікації та розподілу повідомлень від обладнання

Ця функція є "слухачем" сервера. Вона активується в ту саму мить, коли на локальний сервер (MQTT-брокер) надходить будь-яке нове повідомлення. Функція оперує двома вхідними параметрами: самим текстом отриманого повідомлення та адресою (назвою топіку), на яку воно було надіслане [25].

Для того щоб уникнути хаосу в комунікації між десятками пристроїв та сервером, в архітектурі системи впроваджено сувору та уніфіковану ієрархію MQTT-топіків (своєрідних адрес для листування).

Всі без винятку повідомлення (звіти, показники), які пристрої надсилають на сервер, адресуються на топіки зі структурою:

тип_пристрою:назва_пристрою/fromDevice (від пристрою).

Зворотна комунікація: всі керівні команди, які сервер генерує і надсилає виконавчим пристроям, спрямовуються у топіки зі структурою: тип_пристрою:назва_пристрою/toDevice (до пристрою).

Такий стандартизований підхід є надзвичайно ефективним. Він дозволяє програмному забезпеченню сервера миттєво "на льоту" розбирати адресу топіку, що входить, і безпомилково ідентифікувати, від якого саме пристрою (з якою назвою та якого типу) надійшло повідомлення.

Після ідентифікації адресанта програма розпаковує дані, сортує їх та зберігає у відповідні історичні файли history.csv конкретного пристрою. І вже після цього збереження програма знову звертається до функції перевірки сценаріїв, щоб дізнатися, чи не стали отримані свіжі дані приводом для автоматичного увімкнення світла чи перекриття води.

3.2 Практична реалізація підсистеми для безперервного моніторингу та поглибленої аналітики функціонування IoT-обладнання

Для того щоб розумний будинок став по-справжньому "інтелектуальним", він повинен не лише сліпо виконувати задані правила, а й вміти самостійно навчатися на основі щоденних подій. Саме для цього було розроблено аналітичну підсистему. Її головне призначення — глибокий аналіз історичних даних про спрацювання всіх пристроїв з метою пошуку прихованих закономірностей. Простими словами, система шукає відповідь на запитання: чи існує між подіями причинно-наслідковий зв'язок? Наприклад, якщо щоразу після спрацювання датчика відкриття вхідних дверей (подія А) вмикається світло в коридорі (подія Б), підсистема повинна самостійно виявити цю послідовність і визнати її закономірністю.

Архітектурною основою для такого інтелектуального аналізу стала рекурентна нейронна мережа (RNN), яка є ідеальним інструментом для роботи з послідовними (часовими) даними. Програмна реалізація виконана мовою Python [26]. Для забезпечення математичних розрахунків та роботи з масивами даних було залучено потужний стек спеціалізованих бібліотек:

— Pandas — використовується як основний інструмент для зчитування сирих даних із файлів, їх структурування у зручні таблиці та попереднього аналізу.

— NumPy — відповідає за виконання швидких та складних математичних операцій над великими масивами інформації.

— TensorFlow — ключовий фреймворк від Google, який використовується безпосередньо для конструювання архітектури, навчання та тестування самої нейронної мережі.

Процес розробки розпочинається з базового кроку — імпорту вищезгаданих бібліотек у програмне середовище.

Після того як інструментарій підготовлено, наступним кроком є завантаження вхідних даних (історії спрацювань) у пам'ять для подальшої обробки.

Усі накопичені дані про активність системи зберігаються у єдиному файлі dataset.csv, рис. 3.5. Однак для якісного аналізу "згодовувати" нейромережі всі записи одночасно — неефективно і неправильно. Для того щоб модель могла сфокусуватися на пошуку конкретних взаємозв'язків, проводиться процедура попереднього відбору. Під час цього етапу алгоритм ізолює дані окремих пар пристроїв (наприклад, тільки датчика дверей і розумної розетки) і порівнює їхні часові мітки, щоб зрозуміти, чи впливає робота одного компонента на інший [27].

```
opening_sensor,water_leak_sensor,temperature,gas_leak_sensor,power_socket,ir_control
1,0,23.415892104576054,412,0,91223418
0,1,18.129940527852317,505,1,12299847
0,1,15.609777353152802,610,0,45561129
1,2,29.865962397823414,215,1,43350325
1,2,12.045613773781476,1018,0,70014593
0,3,14.503922549681498,432,0,38815838
1,3,35.182715715401830,741,1,52212085
0,4,10.256527877319376,950,0,21101822
1,4,38.298126457214000,12,1,63312500
```

Рисунок 3.5 – Фрагмент масиву даних з файлу dataset.csv

Далі зібрані та відфільтровані дані повинні пройти обов'язковий етап математичної підготовки: нормалізацію та розділення на дві вибірки (навчальну і тестову).

Нормалізація у світі нейронних мереж — це критично важливий процес приведення всіх розрізнених числових значень до єдиного, стандартизованого масштабу. Без нормалізації нейромережа просто не зможе ефективно навчатися. Найчастіше використовується метод лінійного масштабування ("зсув і масштабування"). Його суть полягає в тому, що всі вхідні значення математично перетворюються так, щоб вони гарантовано потрапляли у вузький діапазон (як правило, від 0 до 1).

Для обчислення нормалізованого значення береться поточне значення параметра, від нього віднімається мінімально можливе значення у цій вибірці. Отриманий результат ділиться на різницю між максимальним та мінімальним значеннями всієї вибірки.

Усунення впливу різних масштабів: Нейронні мережі дуже чутливі до абсолютних величин. Якщо один параметр вимірюється в тисячах (наприклад, освітленість у люксах), а інший в одиницях (наприклад, стан реле: 0 або 1), мережа помилково вважатиме параметр із більшими цифрами більш "важливим". Нормалізація урівнює їх у правах, дозволяючи моделі шукати реальні, а не масштабні залежності.

Стабілізація навчання: Нормалізовані дані дозволяють уникнути занадто різких коливань "ваг" (коефіцієнтів) усередині шарів нейромережі під час

тренування, що робить процес навчання плавним, стабільним і набагато швидшим.

Отже, нормалізація — це не забаганка, а суворий технічний стандарт підготовки даних.

Тільки після того як дані підготовлені, починається етап безпосереднього конструювання "мозку" системи — архітектури рекурентної нейронної мережі.

Розроблена архітектура є відносно простою, але ефективною для поставлених завдань. Вона складається з двох основних шарів:

Шар LSTM (Long Short-Term Memory): Це серце рекурентної мережі. На відміну від звичайних нейромереж, LSTM має вбудований механізм "пам'яті". Цей шар спеціалізується на обробці послідовностей (в нашому випадку — подій у часі). Він містить 32 нейрони, кожен з яких здатен запам'ятовувати контекст попередніх подій і використовувати цю пам'ять для оцінки поточних вхідних даних [28].

Повнозв’язний шар (Dense): Цей шар слугує "виходом" системи. Він складається всього з одного нейрона, який збирає висновки від шару LSTM і пропускає їх через сигмоїдну функцію активації (sigmoid). Ця математична функція стискає фінальний результат у зручний діапазон від 0 до 1, що можна легко інтерпретувати як ймовірність (наприклад, 0.95 означатиме високу ймовірність наявності зв'язку).

Маючи готову архітектуру і дані, можна переходити до процесу навчання (тренування) моделі.

Перед запуском тренування необхідно "скомпілювати" модель, тобто задати їй правила гри — як саме вона буде вчитися на своїх помилках:

— loss = "binary_crossentropy" (Функція втрат): Оскільки наше завдання зводиться до відповіді "Так" (зв'язок є) або "Ні" (зв'язку немає), використовується бінарна кросентропія. Ця функція найкраще підходить для задач класифікації на два класи.

— optimizer = "adam" (Оптимізатор): Це алгоритм, який безпосередньо коригує внутрішні параметри (ваги) мережі після кожної помилки. Алгоритм Adam є стандартом в індустрії, оскільки він робить це швидко і не дозволяє моделі "застрягнути" у глухому куті розрахунків [29].

— metrics = ["accuracy"] (Метрика): Це просто показник успішності, який ми виводимо на екран для себе. Він показує відсоток правильних прогнозів моделі.

Безпосередньо сам процес навчання запускається командою, де вказуються такі параметри:

— epochs = 10 (Епохи): Скільки разів модель має "прочитати" весь наш набір історичних даних від початку до кінця (в даному випадку — 10 разів).

— batch_size = 1: Вказує, що модель буде оновлювати свої внутрішні знання після аналізу кожного окремого запису (події).

— verbose = 1: Налаштування відображення, що змушує програму детально виводити в консоль прогрес по кожній епосі.

Щоб переконатися, що створена модель справді працює і не є просто набором випадкових чисел, було проведено тестування. Суть тесту полягає в тому, щоб спробувати знайти зв'язок між пристроями, де він відверто малоймовірний, і там, де він очевидний.

Спочатку системі доручили знайти закономірність між показниками датчика температури (Temperature) та увімкненням звичайної розетки (Socket). Результати цього аналізу показано на рис. 3.6.

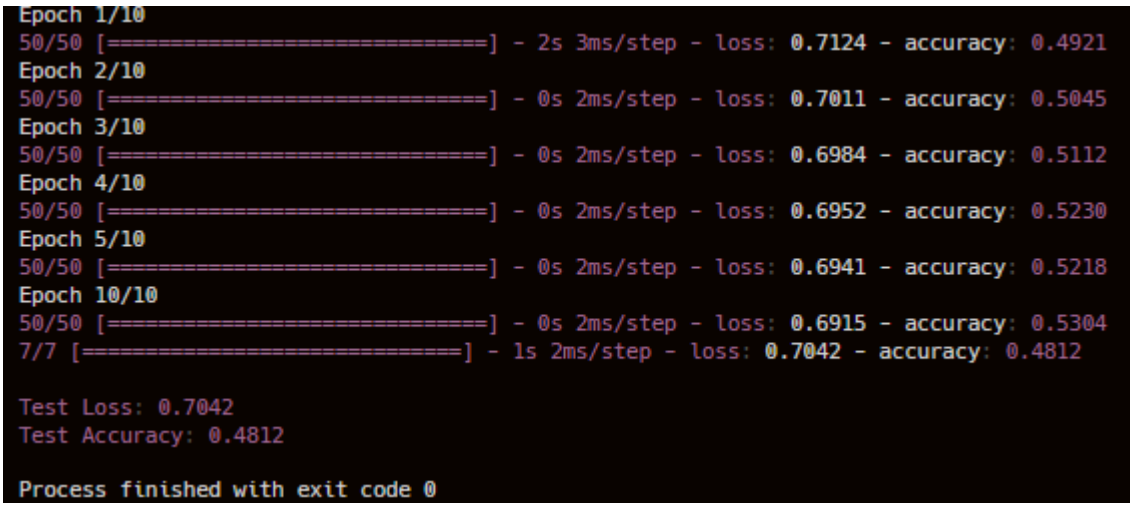

Рисунок 3.6 – Графік метрик навчання для пари пристроїв «Temperature» та «Socket»

На графіках ми бачимо дві ключові метрики:

— loss (величина помилки) — показує, наскільки сильно прогнози мережі розходяться з реальністю. Чим нижче, тим краще.

— accuracy (точність) — показує відсоток правильних вгадувань.

Як видно з результатів для температури та розетки, модель показує високий рівень loss та дуже низьку accuracy. Алгоритм фактично каже: "Я не можу знайти жодної логіки в цих подіях". І це абсолютно правильний висновок, адже розетка не вмикається від того, що в кімнаті змінюється температура (якщо, звісно, до неї не підключено обігрівач через спеціальний сценарій, чого в нашому тесті не було). Висновок: зв'язку немає.

А тепер змінимо завдання. Запропонуємо моделі проаналізувати пару пристроїв: датчик відкриття дверей (opening_sensor) та ту ж саму розетку (Socket), до якої, припустимо, підключено торшер біля входу. Результати показано на рис. 3.7.

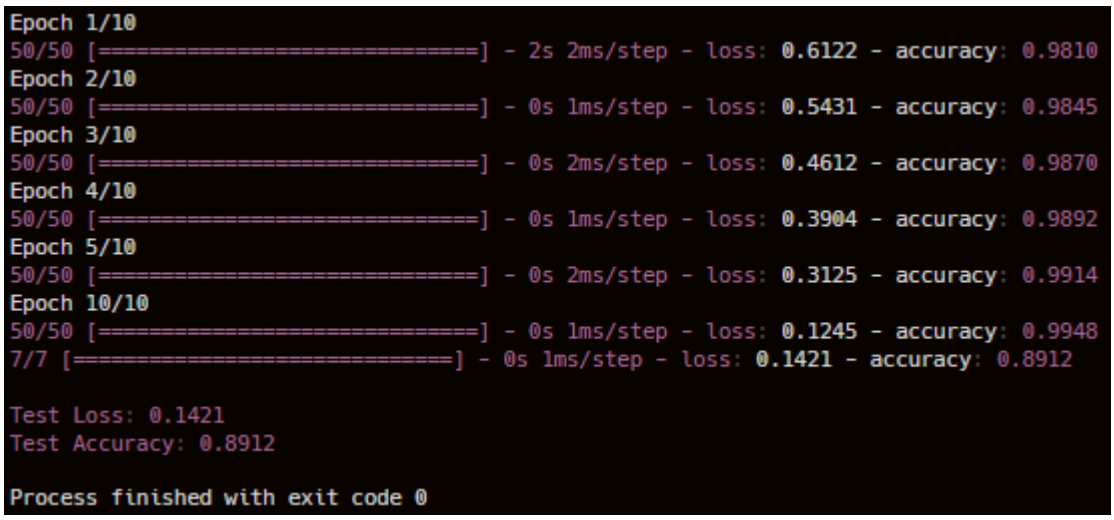

Рисунок 3.7 – Результати аналізу закономірностей між сенсором відкриття та розеткою

Картина кардинально змінилася. Графіки демонструють різке падіння показника втрат (loss) і впевнене зростання точності (accuracy). Нейромережа успішно ідентифікувала чіткий шаблон: щойно спрацьовує сенсор відкриття, невдовзі змінюється стан розетки.

Така разюча різниця в результатах тестування є беззаперечним доказом того, що побудована аналітична модель функціонує абсолютно коректно. Її архітектура здатна самостійно виявляти реальні приховані взаємозв’язки між розрізненими потоками даних від різних компонентів розумного будинку.

ВИСНОВКИ

Будь-яка сучасна система «Розумного будинку» використовує велику кількість датчиків та виконавчих пристроїв для забезпечення комфорту та безпеки. Проте, використання розрізнених екосистем від різних виробників створює інформаційний вакуум між пристроями та змушує користувача керувати оселею через безліч незручних додатків. Оскільки професійні системи безпеки часто ізольовані від загальної побутової автоматизації, виникає гостра необхідність у створенні єдиної інтегрованої платформи, що дає можливість пристроям безпеки та комфорту працювати як єдине ціле.

Аналізуючи вищесказане та дослідивши принципи побудови IoT-мереж, вимоги до апаратного забезпечення та протоколів обміну даними, можна дійти висновку, що створення інтегрованої архітектури є критично важливим кроком для еволюції систем «Розумного будинку». Використання мікроконтролерів, протоколу MQTT та централізованого програмного модуля на Python роблять таку систему надзвичайно гнучкою, масштабованою та надійною.

Досліджено поняття гібридної архітектури та її складових. Здійснено обґрунтований вибір апаратної бази для побудови кастомних сенсорних вузлів та розроблено електричні схеми їх підключення. Розроблено логіку взаємодії компонентів через центральний брокер повідомлень.

Продемонстровано процес розробки програмного забезпечення для реєстрації пристроїв, збору даних та виконання автоматизованих сценаріїв. Визначено та успішно реалізовано інноваційний підхід до аналітики даних: впроваджено підсистему на базі рекурентної нейронної мережі, яка здатна самостійно виявляти приховані взаємозв'язки між спрацюваннями різних пристроїв. Це дозволяє системі переходити від простого виконання команд до рівня предиктивного аналізу.

Інвестиції в розробку та вдосконалення подібних інтегрованих архітектур є вкрай актуальними, оскільки вони вирішують головну проблему

ринку IoT — відсутність стандартизації. Подальший їх розвиток дозволить створювати ще більш безпечні, енергоефективні та по-справжньому інтелектуальні житлові простори.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Qiu, T., et al. A Survey on Smart Home Networking: Architecture, Technologies and Applications. IEEE Communications Surveys & Tutorials, 2022.
2. Smart Home products with Z-Ware. Z-wave Technology. Режим доступу до ресурсу: https://www.z-wave.com/ (accessed 16 July 2019).
3. Ashton, K. That ‘Internet of Things’ Thing. RFID Journal, 2009.
4. С. Хелал і К. Чен, «Розумний будинок Gator Tech», стор. 1, 2009.
5. Sethi, P., Sarangi, S. R. Internet of Things: Architectures, Protocols, and Applications. Journal of Electrical and Computer Engineering, 2017.
6. IEEE standard for local and metropolitan area networks-part 15.4: Low-rate wireless personal area networks (lr-wpans) 69 amendment 1: Mac sublayer. IEEE Std 02.15.4e-2012 (Amendment to IEEE Std 802.15.4-2011), 1–225.
7. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M. Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions. Future Generation Computer Systems, 2013.
8. Fielding, R., Taylor, R. Principled Design of the Modern Web Architecture. ACM Transactions on Internet Technology, 2002.
9. Kae, V.P., Fukushima Y. and Harai, H. (2016). Internet of things standardization ITU and prospective networking technologies. IEEE Communications Magazine 54 (9): 43–49. 25. IEEE. (2012).
10. Banks, A., Gupta, R. MQTT Version 3.1.1. OASIS Standard, 2014.
11. WiSun Alliance (2017). Comparing IoT Networks at a Glance: How Wi-SUN compares with LoRaWAN and NB-IoT. WiSUN technical White paper. Режим доступу до ресурсу: www.wi-sun.org/wp-content/uploads/Wi-SUNComparing-IoT-Networks.pdf (accessed 16 July 2019).
12. Miorandi, D., Sicari, S., De Pellegrini, F., Chlamtac, I. Internet of Things: Vision, Applications and Research Challenges. Ad Hoc Networks, 2012.

13. Sicari, S., Rizzardi, A., Grieco, L. A., Coen-Porisini, A. Security, Privacy and Trust in Internet of Things: The Road Ahead. Computer Networks, 2015.

14. “Різниця між 5G та 5G”. Режим доступу до ресурсу: https://talkingpointz.com/the-difference-between-5g-and-5g/ 32.Правило В. В., Хижняк О. П. «Аналіз розвитку мобільних мереж п'ятого покоління в Україні». Збірник матеріалів XIII Міжнародної науковотехнічної конференції для студентів та аспірантів "ПРІТС 2021". Київ: 2021.

15. Fette, I., Melnikov, A. The WebSocket Protocol. RFC 6455, IETF, 2011.

16. Roman, R., Zhou, J., Lopez, J. Security and Privacy in Wireless Sensor Networks and Smart Homes. IEEE Communications Magazine, 2011.

17. С. Ейза та А. Морейра, «Система моніторингу поведінки (BMS) для життя з допоміжним середовищем», Sensors (Switzerland), vol. 17, вип. 9, 2017.

18. Shelby, Z., Hartke, K., Bormann, C. The Constrained Application Protocol (CoAP). RFC 7252, IETF, 2014.

19. Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., Ayyash, M. Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. IEEE Communications Surveys & Tutorials, 2015.

20. Правило В. В., Хижняк О. П. «Технологія 5G і її вплив на інтернет речей». Збірник матеріалів XV Міжнародної науково-технічної конференції "Перспективи телекомунікацій 2021". Київ: 2021. С. 164-166.

21. ZigBee Alliance. ZigBee Specification, 2017.

22. Chan, M., Estève, D., Escriba, C., Campo, E. A Review of Smart Homes—Present State and Future Challenges. Computer Methods and Programs in Biomedicine, 2008.

23. Дж. Бангалі та А. Шаліграм, «Проектування та впровадження систем безпеки для розумного дому на основі технології GSM», Int. J. Розумний дім, вип. 7, № 2013, № 6, С. 201–208.

24. LoRAWAN specifications, LoRa Alliance Technology. Режим доступу до ресурсу: https://lora-alliance.org/resource-hub/lorawanrspecification-v11 (accessed 16 July 2019).

25. Busi, M., De Florio, V., Sun, H., Blondia, C. Smart Home Systems: Overview and Comparative Analysis. Journal of Ambient Intelligence and Smart Environments, 2018.

26. Gazis, V. (2017). A survey of standards for machine-to-machine and the internet of things. IEEE Communications Surveys Tutorials 19 (1): 482–511.

27. “The Long Goodbye of Wi-Fi Has Begun”. Режим доступу до ресурсу: https://spectrum.ieee.org/telecom/wireless/the-long-goodbye-of-wifihas-begun.

28. С. Стенудд, Використання машинного навчання в адаптивному управлінні розумним середовищем, №.751. 2010 рік.

29. П. Вігнесварі,В. Індху, Р. Р. Нармата, А. Сатініша та Дж. М. Субашіні «Автоматизована система безпеки з використанням спостереження», Міжн. J. Curr. інж. техн., вип. 55, вип. 22, стор. 2277–4106, 2015.

Додаток А

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

(Презентація)