МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління

ВАРХІВ Богдан Ігорович

Інформаційна система інтегрованих модулів знань IoT / Information system of integrated IoT knowledge modules

Спеціальність 122 – Комп'ютерні науки
Освітньо-професійна програма – Комп'ютерні науки

Кваліфікаційна робота

Виконав студент групи КН-43
Б. І.Вархів

Науковий керівник:
к.т.н., доцент, О.Р. Осолінський

Кваліфікаційну роботу допущено до захисту
«___»_____ 2025 р.

В.о. завідувача кафедри
_____ Н.М. Васильків

Тернопіль – 2025

Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «бакалавр»
спеціальність 122 – Комп'ютерні науки
освітньо-професійна програма – Комп'ютерні науки

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Н.М. Васильків
«____» _____2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
ВАРХІВУ Богдану Ігорович
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: Інформаційна система інтегрованих модулів знань IoT / Information system of integrated IoT knowledge modules
керівник роботи О.Р. Осолінський, к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від 20 грудня 2024 р. № 938.
2. Строк подання студентом закінченої кваліфікаційної роботи 25 травня 2025 р.
3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.
4. Основні питання, які потрібно розробити:
- провести аналіз інформаційних систем знань IoT, описати предметну область;
- провести огляд існуючих рішень;
- постановка задачі;
- розробити алгоритмічне та інформаційне забезпечення системи інтегрованих модулів знань IoT;
- розробити архітектуру системи, описати логіку функціонування системи та інформаційне забезпечення системи;
- розробити структуру програмної реалізації системи та реалізувати основну бізнес-логіку;
- описати програмно-технологічне забезпечення та провести тестування.
5. Перелік графічного матеріалу в роботі:
- архітектура інформаційної системи;
- загальний алгоритм функціонування інформаційної системи;

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 20 грудня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Затвердження теми кваліфікаційної роботи, ознайомлення з літературними джерелами та складання плану роботи	до 01.01. 2025 р.	
2	Написання 1 розділу кваліфікаційної роботи	до 01.02. 2025 р.	
3	Написання 2 розділу кваліфікаційної роботи	до 01.04.2025 р.	
4	Написання 3 розділу кваліфікаційної роботи	до 01.05. 2025 р.	
5	Представлення попереднього варіанту кваліфікаційної роботи, перевірка та внесення змін керівником	до 15.05.2025 р.	
6	Опрацювання зауважень та представлення завершеного варіанту кваліфікаційної роботи. Підготовка супроводжуючих документів	до 25.05.2025 р.	
7	Перевірка кваліфікаційної роботи на оригінальність тексту	до 30.05.2025 р.	
8	Оформлення кваліфікаційної роботи та отримання допуску до захисту	до 10.06.2025 р.	
9	Подання кваліфікаційної роботи до захисту на засіданні атестаційної комісії	до 10.06. 2025 р.	

Студент_____ Б. І.Вархів
(підпис) (прізвище та ініціали)

Керівник кваліфікаційної роботи_____ О.Р. Осолінський
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Кваліфікаційна робота на тему «Інформаційна система інтегрованих модулів знань IoT» на здобуття освітнього ступеня «бакалавр» зі спеціальності 122 «Комп'ютерні науки» освітньої програми «Комп'ютерні науки» написана обсягом в 62 сторінки і містить 19 ілюстрацій, 5 додатків та 47 використаних джерел.

Метою роботи є розробка інформаційної системи інтегрованих модулів знань IoT для застосування в освітньому, науковому та прикладному середовищі, яка забезпечує збір, обробку, зберігання, візуалізацію та аналіз даних від сенсорів у режимі реального часу.

Методи дослідження - структурне та модульне проектування, аналізу інформаційних потоків, семантичного моделювання, виявлення аномалій, машинного навчання, побудови багаторівневої архітектури та тестування програмних компонентів у середовищі Flask і SQLite.

Запропоновано інформаційну систему з модульною архітектурою, що включає мікроконтролер Arduino Leonardo, набір сенсорів (температура, вологість, освітлення), серверну частину на Python з вебінтерфейсом на Flask, механізми автоматичної самоорганізації пристроїв, виявлення аномалій (LSTM-модель), адаптивної візуалізації (Plotly), та семантичної інтеграції нових модулів через Thing Description. Система підтримує легкі протоколи обміну даними (MQTT, HTTP) та функціонує в середовищі Linux. Результати апробовано з використанням тестових даних та реальних сенсорів, виконано тестування ключових функцій (модульне, інтеграційне, навантажувальне).

Було розроблено систему яка адаптується до потреб навчального процесу, дозволяє студентам вивчати як апаратну частину IoT, так і програмні принципи побудови сучасних інформаційних систем. Її можна масштабувати під різні предметні області.

Ключові слова: ІНТЕРНЕТ РЕЧЕЙ (IoT), THING DESCRIPTION, ARDUINO LEONARDO, FLASK, MQTT, SQLite, СЕНСОРНА СИСТЕМА, АДАПТИВНА ВІЗУАЛІЗАЦІЯ, ВИЯВЛЕННЯ АНОМАЛІЙ, БАГАТОРІВНЕВА АРХІТЕКТУРА.

ANNOTATION

Qualification Thesis titled “Information System of Integrated IoT Knowledge Modules” for the attainment of the Bachelor's degree in specialty 122 “Computer Science” of the educational program “Computer Science” comprises 62 pages and includes 19 illustrations, 5 appendices, and 47 references.

The purpose of this work is to develop an information system of integrated IoT knowledge modules for use in educational, scientific, and applied environments, which ensures real-time data collection, processing, storage, visualization, and analysis from sensors.

Research methods include structural and modular design, information flow analysis, semantic modeling, anomaly detection, machine learning, development of multi-level architecture, and software component testing in Flask and SQLite environments.

The proposed system features a modular architecture, including an Arduino Leonardo microcontroller, a set of sensors (temperature, humidity, light), a Python-based server with a Flask web interface, mechanisms for automatic device self-organization, anomaly detection using an LSTM model, adaptive visualization with Plotly, and semantic integration of new modules via Thing Description.

The system supports lightweight data exchange protocols (MQTT, HTTP) and operates in a Linux environment. The results were validated using both test data and real sensor input, with modular, integration, and load testing performed on key components.

A system was developed that adapts to the needs of the educational process, allowing students to explore both the hardware components of IoT and the software principles of modern information systems. The solution is scalable and can be tailored to various domains.

Keywords: INTERNET OF THINGS (IoT), THING DESCRIPTION, ARDUINO LEONARDO, FLASK, MQTT, SQLite, SENSOR SYSTEM, ADAPTIVE VISUALIZATION, ANOMALY DETECTION, MULTI-LEVEL ARCHITECTURE.

ЗМІСТ

ВСТУП

Останнім часом у зв’язку зі зростаючим значенням технологій і бізнесу IoT проводилися активні дослідження навчання IoT у рамках стратегії розвитку IoT. В останні роки Інтернет речей (IoT) стрімко розвивається, і, здається, це зростання не зупиниться найближчим часом. Станом на 2024 рік, глобальний ринок Інтернету речей (IoT) демонструє значне зростання та стабілізацію після пандемічних викликів. За даними IoT Analytics [1], кількість підключених IoT-пристроїв досягла 16,6 мільярда у 2023 році, а до кінця 2024 року очікується зростання на 13% до 18,8 мільярда пристроїв. Це зростання дещо нижче попередніх прогнозів через обережніші витрати підприємств, високі інфляційні показники та геополітичні конфлікти.

Інтернет речей – це екосистема фізичних об’єктів, підключених і доступних через Інтернет. IoT дозволяє пристроям спостерігати, розуміти та аналізувати ситуацію чи оточення, не залежачи від втручання людини. Мільярди людей у всьому світі користуються смартфонами, і це число стрімко зростає з кожним днем. Тому логічно, що мобільні програми є кращим каналом для доступу до IoT через легкість їх розробки. Мобільний телефон також є набагато більш гнучкою платформою для передачі даних. За допомогою однієї програми на пристрої можна ефективно керувати та контролювати пристрої IoT. Мобільні додатки відіграють важливу роль у посиленні розвитку IoT. IoT означає широку мережу взаємопов’язаних пристроїв із вбудованими датчиками та програмним забезпеченням для передачі та обміну даними через Інтернет. Це дозволяє зберігати дані (інформацію), зібрані з фізичних об’єктів, на виділених серверах або в хмарі. Аналіз цієї накопиченої інформації дає змогу створити більш безпечне та зручне суспільство, забезпечуючи оптимальний контроль пристрою та методи [2].

Розумний дім IoT – це будинок, який використовує IoT для забезпечення більшої зручності та безпеки. Наприклад, побутову техніку можна підключити до Інтернету та контролювати за допомогою взаємодії людини (голос, додаток для смартфона), а підключення датчиків дозволяє автоматизувати керування, щоб забезпечити високий рівень комфорту та зручності [3].

Розумні міста IoT – це міста, які використовують IoT для досягнення більшої економії енергії при мінімізації впливу на навколишнє середовище. Концепція розумних міст змінилася з часом, але на даний момент багато міст встановлюють численні датчики для отримання даних про стан навколишнього середовища та поведінку споживачів, а також для забезпечення дистанційного керування обладнанням і об'єктами для оптимізації роботи та підвищення зручності для мешканців [4].

Підключені автомобілі на основі IoT – це автомобілі, які включають IoT для досягнення безпечнішого та комфортнішого водіння. Збираючи та аналізуючи дані про водіння, отримані за допомогою бортових GPS, датчиків і камер, а також інформацію про навколишній трафік, можна досягти більш безпечного керування автомобілем і забезпечити оптимальне керування дорожнім рухом [5].

Програми IoT обіцяють принести цінність суспільному життю. З новими бездротовими мережами, розумними датчиками та революційними обчислювальними можливостями IoT може стати наступним рубежем. Програми IoT обіцяють принести величезну цінність повсякденному життю. Рисунок 1.1 показує різноманітність застосувань IoT у реальному світі. IoT — це гігантська мережа, що об'єднує все.

Технологія Інтернету речей базується на використанні розумних пристроїв, підключених до мережі, які оснащені вбудованими системами — процесорами, сенсорами та засобами зв'язку — для збору, передавання та обробки даних з навколишнього середовища. Але більшості розробників комп'ютерних наук або студентів важко вивчити технологію IoT, оскільки вона вимагає різних галузей знань про середовище комп'ютерних технологій [6].

Метою кваліфікаційної роботи є розробка інформаційної системи інтегрованих модулів знань IoT.

Завдання кваліфікаційної роботи полягають у послідовному виконанні наступних етапів:

– провести аналіз інформаційних систем знань IoT, описати предметну область;

– провести огляд існуючих рішень;

– постановка задачі;

– розробити алгоритмічне та інформаційне забезпечення системи інтегрованих модулів знань IoT;

– розробити архітектуру системи, описати логіку функціонування системи та інформаційне забезпечення системи;

– розробити структуру програмної реалізації системи та реалізувати основну бізнес-логіку;

– описати програмно-технологічне забезпечення та провести тестування.

Об’єкт дослідження - інформаційні системи Інтернету речей, які забезпечують інтеграцію гетерогенних пристроїв, обробку потокових даних, семантичну інтерпретацію інформації та взаємодію з кінцевими користувачами через мобільні або веб-інтерфейси.

Предмет дослідження - методи та засоби побудови адаптивної IoT-системи на основі модульної архітектури.

Результати кваліфікаційної роботи апробовані та опубліковані у матеріалах студентської науково-практичної конференції “Інтелектуальні інформаційні технології в прикладних дослідженнях” (IITAR – 2025), м. Тернопіль, Україна, 27-29 травня 2025 р.

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

1 АНАЛІЗ ІНФОРМАЦІЙНИХ СИСТЕМ ЗНАНЬ IoT

1.1 Опис предметної області

Інтернет речей (IoT) є фундаментальною технологією цифрової трансформації, що має значний вплив на різні сфери людської діяльності, включаючи охорону здоров'я, промисловість, транспорт і, що особливо важливо, сферу освіти. Сучасні тенденції розвитку технологій, таких як хмарні обчислення, штучний інтелект, великі дані та обчислення на периферії (edge/fog computing), тісно пов'язані з концепцією Інтернету речей, утворюючи єдиний розумний цифровий простір. IoT дозволяє збирати та передавати дані в режимі реального часу, забезпечуючи автоматизоване управління процесами без постійного втручання людини [7].

Застосування IoT в освіті формує нову парадигму — Освіта 4.0, яка орієнтована на інтеграцію новітніх цифрових технологій у навчальний процес. Це включає в себе розумні освітні середовища, які підтримують безперервний моніторинг, персоналізацію освітнього контенту, аналіз ефективності навчання та гнучке управління ресурсами. В роботі [8] була представлена архітектура багаторівневої системи, де кожен рівень виконує певну функцію: фізичний (сенсори, RFID, камери), сприйняття даних, класифікація, абстрагування та аналітика, прогнозування, та рівень прикладних сервісів.

Однією з ключових переваг IoT в освітньому контексті є можливість безперервного моніторингу студентів, викладачів і просторових умов. Наприклад, за допомогою сенсорів, вбудованих у картки доступу або носимі пристрої, можна фіксувати місце розташування студента, його фізіологічні показники, рівень активності тощо. Така система дає змогу аналізувати участь студентів у навчальному процесі, прогнозувати академічні труднощі та формувати індивідуальні навчальні траєкторії [10].

На рівні аналітики впроваджуються моделі машинного навчання, такі як бінаправлені довгі короткострокові пам'яті (Bi-LSTM), які дозволяють здійснювати предиктивний аналіз на основі часових даних та виявляти відхилення або аномалії у поведінці студентів [9]. Результати такого аналізу передаються в

модулі прийняття рішень, що можуть в режимі реального часу інформувати адміністрацію про потенційні проблеми або критичні ситуації (наприклад, проблеми зі здоров'ям або відсутність активності).

У рамках концепції "інтелектуального кампусу" застосовувались сенсори для вимірювання температури, вологості, шуму, освітлення, присутності в аудиторіях, а також інші показники середовища. Такі дані використовуються не лише для моніторингу комфорту, а й для автоматичного регулювання систем кондиціонування, вентиляції, енергоспоживання. Таким чином, IoT сприяє підвищенню енергоефективності освітніх закладів та зменшенню витрат на обслуговування інфраструктури.

В дослідженні [10] розглянуто взаємодію IoT із хмарними та туманними обчисленнями. Fog-компоненти використовуються для попередньої обробки та агрегації даних поблизу джерел їх збору, зменшуючи навантаження на центральні сервери та забезпечуючи швидший відгук системи. У свою чергу, хмара використовується для глибокої аналітики, довготривалого зберігання даних та централізованого управління системою.

Одним із важливих напрямів є розробка моделей на основі історичних даних. Наприклад, метод HAA (Historical Adversity Approximation) дозволяє на основі попередніх спостережень формувати динамічні порогові значення, що покращують виявлення нетипової поведінки або негативних тенденцій у навчальному процесі.

Автори роботи [7] відзначили, що реалізація подібних систем дозволяє створювати прозорі механізми акредитації та контролю якості освіти. Такі системи можуть автоматично оцінювати відвідуваність, активність студентів, ефективність викладачів, використання ресурсів і навіть прогнозувати успішність випускників. Це дозволяє значно зменшити обсяг ручної роботи та підвищити об'єктивність оцінювання.

Застосування IoT у сфері освіти супроводжується низкою невирішених завдань:

– Забезпечення конфіденційності та захисту персональних даних (особливо біометричних).

– Потреба у стандартизації протоколів взаємодії між пристроями різних виробників.

– Необхідність адаптації педагогічних підходів до нової технологічної реальності.

– Висока вартість впровадження IoT-інфраструктури та її обслуговування.

Попри ці перепони, перспективи впровадження IoT у навчальних закладах залишаються дуже високими. У майбутньому очікується поява ще більш складних і гнучких систем адаптивного навчання, які зможуть не лише враховувати потреби кожного студента, але й передбачати їх, спираючись на індивідуальні траєкторії, емоційний стан і контекст середовища. Імовірним є також поширення IoT-підходів у неформальній освіті, корпоративному навчанні та професійній підготовці.

Таким чином, IoT виступає фундаментом для побудови інтелектуальних освітніх середовищ, що дозволяють гнучко та ефективно реагувати на потреби студентів і викладачів, забезпечуючи новий рівень якості освіти, управління ресурсами та аналітики. Його впровадження потребує міждисциплінарного підходу та тісної співпраці між фахівцями з IT, педагогами, адміністрацією та аналітиками даних.

1.2 Огляд і аналіз існуючих рішень

Остання тенденція створення хмарних додатків обумовлена підключенням різнорідних служб, розгорнутих у кількох центрах обробки даних [11]. Таке розподілене розгортання допомагає підвищити надійність і продуктивність додатків IoT у хмарних/граничних обчислювальних середовищах. Забезпечення високого рівня надійності для завдань трансформації даних IoT, що складаються з безлічі систем, є серйозною проблемою, яку потрібно вирішити з точки зору розгортання. У середовищі IoT виникає значна технічна проблема, коли кілька невеликих сервісів, прийнятих інтелектуальними пристроями та інфраструктурами Edge/Cloud, необхідно об’єднати, щоб створити новий сервіс [12]. Однак ця проблема композиції сервісу є лише підмножиною низки викликів керування конфігурацією ресурсів IoT, оскільки, наприклад, проблеми керування

конфігурацією також розглядаються з точки зору повторного використання та оптимізації в інфраструктурах Cloud/Edge. В наукових колах і промисловості пропонуються різні структури для опису та розгортання ресурсів Cloud/Edge. Кілька хмарних постачальників, таких як CA AppLogic [13] і AWS OpsWorks [14], надають низький опис і розгортання повного стека хмарних програм. Вони пропонують представлення ресурсів, які є специфічними для конкретного постачальника. У периферійних обчисленнях Docker [15] надає рішення для розгортання та керування конфігурацією пристроїв Edge на основі методів контейнерів [16]. Однак, коли мова заходить про IoT, на додаток до рівня Cloud і Edge, керування конфігурацією ресурсів IoT також має враховувати фізичні пристрої, які широко розгортаються в більшості додатків IoT. В IoT усі ресурси з кількох рівнів потрібно розглядати в одній програмі, що призводить до більшої складності в забезпеченні правильності конфігурацій інфраструктур пристроїв IoT у поєднанні з правильними розгортаннями Cloud/Edge.

1.2.1 Концептуальні моделі в IoT

Онтологія семантичних сенсорних мереж представляє концептуальну модель високого рівня для опису фізичних пристроїв, їхніх можливостей і пов’язаних властивостей у семантичних сенсорних мережах в рамках IoT [17]. Автори в [18] надають опис додатків IoT і моделі даних, що фіксує зв’язки між різними постачальниками даних. Вони також ілюструють, як моделі можна асоціювати одна з одною та з різними областями застосування. Проект IoT-A описує послуги, сутності та ресурси як базові поняття в IoT [19]. Сервіси IoT розкривають функціональні можливості ресурсу, розміщеного на пристрої, що пропонує фізичний доступ до об’єкта, який, у свою чергу, представляє пряму взаємодію користувачів із програмними агентами. Підхід у [20] представляє дизайн повної та легкої моделі семантичного опису для представлення знань у сфері IoT. При їх розробці враховуються широко застосовувані правила в інженерії знань та моделюванні онтології. Однак їхня модель розглядає лише фізичний світ домену IoT і не враховує компоненти Cloud/Edge, які важливі для уніфікації представлення

знань додатків IoT для полегшення розгортання. Це особливо актуально, коли розглядається повне уявлення про можливості конфігурації програми.

1.2.2 Контекстно-залежні системи рекомендацій

Поняття усвідомлення контексту для підтримки вибору та розгортання сервісу виникло з початкової пропозиції в [21]. Сьогодні існують різні типи контекстної інформації, які в основному поділяються на три класи: фізичний контекст, контекст користувача та контекст пристрою. Така класифікація базується на прийнятій перспективі (користувача чи програми). Крім того, значення контексту сильно залежить від домену області та структури розглянутої програми. Це спонукало до пошуку змістовних визначень терміну «концепція» в різноманітних областях застосування. Можливі інтерпретації додатків, подібних до електронної комерції, наведено в [22]. Розташування – це звичайна контекстна інформація, проте вона не обов'язково відображає географічне розташування користувачів. Наприклад, визначення географічного розташування є ключовим питанням для забезпечення відповідності джерела потокових даних для перегляду на вимогу відповідним законам про авторські права в даній країні (рекомендовано лише дійсний вміст). В іншому прикладі спільна фільтрація на основі соціальних тегів, наведена в контексті додатків смарт-телевізору [23], є контекстно-залежним підходом, коли враховується інформація як користувача, так і контексту пристрою. У цьому випадку рекомендації розраховуються лише з використанням уподобань користувачів, і рекомендації ранжуються. В [24] автори пердставили структуру для надання декларативних рекомендацій щодо знань на основі контексту для об'єднаної конфігурації хмарних ресурсів. У цій структурі надається рекомендація щодо знань про конфігурацію сутностей на основі заданого контексту. У [25] автори пропонують систему вибору інфраструктури на основі онтології, яка базується на вимогах QoS у реальному часі та використовує аналітичні методи процесу ієрархії для полегшення прийняття багатокритеріальних рішень для рекомендацій. Незважаючи на те, що ці інфраструктури досягли значного поширення з точки зору розгортання та використання в реальному світі, вони

обмежені хмарними обчисленнями та не задовольняють додаткові вимоги додатків Інтернету речей щодо самих пристроїв або їхньої підтримуючої інфраструктури.

1.2.3 Приклади розробки додатків IoT

Є багато платформ, які можна використовувати для розробки програм IoT. Мобільні програми є точкою підключення для багатьох пристроїв IoT.

Авторами в роботі [26] досліджувалась робота зі смартфона, що керував за допомогою мікроконтролера ATMEGA328. У цьому дослідженні вони розробили робота, яким можна керувати за допомогою програми, що працює на телефоні Android. Телефон надсилає команду керування через Bluetooth, який має певні функції, такі як керування швидкістю двигуна, визначення та обмін інформацією з телефоном про напрямок і відстань робота до найближчої перешкоди. В іншій роботі [27] досліджувалось та випробовувалось можливості Bluetooth-робота з керуванням мобільним телефоном Android, використовуючи 8051 мікроконтролер. Відповідно до цього дослідження, Android Bluetooth підтримує телефони та модуль Bluetooth через HC-06 і зв'язок між пристроями Bluetooth. Автори в роботі [28] розробили та реалізували робота з бездротовим керуванням жестами, використовуючи процесор Arduino ATMEGA32 та систему додатків під управлінням Android. Система організовує широко класифіковану функцію розробки на два компоненти. Апаратна частина складається з мікроконтролера Arduino, Adafruit motor shield, модуля HC-05 Bluetooth і Android-смартфона, а програмна частина складається з програми на основі Java, що працює на android. Дослідники в іншому проекті [29] спроектували та розробили роботизований транспортний засіб Android із тристороннім керуванням через Bluetooth. Вони розробили тристороннє керування для роботизованого транспортного засобу, у якому використовувався зв'язок Bluetooth для інтерфейсу мікроконтролера та вбудованих датчиків у смартфоні Android. За командами, що надходять з телефону android, здійснюється управління кінематикою робота. Розроблений робот-транспортний засіб може бути використаний для багатьох застосувань у майбутньому, особливо у сфері спостереження та безпеки.

Автори роботи [30] вивчали та пояснювали техніку автоматизації розумного дому з Raspberry Pi за допомогою технології IoT. Ця запропонована система для техніки автоматизації «Розумного дому» з Raspberry Pi використовує IoT і розроблена шляхом інтеграції камер і датчиків руху у веб-додаток. Для розробки цієї системи вони використовують модуль Raspberry Pi з технікою комп'ютерного зору.

1.2.4 Сервіс та додатки IoT

IoT переносить потужність Інтернету, обробки даних і аналітики в реальний світ фізичних об'єктів. Для споживачів це означає взаємодію з глобальною інформаційною мережею без посередництва клавіатури, миші та екрана; багато їхніх повсякденних предметів і приладів можуть отримувати інструкції з цієї мережі з мінімальним втручанням людини. IoT може створити дуже інтелектуальне середовище за допомогою розумних пристроїв. У поєднанні з мобільними додатками, які дозволяють користувачам дистанційно керувати цими пристроями, ця технологія є однією з найвидатніших технологій цієї епохи. Далі буде представлено корисні технологічні послуги IoT і різні категорії IoT.

Розумний дім на основі технологій IoT — це система, що використовує пристрої з підключенням до Інтернету для дистанційного контролю та керування побутовими приладами й системами, зокрема освітленням і опаленням. Такі рішення набули значної популярності в останні десятиліття завдяки можливості підвищити рівень комфорту та покращити якість повсякденного життя. Зазвичай керування функціями розумного будинку здійснюється через смартфони або мікроконтролери. Смартфонний додаток забезпечує моніторинг і управління домашніми системами за допомогою бездротових технологій зв'язку. У дослідженні [32] розглянуто концепцію інтеграції IoT-сервісів і хмарних обчислень у розумний дім шляхом вбудовування інтелектуальних функцій у сенсори та виконавчі пристрої, об'єднання їх у мережу через відповідні технології, а також застосування хмарних рішень для забезпечення зручного доступу з різних місць, збільшення обчислювальних ресурсів, обсягу зберігання даних і підвищення ефективності обміну інформацією.

Наразі розумний дім, що використовує IoT та науку про дані, виправдовує очікування споживачів, а іноді навіть перевершує їх. Використання датчиків, пристроїв, приладів і всього простору в домі постійно збирає дані про те, як живуть люди в ньому і чим користуються. Ці пристрої дізнаються про звички та визначають моделі споживання за допомогою складних алгоритмів. Потім ця інформація допоможе персоналізувати досвід користувачів на детальному рівні [33]. Розумні домашні пристрої IoT нового покоління використовують дані свого датчика для автоматичного налаштування режимів під розпорядок дня. Вони відстежують місцезнаходження в режимі реального часу та відповідно вмикають і вимикають опалення [34]. Найкраще в такій технології те, що людині дійсно нічого не потрібно робити. Розумні термостати покладаються на свої алгоритми, щоб персоналізувати використання опалення відповідно до потреб і заощадити гроші на зменшеному споживанні енергії [35].

Інтернет речей (IoT) в охороні здоров'я можна використовувати ні тільки для лікування, але в дослідницькій роботі. Дана технологія дозволяє автоматизувати процес догляду за пацієнтами за допомогою мобільних додатків і сучасного обладнання, яке використовується у нових медичних закладах. Велика перевага IoT у тому, що він дає можливість швидко збирати велику кількість медичних даних. IoT-пристрої дозволили організувати дистанційний моніторинг пацієнтів, що зробило лікування безпечнішим, а також допомогло лікарям надавати якіснішу допомогу. Також віддалене спостереження за здоров'ям дозволило зменшити тривалість перебування в лікарні та знизити кількість повторних госпіталізацій. Завдяки цьому охорона здоров'я може скоротити витрати і при цьому покращити результати лікування [36]. Крім того важливе місце займають носимі пристрої, наприклад фітнес-браслети, тонометри, глюкометри та інші бездротові прилади. Вони допомагають пацієнтам отримувати персоналізований догляд. Такі пристрої можна налаштувати для нагадування про вправи, прийом ліків, вимірювання тиску тощо [38].

Особливо корисною ця технологія є для літніх людей. Вона дозволяє постійно стежити за станом їхнього здоров'я. Якщо відбуваються якісь зміни або відхилення у звичній поведінці, система надсилає попередження родичам або

медичному персоналу [39]. Лікарі, в свою чергу, можуть ефективніше слідкувати за станом пацієнтів, перевіряти, чи дотримуються вони плану лікування, і за потреби швидко реагувати. Дані, які збираються цими пристроями, допомагають лікарям обирати найбільш підходящі методи лікування [40].

Крім спостереження за здоров'ям пацієнтів, IoT корисний і для роботи лікарень. Завдяки сенсорам можна в реальному часі «бачити», де знаходиться необхідне медичне обладнання, таке як інвалідні візки, дефібрилятори, кисневі насоси та інше. Також можна визначати, де перебуває медичний персонал [41].

Окрему увагу варто приділити запобіганню інфекціям, що стало дуже актуальним після пандемії COVID-19. Пристрої для контролю гігієни, що працюють на основі IoT, допомагають знизити ризик зараження. Крім того, ця технологія корисна для управління ресурсами: вона допомагає слідкувати за запасами ліків в аптеках, а також контролювати умови зберігання — наприклад, температуру в холодильниках чи вологість у приміщеннях.

Пристрої IoT використовуються в різних сферах для ряду функцій плати. Останнім часом кілька пристроїв IoT використовуються в різних сферах для ряду функцій плати. Згідно з кількома видами досліджень, він генерує та використовує нову назву як додаток IoT.

Промисловий Інтернет речей (IIoT) використовується в промислових галузях. Основна увага приділяється взаємодії між машинами, роботі з великими обсягами даних та використанню методів машинного навчання. Завдяки цьому підприємства можуть працювати ефективніше та надійніше. IIoT охоплює різні сфери, зокрема робототехніку, медичне обладнання та виробничі процеси, які можна налаштовувати програмно [42].

Інтернет медичних речей (IoMT) — це об'єднання медичних пристроїв і програм, які можуть під'єднуватися до систем інформаційних технологій охорони здоров'я за допомогою мережевих технологій. Це може зменшити непотрібні візити до лікарні та зменшити навантаження на системи охорони здоров'я, підключивши пацієнтів до їхніх лікарів і дозволивши передавати медичні дані через безпечну мережу [43]. Автори роботи [44] представили концепцію «SIoT» соціального Інтернету речей у своїй дослідницькій роботі під назвою «Соціальний

Інтернет речей» (SIoT), має потенціал для підтримки нових програм і мережевих служб для Інтернету речей більш активними та корисними для клієнта способами. Конвергенція світів «Інтернету речей» і «Соціальних мереж» можлива або навіть доцільна, набирає обертів. Це пов’язано зі зростаючим усвідомленням того, що парадигма «соціального Інтернету речей» матиме багато бажаних наслідків у майбутньому світі, населеному розумними об’єктами, які проникають у повсякденне життя людей.

1.3 Постановка задачі

В умовах активного розвитку технологій Інтернету речей (IoT) та їх інтеграції в освітнє середовище виникає потреба в створенні інформаційних систем, що здатні забезпечити збирання, обробку, зберігання та аналіз даних у реальному часі. Одним із перспективних напрямів є розробка інформаційної системи на основі інтегрованих модулів знань IoT, яка здатна адаптуватися до різних сценаріїв використання — навчальних, дослідницьких та прикладних. При цьому така система повинна забезпечувати взаємодію великої кількості гетерогенних пристроїв, підтримку легких комунікаційних протоколів (MQTT, CoAP, WebSocket, HTTP/HTTPS), обробку даних на периферії, семантичну інтерпретацію інформації, а також гнучкий інтерфейс для кінцевих користувачів.

Проблема полягає в необхідності побудови масштабованої та модульної IoT-системи, яка б дозволила забезпечити гнучку взаємодію між компонентами, адаптацію до зміни середовища, обробку аномалій, візуалізацію даних у режимі реального часу та підтримку аналітичних і рекомендаційних модулів.

Метою роботи є розробка інформаційної системи інтегрованих модулів знань IoT для застосування в освітньому, науковому та прикладному контексті. Для досягнення цієї мети необхідно вирішити наступні завдання:

1. Розробити алгоритмічне та інформаційне забезпечення системи інтегрованих модулів знань IoT.

2. Розробити архітектуру системи, описати логіку функціонування системи та інформаційне забезпечення системи.

3. Розробити структуру програмної реалізації системи та реалізувати основну бізнес-логіку.

4. Описати програмно-технологічне забезпечення та провести тестування.

2 АЛГОРИТМІЧНЕ ТА ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ ІНТЕГРОВАНИХ МОДУЛІВ ЗНАНЬ IoT

2.1 Розробка архітектури системи

Інтернет речей (IoT) виступає основою інтелектуальних середовищ, де фізичні об’єкти взаємодіють з цифровими системами через вбудовані сенсори, програмне забезпечення та мережеві інтерфейси. В умовах цифровізації освіти, промисловості, міського середовища та охорони здоров’я актуальним стає створення інформаційної системи, що базується на інтегрованих модулях знань IoT, яка є гнучкою, адаптивною, масштабованою та навчально-орієнтованою.

Такі системи мають забезпечувати не лише обмін даними, а й їхній контекстуальний аналіз, адаптацію до поведінки користувачів, самонавчання й підтримку персоналізованих сценаріїв взаємодії. Завдяки модульному принципу та використанню стандартів відкритого програмного забезпечення, розробники можуть комбінувати різні сенсорні системи, обчислювальні вузли та аналітичні компоненти для створення прикладних IoT-рішень.

На основі аналізованих робіт можна визначити, що архітектура такої системи поділяється на кілька логічних рівнів (рисунок 2.1). Першим є фізичний рівень, який охоплює сенсори, що зчитують інформацію про середовище, мікроконтролери (наприклад, Arduino, ESP32), а також виконавчі механізми — двигуни, реле, дисплеї.

Далі йде мережевий рівень, який відповідає за передачу даних за допомогою бездротових технологій, таких як Wi-Fi, Bluetooth, ZigBee, LoRaWAN, або 5G. Для цього використовуються протоколи MQTT, CoAP, HTTP(S), WebSocket, а в якості комунікаційних вузлів — Raspberry Pi або подібні пристрої.

На обчислювальному рівні здійснюється попередня обробка даних: фільтрація, агрегування та кодування, а також їх передача у хмару або на периферійні вузли.

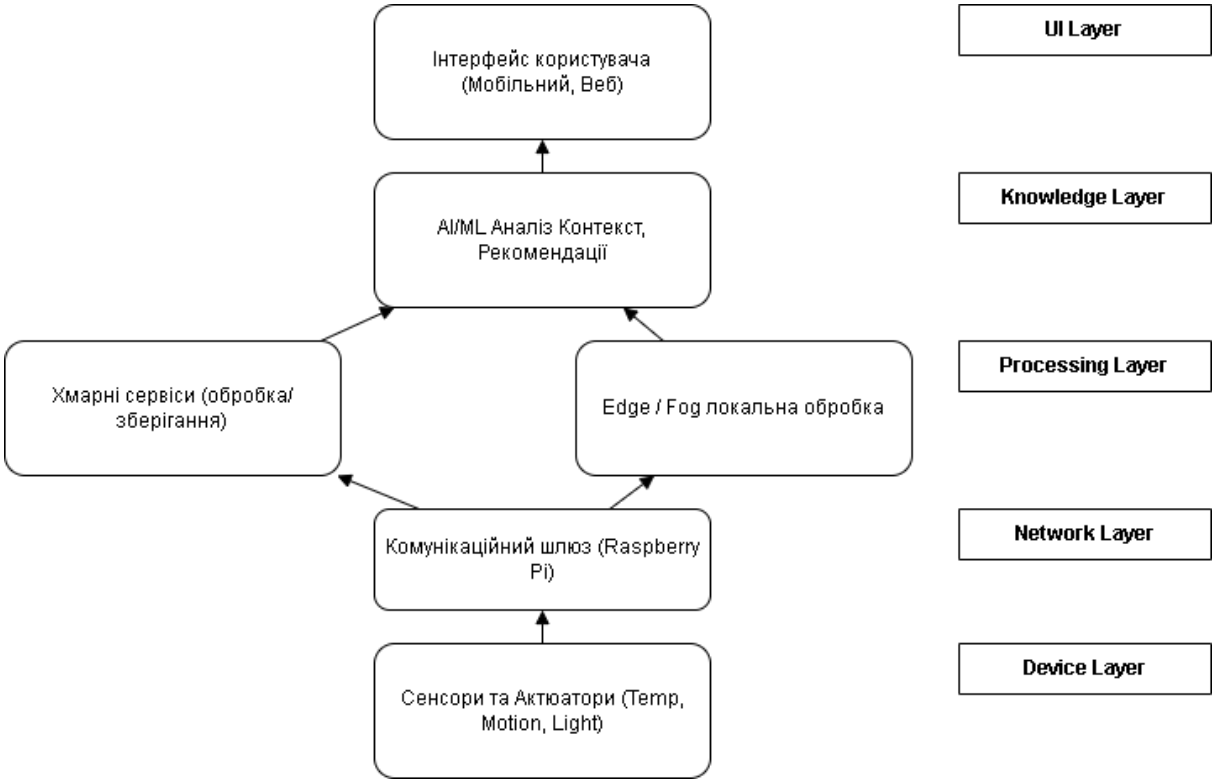


Рисунок 2.1 – Загальна архітектура інформаційної системи

Сучасна система повинна підтримувати як хмарні сервіси, такі як AWS IoT Core, Google Cloud IoT або Firebase, так і edge computing для пришвидшення обробки.

Семантичний і знаннєвий рівень забезпечує контекстуалізацію даних, включає модулі машинного навчання, що реалізують класифікацію, кластеризацію, предиктивну аналітику. Тут також реалізовуються рекомендаційні системи, які адаптують поведінку системи до користувача.

На рівні інтерфейсів важливу роль відіграють мобільні застосунки або веб-панелі, що дозволяють взаємодіяти із системою. Залежно від типу користувача — студент, викладач, адміністратор — інтерфейс може мати різний рівень складності й деталізації. У сучасних реалізаціях дедалі частіше використовуються адаптивні UI/UX рішення, які автоматично підлаштовуються під пристрій, роль користувача, а також контекст використання.

Інформаційна система включає модулі знань, кожен з яких містить набір сенсорів, локальну логіку обробки даних, інтерфейси зв’язку та механізми логування.

Центральний елемент такої системи — інтелектуальний контролер — виконує функції координації між модулями, а також виконує адаптацію під різні режими використання: навчальний, дослідницький або промисловий. В освітньому контексті система включає в себе інтерактивну панель, яка дозволяє візуалізувати дані, моделювати сценарії, запускати симуляції та навіть тренувати прості моделі штучного інтелекту.

2.1.1 Основні алгоритми функціонування

Функціонування системи відбувається за допомогою декількох узгоджених алгоритмів. Першим є алгоритм самоорганізації, при якому кожен модуль надсилає опис своїх можливостей (Thing Description), після чого центральний вузол формує карту взаємодій, визначає залежності та конфігурує канали зв’язку.

Другим є алгоритм виявлення аномалій: на основі даних з історії система формує профіль нормальної поведінки і за допомогою нейромережі (наприклад, LSTM) визначає відхилення. При перевищенні порогів генеруються сповіщення. Третім важливим алгоритмом є адаптивна візуалізація — інтерфейс динамічно змінюється відповідно до пристрою, ролі та сценарію використання.

2.1.2 Основні протоколи зв’язку

В архітектурі активно використовуються легкі протоколи (рисунок 2.2): MQTT для обміну повідомленнями, CoAP для пристроїв з низьким енергоспоживанням, WebSocket для реального часу, а також HTTP/HTTPS для взаємодії з веб-сервісами. Формати передачі даних — JSON, XML, CBOR. Усе це забезпечить взаємодію між модулями, хмарою, мобільними застосунками та системами візуалізації.

Система має високий потенціал для використання в освіті. Вона дозволить не лише демонструвати роботу IoT у реальному часі, а й давати студентам можливість змінювати логіку модулів, проводити досліди, збирати й аналізувати дані, працювати з мобільними застосунками та хмарними технологіями.

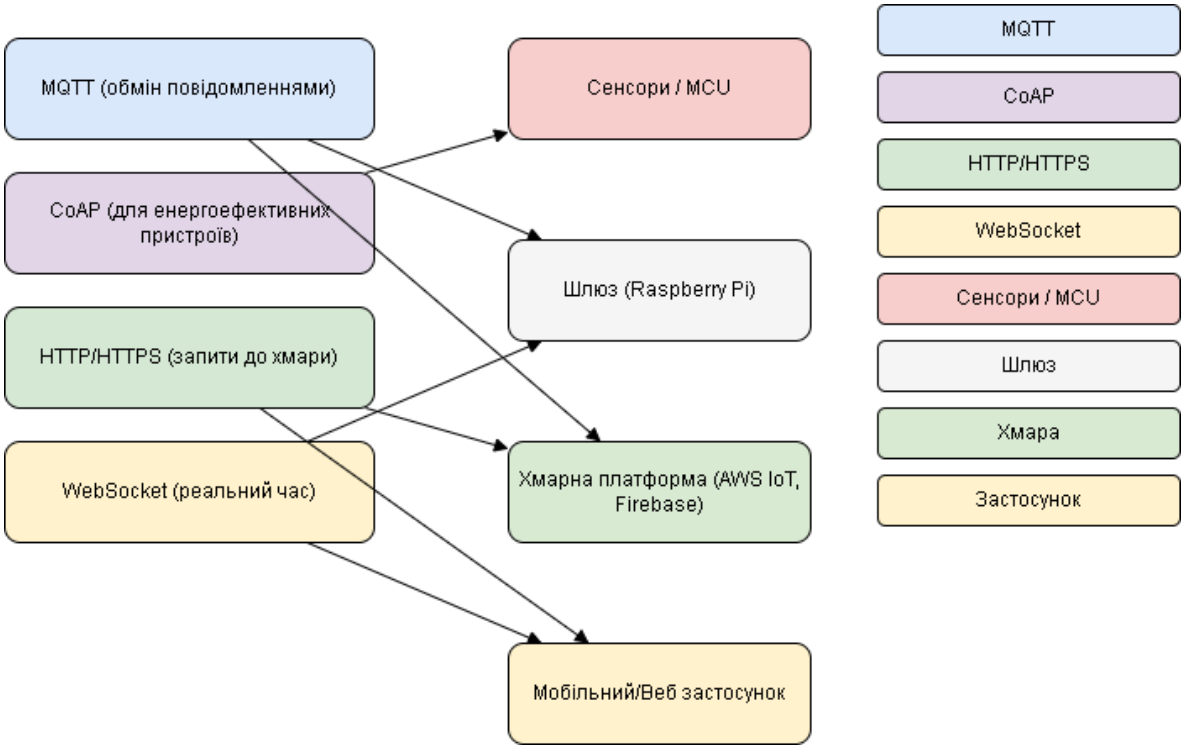


Рисунок 2.2 - Діаграма взаємодії протоколів

Студенти будуть мати змогу вивчати як апаратну частину, так і програмну архітектуру, що є унікальною перевагою в навчальному процесі.

Узагальнюючи, інформаційна система інтегрованих модулів знань IoT буде хорошим інструментом, як для практичного застосування у сфері розумного середовища, так і для навчання, наукових досліджень та прототипування інноваційних рішень. Завдяки модульності, відкритим стандартам, гнучким алгоритмам і підтримці мобільної/хмарної інтеграції, вона забезпечує універсальну платформу для розробки і дослідження IoT-рішень у реальних та симульованих середовищах.

2.2 Узагальнена логіка функціонування системи

Робота інформаційної системи інтегрованих модулів знань IoT побудована за принципом циклічності. Це означає, що система постійно повторює одні й ті ж основні кроки: збирає дані з пристроїв, обробляє їх, аналізує, реагує на виявлені події і адаптується до змін у середовищі (рисунок 2.3). Такий підхід дозволяє

системі бути гнучкою, самостійно оновлювати свою поведінку і швидко реагувати на нові ситуації.

Рисунок 2.3 – Циклічна діаграма роботи інформаційної системи

Коли система запускається або до неї підключається новий пристрій, він автоматично надсилає центральному модулю опис своїх можливостей. Наприклад, якщо це температурний датчик, він вказує, що може вимірювати температуру кожні 10 секунд. Така інформація передається у стандартному форматі, який називається Thing Description (TD). Центральний вузол аналізує ці дані та формує карту зв’язків між усіма підключеними модулями. Потім система налаштовує, як саме ці пристрої будуть обмінюватися даними між собою, як показано на рисунку 2.4.

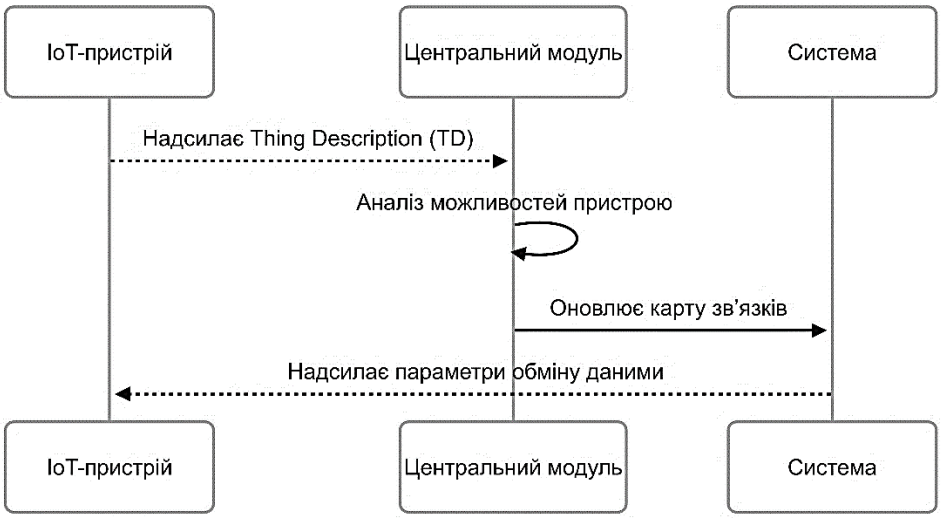


Рисунок 2.4 – Діаграма підключення нового пристрою

Після збору даних система перевіряє, чи є в них якісь незвичні відхилення. Наприклад, якщо температура у теплиці раптово піднялася вище норми або тиск в трубі впав до критичного рівня фіксується ознака несправності або аварійної

ситуації. Щоб такі ситуації виявляти, система повинна використовувати штучний інтелект, а саме, нейромережу типу LSTM. Вона навчається на попередніх даних і "запам’ятовує", якою має бути нормальна поведінка пристроїв. Якщо виявляється щось незвичне, система негайно надсилає сповіщення оператору або запускає відповідну дію, наприклад, включає охолодження чи аварійне відключення.

Щоб користувачеві було зручно працювати з системою, інтерфейс автоматично підлаштовується під конкретну ситуацію. Наприклад, якщо оператор заходить із планшета — відображення буде простішим і компактнішим, а якщо це адміністратор з комп’ютера — будуть доступні додаткові функції. Також, інтерфейс може адаптуватися в залежності від ролі користувача (технік, викладач, аналітик) або від того, яка саме ситуація відбувається (нормальний режим, тривога). Такий підхід дозволяє зробити роботу з системою комфортною, навіть у складних умовах.

2.2.1 Структура обробки даних

Так як інформаційна система працює за багаторівневим принципом — кожен рівень виконує свою частину обробки даних, починаючи від збору інформації з пристроїв і закінчуючи аналітикою (рисунок 2.5).

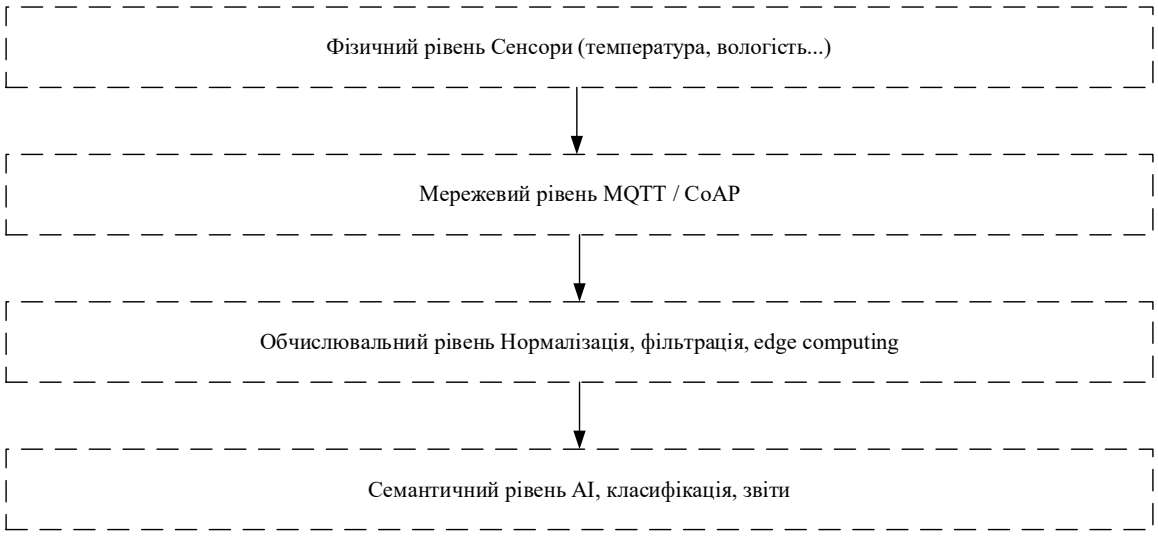


Рисунок 2.5 – Етапи обробки даних

Ця структура допомагає зробити систему більш ефективною, стабільною і зрозумілою.

На першому фізичному рівні працюють сенсори, які вимірюють, температуру, вологість, освітлення або тиск. Вони передають свої дані в цифровому вигляді або перетворюють аналогові сигнали в цифрову форму.

На мережевому рівні вся зібрана інформація надсилається на центральний вузол через протоколи зв'язку. В даній реалізації доцільно використати MQTT або CoAP як легкі протоколи, що дозволяють пристроям швидко і надійно обмінюватися даними.

На обчислювальному рівні система обробляє всі надіслані дані. До функцій цього рівня можна віднести згладжування шумів, нормалізація показників або групування декількох подібних значень. Частина цієї обробки виконуватиметься прямо на пристроях (edge computing), щоб не перевантажувати центральний сервер.

На семантичному рівні підключається штучний інтелект. Дані аналізуються за допомогою алгоритмів класифікації таких, як метод найближчого сусіда або опорних векторів, а також нейромережевих моделей, які допомагають робити прогнози або виявляти тенденції. На основі цього аналізу формуються підказки, звіти або попередження для користувачів.

2.2.2 Алгоритми навчання та адаптації

Так як система не є статичною і навчається тому чим більше даних вона отримує тим кращі результати показує. Тому в ній реалізовані алгоритми, які дозволяють їй пристосовуватися до змін і вдосконалювати свою роботу.

На першому етапі (рисунок 2.6) є навчання з підкріпленням коли система пробує різні варіанти дій і отримує за них "нагороди" або "штрафи". Тобто вона може тестувати різні режими охолодження в теплиці і запам'ятовувати, який з них дав найкращий результат для росту рослин.

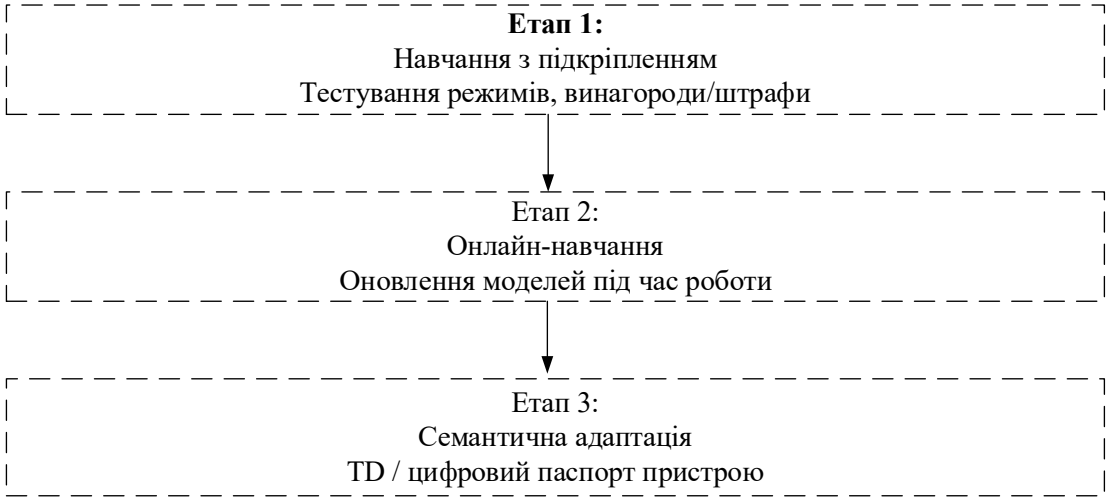


Рисунок 2.6 – Поетапний процес навчання та адаптації

На другому етапі система вчиться не лише на попередньо зібраних даних, а й прямо в процесі роботи. Тобто, якщо з’являється новий тип датчика або змінюється поведінка користувачів, система сама оновлює свої моделі без повного перезапуску чи перепрограмування.

Третім важливим елементом адаптації системи є використання семантичного підходу на основі онтологій. Суть цього підходу полягає в тому, що кожен пристрій, підключений до системи, описується за допомогою спеціального формалізованого документа дескриптора, який містить інформацію про доступні параметри, типи даних, одиниці вимірювання, частоту передачі тощо. Такий опис можна порівняти з "цифровим паспортом" пристрою. Це дозволяє системі автоматично розпізнавати характеристики нових пристроїв, інтегрувати їх у загальну архітектуру та правильно інтерпретувати отримані від них дані. Завдяки цьому забезпечується високий рівень сумісності між модулями від різних виробників без потреби у додатковому ручному налаштуванні. Семантична сумісність підвищує гнучкість, масштабованість і адаптивність інформаційної системи в умовах динамічного середовища.

2.2.3 Узгодження між модулями

Так як система складається з багатьох функціональних модулів, її ефективність значною мірою залежить від узгодженості та синхронності їхньої роботи. Тому для забезпечення стабільного функціонування системи в цілому

важливо, щоб усі компоненти діяли скоординовано, виконуючи свої завдання у визначеному порядку та відповідно до поточного контексту та бути узгодженні (рисунок 2.7).

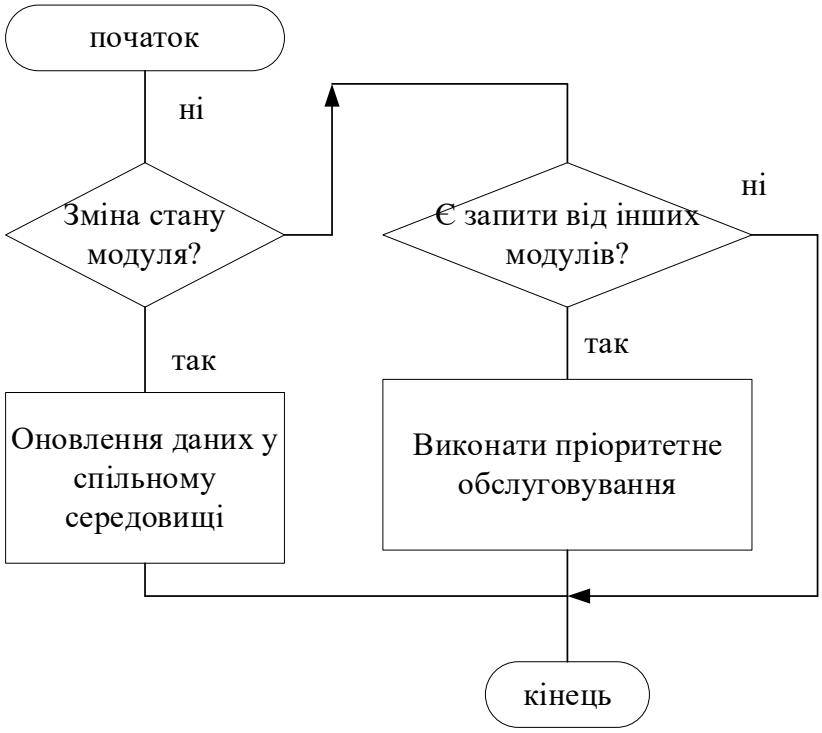


Рисунок 2.7 – Схема алгоритму узгодження між модулями

Одним із ключових алгоритмів, що реалізує таку скоординовану взаємодію, є алгоритм синхронізації станів. Його мета полягає в забезпеченні відповідності конфігурацій усіх модулів системи. До таких моментів відноситься, як версії програмного забезпечення, так і параметри підключення та внутрішніх налаштувань. При виявлені розбіжностей система або автоматично виконує оновлення, або передає повідомлення адміністратору з відповідними інструкціями. Такий механізм дає змогу запобігати конфліктам між компонентами.

Ще одним важливим алгоритмічним компонентом є алгоритм пріоритетного обслуговування запитів. Його завдання полягає в тому, щоб правильно визначати послідовність виконання вхідних повідомлень або дій системи відповідно до їхньої важливості. Якщо надходять одночасно сигнали від кількох сенсорів, система надає перевагу обробці критичного повідомлення (різке перевищення температури), а менш важливі запити можуть бути оброблені з певною затримкою. Для цього

використовується спеціальний механізм черги з ваговими коефіцієнтами, який дозволяє впорядковувати події за ступенем їх пріоритетності.

Застосування цих алгоритмів узгодження дозволяє забезпечити надійність та адаптивність системи, що особливо важливо у середовищах з великою кількістю взаємодіючих IoT-пристроїв і змінними умовами роботи.

2.2.4 Візуалізація та прийняття рішень

Важливою складовою ефективного функціонування системи є можливість своєчасної візуалізації стану об’єктів моніторингу та підтримки процесу прийняття рішень на основі отриманих даних. Інтерфейси взаємодії з користувачем відіграють ключову роль у цьому процесі, оскільки саме через них оператор отримує доступ до результатів аналізу, може відстежувати зміну параметрів у реальному часі та оперативно реагувати на критичні ситуації.

У системі реалізовано механізм побудови інтерактивних панелей моніторингу (рисунок 2.8), які динамічно оновлюються відповідно до змін у даних.

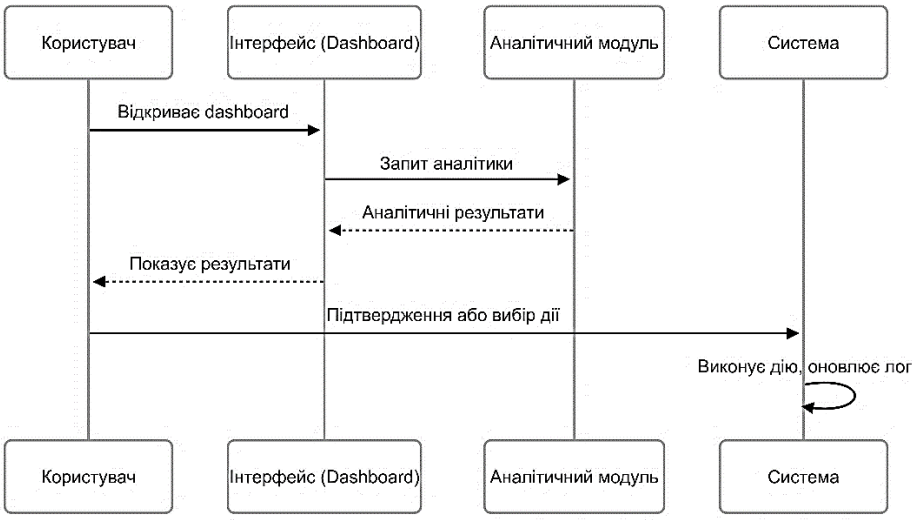


Рисунок 2.8 – Діаграма візуалізації та прийняття рішень

Для візуалізації застосовуються інструменти, що підтримують роботу з потоками даних у реальному часі, зокрема WebSocket-з’єднання або аналогічні засоби передачі подій. У залежності від рівня доступу користувача (технік,

оператор, аналітик) відображаються відповідні блоки з діаграмами, індикаторами стану та повідомленнями про події.

Крім моніторингу, система також підтримує інтелектуальне прийняття рішень. Це означає, що при виявленні подій, які виходять за межі норми, система не лише сповіщає оператора, але й може автоматично ініціювати відповідні дії. Наприклад, у разі фіксації перегріву буде вмикається охолоджувальний модуль або формуватись попередження для персоналу. Прийняття рішень базується на заздалегідь визначених сценаріях або ж на результатах машинного аналізу, що враховує поточний контекст.

Також є можливість запуску сценаріїв симуляції. Користувач може задати гіпотетичну ситуацію, наприклад, різке зростання навантаження на окремий модуль і система візуалізує очікуваний розвиток подій. Це дозволяє не лише планувати дії у разі реальних проблем, а й здійснювати навчання персоналу на основі моделей поведінки системи.

Таким чином, візуалізаційний модуль у поєднанні з механізмами прийняття рішень забезпечує зручний і зрозумілий доступ до інформації, але і високий рівень автономності, що зменшує залежність від людського фактора та підвищує загальну ефективність системи.

2.3 Інформаційне забезпечення системи

2.3.1 Типи даних у системі

Інформаційна система оперує різними типами даних, які відрізняються за своєю структурою, способом зберігання та характером обробки. Для забезпечення повноцінного функціонування, адаптивності та масштабованості будуть використовуватись різні типи даних.

До першого типу будуть відноситись структуровані дані, такі як пакети з показниками сенсорів (температура, вологість, рівень освітлення), які надходять у вигляді JSON-структур з чіткими полями: timestamp, value, unit, device_id. Він підходить для зберігання у базах даних і подальшої аналітики.

Напівструктуровані дані будуть належати до системних журналів, телеметричних звіти та подієвих повідомлень. Їх обробка потребує попереднього аналізу контексту і фільтрації за ключовими ознаками.

І останім типом, яким буде оперувати система є неструктуровані дані які будуть містити зображення з відеокамер, аудіозаписи голосових команд або фото, що надсилаються користувачами.

Окрему категорію становлять метадані, які описують властивості пристроїв та їх можливості. В системі вони представлені у вигляді так званих дескрипторів пристроїв — Thing Description (TD). Ці метадані містять інформацію про тип пристрою, доступні функції, формати даних, частоту передачі, методи доступу до API тощо. Завдяки цьому система може автоматично ідентифікувати підключені пристрої, визначати їх можливості та забезпечувати їхню інтеграцію без ручного втручання.

Поєднання всіх вищезазначених типів даних дає змогу створити гнучку, адаптивну та функціонально багату інформаційну систему, що підтримує роботу з гетерогенними джерелами інформації та відповідає сучасним вимогам до IoT-платформ.

2.3.2 Джерела даних

Для забезпечення повноцінної роботи системи інтегрованих модулів знань IoT залучаються різні джерела даних, які формують основу для прийняття рішень, аналітики, навчання моделей та візуалізації. У системі передбачено декілька основних категорій джерел, кожна з яких має своє функціональне призначення та специфіку інтеграції.

Сенсорні пристрої є головним джерелом первинної інформації. До них належать температурні, вологісні, освітлювальні сенсори, датчики тиску, вібрації, вмісту CO_2 тощо.

Другим джерелом є дії користувачів яке фіксує взаємодію користувачів з інтерфейсом, наприклад, перегляд панелей моніторингу, внесення конфігураційних змін, запуск сценаріїв. Такі дії можуть бути як тригерами для

автоматичних алгоритмів, так і матеріалом для побудови статистики або оптимізації інтерфейсу.

До зовнішніх джерел відносяться сторонні сервіси, такі як мобільні застосунки, хмарні платформи або API-підключення до сторонніх IoT-систем. Через такі інтеграції можуть надходити додаткові аналітичні показники, прогнозні дані, сигнали від партнерських систем або навіть погодні умови

Окрему категорію становлять системні події, які генеруються в процесі роботи самої платформи. Сюди входять повідомлення про збої, відмови зв'язку, оновлення програмного забезпечення, запуск фонових задач, повідомлення від моніторингових служб.

Сукупність різнотипних джерел даних дозволяє системі формувати інформаційний простір, що забезпечує високу точність прийняття рішень, адаптивність до змін у середовищі.

2.3.3 Бази даних та сховища

Через постійне надходження великої кількості даних від сенсорів, подій користувача, системних повідомлень та результатів аналітики в системі застосовується комбінований підхід, який передбачає використання як класичних реляційних баз даних, так і сучасних NoSQL-технологій.

NoSQL-бази використовуються для зберігання неструктурованих та напівструктурованих даних, таких як телеметричні показники, лог-файли, події або документи у форматі JSON.

Реляційні бази даних (SQLite) використовуються для зберігання структурованих даних, що потребують чітких зв'язків між таблицями, транзакційної обробки та схеми. У системі вони застосовуються для збереження профілів користувачів, логів авторизації, описів пристроїв, а також результатів аналітичних обчислень.

У випадках, коли потрібне централізоване та довготривале зберігання, система може використовувати хмарні сервіси збереження даних, що дають змогу зберігати потоки даних у хмарі з можливістю подальшого доступу до них з мобільних застосунків або вебінтерфейсів.

Додатково передбачено тимчасові локальні кеші, які використовуються для буферизації даних на рівні edge-пристроїв у разі втрати з’єднання або затримки в надсиланні..

2.3.4 Семантичні моделі та онтології

Для забезпечення взаєморозуміння між різнорідними модулями IoT-системи, а також для зручного масштабування, у системі застосовуються семантичні моделі та онтології. Вони дають змогу описувати пристрої, процеси та взаємозв’язки у формалізованій формі, яка легко сприймається як людиною, так і комп’ютерними програмами.

В системі використовується Thing Description (TD), стандарт, розроблений в який дає змогу створювати машинно-читабельні описи пристроїв у форматі JSON-LD, де фіксується інформація про доступні сенсори та актуатори, типи даних, способи з’єднання, параметри доступу, інтервали опитування тощо. Семантичне представлення даних дозволяє описати технічні характеристики, та надати значення даним у контексті тобто, зв’язати сенсор температури з конкретним об’єктом у системі.

Застосування семантичних моделей спрощує процес автоматичного підключення нових пристроїв до системи, покращує інтероперабельність компонентів, а також відкриває можливість для інтеграції з іншими IoT-платформами.

3 ПРОГРАМНО-ТЕХНОЛОГІЧНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ ІНТЕГРОВАНИХ МОДУЛІВ ЗНАНЬ IoT

3.1 Структура програмної реалізації системи

Розроблена інформаційна система реалізується з використанням апаратно-програмного комплексу, що включає мікроконтролер Arduino Leonardo (рисунок 3.1), набір сенсорів (температури, вологості та освітлення), а також серверну частину, побудовану на основі мови Python з використанням фреймворку Flask. Система функціонує в середовищі операційної системи Linux і реалізована за модульним принципом, що забезпечує зручність розширення, підтримку та модернізацію окремих компонентів.

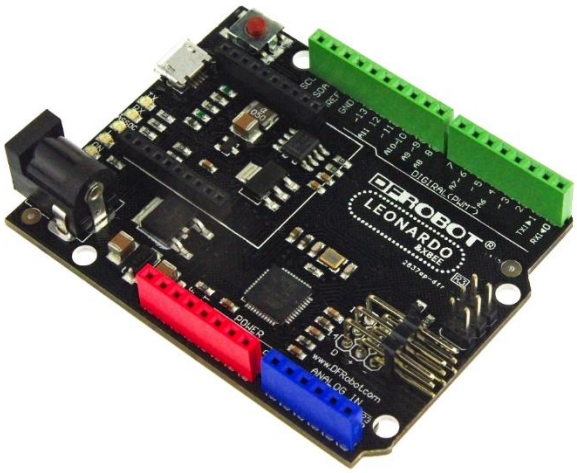


Рисунок 3.1 – Модуль розробки DFRobot Leonardo with Xbee R3

Загальна архітектура системи передбачає три логічні рівні: пристроєвий, серверний та користувацький.

На пристроєвому рівні (edge-level) функціонує мікроконтролер Arduino Leonardo, до якого підключено аналогові сенсори:

- температури LM35;
- вологості, аналоговий сенсор DFRobot;
- освітлення, фоторезистивний світлочутливий сенсор.

Програмне забезпечення, реалізоване на мікроконтролері, забезпечує періодичне зчитування показників із сенсорів, їх попередню обробку (наприклад, усереднення або обмеження за межами діапазону) та передачу результатів на сервер за допомогою інтерфейсу USB у форматі послідовного з'єднання (Serial).

Серверний рівень (gateway-level) представлений Python-застосунком, побудованим із використанням мікрофреймворку Flask (рисунок 3.2).

Рисунок 3.2 – Мікрофреймворк для вебдодатків

Основна роль серверної частини — прийом, перевірка, зберігання та публікація отриманих даних. Дані, що надходять від Arduino, зчитуються через послідовний порт, обробляються (зокрема, перевіряються на валідність та межі), після чого зберігаються у локальну базу даних SQLite. Крім того, сервер реалізує REST API для отримання збережених значень, а також логіку побудови візуалізацій.

Користувацький рівень (user-level) включає вебінтерфейс, що дозволяє користувачеві переглядати поточні та історичні дані в зручному вигляді. Вебінтерфейс створено із використанням шаблонізатора Jinja2, що дозволяє генерувати HTML-сторінки на основі отриманих із бази даних значень. Інтерфейс є адаптивним, сумісним із мобільними пристроями та дозволяє реалізувати такі функції, як перегляд графіків, попереджень, журналів подій тощо.

Вибір використаних технологій пояснюється вимогами до простоти розгортання, автономності, відкритості коду та можливості локального функціонування без підключення до мережі Інтернет:

Arduino Leonardo було обрано завдяки наявності достатньої кількості аналогових входів, підтримці USB-комунікації та простоті інтеграції з Linux-системами;

Flask забезпечує простий механізм створення вебсервісу, підтримку маршрутизації та обробки запитів, що дає змогу швидко розгорнути функціональну серверну частину;

SQLite — вбудована реляційна СУБД, яка не потребує окремого сервера та є оптимальною для локальних проєктів з обмеженим обсягом даних.

Для реалізації функціональних можливостей у Python-середовищі використано такі бібліотеки:

- pyserial — для зчитування даних через послідовний порт;
- sqlite3 — для взаємодії з базою даних;
- plotly або matplotlib — для побудови інтерактивних або статичних графіків;
- threading — для організації паралельного зчитування та обробки даних.

Загальна структура системи є легкою для підтримки, придатною до масштабування, додавання нових сенсорів або модулів виводу та відповідає типовим архітектурним підходам у розробці IoT-рішень.

3.2 Реалізація основної бізнес-логіки

Бізнес-логіка системи визначає основні функціональні сценарії, які реалізуються під час її роботи: зчитування сенсорних даних, обробка інформації, прийняття рішень, збереження даних і формування відповіді для інтерфейсу користувача. В реалізованому рішенні бізнес-логіка розподілена між пристроєвим рівнем (Arduino Leonardo) та серверним рівнем (Flask у Linux-середовищі).

На мікроконтролері реалізовано основний цикл, який забезпечує періодичне зчитування аналогових значень з підключених сенсорів температури, вологості, освітлення.

Значення обробляються у внутрішній логіці прошивки: виконується масштабування (наприклад, переведення значення напруги у градуси Цельсія), перевірка на наявність сигналу, за потреби — усереднення або округлення. Після

цього дані формуються у вигляді одного текстового рядка, в якому значення розділені комами: 22.5,48.7,320

Цей рядок надсилається через USB-інтерфейс у послідовному режимі до комп'ютера.

Серверна частина реалізує обробку вхідного потоку даних за допомогою окремого модуля на Python, що постійно працює у фоновому потоці (threading.Thread). З використанням бібліотеки pyserial здійснюється зчитування кожного рядка, його розділення на складові (температура, вологість, освітлення), та перевірка на коректність. На цьому етапі також реалізується:

Визначення статусу системи — чи знаходяться значення в межах допустимих норм.

Формування об'єкта для збереження — створюється словник або кортеж, який передається у функцію взаємодії з базою даних.

Реакція на порушення — при виявленні аномальних значень, генерується службовий сигнал, який відображається у вебінтерфейсі (рисунок 3.3).

```
def get_status(temp, hum, light):
    if temp > 28 or temp < 18:
        return "Температура поза межами норми"
    elif hum < 30:
        return "Низька вологість"
    elif light < 50:
        return "Низька освітленість"
    else:
        return "Норма"
```

Рисунок 3.3 – Перевірка на допустимість значень

Весь обмін даними побудований на внутрішньому API між модулями Python-програми. Основні класи і функції, які реалізують бізнес-логіку, поділені на:

SerialReader — відповідає за зчитування та попередню обробку;

DatabaseManager — здійснює запис у SQLite;

StatusChecker — визначає поточний стан системи;

WebController — формує відповіді для вебінтерфейсу.

Така структура дозволяє змінювати окремі компоненти, масштабувати функціонал або адаптувати систему до нових типів сенсорів без повного переписування логіки.

3.2.1 Взаємодія з базою даних і зовнішніми сервісами

Ефективне збереження, обробка та подальший аналіз виміряних сенсорами значень передбачає наявність надійного механізму зберігання даних. У розробленій системі для цієї мети використовується вбудована реляційна база даних SQLite, яка оптимально підходить для невеликих автономних застосунків, що функціонують локально без підключення до централізованого сервера.

Структура бази даних відповідає логіці надходження сенсорних даних. Основна таблиця measurements містить поля, необхідні для зберігання даних із сенсорів у поєднанні з часовою міткою:

id — унікальний ідентифікатор запису;

timestamp — дата та час фіксації значень;

temperature — значення температури (у °C);

humidity — рівень відносної вологості (%);

light — рівень освітленості (умовні одиниці або аналогові значення);

sensor_status — службове поле, яке може містити статус вимірювання (наприклад, «норма», «перевищення» тощо).

Збереження даних відбувається на серверній частині після надходження чергового повідомлення з Arduino через послідовний порт. У Python-програмі, реалізованій на Flask, використовується бібліотека pyserial для зчитування рядків, які містять відформатовані значення з сенсорів. Після їх первинної обробки та перевірки на коректність дані записуються

3.3 Програмно - технологічне забезпечення та тестування системи

Для перевірки працездатності запропонованої архітектури інформаційної системи на базі IoT було реалізовано спрощений прототип (рисунок 3.4), що відображає основні функціональні можливості. Його метою є практичне

підтвердження коректності збору, обробки, збереження та виведення сенсорних даних у реальному часі.

```
iot-system/
├── arduino/
│   └── iot_sensors.ino
├── server/
│   ├── app.py
│   ├── templates/
│   │   └── dashboard.html
│   ├── static/
│   │   └── styles.css
│   └── database.db
├── README.md
└── requirements.txt
```

Рисунок 3.4 – Структура проекту

На першому етапі було налаштовано пристроєвий рівень за допомогою плати Arduino Leonardo, до якої підключено три аналогові сенсори: температури (наприклад, LM35), вологості (аналоговий сенсор DFRobot) та освітлення (фоторезистор). Програма, завантажена на Arduino, здійснює циклічне зчитування аналогових значень із сенсорів та формує з них структурований текстовий рядок, який передається через послідовний порт (USB) на комп'ютер кожних 5 секунд (рисунок 3.5).

```
const int tempPin = A0;
const int lightPin = A1;
const int humidityPin = A2;

void setup() {
  Serial.begin(9600);
}

void loop() {
  int tempVal = analogRead(tempPin);
  int lightVal = analogRead(lightPin);
  int humidityVal = analogRead(humidityPin);

  float tempC = (tempVal * 5.0 / 1023.0) * 100;  // LM35
  Serial.print(tempC);
  Serial.print(",");
  Serial.print(humidityVal);
  Serial.print(",");
  Serial.println(lightVal);

  delay(5000);
}

```

Рисунок 3.5 – Зчитування даних з сенсорів

На серверному рівні використано Python-фреймворк Flask, який приймає дані з порту за допомогою бібліотеки pyserial. Отримані значення проходять попередню перевірку, обробку та зберігаються у локальну реляційну базу даних SQLite. У структурі бази створено таблицю data, що містить поля для температури, вологості, освітлення та часової мітки.

Початковим етапом реалізації серверної частини системи є встановлення з'єднання з мікроконтролером Arduino через інтерфейс USB. Для цього у програмі використовується бібліотека serial, яка ініціалізує послідовний порт ser = serial.Serial('/dev/ttyACM0', 9600).

Порт задається '/dev/ttyACM0', який відповідає підключенню Arduino до комп'ютера, а також швидкість передачі даних — 9600 бод. Пристрій періодично надсилає у порт дані у вигляді рядків 23.4,512,701.

Щоб обробляти ці дані без зупинки вебсервера, в коді реалізовано окремий фоновий потік, який безперервно читає рядки з порту, перетворює їх на числові значення та зберігає у базу даних SQLite (рисунок 3.6).

```
def read_from_serial():
    conn = sqlite3.connect('database.db', check_same_thread=False)
    cursor = conn.cursor()
    cursor.execute('''CREATE TABLE IF NOT EXISTS data (temp REAL, humidity REAL, light REAL)'
    conn.commit()
    while True:
        line = ser.readline().decode().strip()
        try:
            temp, humidity, light = map(float, line.split(","))
            cursor.execute("INSERT INTO data VALUES (?, ?, ?)", (temp, humidity, light))
            conn.commit()
        except:
            continue
```

Рисунок 3.6 – Зчитування даних з платформи

Функція read_from_serial() виконує зчитування даних із послідовного порту, розбиває рядок на три числові значення та зберігає їх у таблицю data з трьома полями: температура, вологість та освітлення.

Паралельно з цим у системі працює веб-сервер, реалізований за допомогою фреймворку Flask. Він має один маршрут — головну сторінку (/). При зверненні до

цієї сторінки виконується вибірка останніх 10 записів з бази даних, які потім передаються до HTML-шаблону (рисунок 3.7):

```
@app.route('/')
def dashboard():
    conn = sqlite3.connect('database.db')
    cursor = conn.cursor()
    cursor.execute("SELECT * FROM data ORDER BY rowid DESC LIMIT 10")
    records = cursor.fetchall()
    return render_template('dashboard.html', data=records)
```

Рисунок 3.7 – Зчитування даних з платформи

Користувацький інтерфейс реалізовано у вигляді HTML-сторінки, створеної за допомогою шаблонізатора Jinja2. На цій сторінці відображаються останні 10 записів із бази у вигляді таблиці. Таким чином, користувач отримує доступ до актуальної інформації про стан середовища, зчитаної з сенсорів.

3.3.1 Тестування примітиву

На першому кроці було налаштовано віртуальну машину та порт яка слугувала сервером, як показано на рисунку 3.8.

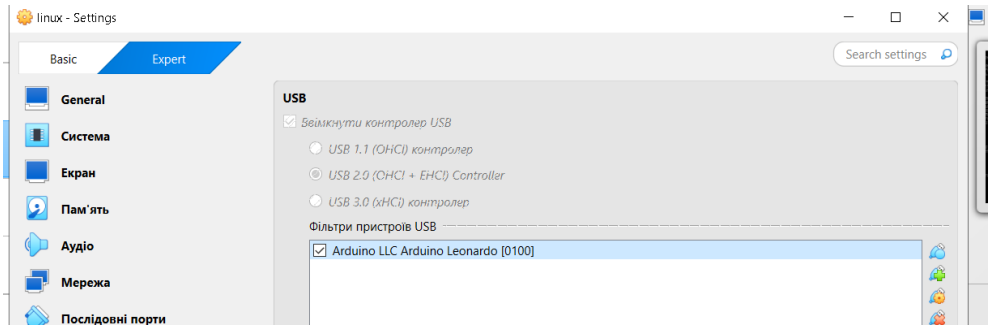


Рисунок 3.8 – Налаштування віртуальної машини та Arduino Leonardo

Далі можна запустити сервер, який отримує запити від браузера та окремим потоком зчитує дані з послідовного порта. Результат роботи представлено на рисунку 3.9.

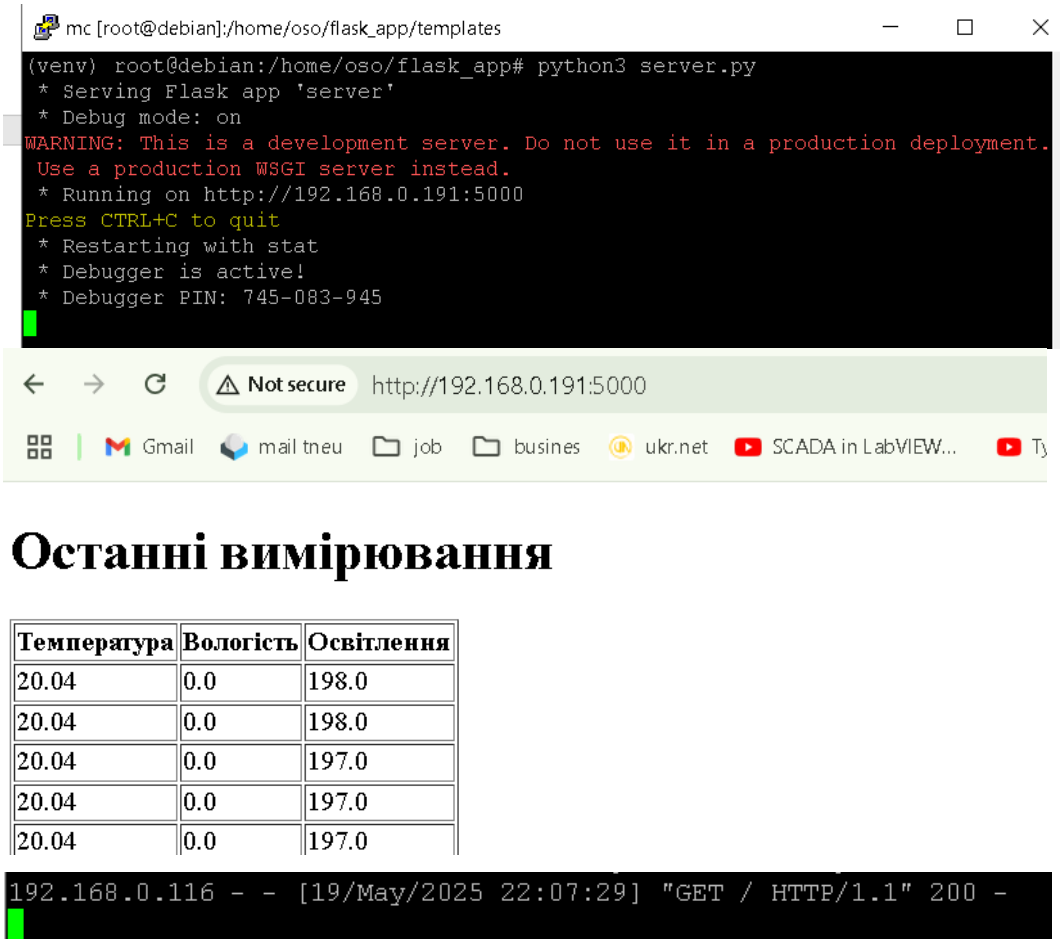


Рисунок 3.9 – Тестування взаємодії усіх компонентів

Було також оновлено примітив системи, як показано на рисунку 3.10.

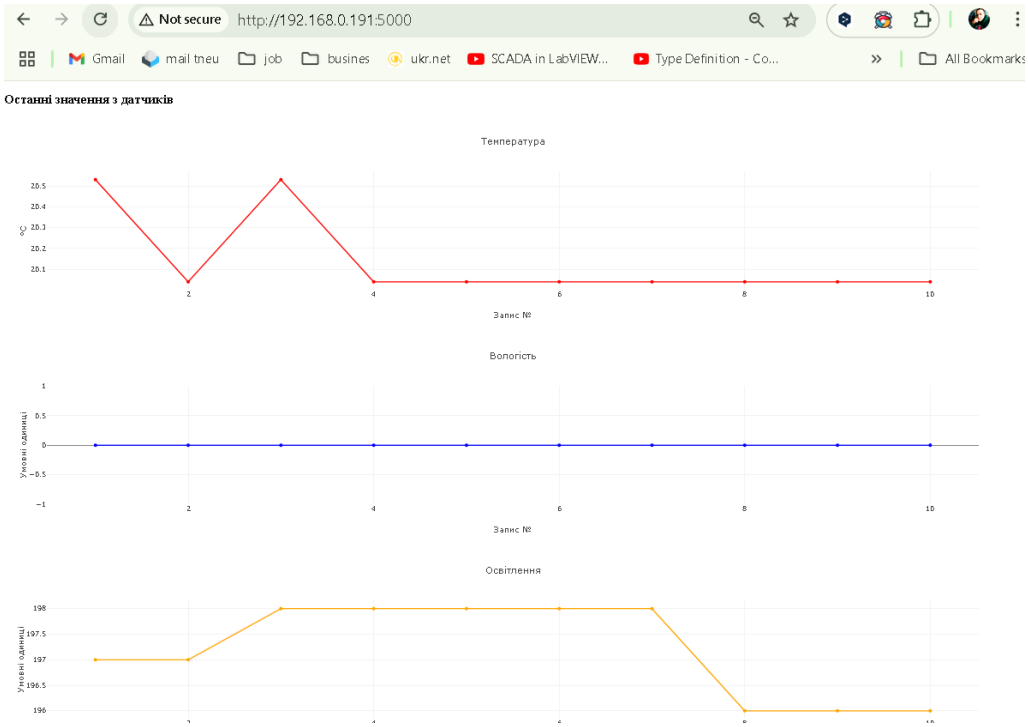


Рисунок 3.10 – Оновлений інтерфейс

Крім того було створено скрипт юніт-тест для тестування перевіряє чи створюється база, чи Flask-роут "/" доступний, чи з бази щось читається, результати якого показано на рисунку 3.11.

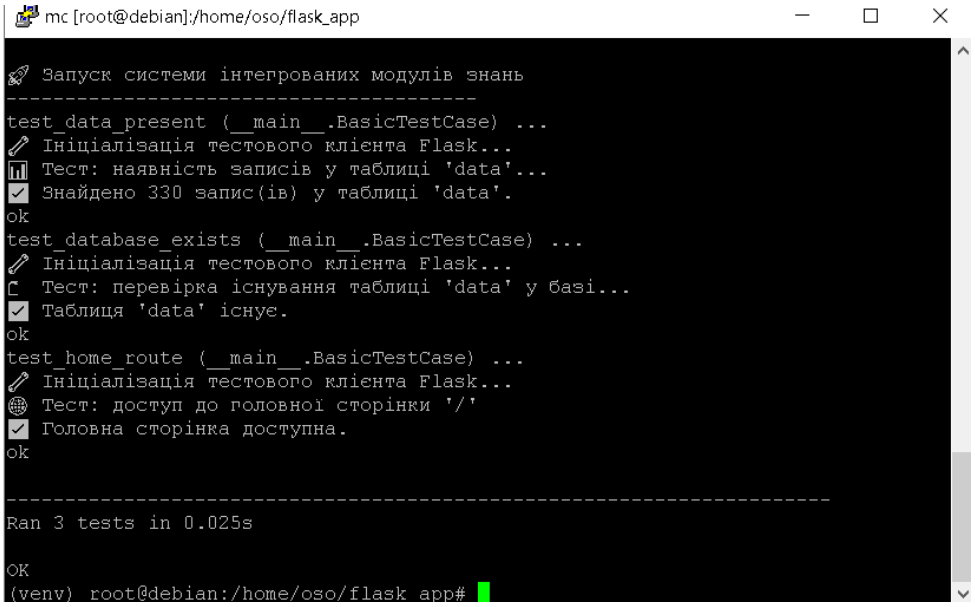


Рисунок 3.11 – Запуск автоматичного тесту

В загальному тестування примітиву інформаційної системи показав, що базові функції відпрацьовуються правильно, отже система може безперешкодно нарощуватись відповідно до основної архітектури та опису всіх алгоритмів функціонування.

ВИСНОВКИ

В даній кваліфікаційній роботі було розроблено інформаційну систему інтегрованих модулів знань IoT, яка дозволяє здійснювати збір, обробку, зберігання, візуалізацію та аналіз сенсорних даних у режимі реального часу з використанням багаторівневої архітектури.

1. Описано предметну область IoT в освітньому, науковому, промисловому та побутовому контекстах. Представлено сучасні концепції, приклади застосувань, аналітику ринку, виклики впровадження та перспективи.

2. Проведено огляд існуючих рішень, які реалізують багаторівневу взаємодію сенсорів, шлюзів, хмарних платформ та користувацьких застосунків. Розглянуто моделі Thing Description, Fog/Cloud архітектуру, платформні рішення, приклади з медицини, освіти та ін.

3. Сформовано постановку задічі та основні вимоги до системи.

4. Реалізовано ключові алгоритми функціонування системи, зокрема алгоритми самоорганізації, виявлення аномалій на основі LSTM, адаптивної візуалізації інтерфейсу, синхронізації модулів та пріоритетного обслуговування запитів.

5. Розроблено архітектуру інформаційної IoT-системи, що базується на принципах модульності, адаптивності та семантичної інтерпретації даних. У структурі системи виділено п’ять основних рівнів: фізичний, мережевий, обчислювальний, семантичний та інтерфейсний.

6. Здійснено програмну реалізацію системи з використанням мікроконтролера Arduino Leonardo, сенсорів температури, вологості та освітлення, а також серверної частини на Python із застосуванням Flask, SQLite, бібліотеки Plotly.js та шаблонізатора Jinja2 для динамічної візуалізації. Забезпечено підтримку легких протоколів обміну повідомленнями (MQTT, CoAP, WebSocket, HTTP/HTTPS) і форматів передачі даних (JSON, CBOR), що дає змогу ефективно взаємодіяти з гетерогенними пристроями та забезпечити масштабованість системи.

7. Проведено модульне тестування системи, що підтвердило її працездатність, стійкість до помилок і відповідність функціональним вимогам.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Number of connected IoT devices, 2024 [Електронний ресурс]. – Режим доступу: https://iot-analytics.com/number-connected-iot-devices/

2. Koreshoff, T. L., Robertson, T., Leong, T. W. Internet of Things: a review of literature and products. *Proceedings of the 25th Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration*. – 2013. – С. 335–344.

3. Abdullaha, A. S., Varolb, A. Smart Home Automation using Internet of Things (IoT). *International Turkish World Engineering and Science Congress*. – 2019. – С. 261–267.

4. Arasteh, H., Hosseinnezhad, V., Loia, V. та ін. IoT based smart cities: a survey. *2016 IEEE 16th International Conference on Environment and Electrical Engineering* (EEEIC). – IEEE, 2016. – С. 1–6.

5. Krasniqi, X., Hajrizi, E. Use of IoT technology to drive the automotive industry from connected to full autonomous vehicles. *IFAC-PapersOnLine*. – 2016. – Vol. 49, № 29. P. 269–274.

6. Ali, F. Teaching the internet of things concepts. *Proceedings of the WESE'15: Workshop on Embedded and Cyber-Physical Systems Education*. 2015. P. 1–6.

7. Verma, A., Singh, A., Anand, D., Aljahdali, H. M., Alsubhi, K., Khan, B. IoT Inspired Intelligent Monitoring and Reporting Framework for Education 4.0. *IEEE Access*. 2021. Vol. 9. P. 131286–131294.

8. Adi, A. Incorporating the Internet of Things (IoT) Learning for Higher Education: A Case Study on Smart Campus Framework. *Preprints*. 2025. № 5010769. – DOI: 10.20944/preprints202501.0769.v1.

9. Antonenko, T., Mykytenko, O., Zaselsky, V. Digital Competence Development and IoT-Driven Personalized Learning Systems. *CEUR Workshop Proceedings*. 2023.

10. Chien, C. Enhancing Higher Education Through IoT-Based Smart Classrooms. *Journal of Education and e-Learning Research*.

11. Alqahtani, A., Li, Y., Patel, P., Solaiman, E., Ranjan, R. End-to-end service level agreement specification for IoT applications. *2018 International Conference on High Performance Computing & Simulation* (HPCS). IEEE, 2018. P. 926–935.

12. Dar, K., Taherkordi, A., Rouvoy, R., Eliassen, F. Adaptable service composition for very-large-scale Internet of Things systems. *Proceedings of the 8th Middleware Doctoral Symposium*. ACM, 2011. P. 2.

13. CA AppLogic [Електронний ресурс]. – 2014. – Режим доступу: https://support.ca.com/us/product-information/ca-applogic.html

14. AWS OpsWorks [Електронний ресурс]. – 2018. – Режим доступу: https://aws.amazon.com/cn/opsworks/

15. Docker [Електронний ресурс]. – 2018. – Режим доступу: https://www.docker.com/

16. Jha, D. N., Garg, S., Jayaraman, P. P., Buyya, R., Li, Z., Ranjan, R. A holistic evaluation of Docker containers for interfering microservices. *2018 IEEE International Conference on Services Computing* (SCC). IEEE, 2018. P. 33–40.

17. Bytyçi, Eliot. SEMDPA: A semantic Web crossroad architecture for WSNs in the Internet of Things. In: Securing the Internet of Things: Concepts, Methodologies, Tools, and Applications. *IGI Global*, 2020. P. 977-998.

18. Nguyen, Lan Anh. Comprehensive survey of sensor data verification in internet of things. *IEEE Access*, 2023, 11: 50560-50577.

19. Boulaares S. Sassi S. Faiz S. Composition Reference Model (CCRM). In: Advances in Model and Data Engineering in the Digitalization Era: MEDI 2022 Short Papers and DETECT 2022 Workshop Papers, Cairo, Egypt, November 21–24, 2022, Proceedings. Springer Nature, 2023. P. 193.

20. Yang, W. Semantic communications for future internet: Fundamentals, applications, and challenges. *IEEE Communications Surveys & Tutorials*, 2022, 25.1: P. 213-250.

21. Islam Mir Salim, KUMAR, Ashok; HU, Yu-Chen. Context-aware scheduling in Fog computing: A survey, taxonomy, challenges and future directions. *Journal of Network and Computer Applications*, 2021.

22. Chen, Z.-L., Raghavan, S., Gray, P., Greenberg, H. J. State-of-the-art decision-making tools in the information-intensive age.

23. Lee, W.-P., Kaoli, C., Huang, J.-Y. A smart TV system with body-gesture control, tag-based rating and context-aware recommendation. *Knowledge-Based Systems*. 2014. Vol. 56. P. 167–178.

24. Zhang, J. Towards data-independent knowledge transfer in model-heterogeneous federated learning. *IEEE Transactions on Computers*. 2023. 72.10. P.2888-2901.

25. Mohapatra, Soumya Snigdha. QoS-aware cloud service recommendation using metaheuristic approach. *Electronics*, 2022, 11.21: 3469.

26. Gadekar S. Arduino Uno-ATmega328 P microcontroller based smart systems. In: Proceedings of the 3rd International Conference on Communication & Information Processing (ICCIP). 2021.

27. Mohanavel V. Diwakaran S. Maheswaran U. Anitha G. & Ramesh S. An Effective Design of Wireless Android based Robotic Operation Control using 8051 Microcontroller. In 2022 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI). P. 1-6.

28. Gupta, S. K. Arduino-based wireless hand gesture controlled robot. *In AIP Conference Proceedings* .Vol. 2916, No. 1.

29. Top, A., Gökbulut, M. Android application design with mit app inventor for bluetooth based mobile robot control. Wireless Personal Communications, 126(2). P.1403-1429.

30. Stolojescu-Crisan C., Crisan C., Butunoi B. P. An IoT-based smart home automation system. *Sensors*. 21(11). 3784.

31. Malche, T., Maheshwary, P. Internet of Things (IoT) for building smart home system. 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC). *IEEE*, 2017. P. 65–70.

32. Alaa, M., Zaidan, A. A., Zaidan, B. B., Talal, M., Kiah, M. L. M. A review of smart home applications based on Internet of Things. *Journal of Network and Computer Applications*. 2017. Vol. 97. P. 48–65.

33. Stojkoska, B. L. R., Trivodaliev, K. V. A review of Internet of Things for smart home: Challenges and solutions. *Journal of Cleaner Production*. 2017. Vol. 140. P. 1454–1464.

34. Jang, W., Chhabra, A., Prasad, A. Enabling multi-user controls in smart home devices. Proceedings of the 2017 Workshop on Internet of Things Security and Privacy. 2017. P. 49–54.

35. Al-Ali, A. R., Zualkernan, I. A., Rashid, M., Gupta, R., AliKarar, M. A smart home energy management system using IoT and big data analytics approach. IEEE Transactions on Consumer Electronics. 2017. Vol. 63, № 4. P. 426–434.

36. Elhoseny, M., Ramírez-González, G., Abu-Elnasr, O. M., Shawkat, S. A., Arunkumar, N., Farouk, A. Secure medical data transmission model for IoT-based healthcare systems. IEEE Access. 2018. Vol. 6. P. 20596–20608.

37. Bhatia, M., Kumari, S. . A novel IoT-fog-cloud-based healthcare system for monitoring and preventing encephalitis. *Cognitive Computation*. 2020. 14(5). P.1609-1626.

38. Verdejo Espinosa Á., Lopez J. L., Mata Mata F., Estevez, M. E. Application of IoT in healthcare: keys to implementation of the sustainable development goals. *Sensors*. (2021). 21(7). 2330.

39. Habibzadeh H., Dinesh K., Shishva, O. R., Boggio-Dandry A., Sharma G., Soyata T. A survey of healthcare Internet of Things (HIoT): A clinical perspective. *IEEE Internet of Things Journal*. 2019. 7(1).P. 53-71.

40. Babu, B. S., Srikanth, K., Ramanjaneyulu, T., Narayana, I. L. IoT for healthcare . International Journal of Science and Research. 2016. Vol. 5, № 2. P. 322–326.

41. Kashani M. H., Madanipour M., Nikravan M., Asghari P., Mahdipour E.. A systematic review of IoT in healthcare: Applications, techniques, and trends. *Journal of Network and Computer Applications*. 2021. 192. 103164.

42. Hossain, M. S., Muhammad, G. Cloud-assisted industrial internet of things (IIoT)–enabled framework for health monitoring. Computer Networks. 2016. Vol. 101. P. 192–202.

43. Joyia, G. J., Liaqat, R. M., Farooq, A., Rehman, S. Internet of Medical Things (IOMT): applications, benefits and future challenges in healthcare domain. *Journal of Communications*. 2017. Vol. 12, № 4. P. 240–247.

44. Malekshahi Rad M., Rahmani A. M., Sahafi A.,Nasih Qader N. Social Internet of Things: vision, challenges, and trends. Human-centric Computing and Information Sciences. 2020. 10(1), 52.

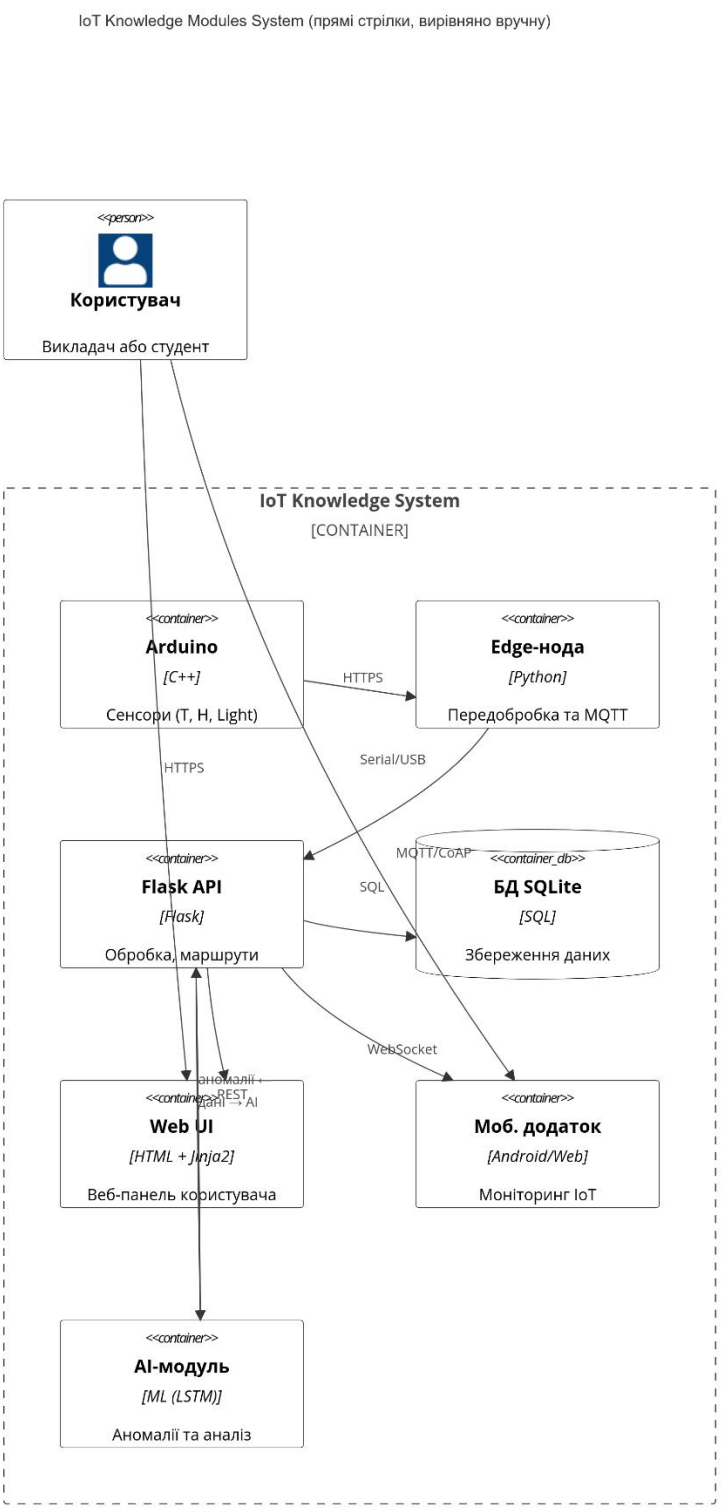
45. Вархів Б., Осолінський О. Інформаційна система інтегрованих модулів знань IoT. Інтелектуальні інформаційні технології в прикладних дослідженнях: тези доп. студентської наук.-практ. конф. (м. Тернопіль, 27–29 травня 2025 р.). Тернопіль, 2025. С.116-119

46. Комар М.П., Саченко А.О., Васильків Н.М., Гладій Г.М., Коваль В.С., Ліп'яніна-Гончаренко Х.В. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної програми «Комп'ютерні науки» спеціальності 122 «Комп'ютерні науки» за першим (бакалаврським) рівнем вищої освіти. Тернопіль: ЗУНУ, 2024. 52 с.

47. В. М. Островерхов, Л. І. Біловус, К. З. Возьний, О. О. Луцишин, Г. Л. Монастирський, С. А. Надвиничний, С. В. Питель,С. К. Шандрук., Загальні методичні рекомендації з підготовки, оформлення, захисту та оцінювання кваліфікаційних робіт здобувачів вищої освіти першого (бакалаврського) і другого (магістерського) рівнів. Тернопіль: ЗУНУ, 2024. 83 с.

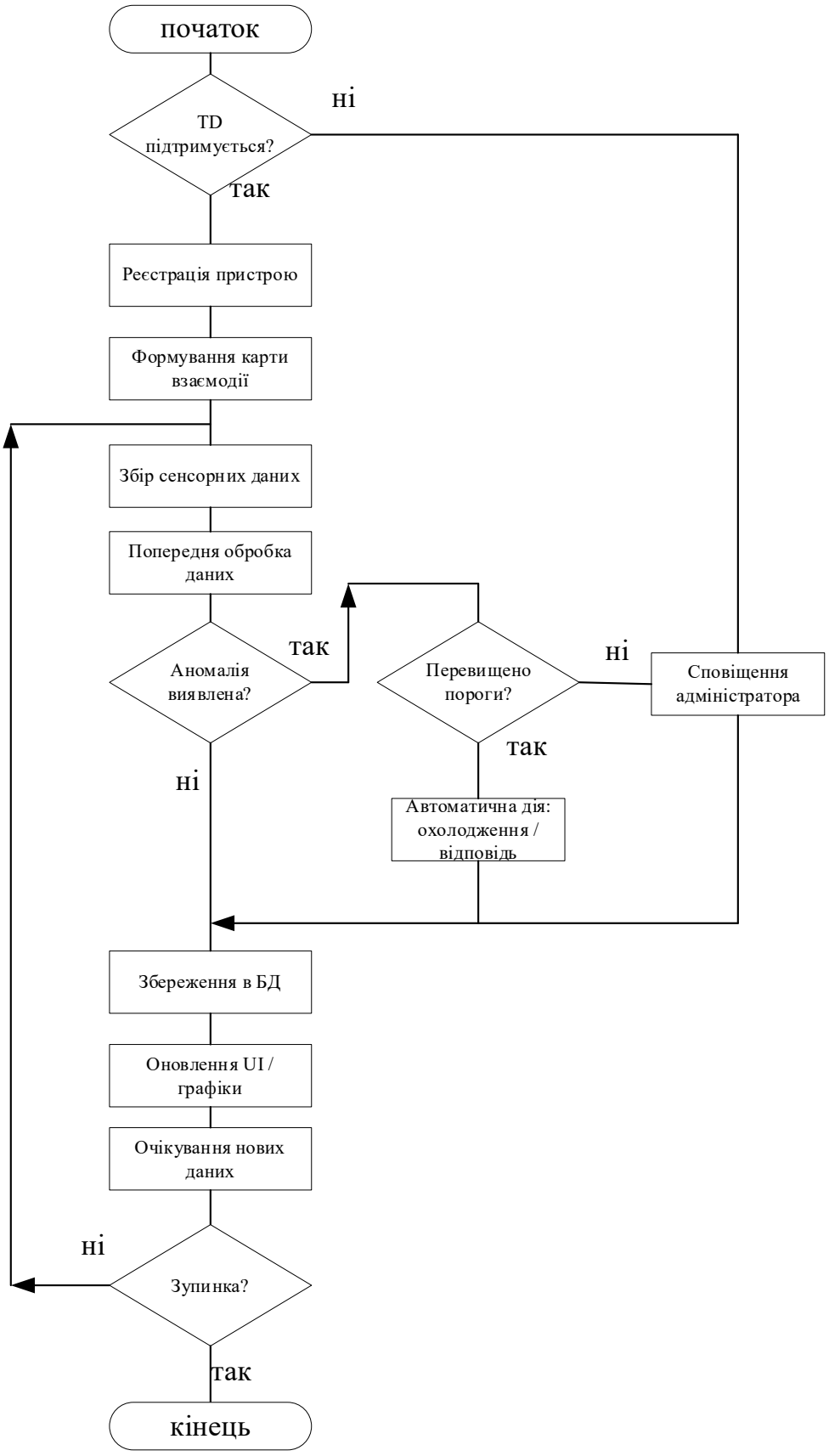
Додаток А

Архітектура інформаційної системи



Додаток Б

Загальний алгоритм функціонування інформаційної системи



Додаток В

Код апаратної та серверної частини інформаційної системи

Arduino.ino

```
const int tempPin = A0;
const int lightPin = A1;
const int humidityPin = A2;

void setup() {
  Serial.begin(9600);
}

void loop() {
  int tempVal = analogRead(tempPin);
  int lightVal = analogRead(lightPin);
  int humidityVal = analogRead(humidityPin);

  float tempC = (tempVal * 5.0 / 1023.0) * 100;  // LM35
  Serial.print(tempC);
  Serial.print(",");
  Serial.print(humidityVal);
  Serial.print(",");
  Serial.println(lightVal);

  delay(5000);
}
```

Server.py

```
from flask import Flask, render_template
import serial, threading, sqlite3

app = Flask(__name__)
ser = serial.Serial('/dev/ttyACM0', 9600)

def read_from_serial():
    conn = sqlite3.connect('database.db', check_same_thread=False)
    cursor = conn.cursor()
    cursor.execute('''CREATE TABLE IF NOT EXISTS data (temp REAL,
humidity REAL, light REAL)''')
    conn.commit()
    while True:
        line = ser.readline().decode().strip()
        try:
            temp, humidity, light = map(float, line.split(","))
            cursor.execute("INSERT INTO data VALUES (?, ?, ?)", (temp,
humidity, light))
            conn.commit()
        except:
            continue
```

```
@app.route('/')
def dashboard():
    conn = sqlite3.connect('database.db')
    cursor = conn.cursor()
    cursor.execute("SELECT * FROM data ORDER BY rowid DESC LIMIT 10")
    records = cursor.fetchall()
    return render_template('dashboard.html', data=records)

if __name__ == '__main__':
    t = threading.Thread(target=read_from_serial)
    t.daemon = True
    t.start()
    app.run(host='192.168.0.191', port=5000, debug=True)
```

Додаток Г

Код тестової програми

```
import sqlite3
import unittest
from server import app
import sys

class BasicTestCase(unittest.TestCase):
    def setUp(self):
        print("\n🔧 Ініціалізація тестового клієнта Flask...")
        self.app = app.test_client()

    def test_home_route(self):
        print("🌐 Тест: доступ до головної сторінки '/'")
        response = self.app.get('/')
        self.assertEqual(response.status_code, 200, "❌ Головна
сторінка недоступна.")
        print("✅ Головна сторінка доступна.")

    def test_database_exists(self):
        print("🗂️ Тест: перевірка існування таблиці 'data' у базі...")
        try:
            conn = sqlite3.connect('database.db')
            cursor = conn.cursor()
            cursor.execute("SELECT name FROM sqlite_master WHERE
type='table' AND name='data';")
            result = cursor.fetchone()
            self.assertIsNotNone(result, "❌ Таблиця 'data' не
знайдена в базі.")
            print("✅ Таблиця 'data' існує.")
        finally:
            conn.close()

    def test_data_present(self):
        print("📊 Тест: наявність записів у таблиці 'data'...")
        try:
            conn = sqlite3.connect('database.db')
            cursor = conn.cursor()
            cursor.execute("SELECT COUNT(*) FROM data;")
            count = cursor.fetchone()[0]
            self.assertGreater(count, 0, "❌ У таблиці 'data' немає
жодного запису.")
            print(f"✅ Знайдено {count} запис(ів) у таблиці 'data'.")
        finally:
            conn.close()

if __name__ == '__main__':
    print("\n🚀 Запуск системи інтегрованих модулів знань \n" + "-" *
40)
    unittest.main(verbosity=2)
```

Додаток Д
Апробація отриманих результатів

Західноукраїнський національний університет

Факультет комп'ютерних інформаційних технологій

Кафедра інформаційно-обчислювальних систем і управління

ЗБІРНИК ТЕЗ ДОПОВІДЕЙ

Студентської науково-практичної конференції

ІНТЕЛЕКТУАЛЬНІ ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ В ПРИКЛАДНИХ ДОСЛІДЖЕННЯХ

(IITAR-2025)

27-29 травня 2025 року

Тернопіль

2025

Збірник тез доповідей студентської науково-практичної конференції «Інтелектуальні інформаційні технології в прикладних дослідженнях» (ІІТАR-2025). (Тернопіль, 27-29 травня 2025 року). Тернопіль: ЗУНУ, 2025. 372 с.

До збірника увійшли тези доповідей учасників студентської науково-практичної конференції "Інтелектуальні інформаційні технології в прикладних дослідженнях" (ІІТАR – 2025), що відбувалась у рамках AI Week на базі кафедри інформаційно-обчислювальних систем і управління Західноукраїнського національного університету. Мета конференції — об'єднати науковців, освітян, представників ІТ-бізнесу та здобувачів освіти для обміну досвідом, презентації сучасних досліджень і впровадження інноваційних рішень у сфері комп'ютерних наук, штучного інтелекту та суміжних галузей.

За зміст наукових праць та достовірність наведених фактологічних і статистичних матеріалів відповідальність несуть автори публікацій та їхні наукові керівники. У збірнику зберігається стилістика та орфографія авторів матеріалів.

Склад організаційного комітету конференції

Керівництво оргкомітету

Мирослав Комар, д.т.н., професор, професор кафедри інформаційно-обчислювальних систем і управління

Христина Ліп'яніна-Гончаренко, д.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління

Надія Васильків, к.т.н., в.о. завідувача кафедри інформаційно-обчислювальних систем і управління

Члени програмного комітету

Василь Коваль, к.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління, заступник декана факультету комп'ютерних інформаційних технологій

Олександр Осолінський, к.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління

Павло Биковий, к.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління

Діана Загородня, к.т.н., доцент, доцент кафедри інформаційно-обчислювальних систем і управління

Ігор Майків, к.т.н., доцент кафедри інформаційно-обчислювальних систем і управління

Члени оргкомітету

Андрій Івасечко, викладач кафедри інформаційно-обчислювальних систем і управління

Дмитро Дюг, викладач кафедри інформаційно-обчислювальних систем і управління

Христина Юрків, студентка ФКІТ ЗУНУ, технік лабораторії з проблем інформаційних технологій

Мар'яна Соя, студентка ФКІТ ЗУНУ, лаборант кафедри інформаційно-обчислювальних систем і управління

Микола Телька, студент ФКІТ ЗУНУ, лаборант кафедри інформаційно-обчислювальних систем і управління, студентський декан факультету комп'ютерних інформаційних технологій

Вархів Богдан
студент групи КН-43
Varhiv446@gmail.com
Осолінський Олександр
к.т.н., доцент
oso@wunu.edu.ua
Західноукраїнський національний університет
Тернопіль, Україна

ІНФОРМАЦІЙНА СИСТЕМА ІНТЕГРОВАНИХ МОДУЛІВ ЗНАНЬ ІОТ

Останнім часом у зв'язку зі зростаючим значенням технологій і бізнесу IoT проводилися активні дослідження навчання IoT у рамках стратегії розвитку IoT. Станом на 2024 рік, глобальний ринок Інтернету речей (IoT) демонструє значне зростання та стабілізацію після пандемічних викликів. За даними IoT Analytics [1], кількість підключених IoT-пристроїв досягла 16,6 мільярда у 2023 році, а до кінця 2024 року очікується зростання на 13% до 18,8 мільярда пристроїв.

Застосування IoT в освіті формує нову парадигму — Освіта 4.0, яка орієнтована на інтеграцію новітніх цифрових технологій у навчальний процес. Це включає в себе розумні освітні середовища, які підтримують безперервний моніторинг, персоналізацію освітнього контенту, аналіз ефективності навчання та гнучке управління ресурсами [2].

Однією з ключових переваг IoT в освітньому контексті є можливість безперервного моніторингу студентів, викладачів і просторових умов. Така система дає змогу аналізувати участь студентів у навчальному процесі, прогнозувати академічні труднощі та формувати індивідуальні навчальні траєкторії [3].

Отже розробка інформаційної системи інтегрованих модулів знань IoT, яка дозволяє збирати, обробляти, зберігати, візуалізувати та аналізувати дані від сенсорів у режимі реального часу, з підтримкою адаптивних інтерфейсів, легких протоколів комунікації, самонавчальних алгоритмів та можливістю інтеграції у навчальне, наукове й прикладне середовище є цікавою та актуальною задечею

Архітектура інформаційної системи

Архітектура системи поділяється на кілька логічних рівнів (рисунок 1). Першим є фізичний рівень, який охоплює сенсори, що зчитують інформацію про середовище, мікроконтролери, а також виконавчі механізми — двигуни, реле, дисплеї.

Далі йде мережевий рівень, який відповідає за передачу даних за допомогою бездротових технологій, таких як Wi-Fi, Bluetooth, ZigBee, LoRaWAN, або 5G. Для цього використовуються протоколи MQTT, CoAP, HTTP(S), WebSocket, а в якості комунікаційних вузлів — Raspberry Pi або подібні пристрої.

На обчислювальному рівні здійснюється попередня обробка даних: фільтрація, агрегування та кодування, а також їх передача у хмару або на периферійні вузли.

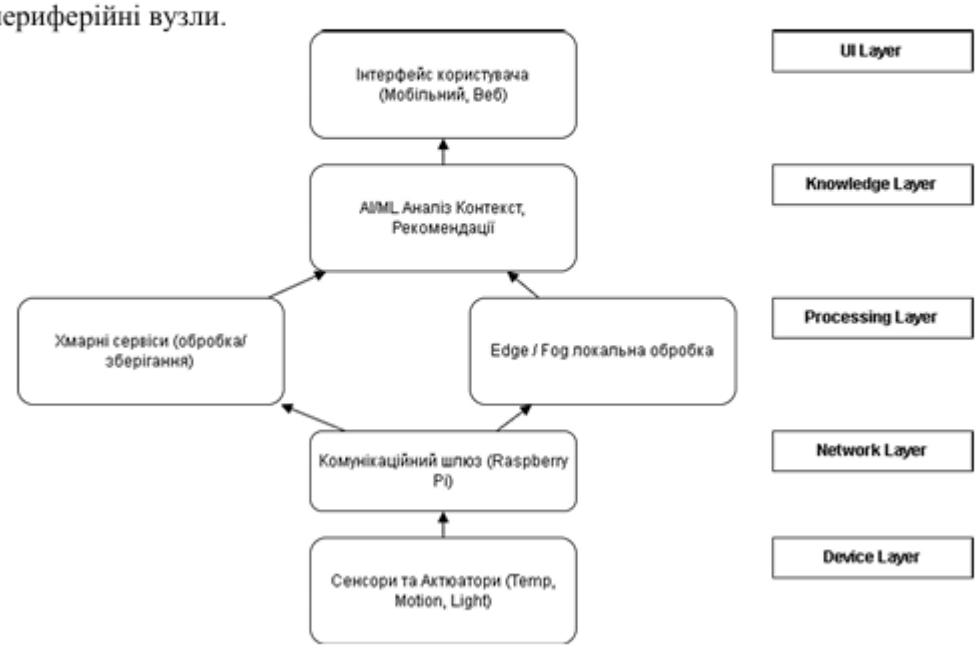


Рисунок 1 – Загальна архітектура інформаційної системи

Семантичний і знаннєвий рівень забезпечує контекстуалізацію даних. На рівні інтерфейсів важливу роль відіграють мобільні застосунки або веб-панелі, що дозволяють взаємодіяти із системою. Залежно від типу користувача — студент, викладач, адміністратор — інтерфейс може мати різний рівень складності й деталізації.

Інформаційна система включає модулі знань, кожен з яких містить набір сенсорів, локальну логіку обробки даних, інтерфейси зв'язку та механізми логування.

Центральний елемент такої системи є інтелектуальний контролер, який виконує функції координації між модулями, а також виконує адаптацію під різні режими використання, такі як навчальний, дослідницький або промисловий.

Реалізація системи

Для перевірки працездатності запропонованої архітектури інформаційної системи на базі IoT було реалізовано спрощений прототип (примітив), що відображає основні функціональні можливості. Його метою було підтвердження коректності збору, обробки, збереження та виведення сенсорних даних у реальному часі.

```
iot-system/
├── arduino/
│   └── iot_sensors.ino
├── server/
│   ├── app.py
│   ├── templates/
│   │   └── dashboard.html
│   ├── static/
│   │   └── styles.css
│   └── database.db
├── README.md
└── requirements.txt
```

Рисунок 2 – Структура проекту

На першому етапі було налаштовано пристроєвий рівень за допомогою плати Arduino Leonardo, до якої підключено три аналогові сенсори: температури, вологості та освітлення.

На серверному рівні використано Python-фреймворк Flask, який приймає дані з порту за допомогою бібліотеки pyserial. Отримані значення проходять попередню перевірку, обробку та зберігаються у локальну реляційну базу даних SQLite.

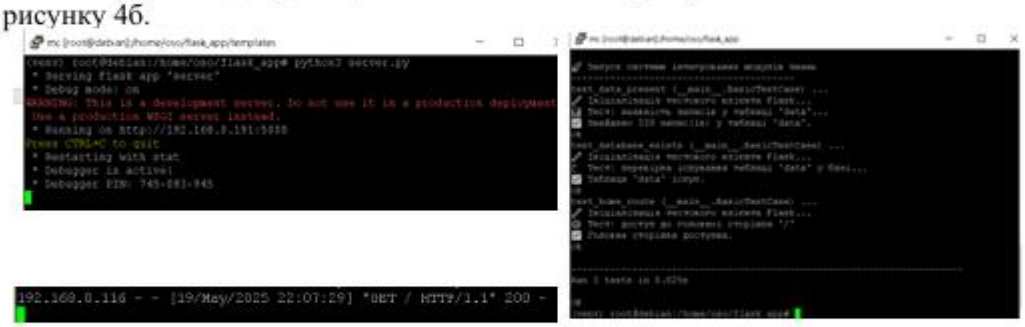
Щоб обробляти ці дані без зупинки вебсервера, в коді реалізовано окремий фоновий потік, який безперервно читає рядки з порту, перетворює їх на числові значення та зберігає у базу даних SQLite.

Користувацький інтерфейс реалізовано у вигляді HTML-сторінки, створеної за допомогою шаблонізатора Jinja2. На цій сторінці відображаються останні 10 записів із бази у вигляді таблиці (рисунок 3а) . Було також оновлено примітив системи (рисунок 3б)

(а) (б)

Рисунок 3 – Користувацький інтерфейс

Крім того протестовано взаємодію усіх компонентів (рисунок 4а) та створено скрипт юніт-тест для тестування, який перевіряє чи створюється база, чи Flask-роут "/" доступний, чи з бази читаються, результати якого показано на рисунку 4б.



(а) (б)

Рисунок 4 – Тестування взаємодії компонентів

Висновок

В загальному тестування примітиву інформаційної системи показало, що базові функції відпрацьовуються правильно і вона дозволяє здійснювати збір, обробку, зберігання, візуалізацію та аналіз сенсорних даних в режимі реального часу з використанням багаторівневої архітектури. Запропоноване рішення відповідає актуальним тенденціям розвитку цифрових освітніх технологій і система може безперешкодно нарощуватись відповідно до основної архітектури та опису всіх алгоритмів функціонування.

Список використаних джерел

1. Number of connected IoT devices, 2024 [Електронний ресурс]. – Режим доступу: https://iot-analytics.com/number-connected-iot-devices/
2. Adi, A. та ін. Incorporating the Internet of Things (IoT) Learning for Higher Education: A Case Study on Smart Campus Framework // *Preprints*. – 2025. – № 5010769. – DOI: 10.20944/preprints202501.0769.v1.
3. Chien, C. Enhancing Higher Education Through IoT-Based Smart Classrooms // *Journal of Education and e-Learning Research*. – 2023.