Міністерство освіти і науки України
Луцький національний технічний університет
(повне найменування закладу вищої освіти)
Факультет комп'ютерних та інформаційних технологій
(повне найменування факультету)
Кафедра комп'ютерної інженерії та охоронних систем
(повне найменування кафедри)

КВАЛІФІКАЦІЙНА РОБОТА ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

СИСТЕМА РАННЬОГО ВИЯВЛЕННЯ ПОРУШЕНЬ РОБОЧИХ ПАРАМЕТРІВ У СЕРВЕРНІЙ КІМНАТІ НА ОСНОВІ RASPBERRY PI
EARLY DETECTION SYSTEM FOR VIOLATIONS OF OPERATING PARAMETERS IN THE SERVER ROOM BASED ON RASPBERRY PI

спеціальність 123 Комп'ютерна інженерія
(шифр і назва спеціальності)

освітня програма Комп'ютерна інженерія
(назва освітньої програми)

Виконав: здобувач вищої освіти
групи КІ-41
Білашук Павло Васильович

(підпис)

Керівник:
к.т.н., доцент
Лавренчук Світлана Василівна

(підпис)

Кваліфікаційну роботу
допущено до захисту
«_____» _____ 2026 р.
Гарант освітньої програми:
к.т.н., доцент
Лавренчук Світлана Василівна

(підпис)

Луцьк – 2026 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

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

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ доц. Т. Терлецький
« 23 » 12 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Білашук Павло Васильович
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи *Система раннього виявлення порушень робочих параметрів у серверній кімнаті на основі Raspberry Pi*

Керівник роботи *к.т.н., доцент Лавренчук Світлана Василівна*

затверджені наказом закладу вищої освіти від «04» січня 2025 року № 11/01-02
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи 28.05.2026р.

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

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити):
Вступ
Теоретичні засади та сучасні підходи до моніторингу серверних кімнат
Розробка апаратної частини системи
Розробка програмного забезпечення системи моніторингу
Висновки

5. Перелік графічного (ілюстративного) матеріалу:
Електрична принципова схема пристрою
Інтерфейс пристрою
Тестування системи моніторингу

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Теоретичні засади та сучасні підходи до моніторингу серверних кімнат	*Лавренчук С.В., доцент*		
Розробка апаратної частини системи	*Лавренчук С.В., доцент*		
Розробка програмного забезпечення системи моніторингу	*Лавренчук С.В., доцент*		
Нормоконтроль	*Багнюк Н.В., доцент*		
Гарант ОП	*Лавренчук С.В., доцент*		
Показник запозичень тексту	***___%***		
Академічна доброчесність	*Міскевич О.І., ст. викладач*		

7. Дата видачі завдання 23.12.2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	*Огляд літератури із досліджуваної проблеми, аналіз предметної області та наявних рішень*	до 10.02.2026 р.	
2.	*Теоретичні засади та сучасні підходи до моніторингу серверних кімнат*	до 02.03.2026 р.	
3.	*Розробка апаратної частини системи*	до 02.04.2026 р.	
4.	*Розробка програмного забезпечення системи моніторингу*	до 10.04.2026 р.	
5.	*Формування списку використаних джерел*	до 15.04.2026 р.	
6.	*Формування додатків*	до 02.05.2026 р.	
7.	*Оформлення ілюстративного матеріалу*	до 10.05.2026 р.	
8.	*Представлення остаточного варіанту кваліфікаційної роботи керівникові*	до 15.05.2026 р.	
9.	*Нормоконтроль*	до 23.05.2026 р.	
10.	*Інструментальна перевірка на академічний плагіат*	до 26.05.2026 р.	
11.	*Здача кваліфікаційної роботи та всіх супровідних документів на кафедру*	до 28.05.2026 р.	

Здобувач вищої освіти _____ Білашук П.В.
(підпис) (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ Лавренчук С.В.
(підпис) (прізвище, ініціали)

АНОТАЦІЯ

Білашук П. В. Система раннього виявлення порушень робочих параметрів у серверній кімнаті на основі Raspberry Pi. Рукопис.

Кваліфікаційна робота бакалавра ОП «Комп'ютерна інженерія» спеціальності 123 Комп'ютерна інженерія. Луцький національний технічний університет. Луцьк, 2026.

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

Перший розділ присвячено теоретичному аналізу предметної області: розглянуто призначення серверних кімнат, нормативні вимоги до параметрів мікроклімату відповідно до стандарту ASHRAE TC9.9, вплив температури та вологості на роботу серверного обладнання, а також здійснено огляд існуючих систем моніторингу.

У другому розділі обґрунтовано вибір апаратної платформи Raspberry Pi Pico 2 W та датчика DHT11, описано їх технічні характеристики, переваги порівняно з аналогами, а також наведено схему підключення пристрою.

Третій розділ присвячено практичній реалізації: описано прошивку мікроконтролера мовою MicroPython, серверну частину на Node.js/Express із підсистемою Telegram-сповіщень та клієнтський веб-інтерфейс із візуалізацією даних у реальному часі. Наведено результати тестування системи.

Ключові слова: Raspberry Pi Pico 2 W, DHT11, MicroPython, IoT, моніторинг мікроклімату, серверна кімната, Node.js, Telegram Bot API.

ANNOTATION

Bilashuk P. Early detection system for violations of operating parameters in the server room based on Raspberry Pi. Manuscript.

Qualifying work of a bachelor of EP «Computer Engineering» specialty 123 Computer Engineering. Lutsk National Technical University. Lutsk, 2026.

The qualification work consists of an introduction, three chapters, conclusions, a list of sources used, and appendices.

The first chapter is devoted to a theoretical analysis of the subject area: the purpose of server rooms, regulatory requirements for microclimate parameters according to ASHRAE TC9.9, the effect of temperature and humidity on server equipment, and an overview of existing monitoring systems.

The second chapter justifies the choice of the Raspberry Pi Pico 2 W hardware platform and the DHT11 sensor, describes their technical characteristics, advantages over analogues, and provides a wiring diagram.

The third chapter is devoted to the practical implementation: MicroPython firmware, Node.js/Express server with a Telegram notification subsystem, and a real-time data visualization web interface. Testing results are presented.

Keywords: Raspberry Pi Pico 2 W, DHT11, MicroPython, IoT, microclimate monitoring, server room, Node.js, Telegram Bot API.

ЗМІСТ

ВСТУП

У сучасному світі інформаційні технології відіграють ключову роль у функціонуванні підприємств, установ та організацій. Основою більшості IT-інфраструктур є серверні кімнати, в яких розміщено обладнання для зберігання, обробки та передачі даних. Надійність роботи серверного обладнання безпосередньо впливає на стабільність функціонування інформаційних систем, а отже – і на ефективність діяльності організацій у цілому [11].

Однією з важливих умов забезпечення безперебійної роботи серверного обладнання є підтримання оптимальних параметрів навколишнього середовища. До таких параметрів належать температура та вологість повітря. Відхилення цих показників від допустимих норм може призвести до перегріву обладнання, зниження його продуктивності, передчасного зношування компонентів або навіть повного виходу з ладу. Зокрема, підвищена температура сприяє перегріванню процесорів, накопичувачів та інших елементів, тоді як надмірна вологість може викликати корозію контактів і короткі замикання.

У зв’язку з цим особливої актуальності набуває задача постійного моніторингу параметрів мікроклімату в серверних приміщеннях. Своєчасне виявлення відхилень дозволяє оперативно реагувати на потенційно небезпечні ситуації та запобігати аваріям. Традиційні промислові системи моніторингу є досить дорогими та складними у впровадженні, що робить їх недоступними для невеликих організацій або домашніх серверних рішень.

Альтернативою є використання недорогих мікрокомп’ютерів, таких як Raspberry Pi, які мають достатню обчислювальну потужність, низьке енергоспоживання та широкий набір інтерфейсів для підключення датчиків. Завдяки цьому вони широко застосовуються в системах Інтернету речей (IoT), автоматизації та моніторингу різноманітних об’єктів.

Використання Raspberry Pi у поєднанні з датчиками температури та вологості дозволяє створити ефективну систему раннього виявлення

небезпечних параметрів у серверній кімнаті. Така система здатна в режимі реального часу здійснювати збір даних, аналізувати їх та повідомляти користувача про перевищення допустимих значень. Це дає можливість значно підвищити надійність роботи обладнання та зменшити ризик аварійних ситуацій.

Таким чином, тема даної кваліфікаційної роботи є актуальною, оскільки спрямована на вирішення важливої практичної задачі забезпечення безпеки серверного обладнання за допомогою доступних та ефективних технічних засобів.

Метою роботи є розробка системи моніторингу температури та вологості у серверній кімнаті на основі Raspberry Pi для своєчасного виявлення небезпечних відхилень параметрів мікроклімату.

Об’єктом дослідження є процес моніторингу параметрів навколишнього середовища у серверних приміщеннях.

Предметом дослідження є методи та засоби вимірювання, обробки та аналізу температури і вологості з використанням мікрокомп’ютерних систем.

Завдання, які необхідно виконати:

– проаналізувати умови функціонування серверних кімнат;

– провести огляд існуючих систем моніторингу таких як Raspberry Pi (локальні системи), Zabbix, Nagios, PRTG Network Monitor та Blynk IoT;

– обґрунтувати вибір апаратної платформи Raspberry Pi;

– обрати та підключити датчик температури і вологості;

– розробити програмне забезпечення для збору та обробки даних;

– реалізувати систему оповіщення про критичні значення;

– провести тестування розробленої системи.

РОЗДІЛ 1
ТЕОРЕТИЧНІ ЗАСАДИ ТА СУЧАСНІ ПІДХОДИ ДО МОНІТОРИНГУ СЕРВЕРНИХ КІМНАТ

1.1 Загальні відомості про серверні кімнати

Серверна кімната (рис. 1.1) – це спеціалізоване приміщення, призначене для розміщення та експлуатації серверного і мережевого обладнання, систем зберігання даних та інженерної інфраструктури [7, 11]. Серверні приміщення забезпечують функціонування корпоративних мереж, вебсервісів, баз даних, хмарних платформ та інших інформаційних систем.

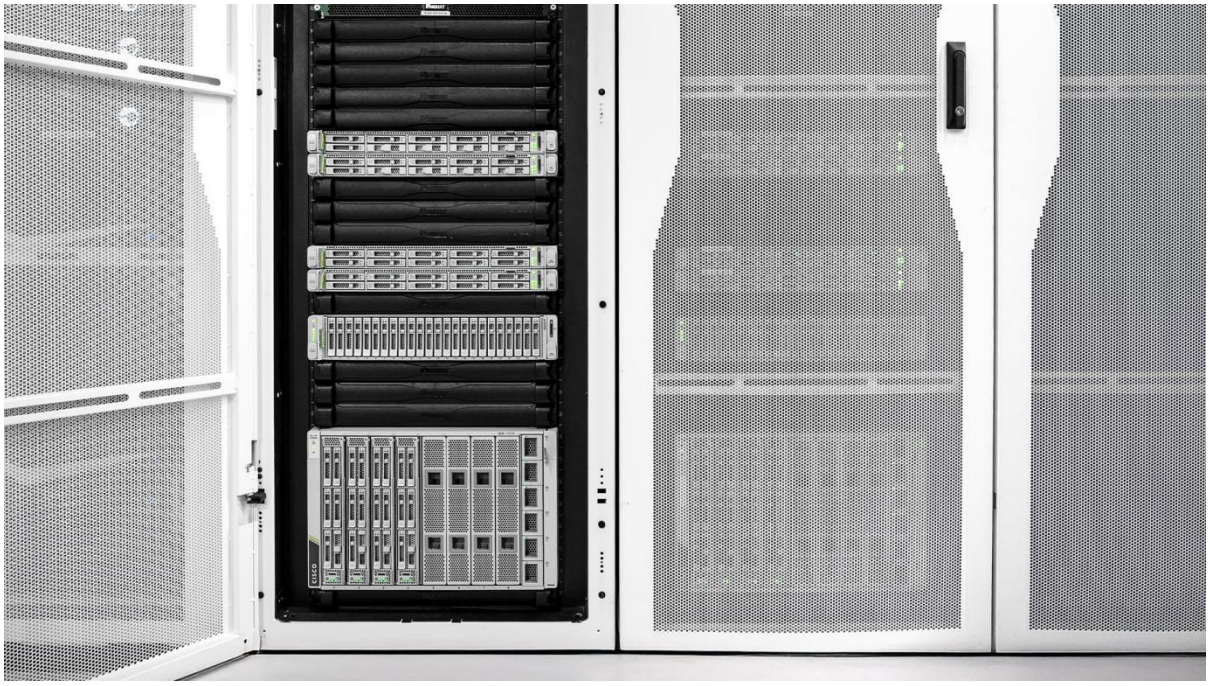

Рисунок 1.1 – Приклад серверної кімнати з серверними стійками [7]

До основного обладнання серверної кімнати належать сервери, комутатори, маршрутизатори, системи зберігання даних, джерела безперебійного живлення (UPS), системи вентиляції та кондиціонування [7, 23]. Надійність роботи серверного обладнання значною мірою залежить від умов експлуатації, зокрема температури, вологості та стабільності електроживлення.

Однією з основних вимог до серверних приміщень є підтримання нормативного мікроклімату. Відповідно до рекомендацій ASHRAE температура у серверній кімнаті повинна перебувати в межах 18-27° С, а відносна вологість у межах 40-60 % [5]. Порушення цих параметрів може призвести до перегріву обладнання, зниження продуктивності, пошкодження компонентів або втрати даних.

Важливою складовою серверної інфраструктури є система моніторингу, яка забезпечує контроль параметрів мікроклімату та стану обладнання у режимі реального часу. Сучасні системи моніторингу використовують датчики температури та вологості, мережеві технології та веб-інтерфейси для збору, аналізу та відображення даних [15, 24].

Для забезпечення безперервної роботи серверних приміщень також використовуються системи резервного живлення, вентиляції, пожежогасіння та контролю доступу [11, 23]. Комплексне використання таких засобів дозволяє підвищити надійність інформаційної інфраструктури та зменшити ризик аварійних ситуацій.

Таким чином, серверна кімната є важливим елементом IT-інфраструктури, а забезпечення належних умов її функціонування є необхідною умовою стабільної та безпечної роботи інформаційних систем.

1.2 Вплив параметрів мікроклімату на роботу серверів

Основними параметрами мікроклімату серверних приміщень, які впливають на надійність роботи обладнання, є температура та вологість повітря [5]. Під час роботи сервери, комутатори та інше мережеве обладнання виділяють значну кількість тепла, тому підтримання нормативних умов є обов’язковою умовою стабільного функціонування інформаційної інфраструктури.

Відповідно до рекомендацій ASHRAE, температура у серверних приміщеннях повинна підтримуватися в межах 18-27° С, а відносна вологість у

межах 40-60 % [5]. Вихід параметрів за допустимі межі може призвести до порушення роботи обладнання та скорочення терміну його експлуатації.

Підвищена температура є однією з основних причин перегріву серверного обладнання. Наслідками перегріву є зниження продуктивності через thermal throttling, збільшення енергоспоживання, прискорений знос компонентів та аварійне вимкнення серверів [5, 10]. Особливо чутливими до температурних перевантажень є процесори, блоки живлення та накопичувачі даних. Підвищення температури на кожні 10° С може суттєво скорочувати термін служби електронних компонентів [5].

Низька температура також є небажаною, оскільки може спричинити утворення конденсату на поверхні електронних компонентів. Це створює ризик короткого замикання та пошкодження обладнання [5].

Вологість повітря безпосередньо впливає на фізичний стан електронних компонентів. Підвищена вологість сприяє корозії контактів та утворенню конденсату, а надто низька – накопиченню статичної електрики та виникненню електростатичних розрядів (ESD), які можуть пошкодити мікросхеми та модулі пам’яті [5, 10].

Причинами порушення температурно-вологісного режиму можуть бути несправність систем кондиціонування, недостатня вентиляція, перевантаження серверного обладнання або неправильна організація повітряних потоків у серверній кімнаті [23]. Крім того, збільшення кількості обладнання у серверному приміщенні призводить до зростання тепловиділення та навантаження на системи охолодження.

Для забезпечення безпечної роботи серверної інфраструктури необхідним є безперервний моніторинг параметрів мікроклімату. Сучасні системи моніторингу використовують датчики температури та вологості, які передають дані до контролера або серверу для подальшого аналізу [15]. У разі перевищення допустимих значень система формує повідомлення адміністратору, що дозволяє оперативно реагувати на аварійні ситуації та запобігати виходу обладнання з ладу.

Таким чином, контроль температури та вологості є одним із ключових завдань під час експлуатації серверних приміщень. Постійний моніторинг параметрів мікроклімату дозволяє підвищити надійність роботи обладнання, зменшити ризик аварій та забезпечити стабільне функціонування інформаційних систем.

1.3 Огляд існуючих систем моніторингу серверних кімнат

У сучасних серверних приміщеннях моніторинг параметрів мікроклімату є обов'язковою складовою забезпечення безперервної роботи обладнання. Для контролю температури, вологості та стану інженерної інфраструктури застосовуються спеціалізовані апаратно-програмні системи моніторингу, які забезпечують збір, передачу, аналіз та візуалізацію даних у режимі реального часу [24].

Існуючі системи моніторингу серверних приміщень можна поділити на три основні групи:

– локальні системи моніторингу;

– мережеві централізовані системи;

– IoT та хмарні системи.

Локальні системи зазвичай будуються на базі мікроконтролерів або одноплатних комп'ютерів, зокрема Arduino, ESP32 або Raspberry Pi. Такі рішення використовують датчики температури та вологості (DHT11, DHT22) для збору параметрів середовища. Основною перевагою локальних систем є низька вартість реалізації, простота налаштування та можливість адаптації під конкретне приміщення. Недоліками є обмежена масштабованість та залежність від локальної мережі [1, 2, 4, 8, 21].

Одним із поширених рішень є системи на базі Raspberry Pi. Одноплатний комп'ютер виконує функції контролера, обробляє дані з датчиків та забезпечує веб-інтерфейс для перегляду параметрів у реальному часі [21].

Для передачі даних часто використовується протокол MQTT, який характеризується низьким мережевим навантаженням та підтримкою пристроїв типу IoT [15].

Мережеві системи моніторингу забезпечують комплексний контроль серверної інфраструктури та мережевого обладнання. До найбільш відомих систем належать Zabbix, Nagios та PRTG Network Monitor. Дані системи підтримують централізований моніторинг великої кількості пристроїв, журналювання подій, побудову графіків та автоматичне формування сповіщень [16, 18, 26].

Система Zabbix є однією з найбільш поширених платформ моніторингу корпоративної інфраструктури. Вона підтримує SNMP, MQTT, агентний моніторинг та має розвинену систему тригерів і сповіщень. Перевагами Zabbix є масштабованість та велика кількість підтримуваних пристроїв, проте налаштування системи є відносно складним для невеликих локальних проєктів [26].

Nagios використовується переважно для моніторингу серверів і мережевих сервісів. Система характеризується високою стабільністю та підтримкою великої кількості плагінів, однак має менш зручний інтерфейс порівняно з сучасними аналогами [16].

PRTG Network Monitor орієнтована на комерційне використання та має зручний графічний інтерфейс, готові шаблони моніторингу та інтегровану систему сповіщень. Недоліком системи є обмеження безкоштовної версії та висока вартість комерційної ліцензії [18].

Окрему групу становлять IoT та хмарні системи моніторингу, які забезпечують віддалений доступ до даних через мережу Інтернет. Для таких рішень використовуються платформу типу Blynk IoT. Дані системи дозволяють реалізувати веб-інтерфейс, мобільний доступ, аналітику та збереження історії вимірювань [6].

У таблиці 1.1 наведено порівняння основних систем моніторингу серверних приміщень.

Таблиця 1.1 – Порівняння існуючих систем моніторингу серверних кімнат

Система	Тип	Переваги	Недоліки
Raspberry Pi + DHT	Локальна IoT-система	низька вартість, простота реалізації	Обмежена масштабованість
Zabbix	Централізована мережева система	масштабованість, велика функціональність	Складність налаштування
Nagios	Мережева система	стабільність, підтримка плагінів	Застарілий інтерфейс
PRTG	Комерційна система	зручний інтерфейс, готові шаблони	Висока вартість
Blynk IoT	Хмарна IoT-платформа	віддалений доступ, мобільний контроль	Залежність від Інтернету

Проведений аналіз показує, що комерційні системи моніторингу мають широку функціональність, однак характеризуються високою вартістю впровадження та складністю адміністрування. Локальні IoT-рішення на базі Raspberry Pi або ESP32 є більш доступними та гнучкими для невеликих серверних приміщень, але потребують самостійної реалізації програмного забезпечення та системи сповіщень [6, 8, 13, 18, 21, 26].

З урахуванням проведеного аналізу для реалізації системи моніторингу у даній роботі доцільно використати одноплатний комп’ютер Raspberry Pi, датчик температури та вологості DHT11, веб-інтерфейс для візуалізації даних та систему сповіщень через Telegram. Такий підхід забезпечує низьку вартість реалізації, можливість віддаленого контролю та достатню функціональність для моніторингу серверного приміщення.

РОЗДІЛ 2
РОЗРОБКА АПАРАТНОЇ ЧАСТИНИ СИСТЕМИ

2.1 Обґрунтування вибору Raspberry Pi Pico 2 W для системи моніторингу

Raspberry Pi є серією одноплатних комп’ютерів та мікроконтролерів, розроблених Raspberry Pi Foundation у Великій Британії, які набули значної популярності у сфері освіти, прототипування та побудови IoT-рішень. Станом на 2021 рік було продано понад сорок мільйонів плат Raspberry Pi різних моделей, що свідчить про широке визнання платформи серед розробників та дослідників [21]. У межах екосистеми Raspberry Pi окремою лінійкою є серія Pico – мікроконтролерні плати, побудовані на базі чіпа RP2040, які орієнтовані на вбудовані застосування та IoT-проєкти. Raspberry Pi Pico 2 W (рис. 2.1), обрана для реалізації системи моніторингу температури та вологості у серверній кімнаті, є вдосконаленою версією цієї лінійки, що поєднує обчислювальні можливості мікроконтролера з вбудованим модулем бездротового зв’язку Wi-Fi, що є критично важливим для IoT-застосувань [21-22].

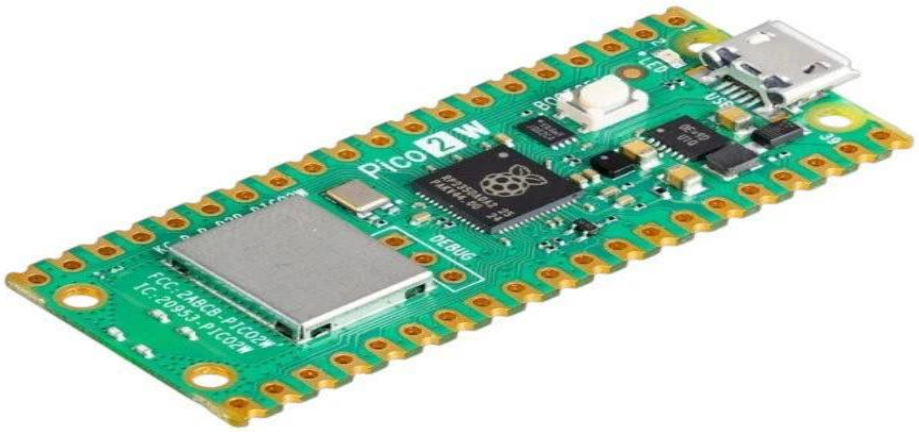

Рисунок 2.1 – Плата Raspberry Pi Pico 2 W [22]

З точки зору технічних характеристик, платформа Raspberry Pi Pico базується на мікроконтролері RP2040, який містить двоядерний процесор ARM Cortex-M0+ та 264 КБ оперативної пам’яті SRAM [21]. Плата має 26 багатофункціональних GPIO-виводів, які можуть бути сконфігуровані для різних функцій, включаючи цифровий ввід/вивід, широтно-імпульсну модуляцію (PWM), а також підтримку інтерфейсів I2C, SPI, UART та аналого-цифрового перетворення (ADC). Така розгалужена система вводу/виводу дозволяє Pico взаємодіяти з широким спектром компонентів, забезпечуючи адаптивність до різноманітних вимог проєктів [21]. Версія Pico W додає можливості бездротового з’єднання завдяки чіпу Infineon CYW43439, який підтримує стандарт Wi-Fi 802.11n на частоті 2,4 ГГц, що дозволяє підключати пристрій до мережі Інтернет та здійснювати передачу даних без дротового з’єднання [22]. Важливою перевагою є те, що розташування 40 GPIO-виводів на платі Pico W є ідентичним до базової версії Pico, що забезпечує повну апаратну сумісність між моделями [21-22].

Можливість підключення датчиків температури та вологості є одним із ключових факторів вибору Raspberry Pi Pico 2 W для розроблюваної системи. Наявність інтерфейсів I2C, SPI та цифрових GPIO-виводів дозволяє безпосередньо підключати широкий спектр сенсорів без необхідності використання додаткових конверторів [21]. Зокрема, датчики серії DHT (DHT11, DHT22) підключаються через цифрові GPIO-виводи та використовують однопровідний протокол передачі даних, що значно спрощує апаратну реалізацію [1-2]. Для більш точних вимірювань можуть використовуватися датчики на базі інтерфейсу I2C, який потребує двох сигнальних ліній – SDA та SCL – для обміну даними між мікроконтролером та сенсором [21]. Крім того, наявність трьох каналів ADC на платі Pico дозволяє підключати аналогові датчики температури, що розширює можливості вибору сенсорного обладнання [21]. Варто зазначити, що Pico також має вбудований датчик температури, який може використовуватися для додаткового контролю температури самої плати [21-22].

Суттєвою перевагою платформи Raspberry Pi Pico є підтримка мови програмування MicroPython, яка є оптимізованою реалізацією Python для мікроконтролерів з обмеженими ресурсами [12, 19]. MicroPython включає вбудовану підтримку доступу та керування апаратними периферійними пристроями, такими як GPIO, UART, I2C, SPI та ADC, що робить цю мову особливо придатною для IoT-застосувань, де взаємодія з датчиками та виконавчими пристроями є критично важливою [12]. Середовище розробки Thonny, яке є безкоштовним та доступним для завантаження, забезпечує зручний інтерфейс для написання коду на Python та його завантаження на плату Pico [12, 21]. Наявність великої кількості готових бібліотек для роботи з різноманітними датчиками та периферійними пристроями значно прискорює процес розробки та знижує поріг входження для розробників [12]. Найбільш зручним способом початкового завантаження програми є підключення Pico до комп’ютера через USB-інтерфейс, після чого плата розпізнається як накопичувач, і користувач може перетягнути файл прошивки безпосередньо у пам’ять пристрою [12, 21].

Порівняння Raspberry Pi Pico 2 W з альтернативними платформами, такими як Arduino та ESP32, дозволяє обґрунтувати доцільність зробленого вибору. Arduino Uno, будучи однією з найпоширеніших платформ для навчання та прототипування, базується на 8-бітному мікроконтролері ATmega328P з тактовою частотою 16 МГц та має лише 2 КБ оперативної пам’яті та 32 КБ флеш-пам’яті [4, 8]. Порівняно з Arduino, Raspberry Pi Pico пропонує значно вищу обчислювальну потужність завдяки двоядерному 32-бітному процесору ARM Cortex-M0+ та 264 КБ SRAM, що дозволяє реалізовувати більш складну логіку обробки даних безпосередньо на мікроконтролері [21]. Крім того, Arduino Uno не має вбудованого модуля Wi-Fi, що вимагає використання додаткових модулів для забезпечення бездротового зв’язку, тоді як Pico W має інтегрований Wi-Fi. ESP32, у свою чергу, є потужним мікроконтролером з вбудованими Wi-Fi та Bluetooth, який широко використовується в IoT-проєктах [4, 8]. ESP32 має GPIO-виводи з підтримкою I2C, SPI, UART та ADC, що

робить його функціонально подібним до Pico. Однак Raspberry Pi Pico 2 W має перевагу у вигляді значно ширшої спільноти користувачів екосистеми Raspberry Pi, що забезпечує кращу документацію, більшу кількість навчальних матеріалів та легший доступ до технічної підтримки. Обмеження у використанні інших пристроїв для реалізації IoT-проєктів пов’язане саме з розміром спільноти, оскільки ширша спільнота користувачів Raspberry Pi забезпечує легші дискусії та застосування у нових сферах, а також платформа пропонує значно більше периферійних пристроїв для комунікації та підключення [4, 8, 22].

З точки зору вартості та енергоспоживання, Raspberry Pi Pico 2 W є одним із найбільш економічно ефективних рішень на ринку мікроконтролерів. Плата Pico спроєктована з урахуванням низького енергоспоживання, що є критично важливим для систем моніторингу, які повинні функціонувати безперервно протягом тривалого часу [21]. Компактний розмір плати (21 × 51 мм) дозволяє розміщувати її у обмеженому просторі серверної кімнати без значного впливу на існуючу інфраструктуру. Живлення здійснюється через порт micro-USB із напругою 5 В, що забезпечує простоту підключення до стандартних джерел живлення [21-22].

Можливість віддаленого доступу та обробки даних є однією з визначальних переваг Raspberry Pi Pico 2 W для системи моніторингу серверної кімнати. Вбудований модуль Wi-Fi дозволяє передавати зібрані дані про температуру та вологість на віддалений сервер або хмарну платформу в реальному часі без необхідності фізичного доступу до пристрою [22]. Завдяки підтримці MicroPython розробник може реалізувати веб-сервер безпосередньо на мікроконтролері, що дозволяє здійснювати моніторинг параметрів через веб-браузер з будь-якого пристрою, підключеного до мережі [12, 17]. Платформа Anvil та інші хмарні сервіси можуть використовуватися для створення інтерфейсів керування та візуалізації даних, що значно розширює функціональні можливості системи. Raspberry Pi Pico можна ефективно використовуватися як IoT-пристрій, забезпечуючи збір даних від сенсорів та їх

передачу через бездротові інтерфейси [21]. Таким чином, поєднання низької вартості, достатньої обчислювальної потужності, вбудованого Wi-Fi, підтримки MicroPython та широкого набору інтерфейсів для підключення датчиків робить Raspberry Pi Pico 2 W оптимальним вибором для реалізації системи моніторингу температури та вологості у серверній кімнаті [12, 22].

2.2 Вибір датчика температури та вологості DHT11

Датчик DHT11 (рис. 2.2) є одним із найпоширеніших цифрових сенсорів для вимірювання температури та відносної вологості повітря, який широко застосовується у навчальних проєктах, системах домашньої автоматизації та простих системах моніторингу навколишнього середовища. Цей датчик являє собою компактний модуль, що поєднує в одному корпусі чутливий елемент для вимірювання вологості резистивного типу та термістор для вимірювання температури, а також спеціалізований 8-бітний мікроконтролер, який здійснює аналого-цифрове перетворення та формує цифровий вихідний сигнал. Завдяки простоті підключення, низькій вартості та достатній для багатьох застосувань точності, DHT11 набув значної популярності серед розробників IoT-систем та студентів, які реалізують навчальні проєкти з моніторингу параметрів середовища [3, 20].

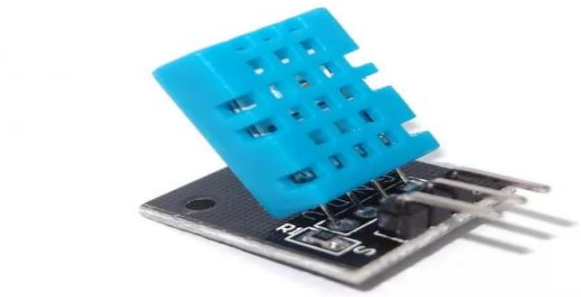

Рисунок 2.2 – Датчик температури та вологості DHT11 [3]

Принцип роботи датчика DHT11 базується на двох фізичних механізмах вимірювання. Для визначення відносної вологості повітря використовується резистивний полімерний елемент, електричний опір якого змінюється залежно від кількості вологи, адсорбованої на його поверхні. При збільшенні вологості повітря полімерний субстрат поглинає молекули води, що призводить до зменшення його електричного опору, і навпаки при зниженні вологості опір зростає. Для вимірювання температури застосовується негативний температурний коефіцієнт (NTC) термістор, опір якого зменшується при підвищенні температури навколишнього середовища. Вбудований мікроконтролер зчитує аналогові значення з обох чутливих елементів, здійснює їх перетворення у цифровий формат та формує пакет даних для передачі зовнішньому мікроконтролеру через однопровідний цифровий інтерфейс. Процес вимірювання ініціюється мікроконтролером, який надсилає стартовий сигнал на лінію даних, після чого DHT11 відповідає послідовністю з 40 біт, що містять значення вологості, температури та контрольну суму для верифікації коректності переданих даних [1, 3, 20].

Основні технічні характеристики датчика DHT11 визначають сферу його застосування [3]. Діапазон вимірювання відносної вологості становить від 20 % до 90 % RH з точністю ±5 % RH, а діапазон вимірювання температури від 0° C до 50° C з точністю ±2° C [3]. Роздільна здатність вимірювання становить 1 % для вологості та 1° C для температури, що є цілочисельними значеннями без десяткової частини. Період опитування датчика становить не менше однієї секунди, тобто нові дані можуть бути отримані не частіше ніж один раз на секунду, що є достатнім для більшості систем моніторингу навколишнього середовища, де параметри змінюються повільно [1, 3]. Робоча напруга живлення датчика становить від 3,3 В до 5,5 В, що забезпечує сумісність як з 3,3-вольтовими мікроконтролерами, такими як Raspberry Pi Pico 2 W, так і з 5-вольтовими платформами на базі Arduino [3]. Струм споживання під час вимірювання не перевищує 2,5 мА, а в режимі очікування становить лише 100–150 мкА, що свідчить про низьке енергоспоживання датчика [1, 4, 21].

Інтерфейс взаємодії DHT11 з мікроконтролерами реалізований через однопровідний цифровий протокол, який використовує лише один сигнальний вивід для двонаправленого обміну даними [3]. Фізичне підключення датчика потребує лише трьох з'єднань: живлення (VCC), земля (GND) та лінія даних (DATA), причому між лінією даних та живленням рекомендується встановлювати підтягуючий резистор номіналом 4,7-10 кОм для забезпечення стабільного рівня сигналу [20]. Протокол передачі даних передбачає, що мікроконтролер ініціює комунікацію, утримуючи лінію даних у низькому стані протягом щонайменше 18 мілісекунд, після чого DHT11 відповідає послідовністю імпульсів, що кодують 40 біт інформації: 8 біт цілої частини вологості, 8 біт дробової частини вологості, 8 біт цілої частини температури, 8 біт дробової частини температури та 8 біт контрольної суми [1, 3]. Контрольна сума обчислюється як сума перших чотирьох байтів і використовується для перевірки цілісності отриманих даних, що підвищує надійність вимірювань. Простота протоколу та мінімальна кількість з'єднань роблять DHT11 особливо зручним для підключення до мікроконтролерів з обмеженою кількістю виводів [20].

Переваги використання DHT11 у навчальних та простих системах моніторингу є багатоплановими. Насамперед, датчик характеризується надзвичайно низькою вартістю, що робить його доступним для студентських проєктів та прототипування [3]. Простота підключення, яка потребує лише одного GPIO-виводу мікроконтролера, значно спрощує апаратну реалізацію системи [20]. Наявність готових бібліотек для MicroPython, C/C++ та інших мов програмування дозволяє швидко інтегрувати датчик у програмний код без необхідності реалізації низькорівневого протоколу обміну даними вручну [12]. Компактні розміри датчика (12 × 15,5 × 5,5 мм) дозволяють розміщувати його у обмеженому просторі без значного впливу на конструкцію системи [3]. Крім того, DHT11 демонструє високу стабільність показань у довготривалій перспективі та має тривалий термін служби, що є важливим для систем безперервного моніторингу [1, 3].

Водночас датчик DHT11 має певні обмеження, які необхідно враховувати при проєктуванні системи [3]. Точність вимірювання температури ±2° C та вологості ±5 % RH є нижчою порівняно з більш дорогими аналогами, що може бути недостатнім для прецизійних застосувань [3]. Обмежений діапазон вимірювання температури (0-50° C) не дозволяє використовувати датчик в умовах від'ємних температур або при значному перегріві обладнання [3]. Роздільна здатність в 1° C та 1 % RH без десяткових значень обмежує можливість виявлення незначних змін параметрів середовища [1, 3]. Мінімальний інтервал опитування в одну секунду може бути недостатнім для застосувань, що потребують високочастотного збору даних, хоча для моніторингу серверної кімнати ця частота є цілком прийнятною [20].

Порівняння DHT11 з альтернативними датчиками дозволяє краще зрозуміти його позиціонування. DHT22 (також відомий як AM2302) є покращеною версією, що пропонує ширший діапазон вимірювання температури від −40° C до 80° C з точністю ±0,5° C та вологості від 0 % до 100 % RH з точністю ±2–5 % RH, а також роздільну здатність 0,1° C та 0,1 % RH [1-3]. Однак DHT22 має вищу вартість та більший мінімальний інтервал опитування дві секунди замість однієї.

DS18B20 є цифровим датчиком температури з високою точністю ±0,5° C у діапазоні від −10° C до +85° C та роздільною здатністю до 0,0625° C, проте він вимірює лише температуру без можливості визначення вологості, що потребувало б використання додаткового сенсора [1].

Датчики серії BME280 забезпечують високу точність вимірювання температури (±1° C), вологості (±3 % RH) та додатково атмосферного тиску через інтерфейс I2C, однак їхня вартість є значно вищою.

Обґрунтування вибору DHT11 для даного проєкту базується на кількох ключових факторах. По-перше, діапазон вимірювання температури 0-50° C є цілком достатнім для моніторингу серверної кімнати, де температура зазвичай підтримується в межах 18-27° C відповідно до рекомендацій стандартів експлуатації серверного обладнання [3, 5]. По-друге, точність ±2° C є

прийнятною для системи раннього попередження, основною метою якої є виявлення суттєвих відхилень температури від норми, а не прецизійне вимірювання [3]. По-третє, низька вартість датчика відповідає бюджетним обмеженням студентського проєкту та дозволяє за необхідності встановити декілька датчиків у різних точках серверної кімнати [3]. По-четверте, простота підключення до Raspberry Pi Pico 2 W через один GPIO-вивід та наявність готових бібліотек для MicroPython забезпечують швидку інтеграцію датчика у розроблювану систему [12, 20, 21]. Нарешті, низьке енергоспоживання DHT11 узгоджується з вимогами до системи безперервного моніторингу, яка повинна функціонувати цілодобово без значного впливу на енергетичний баланс серверного приміщення [1, 3].

Таким чином, DHT11 є оптимальним вибором для реалізації прототипу системи моніторингу температури та вологості у серверній кімнаті, забезпечуючи необхідний баланс між функціональністю, простотою реалізації та економічною доцільністю.

2.3 Підключення датчика DHT11 до Raspberry Pi Pico 2 W

Для реалізації системи моніторингу серверної кімнати використовується макетна плата (breadboard), на якій розміщено Raspberry Pi Pico 2 W та датчик DHT11. Підключення датчика здійснюється трьома провідниками: живлення (VCC – 3,3 В), земля (GND) та лінія даних (DATA – GPIO 10). Між лінією даних та живленням встановлено підтягуючий резистор номіналом 10 кОм для забезпечення стабільного рівня сигналу на шині даних та коректної роботи протоколу однодротового зв’язку DHT11. Живлення Raspberry Pi Pico 2 W здійснюється через порт micro-USB від стандартного блоку живлення 5 В, що забезпечує стабільну напругу для мікроконтролера та підключених периферійних пристроїв. Вбудований стабілізатор напруги мікроконтролера перетворює вхідні 5 В на 3,3 В, необхідні для живлення датчика та логічних

рівнів GPIO. Така конфігурація є типовою для прототипування вбудованих систем збору та моніторингу даних навколишнього середовища.

Загальний вигляд зібраного апаратного вузла системи моніторингу зображено на рисунку 2.3. На фотографії видно Raspberry Pi Pico 2 W, встановлений на макетній платі, та датчик DHT11 (синій модуль у верхній частині), підключений трьома провідниками: помаранчевий – DATA (GPIO 10), жовтий – VCC (3,3 В), сірий – GND.

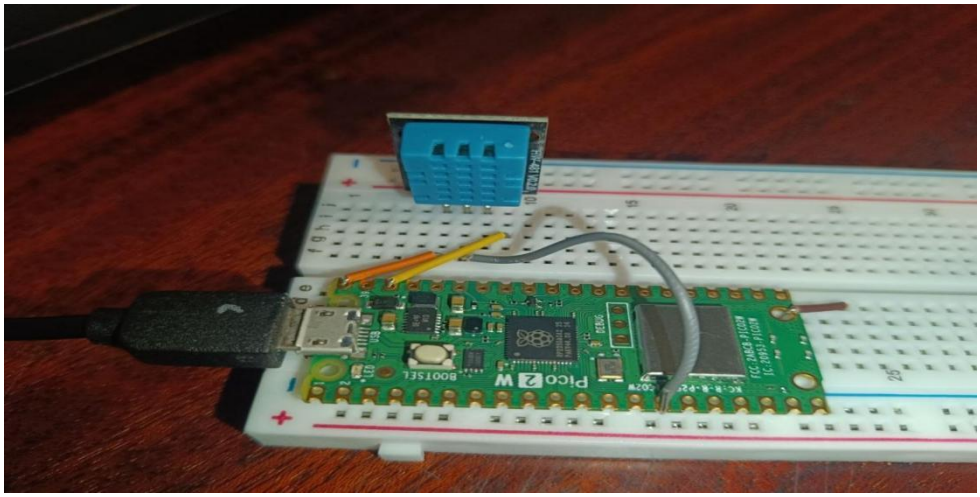

Рисунок 2.3 – Апаратний вузол системи моніторингу: Raspberry Pi Pico 2 W з датчиком DHT11

Компактний розмір плати (21 × 51 мм) та живлення від USB дозволяють розмістити пристрій безпосередньо у серверній кімнаті без значного впливу на існуючу інфраструктуру. Після монтажу та завантаження прошивки система починає автоматично збирати дані з датчика та передавати їх на сервер кожні 5 секунд.

РОЗДІЛ 3
РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ МОНІТОРИНГУ

3.1 Постановка задачі програмної реалізації

Основним завданням програмної реалізації системи раннього виявлення небезпечних параметрів у серверній кімнаті є створення програмного забезпечення, яке забезпечить автоматичний збір, передачу, обробку та відображення даних про стан мікроклімату в режимі реального часу.

Розроблювана система повинна забезпечувати безперервний контроль температури та вологості у серверному приміщенні з використанням мікроконтролера Raspberry Pi Pico 2 W та датчика DHT11. Отримані дані мають передаватися через бездротову мережу Wi-Fi на серверну частину системи для подальшої обробки та збереження.

Програмне забезпечення системи повинно виконувати такі основні функції:

– зчитування показників температури та вологості з датчика DHT11;

– обробка отриманих даних;

– підключення Raspberry Pi Pico 2 W до бездротової мережі Wi-Fi;

– передача даних на сервер у режимі реального часу;

– збереження інформації про параметри середовища;

– відображення поточних значень у веб-інтерфейсі;

– надсилання сповіщень у разі перевищення допустимих значень температури або вологості;

– забезпечення стабільної та безперервної роботи системи моніторингу.

Архітектура програмного забезпечення складається з двох основних частин:

– програмного забезпечення мікроконтролера Raspberry Pi Pico 2 W;

– серверної частини системи.

Програмне забезпечення мікроконтролера відповідає за роботу з датчиком DHT11, отримання даних та їх передачу через Wi-Fi. Для реалізації даної частини системи використовується мова програмування MicroPython.

Серверна частина системи реалізується за допомогою платформи Node.js та забезпечує приймання даних, їх обробку, формування веб-інтерфейсу та реалізацію системи сповіщень через Telegram.

Загальний алгоритм роботи системи можна описати таким чином:

– мікроконтролер зчитує показники температури та вологості з датчика DHT11;

– отримані значення аналізуються програмою;

– дані передаються через Wi-Fi на сервер;

– сервер приймає та обробляє інформацію;

– поточні значення відображаються у веб-інтерфейсі;

– у разі перевищення допустимих параметрів формується аварійне повідомлення;

– користувач отримує сповіщення через Telegram-бота.

Основною метою програмної реалізації є створення простої, надійної та ефективної системи моніторингу, яка дозволить оперативно виявляти небезпечні зміни параметрів мікроклімату у серверній кімнаті та своєчасно реагувати на потенційні аварійні ситуації.

3.2 Розробка програмного забезпечення для Raspberry Pi Pico 2 W

Програмне забезпечення для мікроконтролера Raspberry Pi Pico 2 W було розроблено з метою забезпечення автоматичного збору даних із датчика температури та вологості DHT11, їх обробки та передачі на серверну частину системи через бездротову мережу Wi-Fi. Для програмної реалізації було використано мову програмування MicroPython, яка є спрощеною версією Python для мікроконтролерних систем. На початковому етапі розробки було виконано налаштування середовища програмування та встановлення

MicroPython на мікроконтролер. Для завантаження програмного коду використовувалось середовище Thonny IDE, яке підтримує пряме підключення до Raspberry Pi Pico 2 W та дозволяє швидко редагувати й тестувати програму [12, 22].

Програмне забезпечення мікроконтролера складається з декількох основних модулів:

– модуль підключення до Wi-Fi мережі;

– модуль роботи з датчиком DHT11;

– модуль обробки отриманих даних;

– модуль передачі інформації на сервер;

– модуль контролю аварійних параметрів.

Для зчитування даних із датчика DHT11 використовуються вбудовані бібліотеки MicroPython. Датчик підключається до одного з GPIO-портів Raspberry Pi Pico 2 W та періодично передає значення температури й вологості (лістинг 3.1).

Лістинг 3.1 – Ініціалізація датчика DHT11 та зчитування даних

```
 ===== ДАТЧИК =====
sensor = dht.DHT11(Pin(10))

# ===== ПАРАМЕТРИ =====
TEMP_LIMIT = 30
HUM_LIMIT = 70
```

кінець лістингу 3.1

Після запуску мікроконтролера виконується автоматичне підключення до бездротової мережі Wi-Fi. Для цього у програмі задаються параметри мережі, зокрема назва точки доступу та пароль (лістинг 3.2).

Лістинг 3.2 – Підключення Raspberry Pi Pico 2 W до Wi-Fi мережі

```
# ===== WIFI =====
SSID = "ASUS_78"
PASSWORD = "autumn_3795"

def connect_wifi():
```

```
    wlan = network.WLAN(network.STA_IF)
    wlan.active(True)
    wlan.connect(SSID, PASSWORD)

    print("Connecting to WiFi...", end="")
    while not wlan.isconnected():
        print(".", end="")
        time.sleep(1)

    print("\nConnected:", wlan.ifconfig())
```

кінець лістингу 3.2

Після успішного підключення до мережі мікроконтролер переходить у режим постійного моніторингу параметрів середовища. З певним інтервалом часу програма виконує зчитування температури та вологості, після чого отримані значення аналізуються.

Для підвищення надійності системи у програмі реалізовано перевірку коректності отриманих даних. У випадку помилки зчитування виконується повторна спроба отримання інформації з датчика (лістинг 3.3).

Лістинг 3.3 – Обробка помилок під час зчитування даних

```
# ===== MAIN =====
connect_wifi()
time.sleep(2)
while True:
    try:
        sensor.measure()
        temp = sensor.temperature()
        hum = sensor.humidity()
        status = "OK"

        if temp > TEMP_LIMIT or hum > HUM_LIMIT:
            status = "ALERT"

        print(f"T={temp}C H={hum}% STATUS={status}")

        send_data(temp, hum, status)

    except Exception as e:
        print("Sensor error:", e)

    time.sleep(5)
```

кінець лістингу 3.3

Отримані значення температури та вологості передаються на сервер за допомогою HTTP-запитів. Серверна частина системи приймає ці дані та відображає їх у веб-інтерфейсі моніторингу (лістинг 3.4).

Лістинг 3.4 – Передача даних на сервер

```
# ===== ВІДПРАВКА =====
def send_data(temp, hum, status):
    try:
        data = {
            "temperature": temp,
            "humidity": hum,
            "status": status
        }
        res = urequests.post(SERVER_URL, json=data)
        res.close()
    except Exception as e:
        print("Send error:", e)
```

кінець лістингу 3.4

Одним із важливих елементів програмного забезпечення є реалізація системи раннього виявлення небезпечних параметрів. У програмі задаються гранично допустимі значення температури та вологості. Якщо отримані параметри перевищують встановлені межі, система формує аварійне повідомлення та передає його на сервер.

Для забезпечення безперервної роботи системи використовується циклічне виконання програми. Мікроконтролер постійно повторює процес зчитування даних, їх аналізу та передачі на сервер. Після завершення одного циклу програма робить коротку затримку, після чого повторно виконує вимірювання параметрів середовища. У випадку помилки зчитування даних або втрати Wi-Fi-з’єднання виконується повторна спроба підключення та передачі інформації. Такий підхід забезпечує стабільний моніторинг параметрів серверної кімнати у режимі реального часу.

Середовище розробки програмного забезпечення Raspberry Pi Pico 2 W наведено на рис. 3.1.

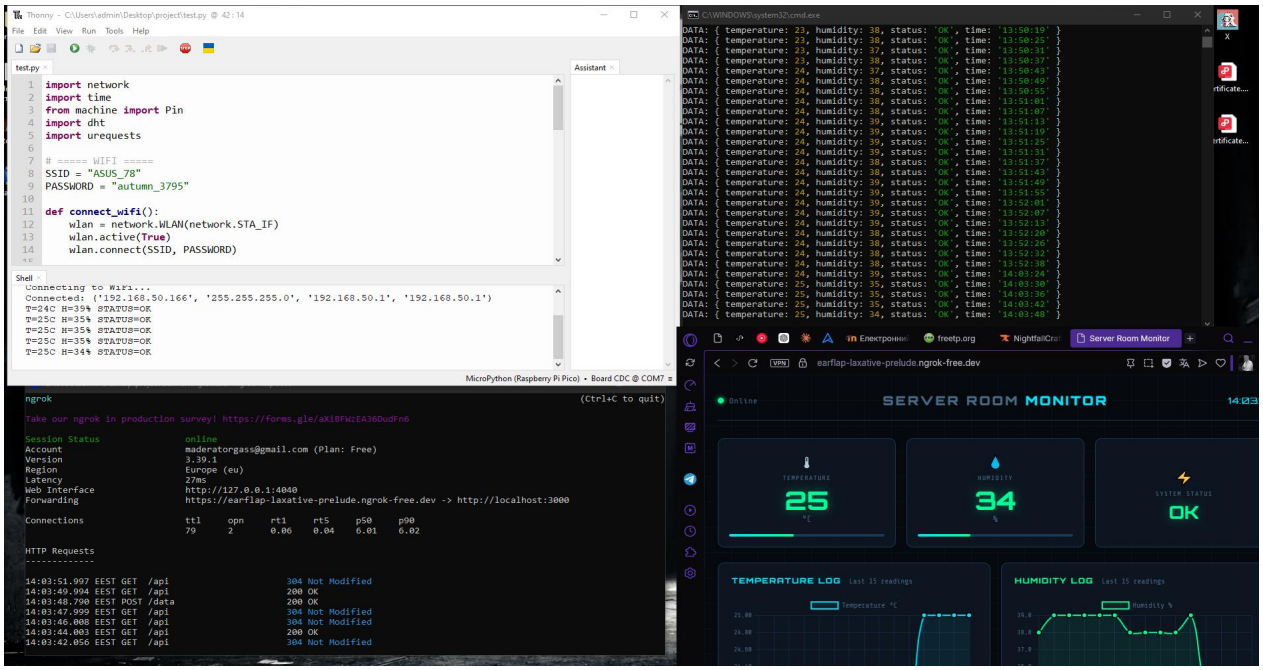

Рисунок 3.1 – Середовище розробки програмного забезпечення мікроконтролера

У результаті розробки було створено програмне забезпечення для Raspberry Pi Pico 2 W, яке забезпечує автоматичний контроль параметрів мікроклімату серверної кімнати, передачу даних через бездротову мережу та можливість оперативного реагування на виникнення небезпечних ситуацій.

3.3 Розробка серверної частини системи на Node.js

Для забезпечення роботи системи раннього виявлення небезпечних параметрів у серверній кімнаті було розроблено серверну частину програмного забезпечення на платформі Node.js. Серверна частина системи виконує функції приймання даних від мікроконтролера Raspberry Pi Pico 2 W, їх обробки, збереження, передачі до веб-інтерфейсу, а також надсилання аварійних повідомлень користувачу через Telegram .

Використання Node.js дозволяє забезпечити швидку обробку HTTP-запитів та підтримку роботи системи у режимі реального часу. Для створення веб-сервера було використано фреймворк Express.js, який значно спрощує реалізацію API та маршрутизації.

На початковому етапі розробки було створено серверний застосунок та підключено необхідні бібліотеки (лістинг 3.5).

Лістинг 3.5 – Підключення бібліотек та створення сервера Node.js

```
const express = require("express");
const path = require("path");
const https = require("https");
const app = express();
app.use(express.json());
app.use(express.static(__dirname));
```

кінець лістингу 3.5

У наведеному фрагменті коду підключаються основні модулі системи. Бібліотека Express використовується для створення веб-сервера, модуль Path забезпечує роботу зі шляхами до файлів, а HTTPS використовується для надсилання запитів до Telegram Bot API. Також реалізовано підтримку JSON-формату та доступ до статичних файлів веб-інтерфейсу.

Для реалізації системи сповіщень було створено Telegram-бота. Сервер використовує Telegram Bot API для автоматичного надсилання повідомлень користувачу у випадку перевищення допустимих параметрів температури або вологості (лістинг 3.6).

Лістинг 3.6 – Налаштування Telegram-бота

```
const BOT_TOKEN = "TOKEN";
const CHAT_ID = "CHAT_ID";
```

кінець лістингу 3.6

Токен бота було отримано за допомогою сервісу BotFather, а ідентифікатор користувача – через Telegram-бот userinfobot. Дані параметри використовуються для надсилання повідомлень у визначений чат.

Для формування та надсилання повідомлень реалізовано окрему функцію (лістинг 3.7)

Лістинг 3.7 – Функція надсилання повідомлень у Telegram

```
sendTelegram().
function sendTelegram(message) {
const body = JSON.stringify({
chat_id: CHAT_ID,
text: message,
parse_mode: "HTML"
});
const options = {
hostname: "api.telegram.org",
path: `/bot${BOT_TOKEN}/sendMessage`,
method: "POST",
headers: {
"Content-Type": "application/json",
"Content-Length": Buffer.byteLength(body)
}
};

const req = https.request(options, res => {
if (res.statusCode !== 200) {
console.error("Telegram error:", res.statusCode);
}
});
req.write(body);
req.end();
}
```

кінець лістингу 3.7

Дана функція формує HTTPS-запит до Telegram API та передає текст повідомлення користувачу. Для покращення зовнішнього вигляду повідомлень використовується HTML-форматування.

Одним із найважливіших елементів серверної частини є система контролю аварійних ситуацій. Для цього було реалізовано функцію maybeAlert(), яка аналізує поточний стан системи та визначає необхідність надсилання аварійного повідомлення (лістинг 3.8).

Лістинг 3.8 – Реалізація системи аварійних сповіщень

```
const lastAlert = {};
```

```
function maybeAlert(status, temp, hum) {
const now = Date.now();
const COOLDOWN = 60000;
if (lastAlert.status === status &&
now - (lastAlert.time || 0) < COOLDOWN) {
return;
}
lastAlert.status = status;
lastAlert.time = now;
sendTelegram(
`Температура: ${temp}°C\n` +
`Вологість: ${hum}%\n` +
`Статус: ${status}`
);
}
```

кінець лістингу 3.8

Для уникнення надмірного надсилання повідомлень використовується механізм затримки (cooldown). Якщо повідомлення одного типу вже було відправлене раніше, повторне повідомлення надсилається лише через визначений проміжок часу.

Приймання даних від Raspberry Pi Pico 2 W реалізовано за допомогою HTTP POST-запиту. Мікроконтролер передає дані у форматі JSON, які містять значення температури, вологості та статус системи (лістинг 3.9).

Лістинг 3.9 – Приймання та обробка даних від Raspberry Pi Pico 2 W

```
let history = [];
app.post("/data", (req, res) => {
const { temperature, humidity, status } = req.body;
const point = {
temperature,
humidity,
status,
time: new Date().toLocaleTimeString("uk-UA")
};
history.push(point);
if (history.length > 100) {
history.shift();
}
maybeAlert(status, temperature, humidity);
res.json({ ok: true });
});
```

кінець лістингу 3.9

Після отримання інформації сервер формує новий запис та додає його до масиву history, який використовується для збереження історії вимірювань. Для економії пам’яті кількість записів обмежується останніми 100 значеннями.

Для передачі інформації у веб-інтерфейс було створено API-маршрут /api, який повертає історію вимірювань у форматі JSON (лістинг 3.10).

Лістинг 3.10 – API для отримання даних системи

```
app.get("/api", (req, res) => {
res.json(history);
});
```

кінець лістингу 3.10

Веб-інтерфейс системи завантажується автоматично через головний маршрут сервера (лістинг 3.11).

Лістинг 3.11 – Відображення веб-інтерфейсу системи

```
app.get("/", (req, res) => {
res.sendFile(path.join(__dirname, "index.html"));
});
```

кінець лістингу 3.11

Для запуску серверної частини використовується команда запуску Node.js сервера (лістинг 3.12).

Лістинг 3.12 – Запуск Node.js сервера

```
const PORT = 3000;
app.listen(PORT, "0.0.0.0", () => {
console.log(`Server running → http://localhost:${PORT}`);
});
```

кінець лістингу 3.12

Для спрощення запуску серверної частини у середовищі Windows було створено bat-файл (лістинг 3.13).

Лістинг 3.13 – bat-файл запуску серверної частини

```
@echo off
cd /d %~dp0
echo Starting IoT Server...
node server.js
pause
```

кінець лістингу 3.13

Для забезпечення віддаленого доступу до системи використовувався сервіс ngrok, який дозволяє створити зовнішній HTTPS-доступ до локального сервера. Завдяки цьому Raspberry Pi Pico 2 W може передавати дані на сервер навіть за межами локальної мережі.

У результаті розробки було створено серверну частину системи моніторингу, яка забезпечує стабільне приймання та обробку даних у режимі реального часу, підтримує веб-інтерфейс користувача та реалізує механізм аварійного сповіщення через Telegram.

3.4 Розробка веб-інтерфейсу моніторингу

Для відображення параметрів серверного приміщення було розроблено веб-інтерфейс системи моніторингу, який забезпечує візуалізацію даних у реальному часі. Інтерфейс побудований із використанням веб-технологій: HTML5, CSS3 та JavaScript, а також бібліотеки Chart.js для побудови графіків.

Основною метою розробки інтерфейсу є забезпечення зручного та наочного відображення таких параметрів, як температура, вологість та стан системи. Веб-сторінка має адаптивну структуру, що дозволяє коректно відображатися як на персональних комп’ютерах, так і на мобільних пристроях.

Інтерфейс складається з таких основних блоків:

1) заголовок (header) – містить індикатор підключення до сервера, назву системи та поточний час;

2) інформаційні картки (cards) – відображають ключові параметри:

– температура (°C);

– вологість (%);

– статус системи (нормальний стан / попередження / аварія);

3) графіки (charts) – відображають історію змін параметрів за останні вимірювання.

Для побудови графіків використовується бібліотека Chart.js, яка дозволяє створювати динамічні діаграми без складної конфігурації. На сторінці реалізовано два графіки:

– графік температури;

– графік вологості.

Вони відображають останні 15 значень, що дозволяє аналізувати динаміку змін параметрів у часі.

Для покращення візуального сприйняття використано кольорові індикатори та прогрес-бари. Наприклад, температура та вологість відображаються не лише числово, але й у вигляді заповнення шкали, що дозволяє швидко оцінити рівень параметра.

Також передбачено індикатор стану підключення до серверної частини, який змінює свій вигляд залежно від наявності зв’язку.

Нижче наведено рис. 3.2 структури веб-сторінки.

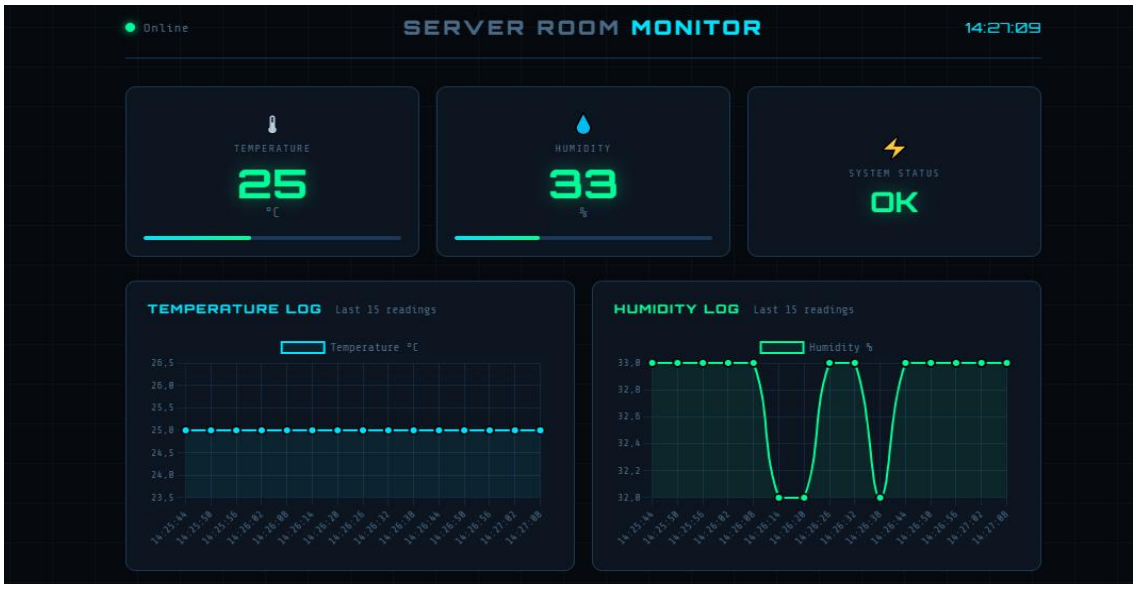

Рисунок 3.2 – Структура веб-сторінки

Розроблений веб-інтерфейс забезпечує зручний моніторинг параметрів серверного приміщення в режимі реального часу. Використання сучасних веб-технологій та бібліотеки Chart.js дозволило реалізувати інформативну та візуально зрозумілу систему відображення даних, яка може бути інтегрована з будь-якою серверною частиною через WebSocket або HTTP API. Архітектура IoT-системи моніторингу серверної кімнати охоплює чотири функціональні рівні: збір даних, обробка, серверна частина та відображення. Датчик DHT11 підключено до мікроконтролера Raspberry Pi Pico 2 W через лінію GPIO 10 із підтягуючим резистором 10 кОм, живлення здійснюється від джерела 5 В через micro-USB.

Мікроконтролер працює під керуванням MicroPython, виконує циклічне опитування датчика та передає дані на сервер через вбудований Wi-Fi модуль за протоколом HTTP. Серверна частина побудована на Node.js з Express.js і забезпечує приймання, обробку телеметричних даних та формування сповіщень при перевищенні порогових значень.

Вихідний рівень системи реалізовано через два канали: Telegram-бот для миттєвої доставки алертів оператору та веб-панель моніторингу з графічним відображенням температури й вологості в режимі реального часу. Загальна архітектура проєкту наведена в додатку Г.

ВИСНОВКИ

У кваліфікаційній роботі було розроблено систему раннього виявлення небезпечних параметрів у серверній кімнаті на основі Raspberry Pi Pico 2 W. У ході виконання роботи проведено аналіз умов функціонування серверних приміщень, визначено основні небезпечні параметри мікроклімату та досліджено їхній вплив на надійність і стабільність роботи серверного обладнання.

У першому розділі виконано аналіз предметної області, розглянуто нормативні вимоги до температури та вологості у серверних кімнатах відповідно до рекомендацій ASHRAE, а також здійснено огляд сучасних систем моніторингу таких як Raspberry Pi (локальні системи), Zabbix, Nagios, PRTG Network Monitor та Blynk IoT. Підтримання стабільних параметрів мікроклімату є необхідною умовою безпечної та безперебійної роботи серверного обладнання.

У другому розділі обґрунтовано вибір апаратної платформи Raspberry Pi Pico 2 W та датчика температури і вологості DHT11. Проведено аналіз технічних характеристик обраних компонентів, розроблено схему підключення та реалізовано апаратний вузол системи моніторингу. Визначено, що використання Raspberry Pi Pico 2 W є доцільним завдяки низькій вартості, підтримці Wi-Fi, компактним розмірам та можливості програмування мовою MicroPython.

У третьому розділі реалізовано програмне забезпечення системи моніторингу. Розроблено прошивку мікроконтролера для зчитування параметрів із датчика DHT11 та передачі даних через мережу Wi-Fi. Також створено серверну частину на базі Node.js та Express із підтримкою Telegram-сповіщень і веб-інтерфейсом для відображення поточних показників у режимі реального часу. Проведене тестування підтвердило працездатність та стабільність функціонування системи.

Результати виконаної роботи показали, що розроблена система забезпечує своєчасне виявлення небезпечних відхилень температури та вологості у серверному приміщенні, дозволяє оперативно інформувати користувача про критичні ситуації та може бути використана як доступне рішення для моніторингу серверних кімнат невеликих організацій або домашніх серверних систем.

Перспективами подальшого розвитку системи є використання більш точних датчиків, реалізація збереження статистики у базі даних, додавання підтримки декількох сенсорів та інтеграція із хмарними сервісами моніторингу.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Aosong Electronics Co., Ltd. Digital relative humidity & temperature sensor DHT11: Product Manual. URL: https://www.electronicoscaldas.com/datasheet/DHT11_Aosong.pdf (date of access: 07.01.2026).

2. Aosong Electronics Co., Ltd. Temperature and humidity module: AM2302 Product Manual. URL: https://www.waveshare.com/w/upload/a/ae/AM2302.pdf (date of access: 23.01.2026).

3. Aosong Electronics Co., Ltd. Temperature and Humidity Module DHT11: Product Manual. URL: https://components101.com/sites/default/files /component_datasheet/DHT11-Temperature-Sensor.pdf (date of access: 24.01.2026).

4. Arduino. Arduino Uno Rev3 Datasheet. URL: https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf (date of access: 03.02.2026).

5. ASHRAE TC9.9. Data Center Power Equipment Thermal Guidelines and Best Practices. 5th ed. Atlanta 2021: ASHRAE. 60 p.

6. Blynk Inc. Blynk Documentation. URL: https://docs.blynk.io/en (date of access: 05.02.2026).

7. Cisco Systems. What Is a Data Center? URL: https://www.cisco.com/c/en/us/solutions/data-center-virtualization/what-is-a-data-center.html (date of access: 14.02.2026).

8. Espressif Systems. ESP32 Series Datasheet. Version 5.2. 2025. URL: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf (date of access: 23.02.2026).

9. Express.js. Express 5.x API Reference. URL: https://expressjs.com/en/5x/api/ (date of access: 24.02.2026).

10. Intel. Data center solutions. URL: https://www.intel.com/content/www/us/en/data-center/overview.html (date of access: 26.02.2026).

11. ISO/IEC 27001:2022. Information security management systems – Requirements. Geneva : ISO, 2022. 23 p.

12. MicroPython. MicroPython documentation. URL: https://docs.micropython.org/en/latest/ (date of access: 01.03.2026).

13. Microsoft Azure. What is IoT? URL: https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-iot/ (date of access: 05.03.2026).

14. Mozilla Developer Network. Overview of HTTP. URL: https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview (date of access: 13.03.2026).

15. MQTT.org. MQTT Version 5.0. OASIS Standard. 2019. URL: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html (date of access: 14.03.2026).

16. Nagios Documentation. Nagios Enterprises. URL: https://www.nagios.org/documentation/ (date of access: 16.03.2026).

17. Node.js Foundation. Node.js v26.1.0 documentation. URL: https://nodejs.org/en/docs (date of access: 17.03.2026).

18. Paessler – The Monitoring Experts PRTG manual. URL: https://www.paessler.com/manuals/prtg (date of access: 19.03.2026).

19. Python Software Foundation. Python 3.14.5 documentation. URL: https://docs.python.org/3/ (date of access: 24.03.2026).

20. Random Nerd Tutorials. Raspberry Pi Pico: DHT11/DHT22 Temperature and Humidity Sensor (MicroPython). URL: https://randomnerdtutorials.com/raspberry-pi-pico-dht11-dht22-micropython/ (date of access: 26.03.2026).

21. Raspberry Pi Foundation. Raspberry Pi Pico Documentation. URL: https://www.raspberrypi.com/documentation/microcontrollers/pico-series.html (date of access: 02.04.2026).

22. Raspberry Pi Foundation. Raspberry Pi Pico W Datasheet. 2024. URL: https://datasheets.raspberrypi.com/picow/pico-w-datasheet.pdf (date of access: 05.04.2026).

23. Schneider Electric. Cooling Strategies for IT Wiring Closets and Small Rooms. White Paper 68. Rev. 1. URL: https://download.schneider-electric.com/files?p_Doc_Ref=SPD_NRAN-6NDTJM_EN (date of access: 09.04.2026).

24. Susnjara S., Smalley I. What is a data center? IBM. URL: https://www.ibm.com/think/topics/data-centers (date of access: 13.04.2026).

25. Telegram. Telegram Bot API. URL: https://core.telegram.org/bots/api (date of access: 19.04.2026).

26. Zabbix Documentation. Zabbix LLC. URL: https://www.zabbix.com/documentation/current/en/manual (date of access: 22.04.2026