

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

Факультет комп'ютерних наук та технологій
Кафедра комп'ютерних систем та мереж

Пояснювальна записка

до дипломного проекту (роботи)

магістра

(ступінь вищої освіти (освітній ступінь))

на тему СИСТЕМА МОНІТОРИНГУ ТА УПРАВЛІННЯ РОЗУМНИМИ
ПРИСТРОЯМИ

Виконав: студент 2 курсу, групи КНТ-513м
спеціальності 123 Комп'ютерна інженерія

Освітня програма (спеціалізація)

Комп'ютерні системи та мережі

(код і назва спеціальності)

КУРІЦІН А.Ю.

(прізвище та ініціали)

Керівник ІЛ'ЯШЕНКО М.Б.

(прізвище та ініціали)

Рецензент МАЛІЙ О.Ю.

(прізвище та ініціали)

2024 рік

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування вищого навчального закладу)

Факультет Комп'ютерних наук і технологій

Кафедра «Комп'ютерні системи та мережі»

Ступінь вищої освіти (освітній ступінь) магістерський

Спеціальність 123 Комп'ютерна інженерія
(код і назва)

Освітня програма (спеціалізація) Комп'ютерні системи та мережі
(назва)

ЗАТВЕРДЖУЮ
Зав. кафедри Кудерметов Р.К.

“ ” 2024 року

ЗАВДАННЯ НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

КУРІЦІН Андрій Юрійович

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Система моніторингу та управління розумними пристроями

керівник проекту (роботи) ІЛЬЯШЕНКО Матвій Борисович, к. т. н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від “18” жовтня 2024 року № 149

2. Строк подання студентом проекту (роботи) 10 грудня 2024 року

3. Вихідні дані до проекту (роботи) міні-ПК Raspberry Pi, периферійні пристрої, додатку Domoticz, модуль Wi-Fi, USB-живлення, програма NOOBS

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) _____

1) Аналіз системи мереж інтернету речей

2) Вибір апаратного забезпечення для функціонування системи

3) Налаштування моніторингу та керування системи

4) Програмне забезпечення системи

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-3	ІЛЛЯШЕНКО М.Б., к. т. н., доцент		
Нормоконтроль	ЩЕРБАК Н.В., ст. викл.		

7. Дата видачі завдання 10.09.2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз технологій та протоколів у IoT	21.09.2024 р.	
2	Аналіз типів архітектури у IoT	07.10.2024 р..	
3	Вибір апаратного забезпечення для системи	13.10.2024 р.	
4	Характеристики мінікомп'ютера Raspberry Pi	21.10.2023 р.	
5	Налаштування програмного забезпечення системи	07.11.2024 р.	
6	Підключення апаратного забезпечення у системі	19.11.2024 р.	
7	Архітектура системи та інструменти розробки системи	25.11.2024 р.	
8	Налаштування системи	01.12.2024	
9	Оформлення ПЗ	10.12.2024	

Студент _____ **Андрій КУРІЩИН**
(підпис) (ініціали та прізвище)

Керівник проекту (роботи) _____ **Матвій ІЛЬЯШЕНКО**
(підпис) (ініціали та прізвище)

РЕФЕРАТ

ПЗ: 81 с., 28 рис., 3 табл., 26 джерел.

DOMOTICZ, RASPBERRY PI, ДАТЧИКИ, МЕРЕЖА ІОТ, МІНІ-ПК

Мета і завдання дипломної роботи – вивчення системи управління та моніторингу компонентів Інтернету речей (ІоТ).

Об’єкт дослідження – дистанційне управління та моніторинг елементів Інтернету речей.

Предмет дослідження – інтеграція апаратного та програмного забезпечення, яка забезпечує управління та моніторинг елементів ІоТ.

Методи дослідження, технічні та програмні засоби – системний аналіз і синтез, використання програмного додатку Domoticz, бездротові та дротові технології передачі даних в мережі ІоТ.

Основні конструктивні та технологічні характеристики об’єкту – система управління та моніторингу з використанням програмного додатку Domoticz, мови програмування Java та міні-ПК Raspberry Pi.

Практична значущість дослідження – аналіз протоколів, засобів та концепцій дистанційного управління й моніторингу ІоТ-компонентів для визначення оптимального мінімального апаратного та програмного забезпечення для роботи таких систем.

Рекомендації щодо застосування результатів дипломної роботи – розроблений об’єкт можна застосовувати у промислових і побутових сферах для управління та моніторингу малих або регіональних виробничих процесів.

Прогноз розвитку об’єкту дослідження – постійне вдосконалення технологій захисту даних у мережах Інтернету речей.

ABSTRACT

Explanatory note to the master's work: 81 p.; 28 figures; 3 table, 26 sources.

DOMOTICZ, RASPBERRY PI, SENSORS, IOT NETWORK, MINI PC

The aim and objectives of the thesis are to study the system of control and monitoring of Internet of Things (IoT) components.

The object of research is remote control and monitoring of the Internet of Things.

The subject of research is the integration of hardware and software that provides management and monitoring of IoT elements.

Research methods, hardware and software - system analysis and synthesis, use of the Domoticz software application, wireless and wired data transmission technologies in the IoT network.

The main design and technological characteristics of the object are the control and monitoring system using the Domoticz software application, the Java programming language and the Raspberry Pi mini-PC.

The practical significance of the research is the analysis of protocols, tools and concepts for remote control and monitoring of IoT components to determine the optimal minimum hardware and software for the operation of such systems.

Recommendations for the application of the results of the thesis - the developed object can be used in industrial and domestic areas to control and monitor small or regional production processes.

Forecast of the object of study - continuous improvement of data protection technologies in the Internet of Things networks.

ЗМІСТ

Вступ	9
1 Аналіз системи мереж інтернету речей	15
1.1 Аналіз технологій та протоколів у IoT	17
1.1.1 Мережі з низьким енергоспоживанням і малим діапазоном	17
1.1.2 Мережі з низьким енергоспоживанням і широкою зоною охоплення (LPWAN)	18
1.1.3 Протоколи IoT	19
1.2 Аналіз пристроїв для використання у мережі IoT	23
1.3 Аналіз типів архітектури у IoT	24
1.4 Аналіз можливих задач для моніторингу у IoT	29
1.5 Висновки до розділу	34
2 Вибір апаратного забезпечення для функціонування системи	34
2.1 Вибір апаратного забезпечення для системи	36
2.3 Характеристики мінікомп'ютера Raspberry Pi	41
2.4 Занальна концепція системи	45
2.5 Висновки до розділу	47
3 Налаштування моніторингу та керування системи	48
3.1 Налаштування програмного забезпечення системи	48
3.2 Налаштування параметрів мережі	51
3.3 Нлаштування бездротового доступу	52
3.4 Налаштування віддаленого доступу до Raspberry Pi	53
3.5 Налаштування конфігурації Domoticz	56
3.6 Підключення апаратного забезпечення у системі	58
3.7 Налаштування API Dpmpticz	61
3.8 Висновки до розділу	63
4 Програмне забезпечення системи	64

4.1 Архітектура системи та інструменти розробки системи	64
4.2 Реалізація збору даних	67
4.3 Налаштування системи	69
4.4 Результати роботи.....	70
4.5 Безпека системи	74
4.2 Висновки до розділу.....	76
Висновки.....	78
Перелік джерел посилання	79

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧКИ

IoT	- Інтернет речей
WSNs	- Бездротові сенсорні мережі
RFID	- Радіочастотна ідентифікація
ITU	- Міжнародний союз електрозв'язку
LPWAN)	- Мережі з низьким енергоспоживанням і широкою зоною охоплення
AMQP	- Розширений протокол управління чергою повідомлень
CoAP	- Обмежений протокол додатків
DDS	- Служба розподілу даних
MQTT	- Протокол для передачі повідомлень
BLE	- Bluetooth Low Energy; Технологія для зменшення енергоспоживання
LTE	- Long-Term Evolution; Стандарт бездротового широкосмугового зв'язку для мобільних пристроїв і терміналів
NFC	- Near Field Communication; Протоколів для обміну даними через електромагнітні поля
MCU	- Мікроконтролер
ЦП	- Центральний процесор
ОЗУ	- Оперативна пам'ять
ПЗУ	- Постійна пам'ять
MPU)	- Мікропроцесор
GPIO	- General Purpose Input Output; Інтерфейс введення/виведення загального призначення
SSH	- Secure Shell; Протокол для встановлення мережевого з'єднання

ВСТУП

Інтернет речей (IoT) активно поширюється завдяки масовій появі пристроїв, оснащених електронними компонентами, програмним забезпеченням та можливістю комунікації. Це можуть бути як смартфони, камери на нафтових установках, так і оптичні датчики на сталепрокатних лініях, які здатні збирати та передавати дані.

У майбутньому, технології та системи на базі Інтернету речей ймовірно будуть широко застосовуватися практично на всіх виробничих підприємствах. Актуальність IoT обумовлена зростанням кількості пристроїв, здатних обмінюватися інформацією без участі людини, що однаково важливо як у побутових, так і промислових умовах.

Розвиток інформаційних технологій суттєво змінює процеси публічного управління, автоматизуючи їх, покращуючи ефективність прийняття рішень і роблячи владу більш прозорою. Україна була піонером у впровадженні автоматизації управлінських процесів ще в 80-х роках XX століття. Український вчений В.М. Глушков запропонував концепцію «Загальнодержавної автоматизованої системи збору та обробки інформації» (ЗДАС) для управління народним господарством. Також в Україні прийнято низку законодавчих актів, які регулюють розвиток та впровадження IoT.

Обрання теми пояснюється стрімким розвитком технологій. Сьогодні багатомільярдні корпорації об'єднуються для створення стандартів та протоколів IoT. Підприємства переходять від ручної праці до автоматизації, а побутові пристрої допомагають полегшити життя користувачів навіть без їх втручання. Індустрія IoT орієнтована на зменшення людської участі в процесах, що дозволяє мінімізувати ризики, знижувати витрати та підвищувати ефективність. Разом із цим необхідні інструменти контролю, адже електроніка може мати апаратні чи програмні помилки, які потребують швидкого вирішення. Для зручності

користувачам пропонуються інтерфейси управління, наприклад, через додатки на смартфоні, але для цього пристрої мають бути об'єднані в єдину мережу.

Інтернет речей – це новий етап цифровізації суспільства, де об'єкти та люди взаємодіють через комунікаційні мережі. IoT дозволяє відстежувати стан пристроїв та навколишнього середовища. Розробники IoT-рішень постійно працюють над удосконаленням технологій. Інтернет речей відкриває нові можливості та створює конкурентні переваги як на традиційних, так і на нових ринках.

У найближчі роки очікується значне зростання обсягу даних, які генеруватимуть підприємства за допомогою IoT. Це позитивно вплине на ефективність бізнесу та його конкурентоспроможність. На відміну від звичайних одноадресних комунікацій, IoT передбачає масову обробку та передачу даних, що вимагає впровадження нових підходів до роботи з інформацією. Інтернет речей вважають третьою хвилею інформаційної революції після поширення персональних комп'ютерів і смартфонів. Наступний етап розвитку, четверта хвиля, пов'язаний із впровадженням штучного інтелекту, який посилить інтеграцію IoT у повсякденне життя.

IoT змінює підходи до управління бізнес-процесами та розвитку ринку. Наприклад, виробники авіадвигунів та будівельної техніки вже можуть в реальному часі відстежувати роботу своїх пристроїв. Комунальні підприємства можуть отримувати дані від лічильників споживачів для оптимізації послуг. За прогнозами IDC, до 2020 року ринок IoT досяг \$7,1 трлн.

Завдяки підключенню десятків мільярдів пристроїв до мережі, IoT генерує величезний обсяг даних. Це зумовлює відмову від традиційних підходів до моніторингу мережевого трафіку, оскільки частота оновлення інформації значно зростає. Технології IoT вимагають платформ, здатних обробляти дані з високою швидкістю, забезпечуючи оперативний моніторинг і контроль.

Інтернет речей трансформує промисловий та споживчий ринки. Новітні технології, такі як автономні транспортні засоби, блокчейн, розширена реальність, цифрові помічники та штучний інтелект, значно прискорюють розвиток IoT-

додатків. Однак разом із перевагами зростає загроза неправомірного використання даних, кібератак та втручання у приватне життя.

Моніторинг і контроль IoT-пристроїв є критично важливими завданнями. Сучасні рішення для віддаленого керування дозволяють автоматизувати процеси у розумних будинках, здійснювати діагностику та модернізацію пристроїв. IoT-системи забезпечують зручність і безпеку, дозволяючи віддалено контролювати стан житлових та промислових об'єктів, а також оперативно реагувати на надзвичайні ситуації, такі як пожежа чи витік газу.

Пропоноване рішення. У роботі описано систему моніторингу та управління IoT-пристроями на базі міні-ПК Raspberry Pi. Запропонована система дозволяє користувачам взаємодіяти з підключеними датчиками через інтернет. Дані зчитуються веб-сервером у реальному часі та відображаються у вигляді зручних інтерфейсів. Система забезпечує високу гнучкість, підтримуючи широкий спектр вимірювальних пристроїв. Серед її переваг – енергоефективність, портативність, низька вартість та висока продуктивність.

Актуальність. IoT залишається перспективною галуззю, яка стимулює автоматизацію в різних сферах. Інноваційні рішення дозволяють підприємствам і споживачам знижувати витрати, підвищувати продуктивність і забезпечувати комфорт у повсякденному житті.

Прогноз. У майбутньому IoT-системи розвиватимуться у напрямі підвищення безпеки даних та інтеграції з новітніми технологіями. Це відкриває нові можливості для досліджень і впровадження більш ефективних рішень.

Основною метою даної роботи є дослідження технологій управління та моніторингу компонентів Інтернету речей (IoT). Завданням є вивчення доступних рішень для розгортання IoT, а також визначення методів управління та моніторингу елементів системи.

Об'єктом дослідження виступає віддалене управління та моніторинг компонентів Інтернету речей.

Предметом дослідження є інтеграція апаратного забезпечення з програмним, за допомогою якого реалізується управління та моніторинг IoT-

систем.

Методи дослідження. Для вирішення поставлених завдань застосовано системний аналіз, синтез та аналіз.

Наукова новизна. У роботі проведено порівняльний аналіз протоколів передачі даних, а також засобів моніторингу та управління компонентами IoT. Це дозволило обґрунтувати найбільш ефективні рішення для їх подальшого впровадження в мережах Інтернету речей.

Практичне значення отриманих результатів. Результати дослідження надають практичну інформацію про оптимальні програмні та апаратні засоби, необхідні для побудови систем моніторингу та управління компонентами IoT. Також розглянуто способи реалізації таких систем із мінімальними фінансовими витратами.

Актуальність теми. Швидкий розвиток технологій автоматизації та віддаленого керування обумовлює актуальність дослідження тематики IoT. У світі спостерігається тренд на автоматизацію побутових і виробничих процесів, що значно полегшує життя людей і підвищує ефективність підприємств. При цьому зростає потреба у доступних і функціональних рішеннях для управління компонентами Інтернету речей, як для підприємств, так і для звичайних споживачів.

Постійний розвиток IoT-технологій та зростання попиту на автоматизовані рішення стали основними причинами вибору цієї теми. Це дозволяє дослідити, як створюються IoT-системи, які протоколи та технології у них використовуються, а також визначити мінімально необхідне програмне й апаратне забезпечення для ефективного функціонування.

Аналіз і результати. У роботі проведено аналіз існуючих IoT-пристроїв, типів мереж, технологій та протоколів, які забезпечують їх роботу. У результаті синтезу отриманих даних визначено оптимальні засоби для безперебійного функціонування IoT-систем.

Прогнозні припущення. З урахуванням швидкого розвитку технологій, предмет і об'єкт дослідження мають великий потенціал для подальшого

вдосконалення. Нові стандарти зв'язку, швидші протоколи передачі даних та оновлення апаратного забезпечення створюють потребу у повторному дослідженні, адаптованому до нових вимог. Крім того, важливим аспектом є розвиток системи захисту даних, оскільки безпека й конфіденційність інформації відіграють ключову роль у прийнятті рішень користувачами.

Інтернет речей надає широкий спектр переваг, включаючи зниження експлуатаційних витрат, підвищення зручності та безпеки, а також можливість моніторингу і керування як у побутовій, так і у виробничій сферах. Завдяки IoT, користувачі отримують можливість контролювати розумні будинки та промислові об'єкти у режимі реального часу.

Дана робота підкреслює важливість інтеграції програмного і апаратного забезпечення для управління IoT-системами. Подальший розвиток і вдосконалення цих систем дозволить забезпечити вищу ефективність, надійність і зручність для користувачів.

1 АНАЛІЗ СИСТЕМИ МЕРЕЖ ІНТЕРНЕТУ РЕЧЕЙ

Інтернет речей (IoT) – це сучасна концепція, у якій Інтернет трансформується від мережі, що об'єднує людей і комп'ютери, до екосистеми, яка з'єднує "розумні" об'єкти або речі. Завдяки постійному розвитку технологій, IoT стає основою глобальної обчислювальної мережі, де всі речі будуть взаємопов'язані через Інтернет (рис. 1.1). Ця концепція активно еволюціонує та залишається однією з найактуальніших тем для досліджень сьогодення. Інтернет у звичному розумінні перетворюється на вдосконалену й інтегровану версію. Щодня зростає кількість пристроїв, які використовують Інтернет-послуги, а їхнє підключення за допомогою дротових або бездротових технологій створює потужний потік інформації, доступний у будь-який момент. Концепція IoT також розширює можливості завдяки взаємодії між інтелектуальними пристроями та передовими технологіями. Водночас технології, що формують основу Інтернету речей, не є новими.

Моніторинг і управління пристроями – один із ключових напрямів IoT, який необхідно активно використовувати в режимі реального часу для забезпечення безпеки, зручності й комфорту людей. Розвиток Інтернет-технологій та бездротових сенсорних мереж сприяв виникненню нової тенденції в епоху цифровізації. Завдяки значному зростанню кількості користувачів Інтернету та модернізації його технологій стало можливим створення мережі повсякденних об'єктів. Споживачі очікують високої ефективності від систем, які підтримують Інтернет, і ці очікування виправдовуються завдяки низці переваг.

Серед таких переваг – зниження витрат на експлуатацію й обслуговування завдяки віддаленому моніторингу, діагностиці, налаштуванню та оновленню мікропрограмного забезпечення. IoT також забезпечує зручність і безпеку завдяки можливості віддалено контролювати стан "розумного будинку" та керувати Інтернет-пристроями навіть за відсутності власника. Це включає дистанційний моніторинг житлових і промислових об'єктів, а також автоматичне повідомлення

аварійних служб у разі пожежі, крадіжки чи витоку газу або рідини [1].

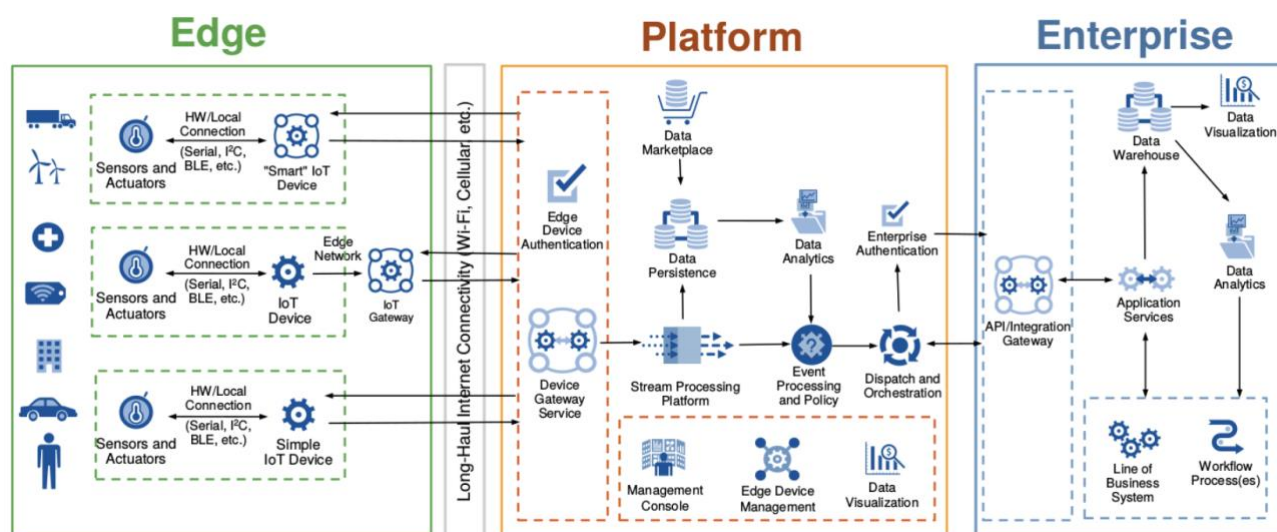


Рисунок 1.1 - Приклад загальної мережі IoT

Інтернет речей (IoT) є концепцією об'єднання інформації з різних джерел у межах віртуальних платформ або існуючої інтернет-інфраструктури. Ідея IoT зародилася ще в 1982 році, коли автомат з газованими напоями був підключений до Інтернету й міг повідомляти про кількість напоїв та їх температуру. У 1991 році Марк Вайзер вперше сформулював сучасну оцінку Інтернету речей. У 1999 році Білл Джой представив ідею зв'язку між пристроями у своїй класифікації Інтернету, а Кевін Ештон у тому ж році вперше ввів термін "Інтернет речей" для опису взаємопов'язаних пристроїв.

Основна ідея IoT полягає в автономному обміні корисною інформацією між унікально ідентифікованими пристроями реального світу. Ці пристрої, оснащені новітніми технологіями, такими як радіочастотна ідентифікація (RFID) і бездротові сенсорні мережі (WSNs), здатні приймати самостійні рішення залежно від автоматизованих дій, які необхідно виконати.

У 2005 році Міжнародний союз електрозв'язку (ITU) оголосив початок ери всепроникних мереж, головною характеристикою яких є інтеграція різних мереж у єдину систему. Основна концепція Інтернету речей передбачає створення середовища, у якому об'єкти можуть виконувати керовані дії, а інформація про

них обробляється для виконання необхідних завдань шляхом навчання пристроїв.

Практичне втілення IoT можна побачити на прикладі Twine – компактного й енергоефективного апаратного забезпечення, яке працює разом із мережевим програмним забезпеченням у реальному часі, втілюючи концепцію IoT у життя. Однак слід зазначити, що різні люди й організації мають власні унікальні підходи до визначення й реалізації Інтернету речей [2].

1.1 Аналіз технологій та протоколів у IoT

У контексті Інтернету речей «речі» здебільшого представлені вбудованими пристроями, які мають такі характеристики, як вузька смуга пропускання, періодичний збір даних і невеликий обсяг переданої інформації. Ці пристрої обмінюються даними між собою та надають доступ до інформації через спеціальні інтерфейси.

Деякі вбудовані IoT-пристрої, такі як охоронні відеокамери високої роздільної здатності, VoIP-відеотелефони та подібні пристрої, потребують широкосмугового зв'язку для передачі потокових даних. Проте більшість пристроїв IoT надсилають дані лише періодично, використовуючи невеликі пакети інформації [3].

Для аналізу мереж, які використовуються у побудові IoT, необхідно розглянути типи мереж, що забезпечують функціонування цих систем. Вони поділяються на два основні типи:

- мережі з низьким енергоспоживанням і малим діапазоном;
- мережі з низьким енергоспоживанням і широкою зоною охоплення (LPWAN).

LPWAN - ці мережі ідеально підходять для невеликих середовищ, таких як будинки чи офіси. Їх робота може забезпечуватись невеликими акумуляторами, а в деяких випадках – без акумуляторів. Вони є економічними в експлуатації.

Нижче наведено найпоширеніші приклади [4]:

Bluetooth - забезпечує високошвидкісну передачу даних і передає сигнали голосу та інформації на відстань до 10 метрів;

- Wi-Fi/802.11 - популярний стандарт для використання вдома і на роботі завдяки низькій вартості, його недоліками є обмежений діапазон і високе споживання енергії;

- Z-Wave - використовується для обміну даними в домашніх пристроях через низькоенергетичні радіохвилі, підтримує взаємодію між системами автоматизації на рівні додатків;

- Zigbee - поширений варіант для домашніх систем автоматизації та медичних пристроїв, є оптимальним для невеликих мереж із низьким споживанням енергії та низькою пропускну здатністю.

1.1.2 Мережі з низьким енергоспоживанням і широкою зоною охоплення (LPWAN)

Ці мережі забезпечують зв'язок на відстані до 500 метрів і використовуються для більшості пристроїв IoT завдяки мінімальному енергоспоживанню. Приклади:

- LoRaWAN - підтримує двонапрямний зв'язок для мобільних пристроїв, що працюють від акумуляторів, забезпечуючи широкий діапазон охоплення;

- 4G LTE для IoT - забезпечує високу потужність і мінімальну затримку, що є чудовим вибором для додатків IoT із потребою в реальному часі;

- 5G для IoT - швидша передача даних і підтримку більшої кількості пристроїв на одиницю площі;

- Cat-0 - економічна LTE-технологія, яка стане базою для Cat-M і замінить 2G;

- Cat-1 - LTE-стандарт, що замінить 3G, ідеально підходить для додатків, яким потрібен голосовий або браузерний інтерфейс;

- LTE Cat-M1 - сумісна з LTE-мережами, дозволяє знизити витрати на друге покоління LTE-мікросхем, створених спеціально для IoT;

- вузькосмугове підключення (NB-IoT - радіотехнологія на основі LTE, яка забезпечує тривалу роботу від акумуляторів і покриття в приміщеннях;
- NB-IoT/Cat-M2 - використовує технологію DSSS для передачі даних без шлюзів, що знижує експлуатаційні витрати;
- Sigfox - глобальний постачальник бездротових мереж для пристроїв IoT із низьким енергоспоживанням, що постійно генерують дані [5].

Ці мережі забезпечують функціональність IoT у різних умовах, залежно від потреб енергоспоживання, діапазону дії та сценаріїв застосування.

Для отримання даних з веб-сайту користувачі використовують веб-браузер. Вони вводять у рядок пошуку URL-адресу цікавого ресурсу і натискають кнопку підтвердження. У цей момент відбувається наступний процес. Клієнт, яким є веб-браузер, надсилає HTTP-запит за вказаною адресою з певними параметрами, отримує відповідь, здійснює її обробку та форматує у зручний для користувача вигляд .

1.1.3 Протоколи IoT

Пристрої IoT здійснюють обмін даними за допомогою спеціалізованих протоколів. Протокол IP визначає набір правил, які регулюють передачу інформації через Інтернет. Протоколи IoT забезпечують коректне зчитування і інтерпретацію інформації між різними пристроями чи сенсорами. Зважаючи на різноманітність IoT-пристроїв, правильний вибір протоколу для кожного конкретного випадку є вкрай важливим.

Тип обраного протоколу залежить від рівня архітектури IoT-системи, на якому відбувається передача даних. У моделі OSI представлені рівні, що визначають процес надсилання та отримання інформації. Протоколи IoT відповідають за взаємодію між пристроями, пристроями та шлюзами, шлюзами та центрами обробки даних, шлюзами та хмарними платформами, а також за обмін інформацією між центрами обробки даних.

Рівень додатків - забезпечує обмін даними між користувачем і пристроєм:

- розширений протокол управління чергою повідомлень (AMQP): забезпечує обмін повідомленнями на рівні апаратного забезпечення, дозволяючи

взаємодію різних систем і додатків, цей протокол створює стандартизовану систему обміну даними у великих масштабах;

- обмежений протокол додатків (CoAP): призначений для пристроїв із низьким енергоспоживанням та обмеженими ресурсами, використовує протокол UDP для передачі даних та забезпечує ефективний обмін інформацією між комп'ютерами;

- служба розподілу даних (DDS): універсальний протокол, який підтримує зв'язок від малих пристроїв до високопродуктивних мереж. DDS спрощує процес розгортання, підвищує надійність і знижує складність;

- MQTT - легкий протокол для передачі повідомлень, оптимізований для низької пропускної здатності та енергоефективності, використовується в пристроях з обмеженим ресурсом енергії і віддалених місцях.

Рівень транспортування - відповідає за безпечну та ефективну передачу даних:

- TCP - основний протокол для більшості інтернет-з'єднань, що забезпечує розподіл великих обсягів даних на пакети та їх повторне надсилання у разі втрат;

- UDP - протокол, що забезпечує швидку передачу даних між процесами, особливо корисний для застосувань, де важлива швидкість, а не гарантія доставки.

Мережевий рівень - допомагає IoT-пристроєм взаємодіяти з маршрутизаторами:

- 6LoWPAN - легка версія протоколу IPv6, оптимізована для пристроїв із низьким енергоспоживанням, що зменшує затримки передачі;

- IPv6 - сучасна версія IP-протоколу, яка забезпечує маршрутизацію трафіку, а також ідентифікацію та виявлення пристроїв у мережі.

Канальний рівень - відповідає за передачу даних у межах системної архітектури та виправлення помилок, які виникають на фізичному рівні:

- IEEE 802.15.4 - стандарт бездротового підключення з низьким енергоспоживанням, що використовується для створення мереж із Zigbee та 6LoWPAN;

- LPWAN - забезпечують зв'язок на великих відстанях із мінімальним енергоспоживанням, що робить їх ідеальними для IoT-застосувань.

Технології кожного рівня забезпечують ефективний зв'язок та обмін даними між пристроями, відповідно до їх призначення та обмежень [6].

На таблиці 1.1 наведена порівняльна характеристика основних мережевих технологій.

Таблиця 1.1 Порівняльна характеристика основних мережевих технологій

Параметр	Zigbee	Wi-Fi	Bluetooth
1	2	3	4
Максимальна дальність	10-100 метрів (в залежності від умов)	50-150 метрів (в залежності від стандарту)	10-100 метрів (в залежності від версії)
Швидкість передачі даних	250 кбіт/с	11-5000 Мбіт/с (в залежності від стандарту)	1-3 Мбіт/с (для класичного Bluetooth)
Споживана потужність	Дуже низька (енергозбереження)	Висока (потрібно більше енергії для зв'язку)	Середня (енергозберігаючий режим)
Пропускна здатність	Низька (для IoT пристроїв)	Висока (для потокових даних і великих файлів)	Середня (для обміну невеликими файлами та даними)
Тип мережі	Мережа зірка, деревоподібна, решітка	Мережа типу клієнт-сервер, інфраструктурна	Мережа з'єднань «точка-точка», «точка-багатоточкове з'єднання»
Діапазон частот	2.4 ГГц (ISM)	2.4 ГГц, 5 ГГц (ISM)	2.4 ГГц (ISM)
Підтримка великої кількості пристроїв	Так, до 65 000 пристроїв	Обмежена кількість підключених пристроїв	Обмежена кількість пристроїв (до 7 пристроїв в мережі)
Тип використання	Призначена для IoT та домашньої автоматизації	Для потокового відео, веб-браузингу, ігор	Для передачі невеликих даних

Продовження таблиці 1.1

1	2	3	4
Основні переваги	Енергозбереження, великі мережі, низька швидкість для IoT	Висока швидкість передачі, широко використовується	Легке налаштування, низьке енергоспоживання в нових версіях (Bluetooth Low Energy)
Тип безпеки	AES-128 bit шифрування	WPA/WPA2/WPA3 (шифрування)	AES-128 bit шифрування

На фізичному рівні встановлюється канал зв'язку, який забезпечує з'єднання пристроїв у визначеному середовищі [7]:

- Bluetooth Low Energy (BLE) – технологія, яка значно зменшує енергоспоживання та витрати. Вона підтримує той самий діапазон підключення, що й традиційний Bluetooth. BLE працює безпосередньо з мобільними операційними системами. Завдяки своїй економічності та здатності працювати від батареї протягом тривалого часу, ця технологія швидко стає популярною в сфері побутової електроніки;

- Ethernet – економічне дротове підключення, яке забезпечує високу швидкість передачі даних і низьку затримку порівняно з бездротовими протоколами. Це дозволяє передавати дані в реальному часі, наприклад, відеопотік, і швидко реагувати на зміни, що відбуваються поза системою, інформуючи користувача;

- LTE (Long-Term Evolution) – стандарт бездротового широкосмугового зв'язку для мобільних пристроїв і терміналів. LTE підвищує швидкість і потужність бездротових мереж, а також підтримує потоки даних для багатоканальних та широкомовних передач;

- NFC (Near Field Communication) – набір протоколів для обміну даними через електромагнітні поля. Це дозволяє пристроям обмінюватися інформацією на відстані до чотирьох сантиметрів. Пристрої, що підтримують NFC, часто використовуються для смарт-карт, безконтактних мобільних платежів і бронювання квитків;

- RFID (Radio Frequency Identification) – технологія, що використовує електромагнітні поля для відстеження електронних міток, інформація про які не може бути отримана іншими способами. Сумісне обладнання забезпечує живлення та взаємодіє з цими мітками, зчитуючи дані для ідентифікації та перевірки автентичності;

- Wi-Fi / 802.11 – стандарт для бездротових мереж у домашніх і робочих умовах. Ця технологія є економічною, але її застосування обмежене через невеликий радіус дії та постійну потребу в енергії.

1.2 Аналіз пристроїв для використання у мережі IoT

Розглянемо, які пристрої використовуються для побудови мережі Інтернету речей:

- вбудовані системи – ці системи включають як апаратне, так і програмне забезпечення, і відповідають за управління функціями, які є частинами більшої системи. Вони базуються на мікропроцесорах або мікроконтролерах і призначені для виконання специфічних завдань у межах системи;

- інтелектуальні системи – ці пристрої здатні виконувати обчислення та часто мають вбудовані мікроконтролери, що дозволяє їм обробляти дані та взаємодіяти з іншими пристроями;

- мікроконтролер (MCU) – це маленькі комп'ютери, що вбудовуються в мікросхеми і оснащені центральним процесором (ЦП), оперативною пам'яттю (ОЗУ) та постійною пам'яттю (ПЗУ). Мікроконтролери призначені для виконання простих завдань, проте їх потужність обмежена порівняно з мікропроцесорами;

- мікропроцесор (MPU) – ці пристрої виконують функції центрального процесора і можуть бути втілені в одній або кількох інтегральних мікросхемах. Мікропроцесори зазвичай потребують додаткових периферійних пристроїв для виконання своїх функцій, але їх витрати на обробку даних значно менші, оскільки

вони включають лише вартість центрального процесора;

- пристрої без обчислювальних функцій – такі пристрої не виконують обчислення, а лише здійснюють підключення і передають дані. Вони зазвичай не мають обчислювальної потужності;

- перетворювачі – фізичні пристрої, які здійснюють перетворення одного виду енергії в інший. У контексті Інтернету речей до них відносяться датчики та виконавчі пристрої, що фіксують зміни в оточуючому середовищі і передають відповідні дані.

1.3 Аналіз типів архітектури у IoT

Для правильного вибору необхідного обладнання та розуміння роботи системи IoT важливо провести аналіз її архітектури[8] .

При обговоренні IoT часто наголошується на його величезному потенціалі. Звістки про те, що IoT може значно полегшити наше життя, поширюються, але для багатьох може здаватися, що ці оптимістичні прогнози не будуть реалізовані так швидко, як це передбачається. Насправді, зміни відбуваються, хоча і не так стрімко, а швидше поступово. Одна з головних причин цього полягає в тому, що різноманітність систем IoT стримує прогрес і часто ускладнює досягнення поставлених цілей.

Однією з основних проблем, з якою стикається IoT (окрім безпеки), є фрагментація, що виникає через різноманітність пристроїв, які повинні бути з'єднані. Для того, щоб реалізувати будь-яку систему IoT, необхідно об'єднати різні ресурси, обладнання, програмне забезпечення та системи в єдину структуру, створюючи інтегроване, надійне та економічно ефективне рішення. Простими словами, кожному розгортанню IoT потрібна сильна архітектура, щоб забезпечити виконання поставлених завдань. При цьому ефективність та здатність системи відповідати вимогам значною мірою залежить від якості розробленої

інфраструктури.

Хоча кожна система IoT має свої особливості, основні принципи архітектури та загальний потік обробки даних залишаються схожими для всіх. Архітектура IoT зазвичай складається з кількох основних компонентів.

По-перше, це самі речі — об'єкти, підключені до Інтернету, які за допомогою вбудованих датчиків і виконавчих механізмів можуть відчувати навколишнє середовище, збирати інформацію та передавати її далі, до шлюзів IoT.

Другий етап охоплює системи збору даних IoT та шлюзи, які отримують великі обсяги необроблених даних, конвертують їх у цифрові потоки, фільтрують і попередньо обробляють, щоб підготувати їх для подальшого аналізу.

Третій шар - це крайові пристрої, що займаються більш глибокою обробкою та посиленням аналізом даних. На цьому рівні можуть застосовуватись такі технології, як візуалізація даних та машинне навчання, для покращення результатів обробки.

Останній етап включає центри обробки даних, які можуть бути як хмарними, так і локально розташованими. Тут дані зберігаються, управляються і аналізуються для отримання цінної практичної інформації. Це чотири основні шари архітектури IoT, детально представлені на рис. 1.2 [9].

Ще одним важливим елементом цієї архітектури є виконавчі механізми, які тісно взаємодіють з датчиками і здатні перетворювати зібрані дані, що генеруються розумними об'єктами, у фізичні дії. Наприклад, у розумній системі поливу, де встановлені необхідні датчики, система в режимі реального часу аналізує сигнали від датчиків і дає команду приводу відкрити водяні клапани в тих місцях, де вологість ґрунту нижча за задане значення. Клапани залишаються відкритими, поки датчики не зафіксують відновлення нормального рівня вологості. Весь цей процес відбувається автоматично, без втручання людини.

Six principles of IoT Cyber Security across the stack

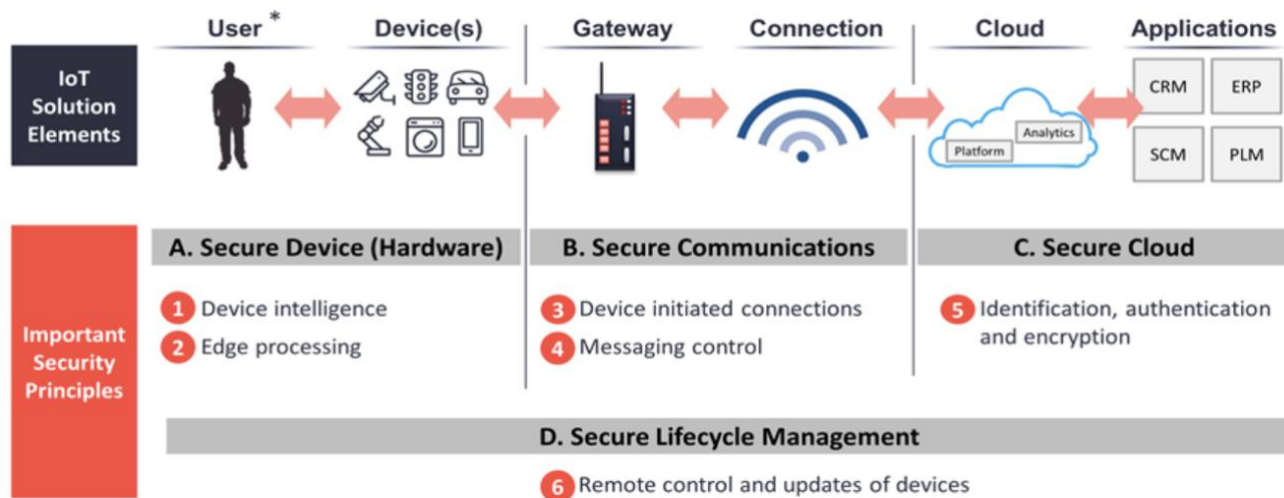


Рисунок 1.2 - Основні складові архітектури IoT

Ще однією важливою характеристикою є те, що підключені об'єкти повинні мати можливість не лише двосторонньо взаємодіяти з відповідними шлюзами або системами збору даних, але й взаємодіяти один з одним. Це дає змогу обмінюватися інформацією та співпрацювати в реальному часі, що необхідно для ефективного функціонування всієї системи (рис.1.3). Це стає особливо важливим у випадках, коли пристрої мають обмежені ресурси або працюють від акумуляторів. Такий зв'язок потребує великих обчислювальних потужностей, а також значного споживання енергії та пропускну здатності, що робить завдання складним.

Тому для досягнення ефективного управління пристроями необхідна надійна архітектура, яка забезпечить використання спеціальних, безпечних і легких протоколів зв'язку, таких як Lightweight M2M. Цей протокол став провідним стандартом для управління малопотужними пристроями, що є характерним для багатьох сценаріїв використання IoT.

Попри те, що цей рівень все ще функціонує в безпосередній близькості від датчиків та виконавчих механізмів на певних пристроях, важливо описати його як окремий етап архітектури IoT, оскільки він має вирішальне значення для процесів

збору даних, фільтрації та передачі до крайової інфраструктури та хмарних платформ

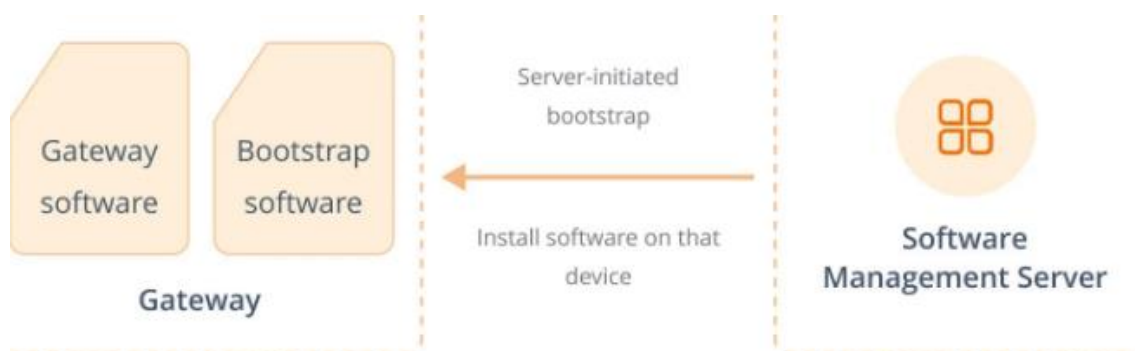


Рисунок 1.3 – Програмні складові для шлюзів

З огляду на величезний обсяг введення та виведення даних, який може створити розгортання мільйонних пристроїв, можливості збору, відбору та транспортування даних повинні бути в центрі уваги. Як посередники між пов'язаними речами та хмарою та аналітикою, шлюзи та системи збору даних забезпечують необхідну точку зв'язку, яка пов'язує решту шарів (рис. 1.4).

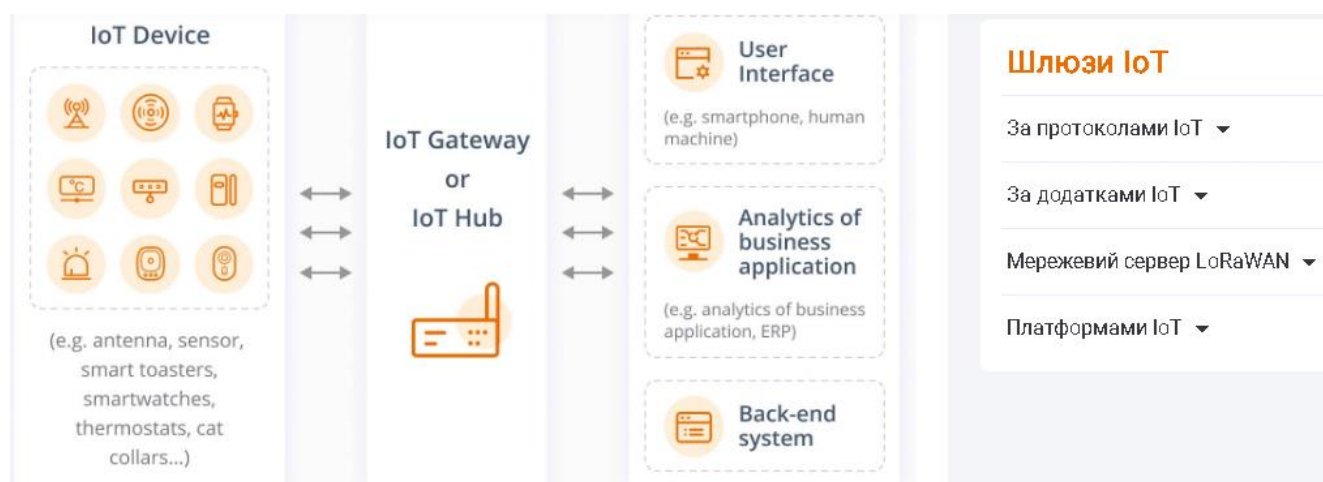


Рисунок 1.4 – Компоненти шлюзи IoT

Шлюзи, розташовані на межі між світами операційні технології та інформаційні технології, сприяють ефективному зв'язку між датчиками та іншими компонентами системи, перетворюючи дані, що надходять від датчиків, у

формати, які легко передаються і використовуються в інших частинах системи. Окрім того, вони здатні контролювати, фільтрувати та вибирати дані, щоб мінімізувати обсяг інформації, яку необхідно передавати в хмару. Це дозволяє знизити витрати на передачу даних та час відгуку системи.

Таким чином, шлюзи виконують роль локальних пристроїв попередньої обробки даних, стискаючи їх у корисні пакети, готові до подальшої обробки та аналізу.

Крім того, шлюзи виконують важливу роль у забезпеченні безпеки. Оскільки вони відповідають за керування інформаційними потоками в обох напрямках, за допомогою ефективних методів шифрування та захисту шлюзи можуть запобігти витоку даних у хмару, а також зменшити ризик атак на пристрої IoT ззовні [10].

Якщо датчики порівнювати з нейронами, а шлюз із основою IoT, то хмара є мозком в тілі Інтернету речей. На відміну від крайових рішень, хмарні системи або центри обробки даних призначені для зберігання, обробки та аналізу великих обсягів даних, забезпечуючи глибше розуміння завдяки потужним механізмам аналізу даних та машинного навчання, які не можуть бути підтримані крайовими системами (рис. 1.5).

За останні кілька років хмарні обчислення стали важливим чинником у покращенні виробничих показників, зменшенні незапланованих простоїв та енергоспоживання, що приносить численні переваги для бізнесу, особливо в промисловій архітектурі IoT.

Крім того, коли хмара оснащена відповідними рішеннями для користувацьких програм, вона здатна надавати бізнес-аналітику та інтерфейси для візуалізації, що дозволяє користувачам взаємодіяти з системою, контролювати її роботу та приймати обґрунтовані рішення на основі звітів, інформаційних панелей і даних у режимі реального часу.

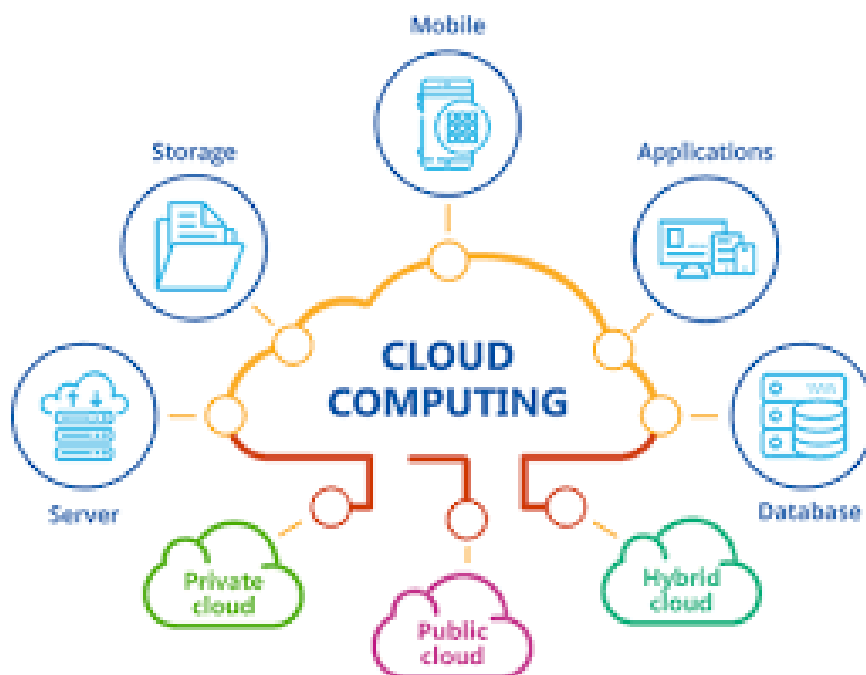


Рисунок 1.5 – Хмарна система обробки даних

1.4 Аналіз можливих задач для моніторингу у IoT

Моніторинг починається зі збору даних шляхом контролю системи, з якою взаємодіє людина. В деяких випадках системою для моніторингу може бути не тільки сам пристрій, а й зовнішнє середовище чи процес, пов'язані з пристроєм. В інших випадках може бути важливим моніторинг працездатності самих пристроїв окремо.

Розглянемо задачу моніторингу людини, наприклад, велосипедиста, який рухається дорогою. У такій системі можна стежити за різними параметрами. Одні з них можуть бути внутрішніми, наприклад, частота серцевих скорочень або рівень пітливості велосипедиста. Інші параметри можуть бути зовнішніми, наприклад, нахил дороги або зовнішня температура та вологість. Ці внутрішні і зовнішні показники можуть співіснувати і використовуватись разом. Методології та інструменти моніторингу можуть збігатися, але різні зацікавлені сторони, такі як лікар і механік, можуть фокусуватися на різних показниках. Інструменти

моніторингу можна адаптувати для створення індивідуальних відображень результатів.

Показники апаратного забезпечення пристрою – це вимірювання самого пристрою або фізичних компонентів, часто з використанням вбудованих датчиків. Ці вимірювання важливі для оцінки продуктивності пристрою, але вони можуть бути менш значущими для кінцевих користувачів пристрою. Прикладом таких показників є напруга акумулятора або потужність радіосигналу.

Програмне забезпечення на пристрої включає не тільки прикладне програмне забезпечення, але й системне, таке як операційна система або частини мережевого стека та драйвера пристроїв. Термін "прошивка" часто використовується для позначення всього програмного забезпечення на пристрої, але в контексті цього опису він застосовується для опису нижчих рівнів програмного забезпечення, які є більш специфічними для конкретного пристрою чи платформи, ніж для конкретної програми. Прикладом показників прошивки є частота перезавантажень або рівень використання пам'яті.

Вимірювання навколишнього середовища за допомогою датчиків часто є тим, що зазвичай мається на увазі у контексті IoT пристроїв. Очевидні приклади – це датчики, які вимірюють температуру або рівень освітленості. Однак, датчики можуть також фіксувати локальний бездротовий зв'язок, як у випадку сенсорів для моніторингу руху або рівня вологості.

Рішення IoT є складними системами, що включають програмні компоненти, які працюють як на самих пристроях, так і в хмарі. Для того, щоб розуміти, як ці системи взаємодіють, необхідно знати, до яких даних має доступ кожна сторона та як вони інтегруються між собою. Наприклад, питання "скільки часу потрібно для того, щоб повідомлення дісталось до хмари?" не можна однозначно відповісти лише з боку хмарної служби, оскільки вона має інформацію тільки про прибуття повідомлення. Щоб отримати точну метрику, можна надіслати час публікації пристрою і порівняти його з часом отримання повідомлення. Прикладом таких вимірювань можуть бути тайм-аути з'єднання,

кількість переданих байтів або помилки автентифікації [11].

Керівник здійснює моніторинг системи, проводячи вимірювання та спостереження, після чого ці дані перетворюються на метрики, що зберігаються у вигляді серій. У контексті IoT для спостереження за навколишнім середовищем часто використовуються датчики. Бібліотеки контрольно-вимірювальних приладів для моніторингу часто включають логічні програмно визначені інструменти, такі як датчики. У системах IoT фізичні датчики, наприклад, датчики температури, можуть безпосередньо корелювати з логічними метриками.

Метрики можуть також відображати більш абстрактні рівні функціональності, що вимагають попередньої обробки даних. Наприклад, датчик акселерометра постійно вимірює силу G, але значущою метрикою буде лише визначення моменту, коли відбувається подія "тремтіння". Локальна обробка даних може забезпечити алгоритм для виявлення таких подій і зберігати лише їх кількість, без необхідності передавати всі вимірювання акселерометра.

Важливо не тільки визначити, які системи підлягають моніторингу, але й зрозуміти, чому він здійснюється. Система, з якою працює користувач, повинна виконувати свою основну функцію, і метою моніторингу є забезпечення належного виконання цієї функції або послуги.

Коли користувач моніторить програмне забезпечення, він звертає увагу на показники, пов'язані з продуктивністю, такі як час відгуку запитів. Якщо моніториться фізичний процес, наприклад, обігрів приміщення, виробництво електрики або фільтрація води, використовуються пристрої для вимірювання таких процесів, як час роботи двигуна або тривалість циклів. Незалежно від того, чи є пристрій лише інструментом для моніторингу фізичних процесів, чи сам здійснює послугу, користувач бажає отримувати вимірювання, що відображають стан самого пристрою.

Метрики можуть бути як низького рівня (необроблені та прямі), так і високого рівня (абстрактні). Метрики вищого рівня можуть бути отримані на основі метрик нижчого рівня. Спочатку слід визначити високоякісні показники,

необхідні для забезпечення надання послуг. Після цього користувач може встановити, які метрики нижчого рівня потрібно збирати для підтримки поставлених цілей моніторингу. Важливо не потрапити в пастку вимірювання того, що не приносить користі, лише через можливість вимірювання або через те, що ці показники можуть здатися вражаючими (так звані "метрики марнославства").

Хороші метрики повинні мати такі характеристики:

- діяльність - мають інформувати користувачів про необхідність змінити поведінку системи чи служби;

- порівнянність - дозволяють порівнювати продуктивність з часом або між різними групами пристроїв, які можуть бути розташовані в різних місцях чи мають різні версії мікропрограм чи обладнання;

- зрозумілість та актуальність - повинні бути актуальними в контексті операційних задач і означає, що крім простих вихідних значень, таких як підсумки, вони можуть надавати додаткову інформацію, наприклад, коефіцієнти або інші показники;

- надання інформації з правильним дозволом - користувач повинен мати можливість визначити, як часто збирати дані, як часто складати звіти, а також як обчислювати середні значення, відставання та будувати графіки.

Ці налаштування повинні відповідати контексту послуги, яку надає система. Наприклад, надання звітності про заповнення SD-карти кожну секунду може створити надмірну кількість деталей та обсягу, в той час як перегляд середнього навантаження центрального процесора за годину може приховати короткочасні стрибки активності. Важливо, щоб роздільна здатність даних відповідала конкретним вимогам моніторингу та діагностики.

Спільнота моніторингу розробила ряд методологій для програмного забезпечення та розподілених систем, як зведено у таблиці 1.2.

Таблиця 1.2 - Методології для програмного забезпечення та розподілених систем

Метод	Сигнали
Чотири золоті сигнали (з книги <i>Google Engineering Reliability Engineering</i>)	Затримка Рух Помилки Насиченість
Червоний метод, який фокусується на вимірюванні речей, про які турбуються кінцеві користувачі	Оцініть Помилки Тривалість
Метод <i>USE</i> , який орієнтований на продуктивність та вузькі місця у системі.	Використання Насиченість Помилки

Моніторинг має на меті надання загального огляду або аналізу того, як показники змінюються з часом, але він не призначений для фіксації кожної важливої точки даних чи події, навіть у процесі операцій. Окремі вимірювання зазвичай не є критичними при створенні системи моніторингу. Хоча користувач часто збирає метрики для оперативного моніторингу, ці дані також можуть використовуватись в інших цілях збору інформації. Наприклад, значення з метрик можуть бути використані для довгострокового аналізу або для демонстрації поточного стану користувачам через мобільний додаток, як частина звичайного використання пристрою [11].

Іноді користувач може захотіти використовувати моніторингові дані для аналітики поза межами самого моніторингу, але це потрібно робити обережно, щоб не вплинути на сам процес моніторингу. Наприклад, він може захотіти визначити тренди використання продукції на основі моніторингових даних, але при цьому не бажає створювати обчислювальну панель в тому ж інструменті, який використовується для критичного моніторингу. Роздільна здатність метрик зазвичай достатньо висока, щоб тривале зберігання даних було непрактичним, і вони часто зберігаються лише протягом кількох тижнів. Для вирішення обох цих проблем користувач може застосувати підхід Pub/Sub, щоб дублювати потоки метрик, що дозволяє ефективно використовувати дані телеметрії для моніторингу та інших завдань.

1.5 Висновки до розділу

У цьому розділі здійснено аналіз мережі Інтернету Речей (IoT) з метою виявлення її специфічних особливостей. Розглянуто, які протоколи, технології та типи мереж використовуються для побудови IoT-мереж, а також проаналізовано архітектуру цих мереж та пристрої, які беруть участь у їх роботі. Окремо розглянуто завдання, які виконує моніторинг у таких мережах, та причини його застосування. Крім того, проведено аналіз основних проблем і викликів кібербезпеки, з якими стикаються мережі Інтернету Речей.

2 ВИБІР АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ФУНКЦІОНУВАННЯ СИСТЕМИ

Для забезпечення ефективної роботи системи управління та моніторингу необхідно обрати таке апаратне забезпечення, яке відповідатиме вимогам для виконання визначених завдань і досягнення поставлених цілей. З цією метою пропонується наступне обладнання.

Інтернет речей - це не лише сукупність різноманітних пристроїв та датчиків, з'єднаних між собою дротовими та бездротовими каналами зв'язку, що підключаються до Інтернету. Це також глибока інтеграція реального та віртуального світів, де відбувається взаємодія між людьми та пристроями [12].

Рішення, засновані на Інтернеті речей, стають все більш популярними завдяки можливості отримувати додатковий прибуток для постачальників «розумних» рішень. «Розумна» поведінка пристроїв може суттєво збільшити їхню споживчу цінність. Наприклад, вентилятор, який автоматично вимикається при досягненні заданої температури, допомагає економити електроенергію, що робить його дорожчим для користувача. А вентилятор, який також може виявляти

присутність людей у приміщенні, має ще більшу цінність.

Як техніка може стати «розумною»? По-перше, через свою конструкцію, яка дозволяє системі демонструвати «розумну» поведінку. По-друге, за допомогою «інтелектуалізації» — оснащення системи пристроями для збору, обробки інформації та прийняття рішень. Такий підхід дає змогу створити складну «розумну» поведінку без необхідності складної конструкції. І третій шлях - це інтеграція системи з іншими системами. Наприклад, для енергозбереження системі опалення може знадобитися короткостроковий прогноз погоди, який можна отримати за допомогою датчиків та обробки їх даних або звернувшись до Інтернету.

Важливо, що з точки зору замовника система виглядає однакою в обох випадках, тому вони готові заплатити за цю функціональність однакову ціну. Однак для постачальника підключення до Інтернету буде значно дешевшим, ніж створення метеостанції.

Інтелект і конективність пристроїв відкривають нові можливості, які можна поділити на чотири категорії:

- моніторинг;
- управління;
- оптимізація;
- автономність.

Кожна з цих функцій є важливою окремо, і служить основою для наступного етапу. Наприклад, моніторинг є передумовою для управління, оптимізації та автономності системи. Моніторинг значно полегшує користування системою та робить її ефективнішою. Компанії можуть вибирати набір функцій для своєї продукції, щоб зробити її більш корисною для споживача, що зміцнює їх конкурентні позиції.

Наприклад, автоматична теплиця, яка самостійно здійснює полив, підтримує потрібну температуру та рівень освітленості, є ідеальним вибором для тих, хто не має часу або можливості доглядати за рослинами, особливо під час відсутності персоналу. Однією з проблем, яку вирішує функція моніторингу, є

усунення занепокоєння щодо стану рослин, коли немає можливості бути присутнім - наприклад, чи є вода в системі, чи не вимкнулося електропостачання, чи працює система вентиляції.

Клієнт готовий заплатити більше, якщо зможе в будь-який момент перевірити умови в теплиці. Тому теплиця з функцією віддаленого моніторингу може мати значно вищу ціну, а її реалізація для виробника буде досить простою. В результаті застосування технології Інтернету речей виробник отримує додатковий прибуток.

Ще більшу споживчу цінність теплиця матиме, якщо додати функцію управління, яка дозволить користувачам не лише отримувати інформацію про умови, але й змінювати їх на свій розсуд. Наприклад, функція оптимізації дозволить економити гроші, уникаючи необґрунтованих витрат, таких як включення опалення, коли на прогнозі погоди очікується підвищення температури.

Нарешті, за допомогою Інтернету речей можна стежити за витратами матеріалів (наприклад, добрив) і автоматизувати їх замовлення, а також контролювати стан елементів, які потребують обслуговування (насоси, вентилятори). Це дозволить здійснювати самодіагностику та обслуговування теплиці, наближаючи її до повної автономії.

2.1 Вибір апаратного забезпечення для системи

Для забезпечення моніторингу та досягнення поставлених цілей пропонується використання міні-ПК Raspberry Pi. Цей пристрій здатен виконувати як базові завдання, такі як вивчення основ роботи з комп'ютером, інтернет-серфінг, відтворення відео та аудіофайлів, так і більш складні функції, зокрема інтеграцію в систему IoT.

Raspberry Pi - це компактний комп'ютер, розміром з кредитну карту, вартість якого становить близько 25 доларів за базову модель і 35 доларів за новішу версію. Він здобув величезну популярність і на сьогоднішній день продано понад 4,5 мільйона одиниць[13].

Однією з ключових переваг міні-ПК Raspberry Pi є його чудове співвідношення ціни та якості. Зокрема, він відмінно підходить для використання в навчальних цілях, допомагаючи студентам освоювати основи інформаційних технологій. Raspberry Pi позиціонується як доступне рішення для початківців у галузі розробки. Завдяки його потужності, низькому енергоспоживанню та невисокій вартості, пристрій може використовуватись для створення особистого міні-сервера.

Щодо статистичного управління процесами - методики моніторингу виробничих процесів для забезпечення якості продукції під час її виготовлення, Raspberry Pi є першим доступним технічним рішенням, яке можна використовувати для програмування на різних мовах та як мікроконтролер для управління роботизованими пристроями.

Одна з найбільш привабливих характеристик Raspberry Pi - наявність апаратних портів GPIO (General Purpose Input Output), що відкриває можливості для використання цього пристрою в робототехнічних проектах та «розумних будинках».

На ринку також є альтернативи Raspberry Pi, такі як Orange Pi Prime, Banana Pi M3, Rock64, ASUS Tinker board S, Libre Computer Renegade, Odroid H2 та Arduino Mega 2560.

Orange Pi Prime є вдосконаленою версією Raspberry Pi 3 завдяки наявності 2 ГБ оперативної пам'яті та вбудованого відеоприскорювача Mali-450 GPU в SoC AllWinner H5, що дозволяє відтворювати 2K відео. Серед особливих характеристик можна виділити наявність ІЧ-приймача, що дає можливість управляти пристроєм за допомогою пульта дистанційного керування або мобільних телефонів з вбудованим ІЧ-світлодіодом. Крім того, Orange Pi Prime має вбудований мікрофон і відеоінтерфейс CSI, який підтримує відео з

роздільною здатністю до 1080p на швидкості 30 кадрів на секунду.

Окремо варто згадати, що Orange Pi Prime має виведений UART із TTL-рівнями, завдяки чому можна спостерігати за процесом завантаження Linux через термінал. У аудіообладнанні, окрім вбудованого мікрофона, є лінійний вихід та аудіовихід через HDMI. Відеоприскорювач підтримує OpenGL ES 2.0 і OpenVG 1.1. Серед підтримуваних операційних систем є Ubuntu, Debian та Android 5.1.

Banana Pi M3 є флагманською моделлю, що побудована на базі восьмиядерного SoC Allwinner A83T (ARM Cortex-A7) з графічним процесором PowerVR SGX544MP1, який працює на частоті до 1,8 ГГц. Пристрій оснащений 2 ГБ оперативної пам'яті, 8 ГБ флеш-пам'яті, гігабітним Ethernet, двома портами USB, Wi-Fi 802.11 b/g/n, Bluetooth 4.0, HDMI, а також портом SATA. Як і Orange Pi Prime, Banana Pi M3 має ІЧ-приймач, відеоінтерфейс CSI, UART, мікрофон, лінійний вихід та аудіовихід через HDMI. Крім того, пристрій оснащений дисплейним інтерфейсом MIPI DSI, інтегрованим з I2C для сенсорних екранів, та GPIO-гребінкою на 40 контактів.

Rock64 виділяється наявністю 4 ГБ оперативної пам'яті, що обслуговує 64-бітний процесор ARM Cortex-A53, та відеопідсистемою, здатною обробляти відео в роздільній здатності 4K при 60 кадрах на секунду. Пристрій підтримує живлення через PoE, а графічна система ARM Mali 450MP2 відповідає стандартам OpenGL ES 2.0 та OpenVG 1.1. Rock64 підтримує операційні системи Debian, CentOS, Fedora та Android 8, а також інші дистрибутиви на базі Linux, завдяки активному ком'юніті та доступній документації. Комп'ютер має 64 контакти GPIO, включаючи сигнали Ethernet, а також порт USB 3.0, що робить його привабливим для проєктів із високими вимогами до обладнання [14].

ASUS Tinker Board S побудований на базі SoC Rockchip RK3288 із чотириядерним ARM Cortex-A17 та відеопроцесором Mali-T760 MP4. Пристрій підтримує OpenGL ES 3.1, OpenCL 1.1, Renderscript і Direct3D 11.1, а його операційною системою є TinkerOS на базі Debian із можливістю встановлення Android.

Libre Computer Renegade є конструктивно подібним до Raspberry Pi, що

дозволяє використовувати його в корпусах, призначених для Raspberry. Пристрій побудований на SoC RK-3328, оснащений чотириядерним ARM Cortex-A53 із частотою до 1,5 ГГц і графічним процесором Mali 450MP2 із частотою 500 МГц. Renegade пропонує варіанти оперативної пам'яті: 1 ГБ (\$35), 2 ГБ (\$50) або 4 ГБ (\$80). Підтримувані ОС включають Ubuntu 18.04, Debian 9, OpenMediaVault 4, Station OS та Android 7.1.

Odroid H2 базується на 64-бітному чотириядерному Intel Celeron Gemini Lake J4105, пропонує x86-сумісну платформу. Пристрій оснащений GPU Intel UHD Graphics 600, шиною PCI-E Gen2, здвоєним SATA 6 Гбіт/с, а також підтримує до 8 ГБ оперативної пам'яті DDR4 (з можливістю встановлення до 32 ГБ). Плата підтримує Windows 10 та Linux x64, а також DirectX 12, OpenGL 4.3 та OpenCL 2.0.

У таблиці 2.1 можна знайти детальну порівняльну характеристику цих пристроїв із Raspberry Pi.

Таблиця 2.1 - Порівняльна характеристика Raspberry Pi та аналогів

Пристрій	Процесор	ОЗУ	GPU	Підтримувані ОС	Особливості	Ціна
1	2	3	4	5	6	7
Raspberry Pi 3/4	ARM Cortex-A53 / Cortex-A72	1-8 ГБ	Broadcom VideoCore IV/V	Raspbian, Ubuntu, Windows 10 IoT	GPIO, низьке енергоспоживання, компактний розмір, популярність у спільноті	\$35-\$55
Orange Pi Prime	AllWinner H5	2 ГБ	Mali-450 GPU	Ubuntu, Debian, Android 5.1	Вбудований мікрофон, ІЧ-приймач, відеоінтерфейс CSI, підтримка 2K відео, OpenGL ES 2.0	\$30-\$40
Banana Pi M3	AllWinner A83T	2 ГБ	PowerVR SGX544MP1	Android, Linux	SATA-порт, ІЧ-приймач, 8 ГБ флеш-пам'яті, відеоінтерфейс MIPI DSI для сенсорного екрану	\$80-\$100

Продовження таблиці 2.1

1	2	3	4	5	6	7
Rock64	ARM Cortex-A53	До 4 ГБ	Mali-450 MP2	Debian, CentOS, Android 8	64 GPIO контакти, підтримка 4K @ 60 fps, USB 3.0, PoE, активне ком'юніті	\$35-\$80
ASUS Tinker Board S	Rockchip RK3288	2 ГБ	Mali-T760 MP4	TinkerOS, Android	Підтримка OpenGL ES 3.1, Direct3D 11.1, продуктивний CPU, підвищена енергоефективність	\$60-\$75
Libre Computer Renegade	RK-3328	1-4 ГБ	Mali-450 MP2	Ubuntu, Debian, Android 7.1	Висока сумісність із Raspberry Pi, корпуси підходять, низька ціна	\$35-\$80
Odroid H2	Intel Celeron J4105	До 32 ГБ	Intel UHD Graphics 600	Windows 10, Linux x64	x86-сумісність, SATA, PCI-E Gen2, до 32 ГБ ОЗУ, GPU підтримує OpenGL 4.3, DirectX 12	\$100-\$120

Отже, на основі проведеного аналізу та порівняльної характеристики визначено, що оптимальним вибором для даного дослідження є міні-ПК Raspberry Pi, оскільки він поєднує доступну вартість із відповідними технічними характеристиками, які задовольняють потреби проєкту .

2.2 Вибір обладнання

На початковому етапі необхідно скласти перелік обладнання, яке буде використовуватись для побудови системи моніторингу та управління елементами IoT. Вибір відповідного обладнання є важливим етапом, адже знадобляться не лише центральний пристрій, такий як міні-ПК Raspberry Pi, а й периферійні компоненти [15].

Для функціонування системи необхідний центральний пристрій - у цьому випадку міні-ПК Raspberry Pi. Для його налаштування потрібні такі компоненти, як клавіатура, миша, монітор і карта пам'яті.

Нижче наведено повний перелік обладнання для налаштування міні-ПК Raspberry Pi:

- одноплатний комп'ютер Raspberry Pi.
- монітор;
- клавіатура;
- маніпулятор типу «миша»;
- адаптер живлення;
- карта пам'яті.

Окрім цього, важливою складовою системи є датчики, які виконують основні функції моніторингу. Повний список необхідних датчиків наведено далі:

- датчик температури DS18B20;
- датчик температури та атмосферного тиску BMP180;
- датчик освітленості TSL2561.

Оскільки міні-ПК Raspberry Pi не оснащений вбудованим Wi-Fi модулем, для забезпечення бездротового підключення необхідно додатково придбати Wi-Fi адаптер, наприклад, модель EDUP Nano EP-N8531. Він підключається через USB-порт і забезпечує більшу мобільність системи, усуваючи необхідність використання мережевого кабелю.

2.3 Характеристики мінікомп'ютера Raspberry Pi

Перші версії міні-ПК Raspberry Pi, а саме моделі «А», «В» (рис. 2.1), «А+» і «В+», були оснащені процесором Broadcom BCM2835 з архітектурою ARM11, що працює на тактовій частоті 700 МГц. Оперативна пам'ять у цих моделях становила 256 або 512 Мбайт, встановлена за технологією package-on-package безпосередньо на процесорі.

Модель «А» мала один порт USB 2.0, модель «В» — два порти USB 2.0, а модель «В+» — чотири таких порти. Крім того, моделі «В» і «В+» були обладнані

портом Ethernet. У складі процесора BCM2835 також є графічне ядро з підтримкою OpenGL ES 2.0, апаратного прискорення та відтворення відео у форматі Full HD, а також цифровий сигнальний процесор (DSP).

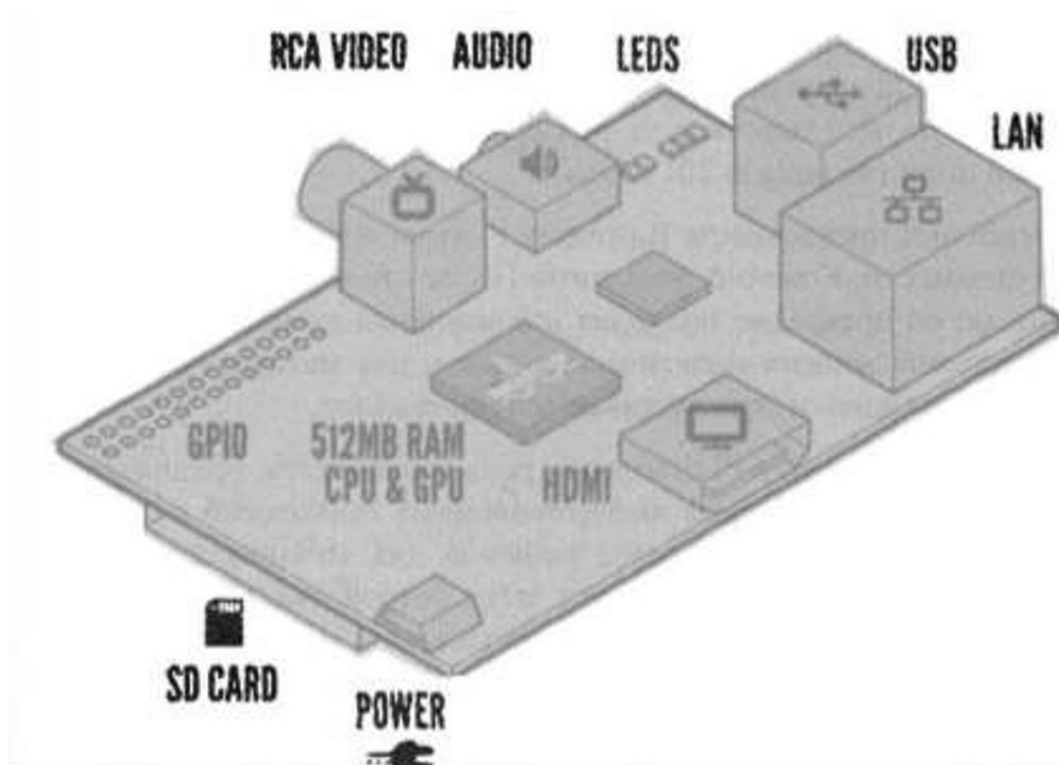


Рисунок 2.1 - Схема міні-ПК *Raspberry Pi*, Модель «В»

Міні-ПК працює через роз'єм MicroUSB, при цьому необхідний струм повинен становити щонайменше 0,5-0,7 А. Якщо сила струму буде нижчою, комп'ютер може запускатися, але перезавантажуватиметься при виконанні ресурсоємних завдань. Тому плату краще підключати безпосередньо до USB-порту комп'ютера або до розетки через спеціальний адаптер, уникаючи використання USB-хабів.

Міні-ПК *Raspberry Pi* не має кнопок увімкнення та вимкнення. Щоб запустити пристрій, достатньо підключити USB-живлення, а для вимкнення - просто витягти кабель. Сподіваємось, що в майбутніх версіях міні-ПК *Raspberry Pi* буде додано живлення через Ethernet, оскільки це одне з найбільш популярних побажань користувачів.

Моделі «А» і «В» оснащені слотом для карт пам'яті формату SD, а версії «А+» і «В+» — для карт формату MicroSD. Міні-ПК Raspberry Pi не працює без карти пам'яті, оскільки на ній зберігається операційна система — це як спроба запустити комп'ютер без жорсткого диска. Оскільки у Raspberry Pi немає вбудованої ОС (як, наприклад, у смартфонах), це дає одну перевагу: пристрій майже неможливо перетворити на "цеглу". Після невдалої спроби експерименту достатньо просто переписати операційну систему на карті пам'яті, і міні-ПК знову буде працювати як новий.

Мікрофон можна підключити до міні-ПК Raspberry Pi, однак для цього необхідно знайти сумісний USB-пристрій. Всі поточні моделі Raspberry Pi не мають вбудованого модуля Wi-Fi, тому для підключення до Інтернету необхідно використовувати Ethernet-порт. Оскільки Ethernet на платі Raspberry Pi підключено через USB 2.0, максимальна швидкість передачі даних обмежена 100 Мбіт/с, а не 1 Гбіт/с.

Процесори та графічні прискорювачі в моделях Raspberry Pi не мають навіть простих радіаторів, і, хоча після кількох годин роботи плата лише злегка нагрівається, це рішення дозволяє ефективно уникати перегріву. Частота процесора становить 700 МГц, і його можна розігнати до 1000 МГц без втрати гарантії, залежно від моделі. Чіп пам'яті виробництва Samsung або Hynix розташований поверх основного чіпсета, тому збільшити обсяг RAM самостійно неможливо. Для моделей з 256 МБ RAM маркування чіпа починається з K4P2G, а для версій з 512 МБ — з K4P4G.

Відеоприскорювач Broadcom VideoCore N дозволяє декодувати відео 1080p h.264 з бітрейтом до 40 Мбіт/с, при цьому для апаратного прискорення MPEG-2 і VC-1 необхідно придбати окремі ліцензії.

Плата має п'ять світлодіодів індикації: три з них показують активність Ethernet та його режим роботи, ще два сигналізують про наявність живлення та роботу з SD-картою.

Що стосується низькорівневих інтерфейсів, то на платі є слоти CSI-2 для підключення камери і DSI для дисплея. Також є колодка для 26 (40 для версії

«B+») ліній вводу-виводу GPIO, що дозволяє підключати UART, консольний порт, шину SPI, I2C і 12S для аудіопристроїв.

Серед недоліків можна відзначити відсутність вбудованих годин реального часу (RTC), тому для отримання часу необхідно використовувати NTP-сервери. Хоча SoC містить цифровий сигнальний процесор (DSP), доступу до його API на даний момент немає.

У новій версії міні-ПК Raspberry Pi 2 Model B замість процесора BCM2835 встановлений чіп BCM2836 (рис. 2.2). Цей чіп має чотири ядра ARM Cortex-A53 з тактовою частотою 1400 МГц, що значно покращує продуктивність у багатозадачних операціях. Обсяг оперативної пам'яті збільшений до 1 ГБ, а відеоядро залишилося незмінним - Broadcom VideoCore IV. Розробники зазначають, що продуктивність у багатопоточних тестах зросла в шість разів, а в однопоточних - у три рази.

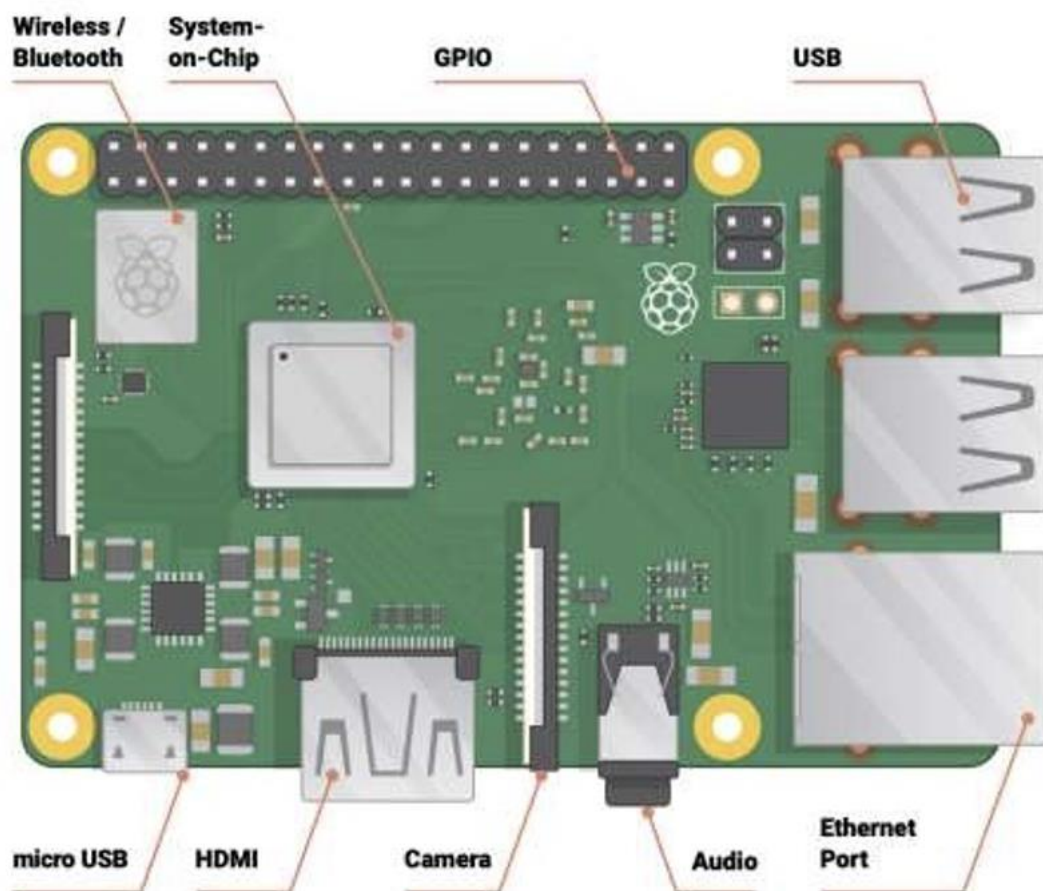


Рисунок 2.2 - Схема міні-ПК Raspberry Pi 2 Model B

2.4 Занальна концепція системи

Оскільки міні-ПК Raspberry Pi не має вбудованої внутрішньої пам'яті, необхідно підготувати зовнішнє сховище для операційної системи. Для цього була обрана карта пам'яті формату MicroSD Transcend Premium microSDHC 16GB Class 10 UHS-I, що має 16 ГБ постійної пам'яті. Цього обсягу достатньо для інсталяції операційної системи та подальших налаштувань. Для встановлення карти пам'яті потрібно просто вставити її в слот MicroSD (рис. 2.3).

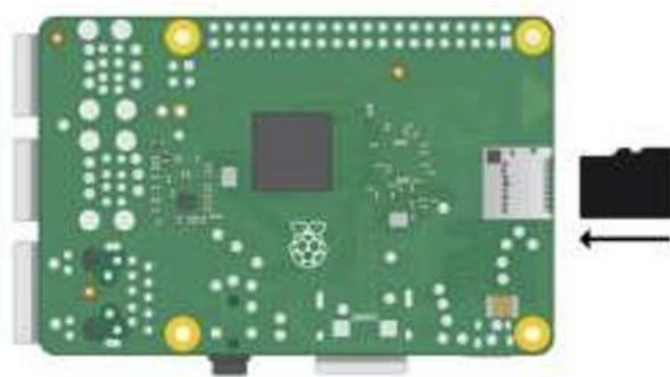


Рисунок 2.3 - Підключення карти пам'яті до міні-ПК Raspberry Pi

Наступним етапом є підключення засобів введення та виведення. Спочатку необхідно підключити клавіатуру та маніпулятор типу «миша», для чого можна скористатися USB-портами міні-ПК Raspberry Pi, через які підключається периферія.

Після цього потрібно підключити монітор. Для цього використовується HDMI-кабель, який слід під'єднати до відповідного порту на міні-ПК Raspberry Pi та на моніторі.

Останнім кроком в апаратному налаштуванні є підключення Raspberry Pi до джерела живлення. Це потрібно робити тільки тоді, коли користувач готовий до налаштування програмного забезпечення, оскільки міні-ПК не має вимикача живлення і автоматично вмикається при підключенні до джерела живлення з відповідною напругою.

У підсумку, в загальному вигляді ми отримуємо пристрій з підключеними периферійними пристроями через USB (білий кабель), монітор через HDMI (червоний кабель) та адаптер живлення через мікро USB (чорний кабель). Більш наочно це відображено на рисунку 2.4.

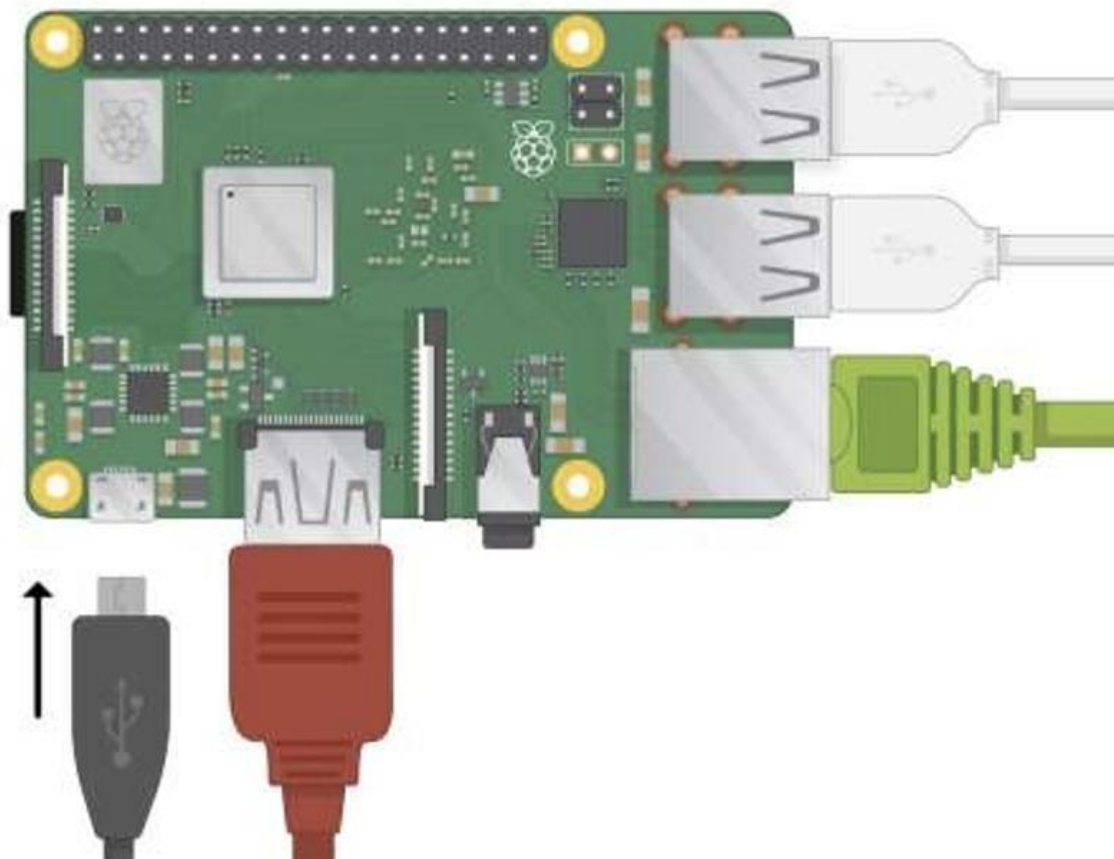


Рисунок 2.4 - Загальний вигляд міні-ПК Raspberry Pi
з підключеними пристроями

У результаті, повна схема системи на основі міні-ПК Raspberry Pi представлена на рисунку 2.5. Крім того, до міні-ПК Raspberry Pi можна підключити необхідні для роботи системи датчики, що дозволяє розширювати можливості моніторингу та управління.

Також є можливість підключення до мережі Інтернет, що забезпечує доступ до віддаленого керування та обміну даними між пристроями в системі.

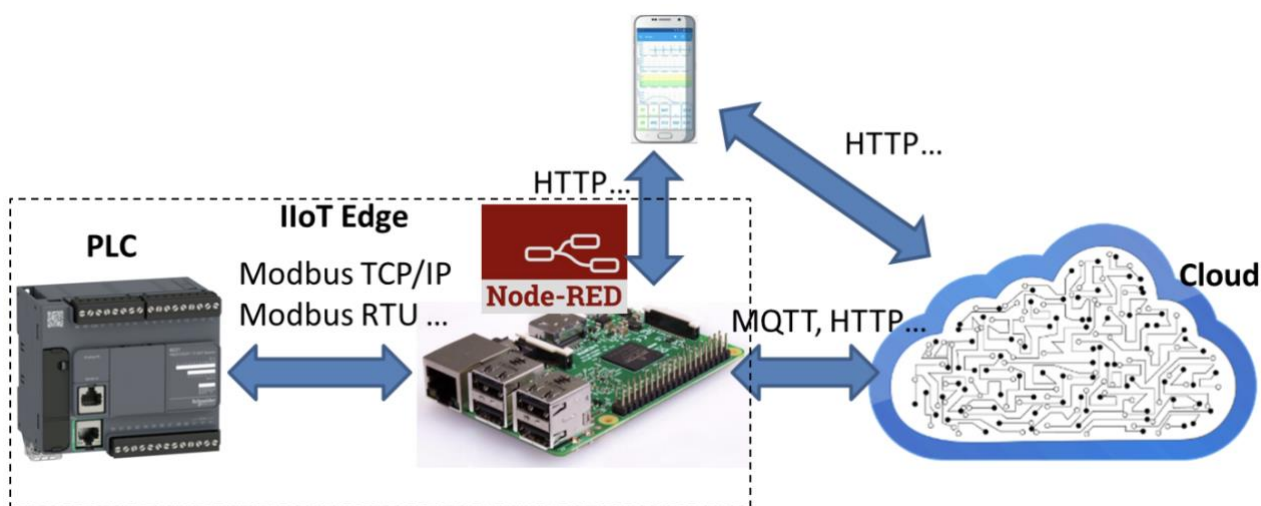


Рисунок 2.5 - Загальна схема системи IoT на основі міні-ПК Raspberry Pi

2.5 Висновки до розділу

У цьому розділі було здійснено аналіз апаратного забезпечення, яке необхідне для функціонування системи управління та моніторингу. Для виконання поставлених задач та досягнення цілей було обрано міні-ПК Raspberry Pi як центральний елемент системи. Цей пристрій здатен виконувати як прості завдання, наприклад, навчання основам роботи з комп'ютером, інтернет-серфінг, відтворення відео та аудіофайлів, так і більш складні функції, такі як інтеграція в систему IoT.

Однією з переваг Raspberry Pi є його економічність, що робить цей міні-ПК ідеальним варіантом для побудови системи управління та моніторингу елементами IoT, що є основною метою цього дослідження.

Також було проаналізовано технічні характеристики міні-ПК Raspberry Pi та додаткового обладнання, необхідного для налаштування цієї системи, включаючи периферійні пристрої, датчики та додаткові модулі. Наприкінці було підсумовано економічні показники для цього обладнання.

3 НАЛАШТУВАННЯ МОНІТОРІНГУ ТА КЕРУВАННЯ СИСТЕМИ

3.1 Налаштування програмного забезпечення системи

Перед початком використання міні-ПК Raspberry Pi необхідно виконати налаштування його програмного забезпечення, включно з установкою операційної системи, яка забезпечує функціонування пристрою. Для спрощення цього процесу створено програму NOOBS – інструмент, що дозволяє легко обрати одну з кількох доступних операційних систем і автоматично її встановити. NOOBS можна завантажити з офіційного сайту Raspberry та встановити на карту пам'яті [16].

Завантажений образ операційної системи слід записати на карту пам'яті microSD. Для цього можна скористатися програмою Win32DiskImager.exe. Після запуску програми в полі *Device* потрібно вибрати карту пам'яті, на яку буде здійснено запис, а потім, натиснувши на піктограму у вигляді синьої папки, вказати шлях до завантаженого образу ОС. Завершальним кроком є натискання кнопки Write (рис. 3.1).

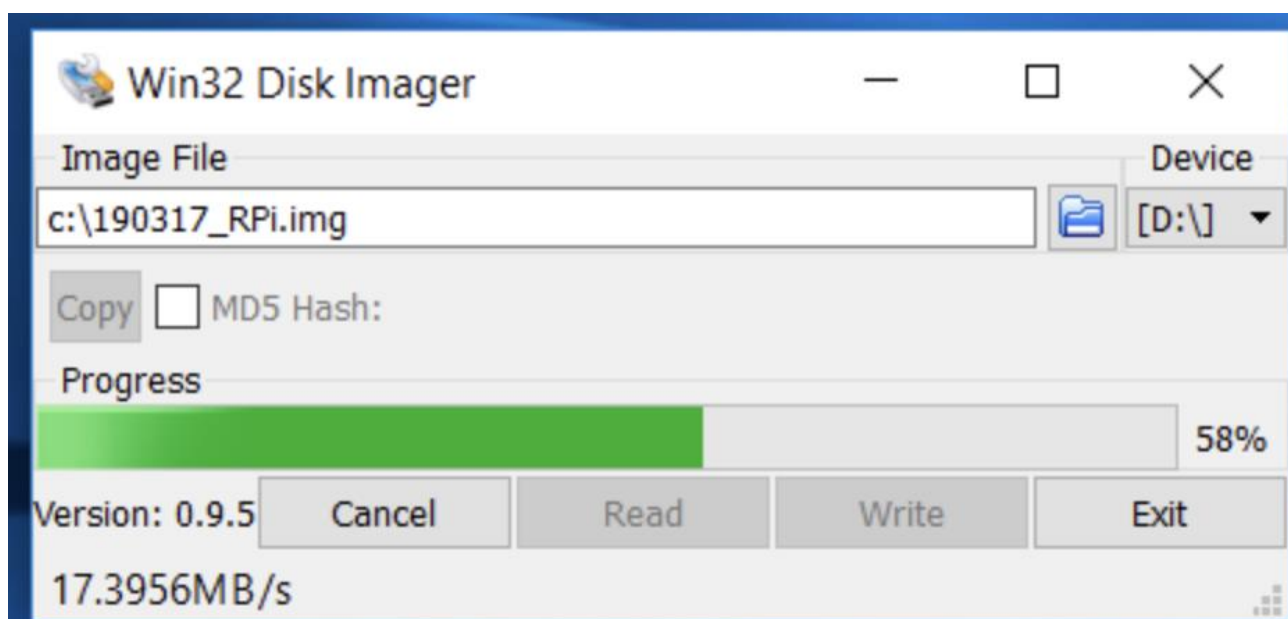


Рисунок 3.1 - Запис ОС raspberrian

Меню NOOBS – це система, яка надає можливість вибору операційної системи для роботи на міні-ПК Raspberry Pi. У стандартній комплектації NOOBS доступні дві операційні системи: Raspbian – версія Debian Linux, адаптована спеціально для Raspberry Pi, та LibreELEC – програмне забезпечення для мультимедійного центру Kodi Entertainment Center.

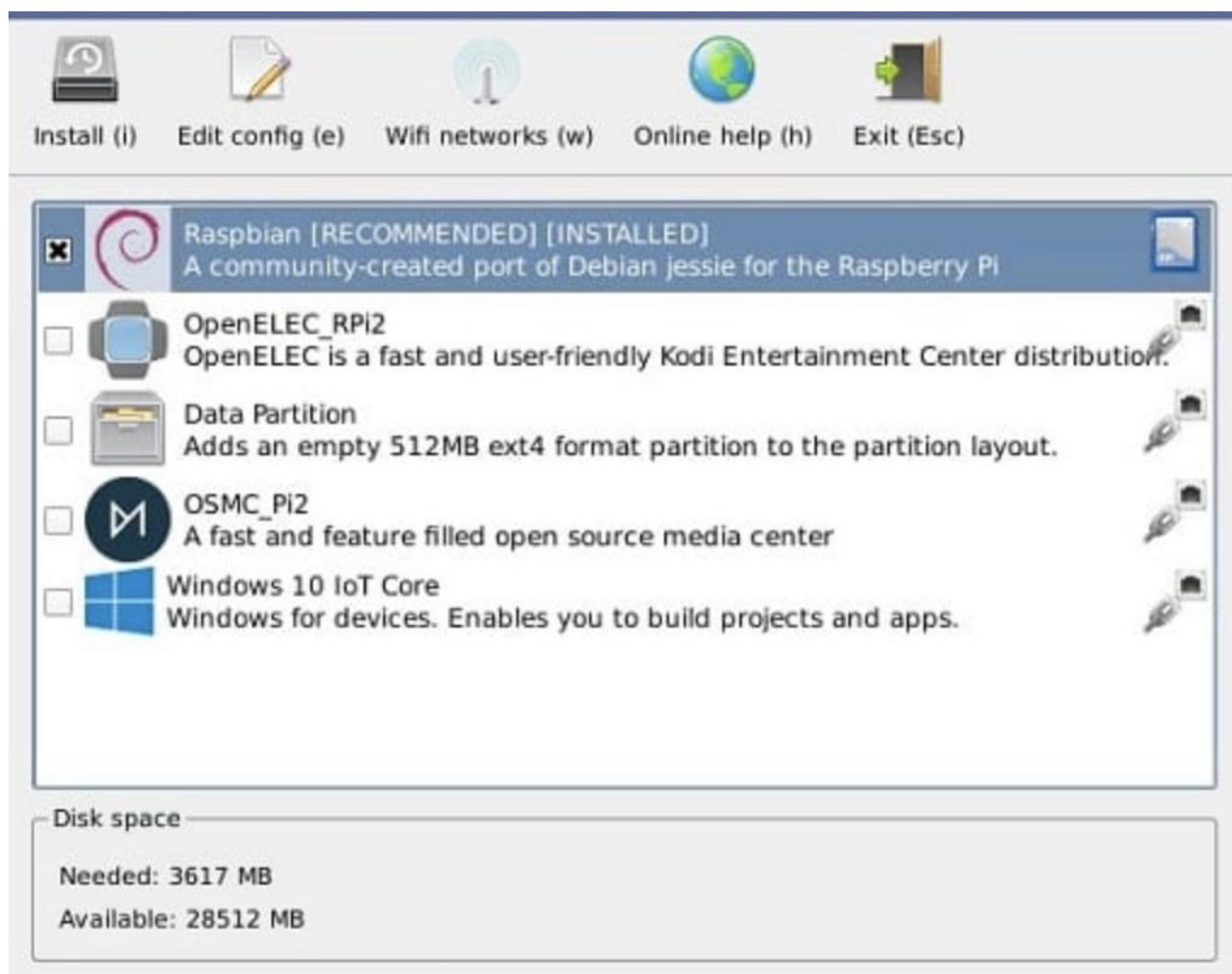


Рисунок 3.2 - Меню вибору NOOBS

Якщо міні-ПК Raspberry Pi підключений до мережі, використовуючи дротове з'єднання або опцію "Мережі Wi-Fi (w)", доступну у верхній панелі піктограм, можна також завантажувати та встановлювати інші операційні системи. У нашому випадку для роботи буде обрана операційна система Raspbian.

Після вибору ОС з'явиться вікно, яке відображає прогрес інсталяції операційної системи (рис. 3.3).

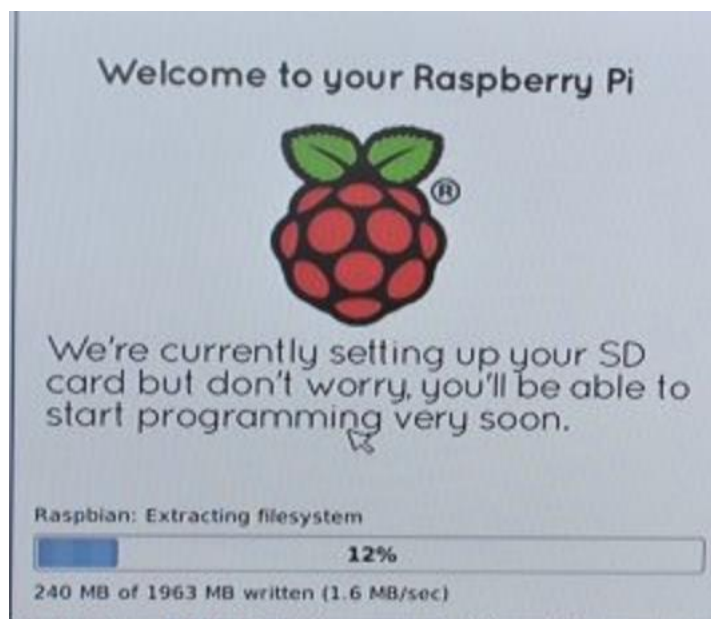


Рисунок 3.3 - Меню встановлення системи Raspbian

Після завершення встановлення необхідно перезавантажити міні-ПК Raspberry Pi. Після цього операційна система буде повністю готова до роботи.

Під час першого запуску користувач побачить робочий стіл та майстер початкових налаштувань операційної системи (рис. 3.4) [17].

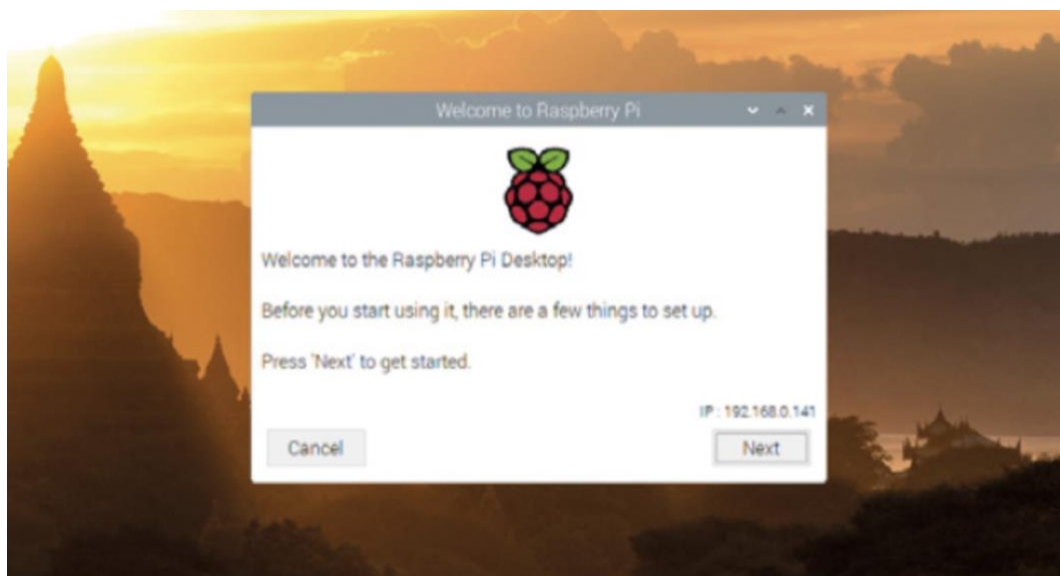


Рисунок 3.4 - Результат встановлення системи

3.2 Налаштування параметрів мережі

Ймовірно, міні-ПК Raspberry Pi не буде використовуватися для виведення інформації на монітор, тому вам, скоріше за все, знадобиться віддалений доступ через SSH. Для цього потрібно налаштувати мережеві параметри пристрою [18].

Налаштування мережевого з'єднання для Raspberry Pi здійснюється через файл конфігурації `/etc/network/interfaces`. Якщо IP-адреса пристрою автоматично отримується через DHCP, змінювати цей файл не потрібно. Однак, якщо необхідно налаштувати статичну IP-адресу, виконайте команду:

```
sudo nano /etc/network/interfaces
```

Після цього додайте до файлу відповідні рядки для налаштування статичної IP-адреси.

Лістинг 3.1 – Налаштування статистичної IP-адреси

```
iface eth0 inet static
address 192.168.0.117
netmask 255.255.255.0
gateway 192.168.0.28
auto eth0
```

Ось пояснення кожного з параметрів:

- `iface eth0 inet static` - вказує, що інтерфейс `eth0` використовує протокол IPv4 (`inet`) зі статичною IP-адресою (`static`);
- `address 192.168.0.117` - встановлює IP-адресу для мережевого інтерфейсу `192.168.0.117`;
- `netmask 255.255.255.0` - вказує маску підмережі, яка має значення `255.255.255.0`;
- `gateway 192.168.0.28` - вказує адресу шлюзу за замовчуванням, що становить `192.168.0.28`;
- `auto eth0` - означає, що інтерфейс `eth0` буде автоматично підключатися при завантаженні системи з зазначеними параметрами.

Для зміни адрес DNS-серверів потрібно виконати команду:

```
sudo gedit /etc/resolv.conf
```

Далі в цьому файлі необхідно прописати наступні рядки:

```
nameserver 192.168.1.1;
nameserver 8.8.8.8.
```

де 192.168.1.1 і 8.8.8.8 - це адреси DNS-серверів. Для додавання інших серверів кожен адресу потрібно вводити з нового рядка, починаючи зі слова `nameserver`.

3.3 Налаштування бездротового доступу

Іноді підключати мережевий кабель до міні-ПК Raspberry Pi буває незручно. Одним із способів вирішення цієї проблеми є використання USB Wi-Fi адаптера. У попередньому розділі згадувалося, що в якості адаптера можна використовувати EDUP Nano EP N8531.

Для підключення адаптера потрібно вставити його в USB-порт міні-ПК Raspberry Pi і перевірити, чи буде він виявлений системою. Для цього виконуємо команду `iwconfig` і перевіряємо, чи з'явиться адаптер у списку доступних мережевих пристроїв (рис. 3.4).



```
GNU nano 2.2.6      File: /etc/wpa_supplicant/wpa_supplicant.conf      Modified
country=DE
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1

# home wifi
network={
    ssid="my_home_network"
    psk="network_password"
    key_mgmt=WPA-PSK
    scan_ssid=1
}

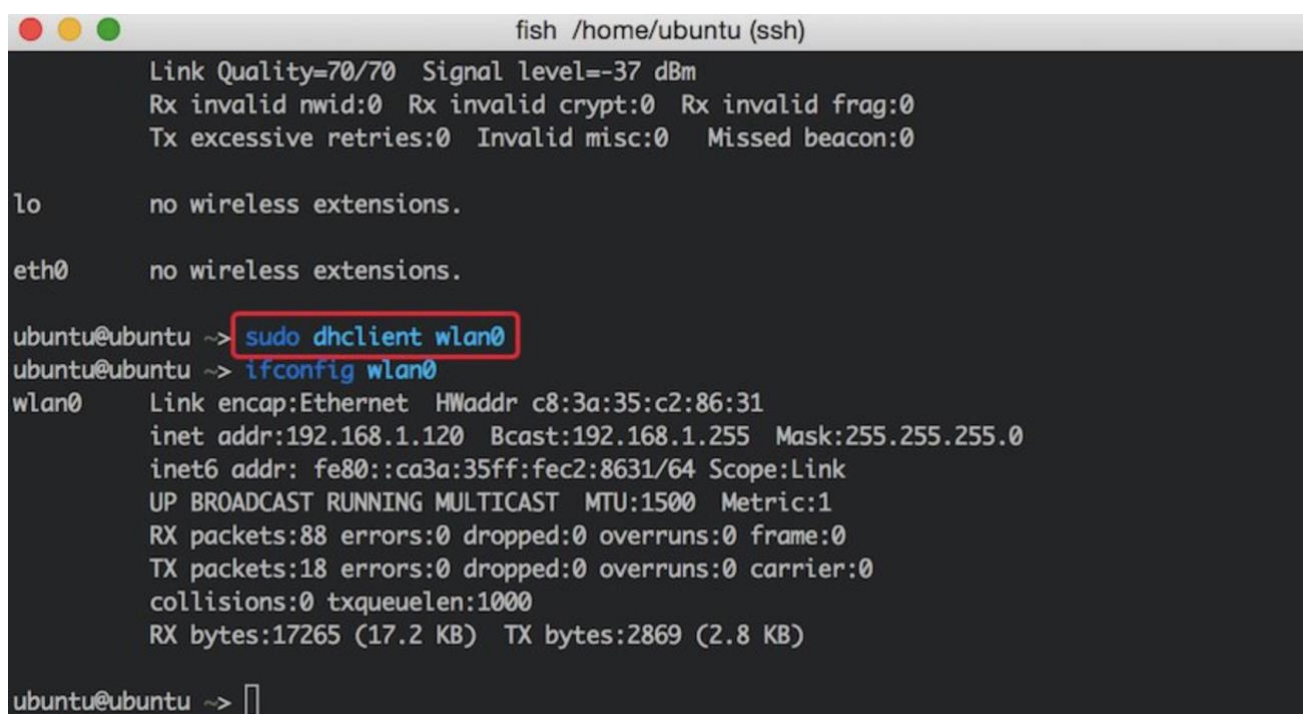
# work wifi
network={
```

Рисунок 3.4 - Вивід команди `iwconfig`

Далі здійснюємо сканування навколишнього простору на наявність доступних бездротових мереж за допомогою команди:

```
sudo iwlist wlan0 scan
```

В результаті цього процесу система виведе список доступних точок доступу, серед яких ми можемо побачити точку з іменем "AndroidAP" (рис. 3.5).



```

fish /home/ubuntu (ssh)
Link Quality=70/70  Signal level=-37 dBm
Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
Tx excessive retries:0  Invalid misc:0  Missed beacon:0

lo      no wireless extensions.

eth0    no wireless extensions.

ubuntu@ubuntu ~-> sudo dhclient wlan0
ubuntu@ubuntu ~-> ifconfig wlan0
wlan0   Link encap:Ethernet  HWaddr c8:3a:35:c2:86:31
        inet addr:192.168.1.120  Bcast:192.168.1.255  Mask:255.255.255.0
        inet6 addr: fe80::ca3a:35ff:fec2:8631/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:88 errors:0 dropped:0 overruns:0 frame:0
        TX packets:18 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:17265 (17.2 KB)  TX bytes:2869 (2.8 KB)

ubuntu@ubuntu ~->

```

Рисунок 3.5 - Налаштування бездротових мереж

3.4 Налаштування віддаленого доступу до Raspberry Pi

SSH (Secure Shell) — це протокол для встановлення мережевого з'єднання між двома віддаленими пристроями, який дозволяє передавати різноманітну інформацію, таку як:

- файли;
- відео та аудіо в реальному часі;
- команди.

SSH має кілька важливих переваг. По-перше, він підтримує стискання даних в процесі передачі. По-друге, забезпечує створення зашифрованого тунелю для захисту даних.

Для налаштування SSH на міні-ПК Raspberry Pi необхідно виконати кілька кроків. Спочатку слід запустити міні-ПК і відкрити термінал. Далі потрібно ввести команду:

```
sudo raspi-config
```

Після натискання Enter з'явиться інтерфейс утиліти конфігурації Raspberry Pi, де необхідно вибрати пункт Interfacing Options (рис. 3.6).

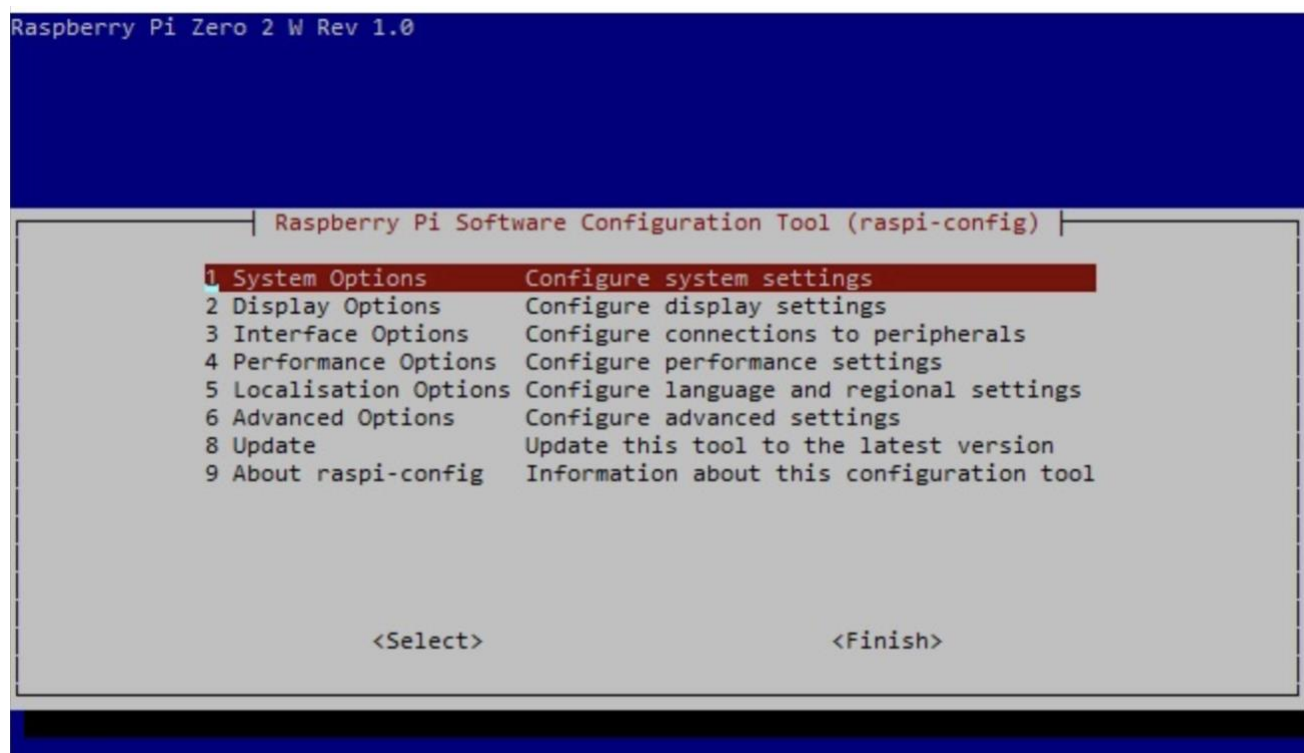


Рисунок 3.6 - Налаштування конфігурації Raspberry

Тепер потрібно увімкнути SSH на міні-ПК Raspberry Pi. Для цього необхідно вибрати пункт SSH в меню, натиснути Enter, вибрати Enable, а потім знову натискати Enter. Після цього можна повернутися до головного екрану і вибрати Finish. Всі зміни будуть застосовані.

Далі розглянемо, як підключитися до Raspberry Pi через SSH на платформі Windows.

Перш за все, вам знадобиться спеціальне програмне забезпечення, оскільки Windows не має вбудованої програми для роботи з протоколом SSH. Одним із найбільш популярних і безкоштовних SSH-клієнтів є PuTTY.

Щоб підключитися до Raspberry Pi через PuTTY, виконайте такі кроки:

а) відкрийте вікно налаштувань PuTTY (рис. 3.7);

б) на вкладці Session вкажіть наступні параметри:

1) у поле Host Name (or IP address) введіть IP-адресу вашого Raspberry Pi;

2) у поле Port введіть номер порту (за умовчанням для SSH це порт 22).

в) в розділі Connection type виберіть SSH.

Після цього натисніть Open для встановлення з'єднання (рис.3.7)..

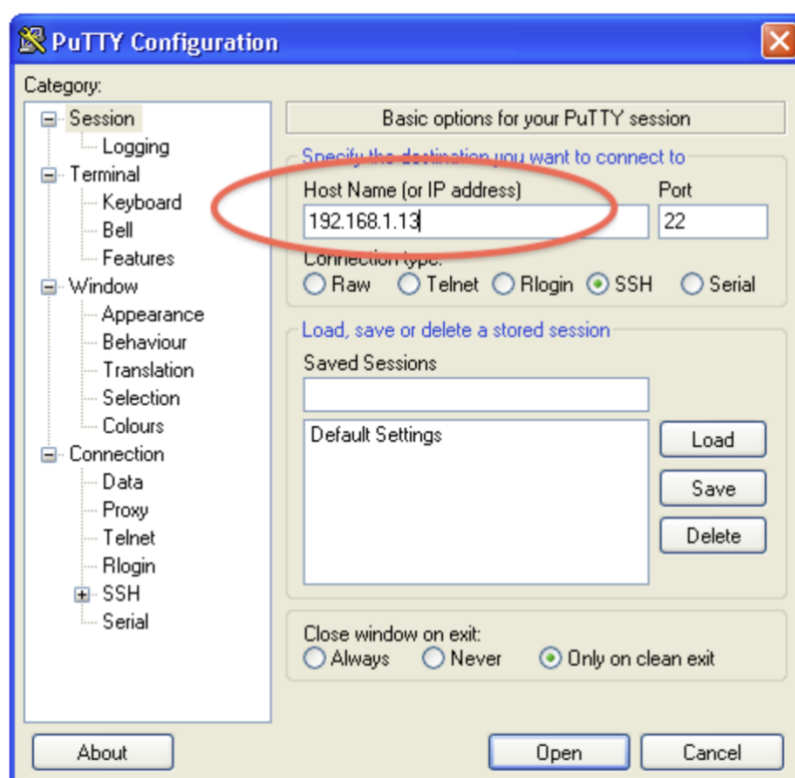


Рисунок 3.7 - Доступ з налаштуванням PuTTY

Коли всі попередні налаштування завершено, необхідно натиснути кнопку "З'єднатися" в програмі PuTTY. Якщо правильно вказані адреса, порт і тип з'єднання, а також належна працездатність мережі на обох пристроях, через кілька секунд з'явиться запит командного рядка Windows. Через цей інтерфейс ви

зможете взаємодіяти з міні-ПК Raspberry Pi, і всі доступні команди стануть доступними для використання.

Після відкриття консолі необхідно пройти авторизацію в системі. Для цього вам потрібно ввести логін (ім'я користувача) та пароль, які були встановлені для відповідного користувача. Якщо введені дані правильні, система надасть повний доступ до Raspberry Pi.

3.5 Налаштування конфігурації Domoticz

Domoticz — це легка система для домашньої автоматизації, яка дає можливість керувати різними пристроями, такими як освітлення, вимикачі, датчики та лічильники, зокрема для вимірювання температури, опадів, швидкості вітру, ультрафіолетового (УФ) випромінювання, а також для моніторингу споживання електроенергії, газу, води та багато іншого. Система дозволяє надсилати повідомлення або оповіщення на мобільні пристрої або електронну пошту[20].

Domoticz підтримує різні операційні системи, такі як Raspberry Pi, Windows, Linux, Mac OS X та вбудовані пристрої. Вона має власний мобільний додаток та інтерфейс у вигляді масштабованого веб-інтерфейсу HTML5, що автоматично адаптується під настільні та мобільні пристрої. Система сумісна з усіма сучасними браузерами.

Для встановлення Domoticz на систему, зокрема на Raspberry Pi з операційною системою Raspbian, необхідно ввести наступну команду в терміналі:

```
sudo curl -L install.domoticz.com | sudo bash
```

Після цього на екрані з'явиться виведення, подібне до того, що показано на рисунку 3.8.

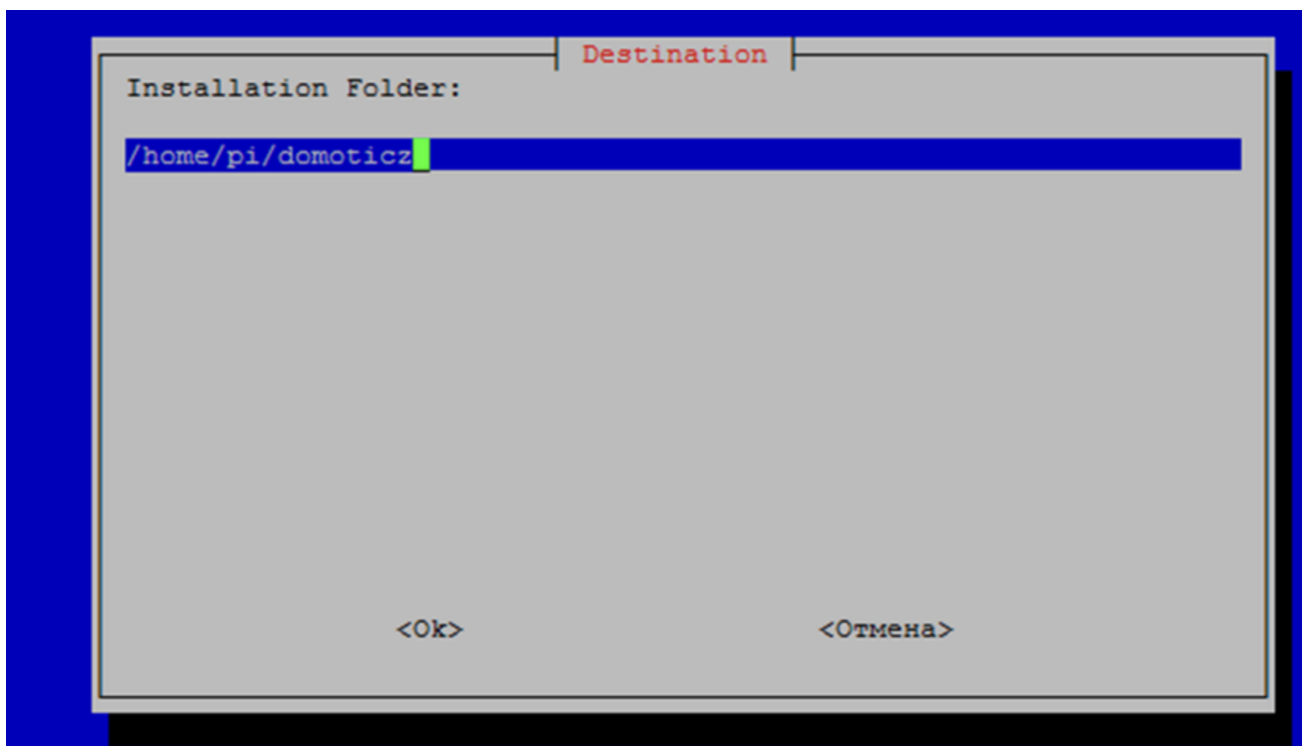


Рисунок 3.8 - Завершення установки системи Domoticz

Після завантаження всіх необхідних компонентів, з'явиться допомога по установці Domoticz, де рекомендовано залишити значення за замовчуванням або внести зміни в наступні параметри:

- протокол (HTTP/HTTPS);
- порт для системи (за замовчуванням 8080);
- шлях до каталогу для Domoticz (за замовчуванням /home/pi/domoticz).

Для доступу до Domoticz після встановлення потрібно ввести в браузері наступну адресу:

`http://IP_адреса_Raspberry:8080`

Після цього з'явиться інтерфейс Domoticz (рис. 3.9). Установка завершена. Domoticz буде додано до автозапуску, тому після наступних перезавантажень система автоматично запуститься.

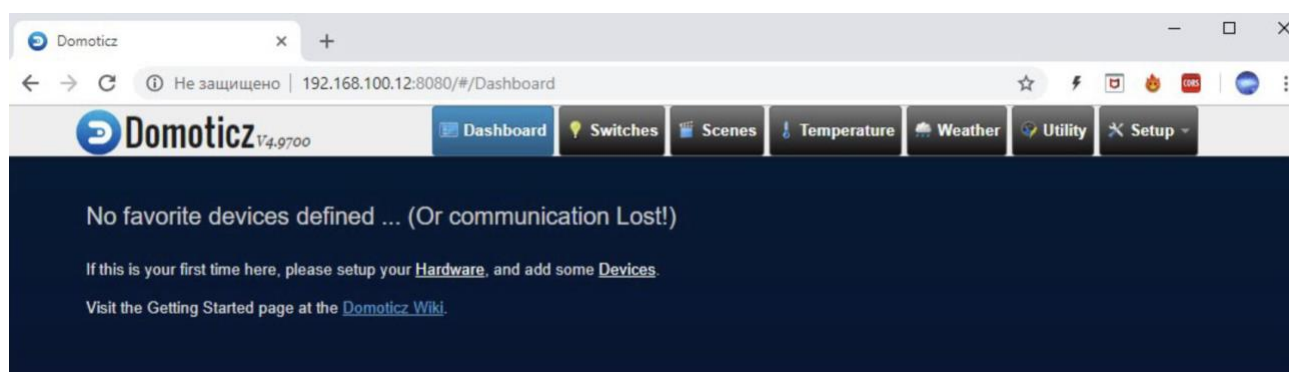


Рисунок 3.9 - Початкова сторінка Domoticz

3.6 Підключення апаратного забезпечення у системі

Після налаштування системи Domoticz, наступним кроком є підключення датчиків до системи. Для цього було обрано два датчики температури DS18B20, які підключаються через інтерфейс 1-Wire, датчик температури та атмосферного тиску BMP180, а також датчик освітленості TSL2561, який використовує інтерфейс I2C [21].

Процес підключення здійснюється через веб-інтерфейс. Для цього потрібно перейти на вкладку Setup і вибрати Hardware. У вікні, що відкрилося, з'явиться список вже підключених пристроїв, а також можливість додавати нові (рис. 3.10).

Для додавання нового пристрою, наприклад, датчика температури DS18B20, необхідно вказати його ім'я — DS1820 #1, вибрати тип інтерфейсу 1-Wire (System), а решту налаштувань залишити за замовчуванням. Після цього, для збереження змін і додавання пристрою, потрібно натиснути кнопку Add. Аналогічним чином можна підключити і налаштувати другий датчик.

Далі потрібно налаштувати датчик температури та атмосферного тиску BMP180. Для цього у пункті Type слід вибрати I2C sensors, а в вкладці SubType обрати I2C sensors BPM180/180 Temp+Baro.

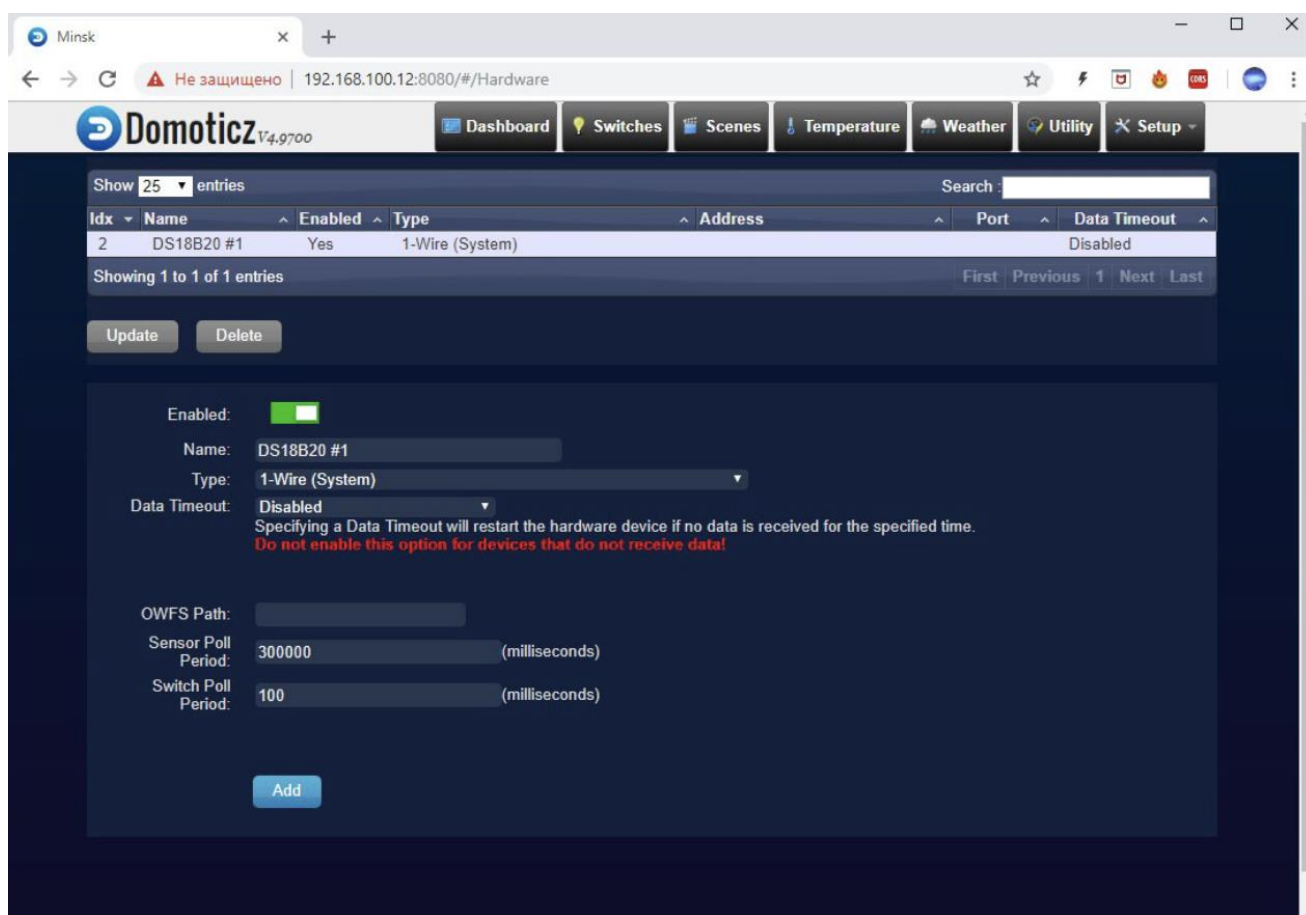


Рисунок 3.10 - Список під'єднаних пристроїв

Аналогічним чином додається датчик освітлення TSL2561. У цьому випадку в полі SubType потрібно вибрати I2C sensor TSL2561 Illuminance (рис. 3.11).

Таким чином, після правильного вибору типу та підтипу інтерфейсу, кожен датчик буде додано до системи, і всі необхідні параметри будуть налаштовані для подальшого використання.

Після підключення датчиків вони знаходяться в неактивному стані, тому для їх активації необхідно виконати кілька додаткових кроків. Спочатку потрібно перейти в меню setup, обрати пункт settings та активувати датчики.

У системі автоматично відбувається угруповання датчиків у відповідні категорії. Наприклад, датчики температури потрапляють у розділ temperature sensors, а датчик температури та атмосферного тиску буде відображатися в цій категорії лише для температури, в той час як показник тиску буде розміщений в іншому розділі – weather sensors.

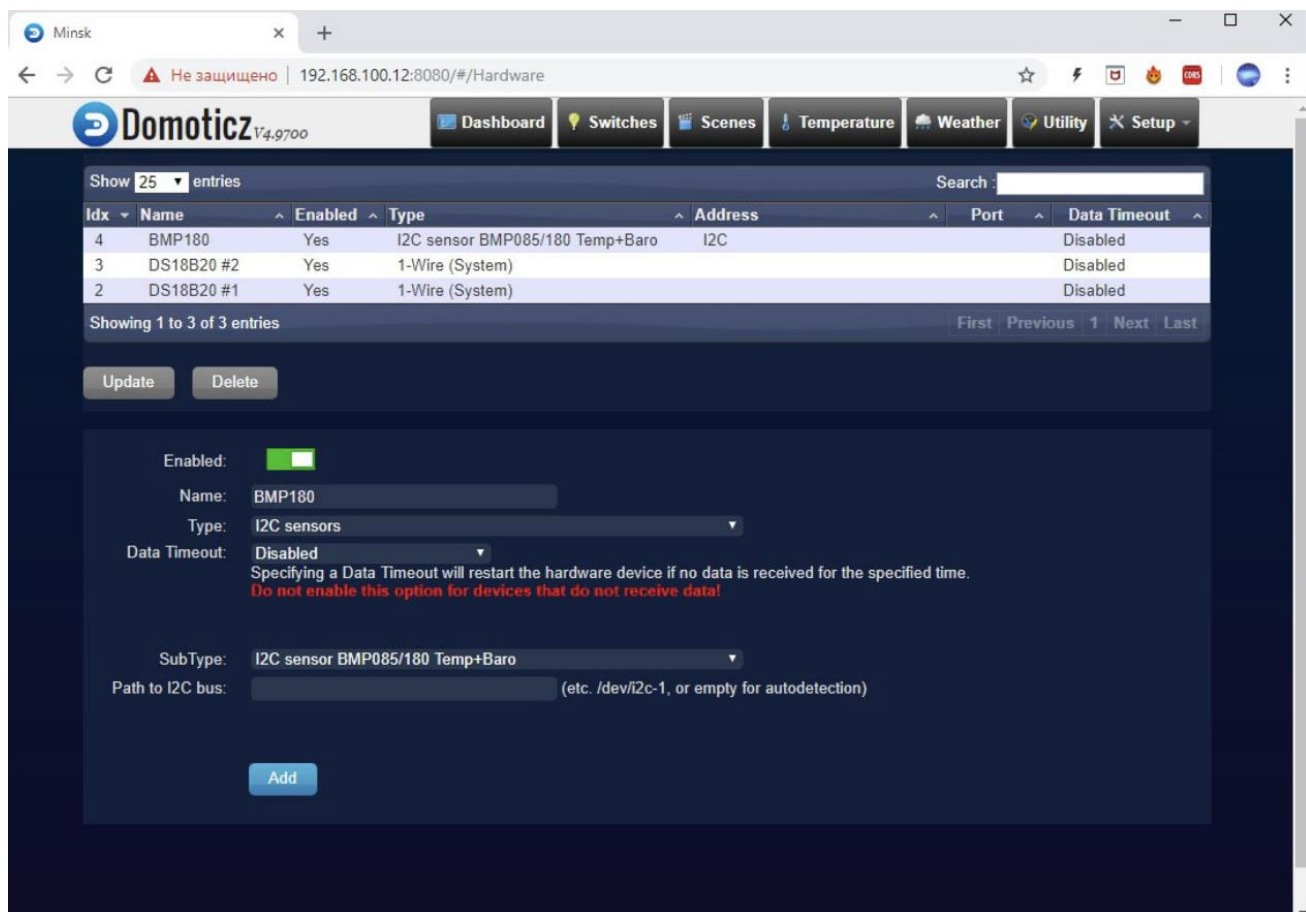


Рисунок 3.11 - Додавання датчика атмосферного тиску та температури ВРМ180

Датчик освітлення групується в розділ utility sensors. Крім того, в системі є можливість додавати часто використовувані датчики до розділу «обране». Після цього вибрані пристрої з'являться на дашборді, що дозволяє швидко переглядати їх стан (рис. 3.12).

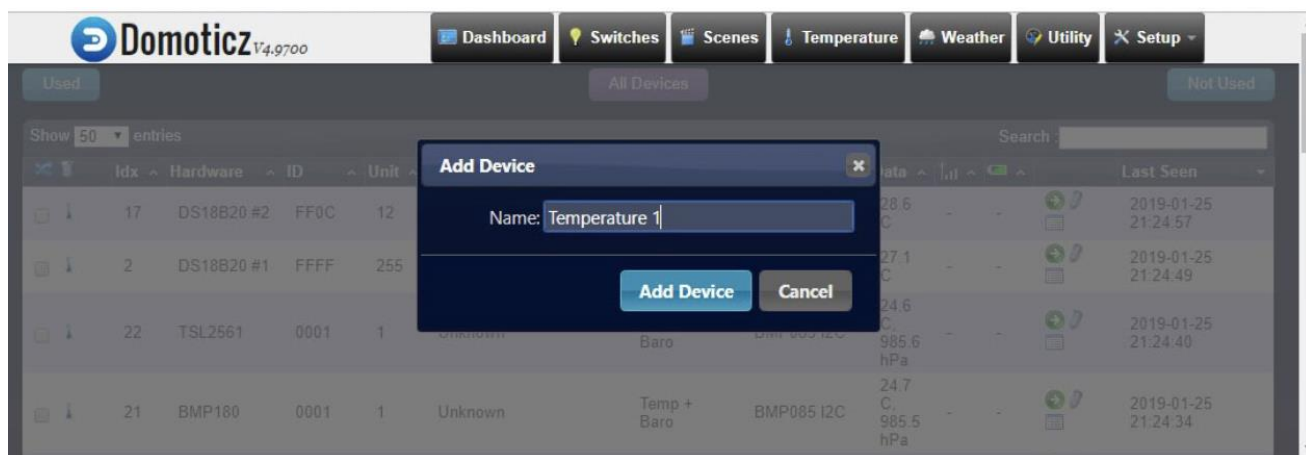


Рисунок 3.12 - Панель додавання пристрою

Система Domoticz має можливість відображати динаміку змін показань датчиків у графічному вигляді, що дозволяє зручніше відстежувати зміни параметрів. Наприклад, на рисунку 3.13 можна побачити, як змінювалась температура протягом певного проміжку часу, що дає змогу аналізувати дані за певний період і оцінювати тенденції змін температури. Це особливо корисно для моніторингу середовища та виявлення аномальних змін, які можуть потребувати уваги.

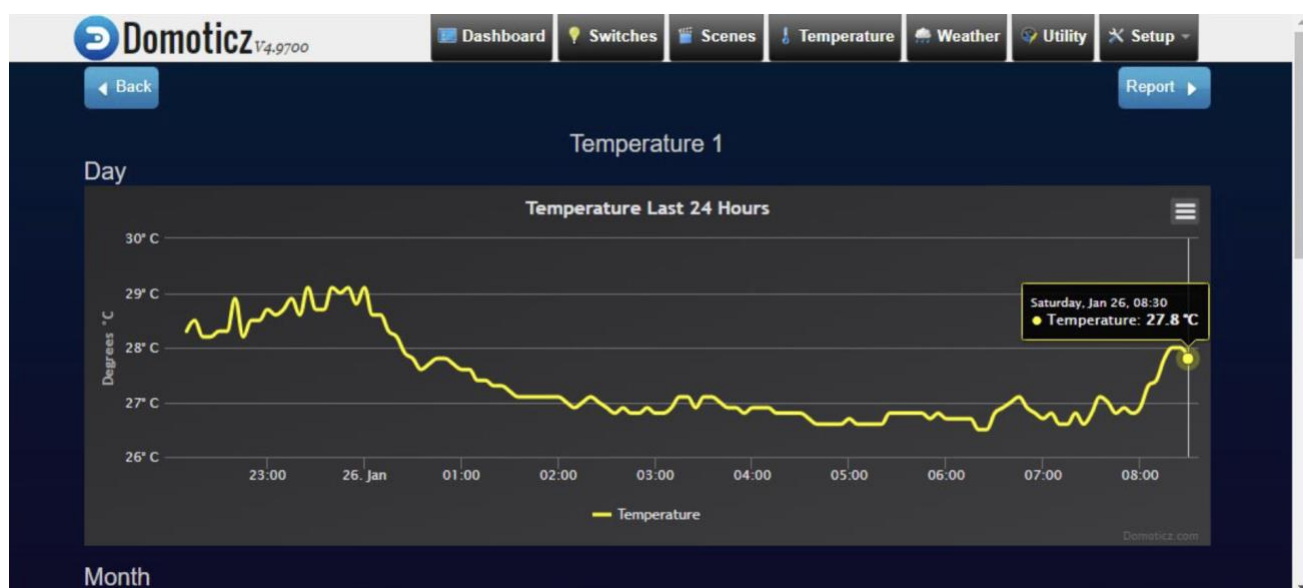


Рисунок 3.13 - Динаміка зміни температури у датчика *DS18B20*

3.7 Налаштування API Dpmpticz

Domoticz надає можливість взаємодіяти з усіма підключеними пристроями, включаючи перемикачі та датчики, як через інтерактивний веб-інтерфейс, так і за допомогою програмних сценаріїв, зокрема через JSON запити. Це дає змогу автоматизувати управління пристроями і створювати складні сценарії для домашньої автоматизації [21].

Запити до системи Domoticz мають такий формат:

`http://<username:password@>domoticz-ip<:port>/json.htm?api-call,`

де `<username:password@>` - це логін і пароль для доступу до Domoticz;
`domoticz-ip` - IP-адреса або ім'я хосту вашої інсталяції Domoticz;
`<:port>` - порт вашої інсталяції Domoticz (якщо використовуєте порт за замовчуванням, то його можна не вказувати).

Цей формат запитів дозволяє вам здійснювати такі операції, як:

- перевірка статусу пристроїв;
- вмикання та вимикання пристроїв;
- створення власних сценаріїв автоматизації.

Приклад сценарію для вмикання лампи:

```
/json.htm?type=command&param=switchlight&idx=99&switchcmd=On
```

У цьому прикладі:

- `idx=99` - ідентифікатор вашого пристрою (в даному випадку лампи);
- `switchcmd=On` - команда для ввімкнення пристрою.

У випадку успішного виконання запиту, система поверне відповідь у вигляді:

```
{
  "status": "OK",
  "title": "SwitchLight"
}
```

Приклади сценаріїв для оновлення показань датчиків:

- датчик температури

```
/json.htm?type=command&param=udevice&idx=IDX&nvalue=0&svalue=TEMP,
```

де `IDX` — ID вашого пристрою;

`TEMP` — значення температури;

- датчик атмосферного тиску та температури

```
/json.htm?type=command&param=udevice&idx=IDX&nvalue=0&svalue=TEMP;BAR;BAR_FOR;ALTITUDE,
```

де `IDX` - ID вашого пристрою;

`TEMP` – температура;

`BAR` - барометричний тиск;

`BAR_FOR` - прогноз для барометра (може бути одне з: 0 - немає інформації, 1 - сонячно, 2 - похмуро, 3 - хмарно, 4 - дощ);

ALTITUDE - висота (не використовується в даному випадку, можна поставити 0);

- датчик освітленості

/json.htm?type=command¶m=udevice&idx=IDX&svalue=VALUE

де IDX - ID пристрою;

VALUE - значення освітленості в люксах (lux).

Ці сценарії дозволяють вам інтегрувати зовнішні пристрої з вашим домом, створюючи розумну систему автоматизації, що реагує на зміни в реальному часі.

3.8 Висновки до розділу

У цьому розділі було здійснено налаштування засобів моніторингу для створення системи автоматизації та віддаленого доступу на міні-ПК Raspberry Pi. Спершу було завантажено та встановлено операційну систему на міні-ПК, що є основою для подальшої роботи. Наступним кроком стало налаштування мережевих параметрів Raspberry Pi, а також забезпечення віддаленого доступу до пристрою за допомогою SSH.

Крім того, важливим етапом стало встановлення та конфігурація системи Domoticz. Це легка система домашньої автоматизації, яка дозволяє контролювати та налаштовувати різноманітні пристрої, включаючи освітлення, вимикачі, датчики та лічильники, такі як температура, опади, вітер, ультрафіолетове випромінювання, а також споживання енергії, газу та води.

Ці кроки є невід'ємною частиною дослідження, яке спрямоване на виявлення мінімальних необхідних засобів та кроків для побудови власної системи управління та моніторингу елементами IoT. Тому налаштування мережі та автоматизації, описані вище, є важливими етапами в розробці ефективної та надійної системи для контролю за різними параметрами та пристроями в межах домашньої чи іншої автоматизованої інфраструктури.

4 ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

4.1 Архітектура системи та інструменти розробки системи

У результаті аналізу архітектури IoT-мережі, було обрано архітектуру клієнт-сервер, що є найбільш підходящою для подальшої розробки системи управління та моніторингу. У такій архітектурі клієнт є додатком, встановленим на користувацькому пристрої, а сервер може бути розміщений на віддаленій машині або хостингу. Перевага цієї архітектури полягає в тому, що сервер працює автономно і не залежить від клієнта, в той час як клієнт може зручно підключатися до сервера і не залежить від інших клієнтів.

Для розробки додатку для управління цією системою було обрано мову програмування Java та фреймворк Spring Boot [22, 23].

Причини вибору Java:

- Java Virtual Machine (JVM) дозволяє писати код, який буде працювати на різних платформах, що значно підвищує швидкість розробки і знижує її вартість.
- простіший об'єктно-орієнтований синтаксис;
- безпечна робота з пам'яттю і висока надійність;
- незалежність від платформ і архітектури;
- висока продуктивність.

Spring Boot був обраний через його здатність значно спростити розробку додатків на Java. Це дає змогу швидко створювати високопродуктивні Spring-додатки без потреби в додатковій конфігурації. Завдяки цьому розробники можуть зосередитися на бізнес-логіці, замість налаштувань середовища.

Можливості Spring Boot:

- легке створення самостійно працюючих додатків;
- вбудовані контейнери сервлетів (наприклад, Tomcat або Jetty);
- автоматична конфігурація і моніторинг;
- можливість без зайвого коду для написання XML-файлів.

Для зберігання даних обрана СУБД PostgreSQL, яка є реляційною та об'єктною системою управління базами даних. Це потужна, надійна система з відкритим кодом, що підтримує безліч типів даних і дозволяє створювати власні типи. Вона підтримує основні принципи ACID (атомарність, узгодженість, ізолюваність, стійкість), що забезпечує цілісність та безпеку даних [24].

Основні можливості PostgreSQL:

- підтримка складних запитів і підзапитів (JOIN, UNION тощо);
- реплікація;
- висока швидкість обробки запитів;
- підтримка рекурсивних запитів та загальних табличних виразів.

Для спрощення роботи з даними та зберіганням інформації було вибрано Hibernate - ORM (Object-Relational Mapping) фреймворк для Java. Hibernate дозволяє конвертувати Java-класи у таблиці в базі даних і значно спрощує розробку, оскільки багато запитів уже реалізовані в фреймворку. Також Hibernate має власну мову запитів HQL (Hibernate Query Language), яка є спрощеним, об'єктно-орієнтованим аналогом SQL.

Основні переваги Hibernate:

- автоматичне відображення класів на таблиці в базі даних;
- спрощений доступ до даних через об'єктно-орієнтовані інтерфейси;
- мінімізація необхідності написання SQL-запитів.

Для розробки веб-інтерфейсу було обрано Freemarker - шаблонізатор, який дозволяє створювати динамічні веб-сторінки без необхідності знань JavaScript або TypeScript. Freemarker надає макроси для зменшення кількості коду та повторного використання елементів. Це дозволяє швидко створювати зручний та адаптивний інтерфейс користувача.

Проаналізувавши всі перелічені інструменти можна зробити наступні висновки щодо необхідності їх обрання:

- Java та Spring Boot надають надійну основу для створення додатка для управління IoT-пристроями, з великою кількістю готових інструментів для зручності розробки;

- PostgreSQL є надійною і потужною СУБД для зберігання даних, з підтримкою складних запитів і високою продуктивністю;
- Hibernate дозволяє зручно працювати з базою даних через об'єкти і значно спрощує доступ до інформації;
- Freemarker дає можливість створювати інтерактивні веб-інтерфейси з мінімальними зусиллями.

Таким чином, використання цих технологій дозволяє створити ефективну та масштабовану систему для моніторингу та управління IoT-пристроями, яка є незалежною від платформ та може працювати на різних пристроях і операційних системах.

Для розробки додатку пропонується використання архітектурного шаблону MVC (Model-View-Controller). Цей підхід дозволяє знизити ступінь зв'язку між модулями, що забезпечує легкість у підтримці та зміні окремих компонентів системи [25].

Детальніше про основні модулі MVC:

- model - це робота з базою даних, обробка та перевірка правильності даних;
- view - компонент, що приймає вхідні дані від користувача та відправляє їх після обробки;
- controller - відповідальний за основну бізнес-логіку, обробку запитів і виконання операцій з даними.

Model інкапсулює основну логіку обробки даних і не залежить від того, як саме ці дані вводяться чи виводяться.

View може включати кілька взаємопов'язаних елементів, таких як таблиці чи поля форм, в яких відображаються дані.

Controller відповідає за відстеження подій, що виникають в результаті дій користувача, і забезпечує структурування коду, групуючи подібні функції в окремі класи. Наприклад, у типового MVC-додатку може бути контролер користувача, який містить методи для реєстрації, авторизації, редагування профілю тощо.

Коли користувач виконує дії, події реєструються і передаються в Model для обробки або в View для оновлення інформації. Відокремлення Model від View дозволяє незалежно змінювати компоненти, що відображають дані. Таким чином, якщо користувач змінює дані через контролер, зміни автоматично відображаються в інтерфейсі без необхідності додаткових налаштувань.

У клієнтському додатку пропонується реалізація монолітної архітектури, оскільки для нього немає потреби інтегрувати базу даних або веб-інтерфейс. Такий підхід обрано через простоту і відповідність завданням програми.

Основна функціональність додатку полягає у встановленні з'єднання з віддаленим сервером, його підтримці, а також у відправці та отриманні повідомлень. У зв'язку з цим використання складніших архітектурних підходів, таких як мікросервісна або MVC, є зайвим і невиправдано ускладнює розробку.

4.2 Реалізація збору даних

Основна мета додатку полягає в зборі даних з декількох серверів, на яких працює Domoticz, їх об'єднанні та впорядкуванні в одному місці, зборі статистичних даних, а також можливого здійсненні контролю.

Система Domoticz надає API, яке дозволяє отримувати певну інформацію або надсилати команди через JSON-запити. Для виконання цих запитів у додатку буде використовуватися RestTemplate - стандартний інструмент, інтегрований у Spring Boot. Цей інструмент спрощує передачу даних, позбавляючи необхідності здійснювати ручне перетворення.

Для організації запитів у додатку реалізовано клас HttpUtil, який є сервісом для відправлення запитів. Даний клас успадковується від абстрактного класу ClosableHttpUtil (рис. 4.1), що забезпечує можливість зручного управління з'єднаннями та викликами API.

```

public static HttpClientResponse executeRequest(HttpUriRequest request, CloseableHttpClient closeableHttpClient) {
    String jsonText = null;
    int statusCode;

    try (CloseableHttpResponse response = closeableHttpClient.execute(request)) {
        statusCode = response.getStatusLine().getStatusCode();
        log.debug("Status code: {}", statusCode);
        HttpEntity entity = response.getEntity();
        if (entity != null) {
            jsonText = extractJsonStringFromHttpResponse(entity);
            EntityUtils.consume(entity);
        }
    }
}

```

Рисунок 4.1 - Реалізація методу для надсилання запитів

Для розробки користувацького інтерфейсу пропонується використати шаблонізатор Freemarker, який має суттєву перевагу у вигляді підтримки макросів. Макроси дають змогу повторно використовувати вже створений код, модифікуючи лише невеликі його частини, що суттєво спрощує розробку та зменшує обсяг необхідного коду.

Контролери у додатку оголошуються за допомогою анотацій `@Controller` або `@RestController` (рис. 4.2). Шлях, який обробляє певний контролер, задається за допомогою анотації `@RequestMapping(path)`. Методи всередині контролера можуть мати власні шляхи, які додаються до основного шляху контролера.

```

@RestController
@RequiredArgsConstructor
@RequestMapping("/device")
public class DeviceController {
    private final DeviceService deviceService;

    @GetMapping("all")
    public ResponseEntity<List<Device>> getAll() {
        List<Device> all = deviceService.getAll();
        return ResponseEntity.ok(all);
    }
}

```

Рисунок 4.2 - Реалізація оголошення шляху, при запиті до HTTP метод GET

Наприклад, якщо для контролера задано шлях `"/device"` за допомогою `@RequestMapping("/device")`, а метод контролера позначено анотацією `@GetMapping("all")`, то кінцевий шлях, який буде обробляти цей метод,

виглядатиме як `"/device/all"` і відповідатиме HTTP-запиту методом GET.

Цей підхід дозволяє чітко структурувати маршрутизацію в додатку, забезпечуючи зрозумілість і гнучкість у визначенні шляхів для обробки запитів.

4.3 Налаштування системи

Для розробки користувацького інтерфейсу пропонується використати шаблонізатор Freemarker, який має суттєву перевагу у вигляді підтримки макросів. Макроси дають змогу повторно використовувати вже створений код, модифікуючи лише невеликі його частини, що суттєво спрощує розробку та зменшує обсяг необхідного коду [26].

Контролери у додатку оголошуються за допомогою анотацій `@Controller` або `@RestController` (рис. 4.3). Шлях, який обробляє певний контролер, задається за допомогою анотації `@RequestMapping(path)`. Методи всередині контролера можуть мати власні шляхи, які додаються до основного шляху контролера.

Наприклад, якщо для контролера задано шлях `"/device"` за допомогою `@RequestMapping("/device")`, а метод контролера позначено анотацією `@GetMapping("all")`, то кінцевий шлях, який буде обробляти цей метод, виглядатиме як `"/device/all"` і відповідатиме HTTP-запиту методом GET.

Цей підхід дозволяє чітко структурувати маршрутизацію в додатку, забезпечуючи зрозумілість і гнучкість у визначенні шляхів для обробки запитів.

```
datasource.url=jdbc:postgresql://localhost/iot-control-system
datasource.username=postgres
datasource.password=123
```

Рисунок 4.3 - Конфігурація для доступу до бази даних

Для налаштування клієнтського застосунку необхідно додати у файл конфігурації `application.properties` наступні параметри:

- `remote-server.ip` – IP-адреса віддаленого сервера, на який надсилатимуться дані;
- `remote-server.port` – необов'язковий параметр, який вказується лише у випадку, якщо віддалений сервер використовує порт, відмінний від стандартного 8080;
- `remote-server.key` – ключ безпеки, без якого сервер відхиляє всі запити;
- `remote-server.local-id` – унікальний ідентифікатор клієнта;
- `domoticz.login` – логін для доступу до системи Domoticz;
- `domoticz.password` – пароль для підключення до Domoticz;
- `domoticz.port` – необов'язковий параметр, який вказується лише, якщо Domoticz використовує порт, відмінний від стандартного 8080.

Процес запуску клієнтського застосунку:

- після завершення налаштувань потрібно запустити клієнтську частину;
- додаток автоматично встановить з'єднання з Domoticz, знайде всі підключені пристрої, а за успішного виконання цих операцій – підключиться до віддаленого сервера;
- у разі недоступності віддаленого сервера програма перейде в режим очікування. Вона періодично перевірятиме можливість відновлення з'єднання і продовжить роботу після успішного підключення.

4.4 Результати роботи

Інструкція з встановлення системи:

а) підготовка до встановлення:

- 1) переконайтесь, що на машині встановлена Java Virtual Machine (JVM) версії не нижче 8-ї;

б) встановлення клієнтського додатку:

1) скопіюйте файл IoT-control-system-client.jar на пристрій із встановленим Domoticz;

2) запустіть клієнтський додаток командою:

```
- java -jar IoT-control-system-client.jar
```

в) встановлення серверного додатку:

1) підготуйте базу даних із попередньо створеним користувачем;

2) скопіюйте файл IoT-control-system-server.jar на сервер;

3) запустіть серверний додаток командою:

```
- java -jar IoT-control-system-server.jar
```

г) використання системи:

1) система доступна як із персональних комп'ютерів, так і з мобільних пристроїв, основна вимога – підтримка браузера;

2) для початку роботи перейдіть за адресою:

```
- https://<ip-адреса-сервера>:8080
```

3) у браузері відкриється вікно входу в систему (рис. 4.4).

Тепер система готова до роботи.



Рисунок 4.4 - Форма входу в систему

Під час першого запуску слід увійти до системи, використовуючи стандартні облікові дані “admin:admin”. Після авторизації потрібно змінити пароль адміністратора, а за потреби - додати нових користувачів. У системі передбачена панель керування користувачами, яка дозволяє редагувати

інформацію та права доступу існуючих користувачів, а також створювати нові облікові записи.

Для підключення клієнтських додатків до системи необхідно спочатку створити клієнта в панелі створення клієнтів (рис. 4.5). У цьому процесі потрібно вказати назву клієнта, яка буде використовуватися як ідентифікатор у системі, та згенерувати ключ безпеки. Після додавання клієнта його слід налаштувати й активувати. Під час налаштування необхідно використовувати отримані ідентифікатор і ключ.

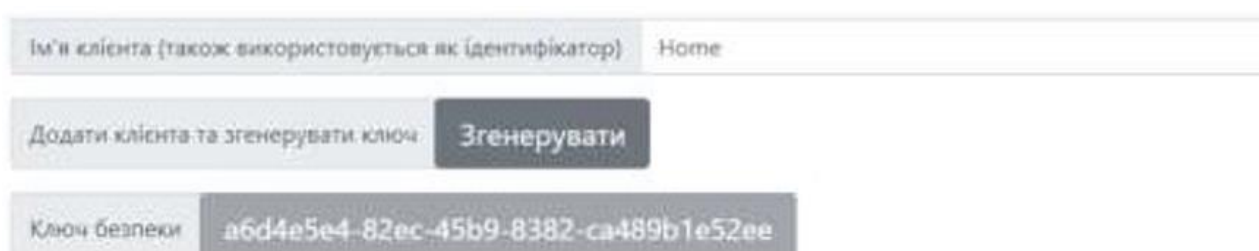


Рисунок 4.5 - Етап додавання нового клієнта

Коли підключення буде успішним, клієнт з'явиться у списку зі статусом "активний" (рис. 4.6). У випадку, якщо підключення ще не здійснено, клієнт матиме статус "новий". Якщо з'єднання було встановлено, але згодом перервалося, або якщо пристрій відключений, статус клієнта зміниться на "неактивний". Для отримання докладної інформації про клієнта потрібно натиснути на нього. Відкриється вікно з детальною інформацією, зокрема: назва пристрою, його IP-адреса, тривалість роботи, кількість підключених пристроїв із можливістю перегляду даних про кожен з них окремо, а також інші корисні відомості.

У вкладці "Домашня" відображаються всі пристрої, які на даний момент підключені до системи. Є можливість сортування пристроїв за клієнтами або типами, наприклад, датчики температури чи лампочки.



Рисунок 4.6 - Опис записів

Також з домашньої сторінки можна перейти до детального огляду пристроїв. У цьому розділі є можливість відключити певний пристрій. Відключені пристрої не відображаються на домашній сторінці, і для їх повторного включення необхідно зайти в налаштування клієнта.

При натисканні на посилання "Детальніше" для конкретного пристрою відкривається вікно з його детальною інформацією, зокрема назва пристрою, його показники, а також список доступних команд. Наприклад, якщо це датчик температури, можна переглянути графік змін температури за останні 24 години (рис. 4.7).

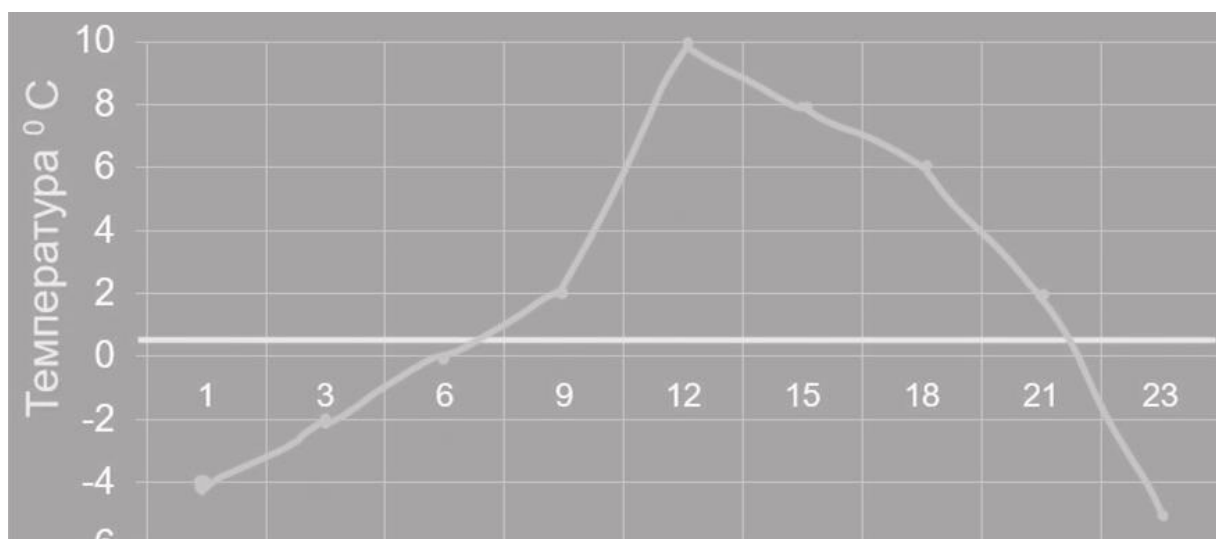


Рисунок 4.7 - Інформація про графік зміни температури

4.5 Безпека системи

Ключовим аспектом Інтернету речей (IoT) є кібербезпека. Для моніторингу IoT багато держав запроваджують спеціальні правила та норми використання таких пристроїв.

IoT спрощує наше життя, надаючи можливість швидко знаходити людину чи об'єкт. Проте, без належного рівня безпеки та конфіденційності даних користувачів, ця технологія може не отримати масового визнання. Для широкого впровадження IoT необхідно забезпечити потужну інфраструктуру захисту [2].

Як кіберфізична система, IoT-мережа стикається з низкою загроз:

- підміна особистості (імперсонація) - вид атаки, при якій зловмисник імітує легітимний пристрій, використовуючи його ідентифікаційні дані. Наприклад, зловмисник може підробити дані розумного лічильника для маніпуляції інформацією про споживання енергії;

- несанкціоноване вилучення інформації - через використання спільної інфраструктури зв'язку зловмисник може отримати доступ до переданих даних між IoT-пристроями;

- втручання у дані - зловмисник може змінювати дані обміну, наприклад, змінювати ціни в пікові періоди, що може призвести до збільшення витрат енергії для домогосподарств та перенавантаження енергосистеми;

- Авторизація та контроль доступу - дистанційне управління деякими пристроями (наприклад, розумними лічильниками чи сенсорами на підстанціях) створює ризик, що зловмисник або незадоволений працівник отримає несанкціонований доступ для пошкодження фізичних об'єктів або спричинення перебоїв у подачі електроенергії;

- проблеми конфіденційності - розумні лічильники та побутові прилади можуть збирати дані, які порушують приватність користувачів, наприклад, інформацію про їхні звички, час перебування вдома чи у відпустці тощо;

- компрометація та шкідливий код - пристрої IoT вразливі до фізичного чи віддаленого втручання, а також до зараження шкідливим програмним забезпеченням, що дозволяє зловмиснику контролювати їхню роботу, особливо ризикованими є пристрої з обмеженими ресурсами, які легко піддаються компрометації;

- DoS-атаки - нападник може вивести з ладу частину мережі, скориставшись протоколами IP, які використовують більшість пристроїв IoT;

- кібератаки - зловмисники можуть використати цю взаємодію для здійснення атак.

Небезпека зчитування RFID-міток. RFID-мітки, що містять ідентифікаційну інформацію, можуть бути прочитані, змінені або знищені зловмисниками. Це створює ризики розголошення конфіденційної інформації про користувача. Серед поширених атак на RFID - використання RFID-вірусів, злом через мобільні пристрої та маніпуляції SpeedPass.

Атаки на вузли датчиків WSN. Мережі датчиків вразливі до таких атак, як:

- забивання каналів, що порушує роботу мережі;
- втручання, при якому дані з вузлів можуть бути вилучені або змінені;
- атака Сібілли, яка полягає у створенні фальшивих ідентифікаторів для дестабілізації системи;

- заповнення каналів, що призводить до витрати пам'яті через збільшення обсягу трафіку.

Зловживання хмарними обчисленнями. Хмарні обчислення, що забезпечують розподіл ресурсів, є об'єктом атак, таких як:

- шкідливий інсайдер, який може маніпулювати даними;
- втрата даних, коли зловмисник видаляє або змінює інформацію;
- man-in-the-middle, коли нападник перехоплює або змінює повідомлення;
- зловмисне використання ресурсів, наприклад, через завантаження шкідливих програм, які надають зловмиснику контроль над пристроями.

Виклики для забезпечення безпеки IoT. Розробка безпечних IoT-мереж стикається з низкою завдань, серед яких:

- масштабованість, враховуючи величезну кількість пристроїв;
- мобільність об'єктів, які потребують захисту в динамічному середовищі;
- розгортання мереж на великих територіях, що робить пристрої вразливими до фізичного втручання;
- інтеграція старих систем, які не підтримують сучасні засоби безпеки;
- обмежені ресурси деяких пристроїв, що потребує оптимізованих рішень;
- неоднорідність протоколів, які використовують різні пристрої;
- проблеми сумісності між пристроями, що працюють на різних протоколах;
- початкове налаштування мільйонів пристроїв з безпековими матеріалами (ключами, алгоритмами тощо);
- управління довірою між пристроями, якими керують різні організації;
- забезпечення реального часу для систем, таких як SCADA, де затримки можуть призвести до аварій.

Задля цього у 2019 році в світі було прийнято законопроект, який зобов'язує розробників IoT-пристроїв передавати владі інформацію про прошивки, потенційні ризики та способи їхнього усунення. Пристрої, що не відповідають цим вимогам, будуть заборонені. У Великій Британії також запроваджено закон, спрямований на підвищення безпеки користувачів IoT.

Ці заходи підтверджують, що питання моніторингу та безпеки IoT залишатиметься у пріоритеті, оскільки захист громадян є важливою державною задачею.

4.2 Висновки до розділу

У цьому розділі було проведено аналіз архітектури мережі IoT і вибрано відповідну архітектуру для подальшої розробки системи.

На основі проведеного дослідження було визначено, що найбільш

доцільною для розробки є архітектура клієнт-сервер. Такий підхід дозволяє забезпечити централізоване керування системою, ефективний обмін даними між компонентами мережі IoT та підтримку масштабованості.

Для реалізації серверного додатку було обрано мову програмування Java, яка має такі переваги:

- простий та об'єктно-орієнтований синтаксис, що прискорює розробку;
- безпечність та безвідмовність у роботі з пам'яттю, що особливо важливо для мереж IoT, де надійність є критичним фактором;
- незалежність від платформи завдяки використанню JVM (Java Virtual Machine);
- висока продуктивність, що дозволяє обробляти великі обсяги даних у реальному часі.

Для спрощення та прискорення розробки додатка на Java було обрано Spring Boot Framework, який дозволяє:

- легко створювати продуктивні серверні додатки;
- використовувати вбудовані механізми конфігурації, що зменшує час налаштування системи;
- забезпечити зручну інтеграцію з іншими технологіями та бібліотеками.

На основі вибраних технологій та архітектурних підходів була реалізована система, яка демонструє свою функціональність. У підсумку розділу представлено роботу системи, яка показує ефективність обраного підходу, а також можливості керування та моніторингу IoT-пристроїв за допомогою клієнтського та серверного додатків.

ВИСНОВКИ

В роботі проведено аналіз мережі IoT, визначено типи використовуваних протоколів, технологій та архітектури мереж, а також пристроїв, що беруть участь у роботі таких систем. Розглянуто завдання моніторингу та його важливість для ефективного контролю мереж, а також основні проблеми кібербезпеки в контексті IoT. Це дозволяє обрати необхідні протоколи і технології для побудови майбутньої системи.

У роботі запропоновано систему моніторингу та управління на основі Raspberry Pi. Ця компактна мережа дозволяє контролювати стан датчиків через Інтернет, з оновленням інформації на веб-сервері, і підтримує високий рівень захисту. Користувач може взаємодіяти із системою в реальному часі. Система забезпечує енергоефективність, інтелектуальні функції, низьку вартість та високу продуктивність.

Аналіз апаратного забезпечення показав, що для забезпечення роботи системи управління та моніторингу необхідно вибрати відповідні пристрої. Як центральний компонент системи було обрано міні-ПК Raspberry Pi, який є економічним і портативним рішенням для побудови систем IoT.

Що стосується налаштування засобів моніторингу, було вибрано систему Domoticz для домашньої автоматизації, яка дозволяє контролювати різні пристрої, такі як світло, датчики температури, вологості, споживання електроенергії та інше. Це рішення має велике практичне значення для побудови ефективної та зручної системи управління.

Також було визначено, що для подальшого розвитку найбільше підходить архітектура клієнт-сервер, а для серверної частини було обрано мову програмування Java, яка має об'єктно-орієнтований синтаксис, високий рівень безпеки, незалежність від платформи та архітектури. Пропоновано серверний додаток за архітектурним шаблоном MVC та монолітну архітектуру для клієнтського додатка, а також розробку користувацького інтерфейсу.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Management information systems for the information age Stephen Haag, Maeve Cummings, Amy Phillips, McGraw-Hill/Irwin, Boston, 2020 – 672 p.
2. Методи аналізу даних. URL: http://web.kpi.kharkov.ua/auts/wpcontent/uploads/sites/67/2017/02/DAMAP_Ivashko_posobie2.pdf (дата звернення 13.10.2024).
3. The Universal Parser Compiler and Its Application to a Speech Translation System Masaru Tomita, Marion Kee, Hiroaki Saito, Teruko Mitamura and Hideto Tomabechi. URL: <https://aclanthology.org/1988.tmi-1.9/> (Last accessed: 09.10.2024).
4. ZigBee® Network Protocols and Applications. Taylor & Francis, 2019 - 378 p.
5. E. Turban Information Technology for Management: Advancing Sustainable, Profitable Business Growth Paperback – Realtime Publishers, 2020 – 92p.
6. The Universal Parser Compiler and Its Application to a Speech Translation System Masaru Tomita, Marion Kee, Hiroaki Saito, Teruko Mitamura and Hideto Tomabechi. URL: <https://aclanthology.org/1988.tmi-1.9/> (Last accessed: 03.10.2024).
7. Xhafa L. Modeling and Processing for Next-Generation Big-Data Technologies / L.Xhafa, L. Barolli, A. Barolli, P. Papajorgji. – Springer, 2016. – 473–497 pp.
8. Internet of Things for Industry and Human Application. In Volumes 1-3. Volume 1. Fundamentals and Technologies /V. S. Kharchenko (ed.) - Ministry of Education and Science of Ukraine, National Aerospace University KhAI, 2019 - 605p.
9. Vorobets H. I., Kharchenko V. S., Kudermetov R. K., Klyatchenko Ya., M., Horditsa V. E., Pshenychnyi O. O., Khamula I. S., Lobachev I. M., Lobachev M. V., Tiahunova M. Y., Polska O. V. Internet of Things Technologies for Cyber Physical Systems: Practicum / Vorobets H. I. and Kharchenko V. S. (Eds.) – Ministry of Education and Science of Ukraine, Yuriy Fedkovych Chernivtsi National University, National Aerospace University “KhAI”, Zaporizhzhia National Technical University,

2019. – 172 p.

10. Internet of Things for Industry and Human Application. Fundamentals of Internet of Things /V.S. Kharchenko (ed.). – Ministry of Education and Science of Ukraine, National Aerospace University KhAI, 2019. – 95 p.

11. Steffen Schenkluhn. Device and event management as foundation of IoT applications URL: <https://blog.bosch-si.com/internetofthings/device-and-event-management-as-foundation-of-iot-applications/> (Last accessed 11.11.2024).

12. Skarga-Bandurova I.S., Gorbenko A.V., Biloborodova T.O., Koval V.S., Sachenko A.O., Tarasyuk O. M. Data Science for Internet of Things and Internet of Everything: Practicum / Skarga-Bandurova I.S. and Gorbenko A.V. Ministry of Education and Science of Ukraine, Volodymyr Dahl East Ukrainian National University, National Aerospace University “Kharkiv Aviation Institute”, Ternopil National Economic University, 2019. – 169 p.

13. The Official Raspberry Pi Handbook 2025: Astounding projects with Raspberry Pi computers. Raspberry Pi Press.2024.. P. 200.

14. С. Могильний. Мікрокомп'ютер Raspberry Pi - інструмент дослідника. 2014. 540 с.

15. Simon Monk. Raspberry Pi Cookbook: Software and Hardware Problems and Solutions 4th Edition. O'Reilly Media. 2023. P.613.

16. Майк Кук, Шон МакМанус. Raspberry Pi For Dummies. Wiley. John Wiley & Sons, LTD. 2021. P.512.

17. Брок Крафт, Джонатан Еванс, Майк Кук. Raspberry Pi Projects For Dummies. 2015. P. 512.

18. Роберт М. Корецьки. Raspberry Pi OS Text Editors, git, and LXC: A Practical Approach. Taylor & Francis. 2024. P. 312.

19. Domoticz. URL: <https://sprut.ai/article/domoticz-ustanovka> (Last accessed 23.11.2024).

20. Domoticz Home Automation Lite. URL: <https://play.google.com/store/apps/details?id=nl.hnogames.domoticz&hl=uk> (Last accessed 29.10.2024).

21. Domoticz api/json url's URL: https://www.domoticz.com/wiki/domoticz_api/json_url%27s (Last accessed 27.11.2024).
22. JVM, JRE та JDK. URL: <https://w3schoolsua.github.io/hyperskill/jvm-jre-jdk.html#gsc.tab=0> (Last accessed 31.10.2024).
23. Christian Ullenboom. Spring Boot 3 and Spring Framework 6 First Edition. Rheinwerk Computing. 2023. P. 934
24. Luca Ferrari, Enrico Pirozzi. Learn PostgreSQL - Second Edition: Use, manage and build secure and scalable databases with PostgreSQL 16 2nd ed. Edition. Packt Publishing. 2023. P.744.
25. Колін Кемпбелл. Model-Based Software Testing and Analysis with C#. Cambridge University Press. 2017. P.368.
26. Domoticz.Application Settings. URL: https://wiki.domoticz.com/Application_Settings#Log_History (Last accessed 08.11.2024).