ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ АВТОМАТИЗОВАНИХ СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Застосування IoT для моніторингу та контролю якості харчових продуктів під час транспортування та зберігання»

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

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

_____ Олександр БОВКУН
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконав: здобувач вищої освіти гр. ІСД- 42
Олександр БОВКУН
Ім'я, ПРІЗВИЩЕ

Керівник: *PhD* Владислав ХОМЕНЧУК
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ

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

Київ 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра Інженерії програмного забезпечення автоматизованих систем
Ступінь вищої освіти Бакалавр
Спеціальність 126 Інформаційні системи та технології
Освітньо-професійна програма Інформаційні системи та технології

ЗАТВЕРДЖУЮ
Завідувач кафедрою ІПЗАС
_____ Каміла СТОРЧАК
«____» _____20___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Бовкуну Олександру Сергійовичу
(прізвище, ім'я, по батькові здобувача)

1.Тема кваліфікаційної роботи: Застосування IoT для моніторингу та контролю якості харчових продуктів під час транспортування та зберігання
керівник кваліфікаційної роботи Владислав Хоменчук, доктор філософії, доцент
(Ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)

затверджені наказом Державного університету інформаційно-комунікаційних технологій від « 27 » лютого 2024 р. № 36

2. Строк подання кваліфікаційної роботи « 10 » травня 2024 р.

3. Вихідні дані до кваліфікаційної роботи:
Середовище розробки Arduino IDE
Програма віртуалізації VirtualBox
Науково-технічна література з питань, пов'язаних з інтернетом речей (IoT)

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Здійснити аналіз процесу моніторингу та контролю якості харчових продуктів
2. Провести аналіз наявних рішень моніторингу та контролю якості
3. Обрати компоненти для розробки та обґрунтувати цей вибір
4. Розробити серверну частину для отримання даних пристрою
5. Провести аналіз отриманих результатів

5. Перелік ілюстративного матеріалу: *презентація*

6. Дата видачі завдання: « 27 » лютого 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної літератури	15.03.24 – 30.03.24	
2	Аналіз процесу моніторингу та контролю якості харчових продуктів	01.04.24 – 14.04.24	
3	Обирання компонентів для розробки пристрою	15.04.24 – 22.04.24	
4	Написання коду для пристрою	23.04.24 – 27.04.24	
5	Розробка системи моніторингу та контролю якості харчових продуктів	28.04.24 – 15.05.24	
6	Тестування розробленої моделі та висновки	16.05.24 – 19.05.24	
7	Розробка демонстраційних матеріалів (презентація)	20.05.24 – 25.05.24	

Здобувач вищої освіти _____ Олександр БОВКУН
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник
кваліфікаційної роботи _____ Владислав ХОМЕНЧУК
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 59 стор., 21 рис., 20 джерел.

Мета роботи – проаналізувати та створити модель пристрою, який буде збирати інформацію про внутрішній стан (температуру, розташування) та надсилати дані на сервер для відображення.

Об'єкт дослідження – процес моніторингу та контролю якості харчових продуктів під час транспортування та зберігання.

Предмет дослідження – пристрій для моніторингу та контролю якості харчових продуктів.

Короткий зміст роботи:

У ході роботи над завдання було проаналізовано вже існуючі варіанти для моніторингу та контролю якості харчових продуктів під час транспортування та зберігання з подальшим вдосконаленням.

Щоб дослідити як воно буде працювати, була розроблена симуляція роботи пристрою з імітацією надсилань даних про розташування та температуру. На віртуальній машині було встановлено Ubuntu Server – операційну систему для серверного обладнання, разом з пакетом додатків для приймання даних та їх подальшого відображення.

Було зроблено висновки щодо роботи пристрою.

КЛЮЧОВІ СЛОВА: IOT, ARDUINO, MQTT, UBUNTU, VICTORIA METRICS, GRAFANA, ПЕЛЬТЬЄ, ТЕМПЕРАТУРА, GPS, GSM.

ABSTRACT

Text part of the bachelor level qualification work: 59 pages, 21 pictures, 20 sources.

The purpose of the work - is to analyze and create a model of a device that will collect information about the internal state (temperature, location) and send data to the server for display.

Object of research - the process of monitoring and controlling the quality of food during transportation and storage.

Subject of research is a device for monitoring and controlling the quality of food products.

Summary of work:

During the work on the task, we analyzed the existing options for monitoring and controlling the quality of food during transportation and storage with further improvement.

To investigate how it would work, a simulation of the device was developed to simulate sending location and temperature data. Ubuntu Server, an operating system for server hardware, was installed on a virtual machine, along with a package of applications for receiving data and displaying it.

KEYWORDS: IOT, ARDUINO, MQTT, UBUNTU, VICTORIA METRICS, GRAFANA, PELTIER, TEMPERATURE, GPS, GSM.

ЗМІСТ

Стор.

ВСТУП

Найважливішим завданням у сучасному світі є забезпечення безпечності та якості харчових продуктів. Зростання чисельності населення планети та збільшення обсягів світової торгівлі призвели до збільшення обсягів перевезень харчових продуктів на великі відстані, а отже, до ризику їх псування та втрати якості.

Актуальність моєї теми визначається необхідністю удосконалення методів моніторингу та контролю якості харчових продуктів і перспективами використання Інтернету речей (IoT). Традиційні методи контролю якості часто є повільними та дорогими, тоді як нові технології можуть забезпечити швидший і точніший зворотний зв'язок. Використання IoT в транспортуванні та зберіганні харчових продуктів дозволяє відстежувати їх стан в режимі реального часу, розробляти ефективні методи утилізації, зменшувати псування та підвищувати безпеку і поживну цінність продуктів за допомогою розумних контейнерів і сучасних датчиків.

Метою цього дослідження є розробка інноваційного контейнера, що володітиме інтелектуальними функціями для моніторингу та забезпечення свіжості та безпечності продуктів харчування протягом транспортування та зберігання. Цей контейнер можна уявити як "розумного помічника", який буде дбати про продукти харчування, гарантуючи їх якість та безпеку.

Для досягнення мети дослідження було визначено наступні завдання: аналіз існуючих систем моніторингу IoT в харчовій промисловості, які значно покращують контроль температури та вологості, забезпечують точні дані в реальному часі, постійний надійний моніторинг, економію коштів і зниження витрат на відходи та ручну працю. Розробка функціональної схеми розумного контейнера включає оснащення датчиками, збір і аналіз даних, сповіщення про відхилення. Переваги IoT для харчової промисловості полягають у підвищенні

безпеки продуктів, зменшенні відходів, підвищенні якості, економії коштів, прозорості ланцюжка поставок, оптимізації маршрутів і підвищенні конкурентоспроможності. Виклики впровадження IoT включають забезпечення сумісності систем, безпеку даних, необхідність навчання персоналу та значні початкові інвестиції.

Об'єктом дослідження є процес моніторингу та контролю якості харчових продуктів під час транспортування та зберігання. Цей комплексний процес включає визначення та контроль параметрів середовища, таких як температура, вологість, освітленість і вібрація. Контроль температури забезпечує збереження свіжості продуктів, запобігаючи псуванню та розвитку бактерій. Оцінка стану харчових продуктів включає візуальний огляд для виявлення явних ознак псування, лабораторні аналізи для детального дослідження хімічного складу та вмісту мікроорганізмів, а також використання спеціальних приладів, таких як термометри, гігрометри, люксметри та віброакустичні вимірювачі. Забезпечення відповідності харчових продуктів встановленим національним та міжнародним стандартам якості є важливим аспектом цього процесу.

Предметом дослідження є пристрій для моніторингу та контролю якості харчових продуктів. Він обладнаний датчиками, засобами бездротового зв'язку та мікропроцесором, і призначений для збору даних про параметри середовища, такі як температура, вологість, освітленість і вібрація. Зібрані дані передаються на сервер для зберігання, обробки та аналізу, а також для інформування користувачів про стан харчових продуктів та можливі проблеми. Пристрій підвищує точність, надійність та економічність моніторингу і контролю якості продуктів, забезпечує їх безпечність та якість, оптимізує умови транспортування та зберігання, зменшує псування продуктів і пов'язані з цим економічні втрати, а також підвищує конкурентоспроможність харчових підприємств.

Методом дослідження буде створення симуляції для бачення прикладу як буде працювати пристрій в реальному світі.

Робота була апробована на V Міжнародній науково-технічній конференції «Сучасний стан та перспективи IoT», яка проходила 18 квітня 2024 року. Тезу на тему «Застосування IoT для моніторингу та контролю якості харчових продуктів під час транспортування та зберігання» було опубліковано у збірнику, присвяченому цій конференції.

1 АНАЛІЗ ВИКОРИСТАННЯ ТЕХНОЛОГІЙ ІОТ ДЛЯ МОНІТОРИНГУ СТАНУ ЯКОСТІ ХАРЧОВИЧ ПРОДУКТІВ

1.1 Визначення та основні принципи функціонування Інтернету речей

На перший погляд, IoT може здатися простою ідеєю підключення фізичних об'єктів до Інтернету, але його засновники й прихильники розглядають це як більш глибоку та складну технологічну парадигму. Одним із ключових принципів IoT є підключення об'єктів до Інтернету за допомогою різних засобів зв'язку, таких як Wi-Fi, Bluetooth, RFID, NFC тощо. Це дає змогу цим об'єктам збирати та обмінювати дані, а також приймати команди з центральної системи чи взаємодіяти з іншими підключеними об'єктами.

Однак IoT - це не просто підключення до Інтернету. Його сутність полягає в тому, що ці об'єкти можуть бути обладнані різними сенсорами та актуаторами, які дозволяють збирати дані про навколишнє середовище та впливати на нього. Це означає, що об'єкти можуть виявляти своє становище, середовище, температуру, вологість тощо, і відповідно реагувати на ці дані, виконуючи певні дії. Інший важливий аспект IoT - це аналіз та обробка даних. Зібрані сенсорами дані потребують обробки, аналізу та інтерпретації, щоб винести корисну інформацію, яка може бути використана для прийняття рішень. Для цього використовуються різні технології та алгоритми, включаючи штучний інтелект та машинне навчання.

У цілому, IoT - це комплексна концепція, яка об'єднує в собі технології зв'язку, сенсори, актуатори та аналітику для створення "розумних" об'єктів, які можуть взаємодіяти між собою та з оточуючим середовищем.

1.2 Історія виникнення та еволюція концепції Інтернету речей

Інтернет речей (IoT) - це концепція, яка з'явилася не так давно, але пройшла значний шлях від свого виникнення до сучасного розвитку. Історія IoT починається з простих ідей та концепцій, які виникали в наукових та технологічних лабораторіях, і перетворювалися на революційні технології, які зараз формують наше повсякденне життя. Інтернет речей (IoT) включає в себе багато “розумних”, комп’ютероподібних пристроїв, які сьогодні є повсякденними, і які можуть підключатися до Інтернету або взаємодіяти через бездротові мережі. Хоча приклади взаємопов’язаних електронних пристроїв існують ще з початку 19-го століття, з винаходом телеграфу та його здатності передавати інформацію за допомогою кодованого сигналу на відстань. Одним з перших прикладів IoT був холодильник Coca Cola, розташований в Університеті Карнегі-Меллона на початку 1980-х. Місцеві програмісти підключалися через Інтернет до холодильного пристрою і перевіряли, чи є доступний напій, і чи він холодний, перш ніж йти купувати його.

Термін “Інтернет речей” був запропонований Кевіном Ештоном, виконавчим директором Auto-ID Labs в MIT, у 1999 році. Він був першим, хто описав IoT, під час презентації для Procter & Gamble, де він розповідав про те, як технологія RFID може використовуватися для відстеження товарів у ланцюжках постачання. З розвитком Інтернету та бездротових технологій у 2000-х роках, концепція IoT стала ще актуальнішою. У цей час почали з'являтися перші бачення масштабного застосування IoT у різних сферах життя, від промисловості до медицини та сільського господарства.

Сьогодні число підключених пристроїв, які складають IoT, нараховує мільярди. IoT спрощує та автоматизує завдання, які є складними і іноді виходять за межі людських можливостей. Інтернет речей став не просто технологічним поняттям, а цілою галуззю, яка впливає на економіку, соціальні взаємини та повсякденне життя людей по всьому світу. Його потенціал ще далеко не

вичерпаний, і очікується, що в майбутньому IoT продовжить розвиватися та трансформувати наше життя ще більше.

1.3 Ключові компоненти та технології які використовуються в Інтернеті речей

Інтернет речей (IoT) складається з різноманітних компонентів та технологій, які спільно працюють для збору, обробки та обміну даними між фізичними об'єктами та інтернет-платформами. Ось деякі з найважливіших компонентів IoT:

1. Сенсори та датчики: Це невеликі пристрої, які вимірюють фізичні параметри навколишнього середовища, такі як температура, вологість, освітленість, тиск тощо. Сенсори можуть бути вбудованими безпосередньо в об'єкти або підключені до них, і вони забезпечують постійне збирання даних, які потім використовуються для аналізу та управління.

2. Актуатори: Це пристрої, які реагують на отримані дані та виконують певні дії. Наприклад, актуатори можуть включати або вимикати пристрої, регулювати рівень освітлення, керувати механізмами або виконувати інші дії в залежності від умов навколишнього середовища.

3. Мікроконтролери або мікропроцесори: Це обчислювальні пристрої, які вбудовані в об'єкти IoT та відповідають за обробку отриманих даних та керування пристроями. Вони забезпечують виконання різних функцій та алгоритмів, що дозволяють об'єктам IoT взаємодіяти зі своєю оточуючою середовище.

4. Мережеві засоби зв'язку: Це технології, які дозволяють об'єктам IoT підключатися до мережі Інтернет та обмінюватися даними. До них відносяться бездротові технології, такі як Wi-Fi, Bluetooth, Zigbee, а також провідники, такі як Ethernet.

5. Хмарні технології та аналітика: Це інфраструктура, яка дозволяє зберігати великі обсяги даних в хмарних серверах та використовувати аналітичні інструменти для обробки цих даних та отримання корисної інформації. Хмарні

сервіси дозволяють забезпечити масштабованість та доступність даних для різних застосувань IoT.

6. Блокчейн: Технологія, яка дозволяє створювати безпечні, недійсні та недоступні до модифікації записи даних. В IoT блокчейн може використовуватися для забезпечення безпеки та надійності збереження та обміну даними між різними пристроями та системами.

7. Штучний інтелект та машинне навчання: В IoT вони можуть бути використані для автоматизації процесів, виявлення аномалій та управління системами в реальному часі.

Ці компоненти та технології утворюють основу для створення різноманітних застосувань Інтернету речей в різних сферах життя, від промисловості та транспорту до медицини та побутових пристроїв. Вони дозволяють забезпечити збір, обробку та аналіз даних у реальному часі, що відкриває нові можливості для оптимізації процесів та покращення якості життя.

1.4 Сфери застосування IoT поза харчовою промисловістю

Інтернет речей (IoT) знаходить застосування в різних галузях життя, поза харчовою промисловістю. Ось деякі з найбільш важливих сфер застосування IoT:

1. Медицина та охорона здоров'я: Перш за все, носимі пристрої Інтернету речей дозволяють лікарням контролювати стан здоров'я своїх пацієнтів вдома, тим самим скорочуючи час перебування в лікарні, надаючи при цьому актуальну інформацію в режимі реального часу, яка може врятувати життя. У лікарнях "розумні ліжка" інформують персонал про наявність вільних місць, тим самим скорочуючи час очікування на них. Встановлення датчиків Інтернету речей на критично важливе обладнання означає меншу кількість поломок і підвищену надійність, що може означати різницю між життям і смертю.

Догляд за людьми похилого віку стає значно комфортнішим завдяки IoT. На додаток до вищезгаданого домашнього моніторингу в режимі реального часу,

датчики також можуть визначити, чи впав пацієнт або чи страждає він від серцевого нападу.

2. Смарт-будинок: Пристрої Інтернету речей у вигляді натільного одягу та "розумних" будинків полегшують життя приватних осіб. Носимі пристрої охоплюють такі аксесуари, як Fitbit, смартфони, годинники Apple, монітори здоров'я та багато іншого. Ці пристрої покращують розваги, підключення до мережі, здоров'я та фітнес. Розумні будинки піклуються про такі речі, як активація екологічного контролю, щоб ваш будинок був у максимальному комфорті, коли ви повертаєтесь додому. Вечерю, для приготування якої потрібна духовка або горщик, можна запустити дистанційно, щоб їжа була готова, коли ви повернетесь. Безпека також стає доступнішою: споживач має можливість дистанційно керувати приладами та світлом, а також активувати розумний замок, щоб дозволити відповідним людям увійти в будинок, навіть якщо у них немає ключа.

3. Містобудування та розумні міста: Використання IoT у містобудуванні дозволяє вдосконалити управління транспортними потоками, використанням енергії, управлінням водопостачанням та відходами.

4. Транспорт і логістика: До цього часу більшість людей чули про прогрес, досягнутий у створенні безпілотних автомобілів. Але це лише частина величезного потенціалу в галузі транспорту. GPS, який, якщо подумати, є ще одним прикладом Інтернету речей, використовується, щоб допомогти транспортним компаніям прокладати більш швидкі та ефективні маршрути для вантажівок, що перевозять вантажі, тим самим прискорюючи час доставки. Вже досягнуто значного прогресу в навігації, що знову ж таки нагадує про GPS в телефоні чи автомобілі. Але містобудівники також можуть використовувати ці дані, щоб допомогти визначити схеми руху, попит на місця для паркування, а також будівництво та обслуговування доріг.

5. Сільське господарство: У сільському господарстві IoT використовується для моніторингу росту рослин, контролю за вологою та рівнем родючості ґрунту, автоматизації процесів поливу та дозування добрив.

Це лише деякі приклади того, як Інтернет речей трансформує різні аспекти нашого життя, полегшуючи його та роблячи більш ефективним та зручним.

1.5 Основні аспекти та принципи функціонування харчової промисловості

Харчова промисловість – це галузь економіки, що займається виробництвом, переробкою, зберіганням, розподілом та продажем харчових продуктів. Одним з ключових елементів цього процесу є “холодовий ланцюг” - система, що забезпечує постійне зберігання та транспортування продуктів при визначеній температурі. Ось основні принципи харчової промисловості, які включають використання холодового ланцюгу:

- Безпека продуктів: Холодовий ланцюг допомагає забезпечити безпеку харчових продуктів, зокрема шляхом запобігання розвитку бактерій, що можуть спричинити хвороби.
- Якість продуктів: Завдяки холодовому ланцюгу продукти зберігають свою якість, свіжість та поживність.
- Інновації та технології: Холодовий ланцюг використовує новітні технології для моніторингу та контролю температури продуктів на всіх етапах виробництва, зберігання та доставки.
- Стійкість до зовнішніх впливів: Холодовий ланцюг допомагає забезпечити стійкість харчової промисловості до зовнішніх факторів, таких як зміни клімату, погодні умови, ринкові та економічні коливання.
- Споживчі тенденції: Завдяки холодовому ланцюгу споживачі мають доступ до свіжих продуктів незалежно від їх місцезнаходження.

З появою Інтернету та Інтернету речей (IoT), холодовий ланцюг переживає новий рівень інновацій та технологічного прориву. IoT вносить значний внесок у підвищення ефективності та безпеки холодового ланцюгу, дозволяючи виробникам моніторити та керувати температурою продуктів в реальному часі.

1.6 Використання IoT в харчовій промисловості

Харчова промисловість завжди стояла на передніх лініях інновацій, використовуючи нові технології для поліпшення якості та безпеки продуктів. Протягом років вона пройшла значний шлях від ручної виробництва до високотехнологічних систем виробництва та дистрибуції. З появою Інтернету речей харчова промисловість знову переосмислює свої можливості та здійснює технологічну трансформацію. Інтернет речей став невід'ємною частиною харчової промисловості, забезпечуючи не лише стандартні аспекти автоматизації та відстеження, але й впровадження передових технологій у сфері безпеки, якості та ефективності.

IoT в харчовій промисловості виявляється надзвичайно важливим, оскільки надає безліч можливостей для поліпшення ефективності, якості та безпеки продукції. Технологія Інтернету речей дозволяє підприємствам у галузі харчового виробництва оптимізувати свої процеси, вдосконалювати системи контролю якості та забезпечувати безпеку для споживачів. Ось кілька основних застосувань цієї технології:

1. Моніторинг умов зберігання та транспортування: Системи IoT використовуються для моніторингу температури, вологості та інших параметрів умов зберігання та транспортування харчових продуктів. Це дозволяє підприємствам вчасно виявляти та усувати проблеми, що можуть призвести до псування продуктів.

2. Оптимізація виробництва: Інтернет речей дозволяє впроваджувати системи автоматизації та моніторингу виробничих процесів. Це допомагає знижувати витрати, підвищувати ефективність та забезпечувати стабільну якість продукції.

3. Відстеження походження продуктів: Завдяки технології блокчейн та системам відстеження, IoT дозволяє споживачам відстежувати шлях харчових продуктів від постачальника до полиці магазину, забезпечуючи прозорість та впевненість у їх якості та безпеці.

4. Боротьба зі збитками та втратами: Системи IoT допомагають підприємствам зменшувати збитки та втрати, виявляючи ризики псування чи крадіжок у реальному часі та дозволяючи швидко реагувати на них.

Інтернет речей (IoT) став не лише просто технологічним трендом, але й перетворився на ключовий компонент в сучасній харчовій промисловості. Його застосування вирішує багато проблем, з якими стикаються підприємства у цій галузі, починаючи від моніторингу умов зберігання та транспортування продуктів, і закінчуючи оптимізацією виробничих процесів. Це дозволяє підприємствам забезпечувати високу якість продукції, підвищувати ефективність роботи та знижувати витрати. Особливо важливим є застосування IoT в холодовому ланцюгу, що дозволяє контролювати температуру продуктів на всіх етапах від виробництва до споживача, що є критично важливим для забезпечення безпеки та якості харчових продуктів.

1.7 Температурно-контрольований ланцюг харчової промисловості та роль IoT в управлінні

Холодовий ланцюг - це температурно-контрольований ланцюг поставок, який включає виробництво, транспортування та зберігання товарів, що швидко псуються. Цей тип логістичної системи працює з метою збереження якості та безпеки продуктів харчування, фармацевтичних препаратів, хімічних речовин та інших товарів, які чутливі до умов навколишнього середовища, таких як тепло або холод.

Холодовий ланцюг включає в себе такі етапи:

1. Виробництво: На цьому етапі товари, що швидко псуються, виробляються або збираються. Вони потім поміщаються в контрольоване середовище для зберігання.

2. Транспортування: Товари транспортуються від виробника до дистриб’ютора або роздрібного продавця. Під час транспортування

використовуються спеціальні транспортні засоби, які можуть підтримувати контрольовані температурні умови.

3. Зберігання: Після доставки товари зберігаються в контрольованих умовах до того, як їх продадуть кінцевому споживачеві.

Холодовий ланцюг важливий для забезпечення безпеки та якості продуктів харчування, а також для забезпечення їх свіжості та смаку. Він також важливий для зберігання фармацевтичних препаратів, таких як вакцини, які повинні зберігатися при певних температурних умовах. Однак, управління холодовим ланцюгом може бути складним. Це включає в себе виклики, такі як забезпечення правильного зберігання та транспортування товарів, відстеження температури протягом всього ланцюга поставок, а також використання технологій, таких як IoT, для моніторингу та контролю температури. Завдяки Інтернету речей (IoT), можливості управління холодовим ланцюгом значно розширюються. IoT-пристрої, такі як датчики температури, вологості та GPS-трекери, дозволяють відстежувати стан товарів у реальному часі та надсилати дані на сервер для аналізу та прийняття рішень. Це дозволяє підприємствам швидко реагувати на будь-які проблеми, що можуть виникнути, та запобігати псуванню продуктів. На прикладі блок схеми системи моніторингу (Рис. 1.1), можна побачити, як датчики температури, вологості та освітленості (LDR) інтегровані з мікроконтролером, що управляє системою, та GSM-GPS-GPRS модулем для передачі даних. Користувачі можуть отримувати інформацію через веб-сервер, мобільний додаток або SMS-повідомлення, що забезпечує ефективне управління холодовим ланцюгом.

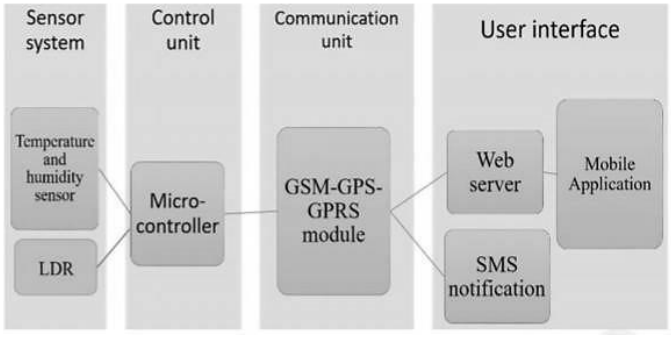


Рис. 1.1 Блок-схема системи моніторингу

Проаналізувавши сучасні методи транспортування та зберігання, стає зрозуміло, що існуючі системи можуть бути покращені за допомогою IoT. Інтеграція IoT не тільки дозволяє відстежувати температуру та місцезнаходження продуктів, але й забезпечує постійне підтримання оптимальних умов зберігання через автоматичне регулювання температури. Це знижує витрати, пов’язані з втратами продуктів, та підвищує загальну ефективність холодового ланцюга.

2 РОЗРОБКА ІОТ ПРИСТРОЮ НА ОСНОВІ МІКРОКОНТРОЛЕРА ATMEGA

У цьому розділі буде розглянуто як технологію Інтернету речей можна використовувати для розробки пристроїв для моніторингу якості продуктів харчування під час транспортування та зберігання. Уявімо, що у нас є інструмент, який може надавати вам точну і надійну інформацію про стан продуктів харчування на етапі ланцюга поставок. Цей інструмент може допомогти гарантувати, що їжа, яку ви отримуєте, буде найвищої якості. Тепер треба зануритися в різні частини, з яких складається цей пристрій, їх функції та переваги, які вони приносять при використанні в цьому конкретному застосуванні.

2.1 Формулювання вимог до IoT пристрою

Перед початком розробки пристрою для моніторингу та контролю якості харчових продуктів під час транспортування та зберігання треба було чітко визначити вимоги до його функціоналу та характеристик. Наведу ключові вимоги до пристрою:

1) Моніторинг температури: Основна функція цього пристрою - відстежувати температуру в внутрішньому середовищі за допомогою температурного датчика. Швидко збираючи та аналізуючи ці дані, він дозволяє оперативно виявляти будь-які відхилення від оптимальних умов зберігання харчових продуктів.

2) Дисплей для виведення даних: Наявність дисплея на пристрої необхідна для візуального відображення виміряних даних, таких як температура. Це дозволить легко відстежувати поточні умови і реагувати на будь-які відхилення.

3) Трекер для передачі даних на сервер: Щоб забезпечити віддалений моніторинг температури, необхідно зробити трекер для передачі даних на сервер.

Це дозволить відстежувати стан харчових продуктів в режимі реального часу і вживати необхідних заходів для їх збереження.

4) Використання мікроконтролера Arduino: Для керування роботою пристрою та обробки зібраних даних я обрав мікроконтролер Arduino. Його зручний інтерфейс і широкі можливості програмування роблять його ідеальним вибором для цього завдання.

5) Контроль температури за допомогою елемента Пельтьє: Для забезпечення контролю температури в окремому просторі використовується пристрій, який називається елементом Пельтьє. Цей елемент має здатність активно охолоджуватися або нагріватися, що дозволяє підтримувати бажану температуру в певних межах.

2.2 Аналіз та обґрунтування вибору компонентів для IoT пристрою

В цьому підрозділі буде розглянуто які компоненти потрібні для подальшої розробки пристрою:

1) Датчики

У сучасному світі пристрої Інтернету речей стають все більш важливими для відстеження та управління якістю продуктів харчування під час їх переміщення та зберігання. Ці пристрої покладаються на спеціальну систему, що називається датчиками, які збирають інформацію про різні аспекти навколишнього середовища, такі як температура, вологість і світло. Вибір правильного датчика дуже важливий, оскільки він визначає якість і точність зібраних даних.

Я вирішив обрати популярний датчик DS18B20 (Рис. 2.1), який може працювати в широкому діапазоні температур, що робить його ідеальним для ситуацій, коли температура може бути як низькою -55°C, так і високою +125°C. DS18B20 має простий цифровий інтерфейс, який вимагає лише одного дроту для підключення до мікроконтролера. Через це він дуже зручний і простий у використанні. Щоб датчик DS18B20 працював правильно, його потрібно

підключити до мікроконтролера та джерела живлення. Однак також потрібно використовувати резистори, щоб переконатися, що датчик працює належним чином. Зазвичай для підключення DS18B20 використовують один резистор 4,7 кОм або два резистори по 10 кОм, з’єднані паралельно.

Рис. 2.1 Датчик температури DS18B20

Загалом, DS18B20 – це гарний вибір для пристрою. Його точність, температурний діапазон і простота використання роблять його ідеальним рішенням для збереження свіжості та безпеки ваших продуктів. Датчик DS18B20 відіграє важливу роль в системі моніторингу, яка допомагає підтримувати якість і надійність температурних даних при транспортуванні і зберіганні харчових продуктів.

2) Мікроконтролер

Розробка пристрою для моніторингу та контролю якості харчових продуктів потребує мікроконтролера, здатного збирати та обробляти дані з датчиків і передавати їх до центральної системи або хмарного сервісу. Вибір правильного мікроконтролера має вирішальне значення для забезпечення ефективності та надійності пристрою. У моєму випадку Arduino Uno (Рис. 2.2), як мініатюрний, але

водночас потужний мікроконтролер, є ідеальним рішенням для створення пристрою, що вимагає точного моніторингу та контролю якості харчових продуктів. Цей мікроконтролер вирізняється своєю здатністю до збору даних з широкого спектру датчиків, їх обробки та передачі до централізованих систем управління або хмарних сервісів, що є критично важливим для забезпечення безперебійної роботи та високої якості кінцевого продукту.

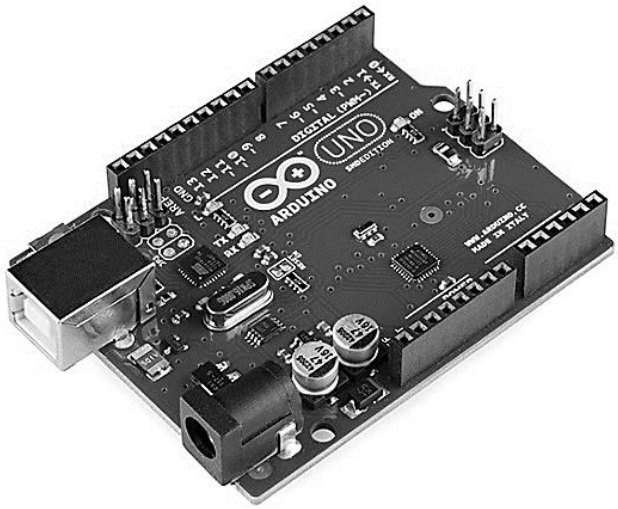


Рис. 2.2 Arduino Uno

В основі Arduino Uno лежить мікроконтролер Atmega328P, який працює на тактовій частоті 16 МГц і забезпечує достатню обчислювальну потужність для більшості завдань. З 32 кілобайтами флеш-пам’яті, з яких 0.5 кілобайтів використовуються для завантажувача (bootloader), та 2 кілобайтами SRAM, Arduino Uno може зберігати та обробляти значні обсяги даних. Додатково, 1 кілобайт EEPROM дозволяє зберігати дані, які не будуть втрачені при вимкнені живлення. Щодо інтерфейсів, Arduino Nano пропонує 14 цифрових входів/виходів (з яких 6 можуть використовуватися для PWM виходів) та 8 аналогових входів, що робить його надзвичайно гнучким для підключення до різноманітних датчиків та виконавчих механізмів. Підтримка протоколів UART, SPI та I2C розширює можливості комунікації та інтеграції з іншими пристроями та системами.

Arduino Uno може живитися від USB-B з’єднання, нерегульованого зовнішнього джерела живлення (рекомендована напруга живлення 7 – 12V) або регульованого зовнішнього джерела живлення (5V), з автоматичним вибором джерела живлення з найвищою напругою. Це забезпечує гнучкість у виборі джерел живлення, що є важливим для портативних або віддалених пристроїв.

3) Дисплей

При створенні пристрою, який відстежує та управляє якістю продуктів харчування під час їх транспортування та зберігання, однією з найважливіших частин є екран, на якому відображатиметься вся інформація та стан пристрою. Вибір правильного екрану дуже важливий, оскільки він повинен бути простим у використанні і чітко відображати всю важливу інформацію для людей, які ним користуються. Я обрав дисплей LCD I2C QC1602A (Рис. 2.3) через його переваги:

- Зручність інтерфейсу: цей дисплей оснащений I2C інтерфейсом, що дозволяє підключити його до мікроконтролера за допомогою всього двох проводів. Це значно спрощує процес підключення та зменшує кількість потрібних компонентів, зменшуючи загальну складність системи.
- Компактні розміри: дисплей має невеликі розміри, що дозволяє легко вбудовувати його в корпус пристрою без значного збільшення його обсягу. Це важливо для наших цілей, оскільки дозволяє створити компактний та ергономічний дизайн.
- Чіткість відображення: дисплей використовує синій колір, що забезпечує контрастне та чітке відображення символів, що робить інформацію легко читабельною навіть у різних умовах освітлення.
- Доступність та вартість: дисплей є доступним та вартісно-ефективним варіантом, що робить його привабливим вибором для нашого проекту, зокрема при великих обсягах виробництва. Це важливо для забезпечення економічності проекту та його доступності для широкого кола користувачів.

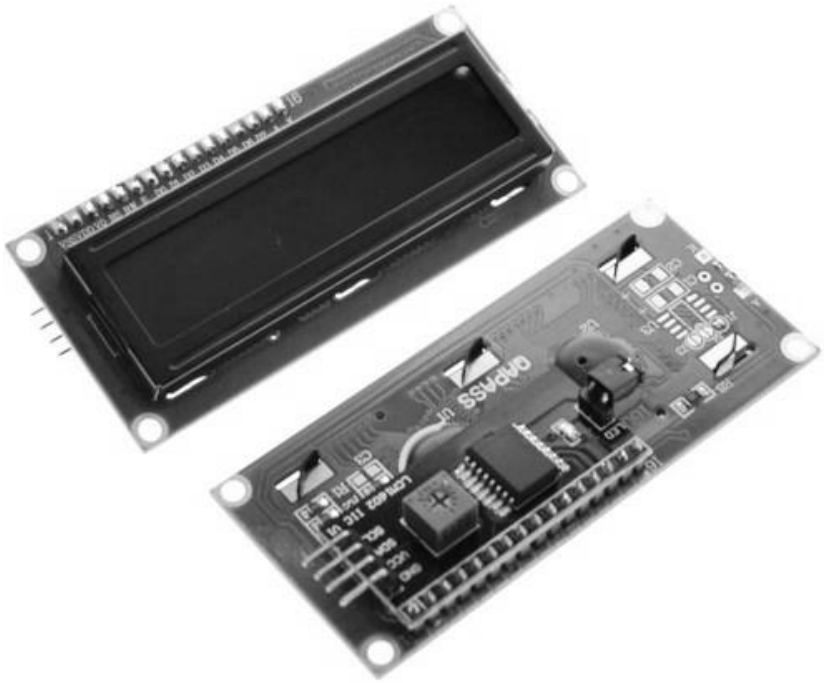


Рис. 2.3 Дисплей LCD I2C QC1602A

Загалом, обраний дисплей відповідає вимогам мого проекту з точки зору функціональності, зручності використання та ефективності, тому є оптимальним варіантом для використання у розробленому пристрої.

4) Елемент Пельтьє

Елемент Пельтьє TEC1-12706 (Рис. 2.4) є ключовим компонентом у моєму пристрої. Він використовується для регулювання температури в межах контейнера чи приміщення, де знаходяться продукти, та забезпечення оптимальних умов для їх збереження. Але спочатку треба знати, що таке цей елемент Пельтьє.

Елемент Пельтьє – це електронний пристрій, який використовується для перетворення електричної енергії в теплову енергію та навпаки. Він працює на основі принципу термоелектричного ефекту, що полягає у зміні температури матеріалу при проходженні через нього електричного струму. Елемент Пельтьє складається з пари напівпровідникових матеріалів, з’єднаних між собою при певному температурному градієнті. При подачі струму на елемент Пельтьє один з його кінців нагрівається, а інший охолоджується. Елемент використовуються у

багатьох пристроях для охолодження чи нагріву, таких як холодильники, кондиціонери, теплові насоси, термоелектричні охолоджувачі, а також у виробництві електроніки та наукових дослідженнях. Вони є ефективними та економічними з точки зору енергоспоживання, що робить їх популярними в різних галузях техніки і промисловості. Я його обрав через наступне:

- Контроль температури: Елемент Пельтьє дозволяє точно регулювати температуру в зоні зберігання продуктів, забезпечуючи оптимальні умови для їх консервації. Це особливо важливо для продуктів, які вимагають певного температурного режиму для збереження своїх якісних властивостей.
- Стабільність температури: За допомогою елемента Пельтьє можна забезпечити стабільність температури в межах зазначених значень, що важливо для попередження перепадів температури, які можуть негативно вплинути на якість харчових продуктів.
- Енергоефективність: Елемент Пельтьє відзначається високою енергоефективністю та швидкістю реакції на зміни температури. Це дозволяє знижувати витрати енергії на утримання необхідного температурного режиму, що є важливим аспектом в системах IoT.

Рис. 2.4 Елемент Пельтьє TEC1-12706

Але у використанні з Arduino є один недолік. Оскільки елемент Пельтьє – це досить «ненажерливий» модуль, то його пряме під'єднання до виводів Arduino суворо заборонено. Логічно припустити, що в цьому разі необхідно мати проміжний силовий комутатор, здатний керуватися логічним рівнем мікроконтролера і витримувати струм до 6А. Перше, що спадає на думку – це комутувати Пельтьє за допомогою релейного модуля. Однак перед складанням схеми слід усвідомити кілька важливих правил, які допоможуть продовжити життя термоелектричного перетворювача:

- Пульсація напруги живлення понад 5% знижує ефективність роботи елемента Пельтьє в середньому на 30-40%;
- Вкрай не рекомендується використовувати релейні регулятори, які виконують часті цикли вмикання/вимикання для живлення елемента Пельтьє. Така динаміка призведе до прискореної деградації напівпровідникових пар, і модуль вийде з ладу через 1-2 місяці. Середній цикл "старт-стоп" побутових елементів становив приблизно 5000 перемикань.
- Деякі умільці регулюють температуру елементів Пельтьє за допомогою широтно-імпульсної модуляції (ШІМ). Такий підхід аналогічно релейним регуляторам сприяє прискореній деградації внутрішніх складових модуля.

На підставі вищевикладених фактів найлогічніше комутувати живлення елемента Пельтьє за допомогою польового транзистора, здатного відкриватися від логічного рівня, наприклад IRL540N. Він здатний пропускати струм до 36А при 4.5В на затворі, що для модуля TEC1-12706 вистачить з лишком.

5) Транзистор

Перед тим як обрати транзистор треба знати що це таке. Транзистори - це електронні компоненти, які широко застосовуються в різних сферах, таких як підсилення, регулювання та перемикання електричних сигналів та енергії. Вони необхідні для ефективного керування електронними схемами. Транзистор, який я вибрав для свого пристрою, є IRL540N (Рис. 2.5) - потужний MOSFET-транзистор, здатний витримувати високу напругу (до 100 В) і струм (до 28 А). Його низький

внутрішній опір робить його придатним для використання в різних схемах, які потребують підсилення та перетворення електричної енергії. Однією з важливих особливостей транзистора IRL540N є його широкий температурний діапазон, який простягається від -55°C до +175°C. Це робить його надійним вибором для використання в схемах підсилення та перетворення та для використання в різних умовах експлуатації, в тому числі в екстремальних умовах.

Обираючи транзистор IRL540N для свого пристрою, я можу бути впевненим, що він відповідатиме вимогам моєї системи з точки зору надійності, ефективності та продуктивності.

Рис. 2.5 Транзистор IRL540N

6) Трекер

Трекер є важливою частиною системи моніторингу, яка дозволяє відстежувати місцезнаходження та інформацію про стан продуктів під час транспортування. Для цього я буду використовувати наступні два компоненти:

- GPS модуль NEO-6M

GPS модуль NEO-6M (Рис. 2.6) є високоточним інструментом, який забезпечує детальну інформацію про положення пристрою. Він використовує 50 каналів для прийому сигналів GPS і підтримує SBAS для підвищення точності. З часом до першого фіксу від 32 секунд при холодному старті до менше ніж 1 секунди

при гарячому старті, NEO-6M забезпечує швидке визначення положення, що є важливим для оперативного моніторингу транспортування харчових продуктів. Зі стандартною швидкістю передачі даних 9600 біт/с та чутливістю -160 дБм, цей модуль забезпечує надійне відстеження навіть у складних умовах. Його робочий діапазон напруги від 3V до 5V та низький струм виходу роблять його чудовим варіантом для розробки пристрою.

Рис. 2.6 GPS модуль NEO-6M

- GSM модуль SIM800C V2

GSM модуль SIM800C V2 (Рис. 2.7), як компактний і енергоефективний засіб зв’язку, є чудовим вибором для системи моніторингу, що вимагає надійного передавання даних. Цей модуль вирізняється своєю здатністю до швидкої передачі даних через GPRS, а також можливістю інтеграції з широким спектром датчиків через свої інтерфейси UART, USB2.0, та GPIO. Це дозволяє модулю забезпечувати безперервний зв’язок між пристроєм та централізованими системами управління або хмарними сервісами, що є критично важливим для точного відстеження стану харчових продуктів. З низьким споживанням енергії, SIM800C V2 забезпечує

тривалу роботу пристрою, що є важливим для систем, які повинні функціонувати протягом тривалого часу без підзарядки.

Рис. 2.7 GSM модуль SIM800C V2

7) Живлення

В будь-якій електронній системі, ефективне живлення відіграє важливу роль у забезпеченні безперебійної та надійної роботи всіх компонентів. Особливо це стосується систем відстеження, де точність та надійність є вирішальними аспектами для досягнення поставлених цілей. У моєму пристрої важливо забезпечити стабільне та безперебійне живлення всіх компонентів. Зважаючи на те, що Arduino не може подавати достатньо потужного живлення для елемента Пельтьє без ризику збоїв, використання модуля роз'єму підключення живлення є логічним рішенням.

Модуль роз'єму підключення живлення (5.5 x 2.1 мм) (Рис. 2.8) дозволяє подавати стабільний струм не лише на саму Arduino та її модулі, але й на елемент Пельтьє, що потребує додаткової потужності для своєї роботи. Цей модуль забезпечує надійну з'єднаність та захист від короткого замикання, що робить його ідеальним вибором для живлення системи відстеження. Правильне живлення не лише забезпечує безперебійну роботу всієї системи, але і дозволяє підтримувати оптимальні умови для функціонування кожного компонента. Модуль роз'єму підключення живлення забезпечує потрібну потужність для Arduino, GPS-GSM трекера, датчика температури, дисплея та елемента Пельтьє,

забезпечуючи їх безперебійну та ефективну роботу навіть в умовах зміни температур та інших екстремальних умов.

Рис. 2.8 Модуль роз'єму підключення живлення

2.3 Проектування схеми роботи IoT пристрою

Система, яку планується розробити, є комплексним рішенням для моніторингу та контролю якості продуктів харчування під час транспортування та зберігання. Вона використовує набір різних компонентів, включаючи Arduino Uno, трекер на основі GPS та GSM модулів, датчик температури, модуль Пельтьє, транзистор IRL540N, дисплей LCD I2C QC1602A та модуль роз’єму підключення живлення. Її мета полягає в тому, щоб забезпечити надійний та ефективний спосіб моніторингу та контролю якості продуктів харчування. Вона використовує сучасні технології та протоколи, такі як MQTT, для передачі даних в реальному часі та забезпечення точного моніторингу. Для більш кращого уявлення про те як система буде виглядати, була зроблена принципіальна схема (Рис. 2.9).

Цільова аудиторія системи включає виробників продуктів харчування, дистриб’юторів, перевізників та будь-яких інших сторін, які зацікавлені в моніторингу та контролі якості продуктів харчування під час транспортування та зберігання. Система збирає різні типи даних для моніторингу та контролю якості

продуктів харчування. Це включає дані про місцезнаходження, які надаються модулем GPS, та дані про температуру, які збираються датчиком температури. Ці дані потім передаються на сервер через GPRS/GSM модуль.

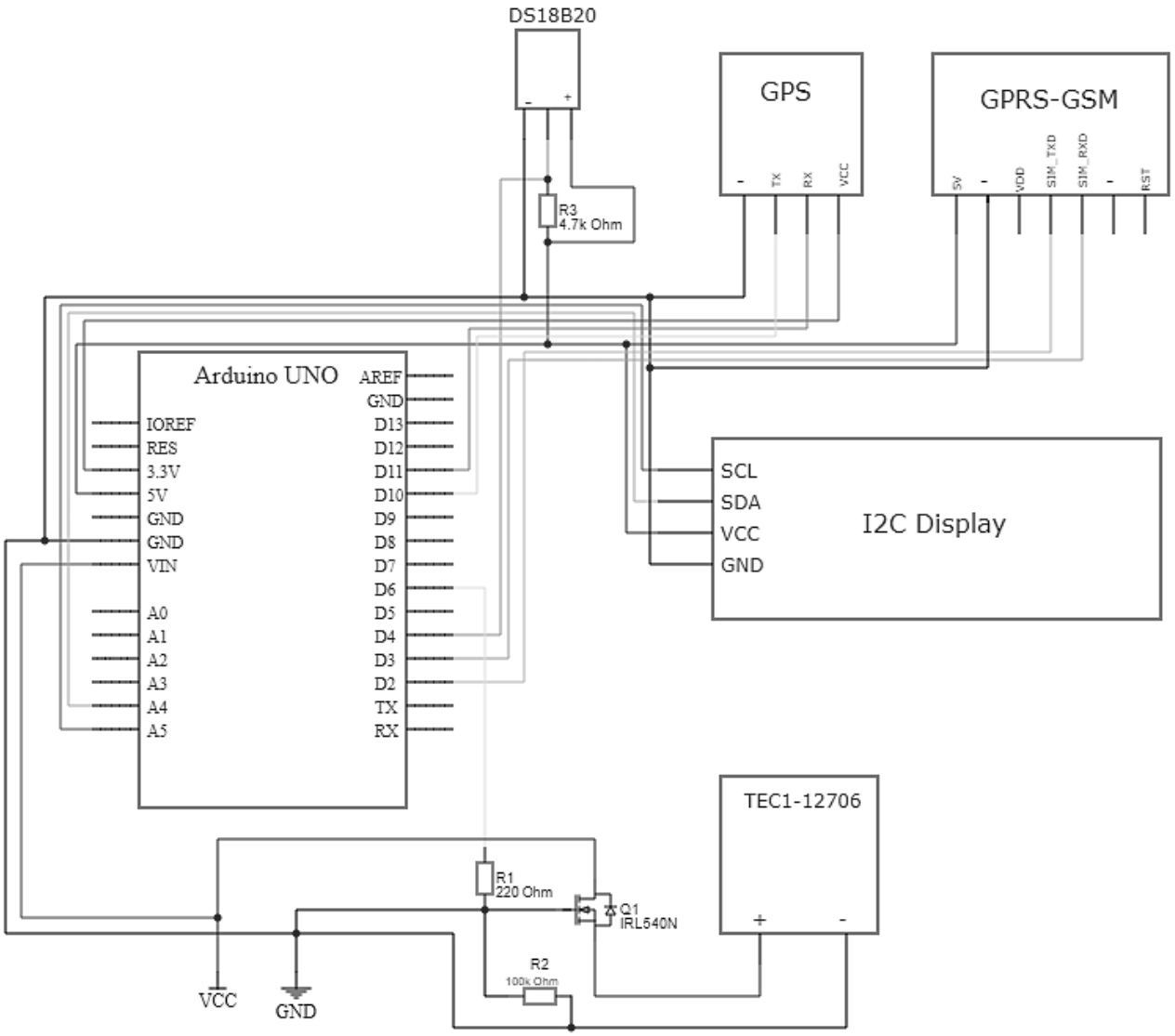


Рис. 2.9 Принципіальна схема пристрою

Дані про місцезнаходження використовуються для відстеження маршруту транспортування продуктів харчування. Це дозволяє виявляти будь-які відхилення від запланованого маршруту та вживати відповідних заходів. Дані про температуру використовуються для моніторингу умов зберігання продуктів харчування. Якщо температура перевищує або знижується нижче встановлених границь, система може вжити відповідних заходів, таких як активація модуля Пельтьє для корекції температури. Всі ці дані збираються, обробляються та аналізуються для забезпечення найвищого рівня контролю якості продуктів харчування. Це дозволяє забезпечити безпеку та свіжість продуктів харчування, що в свою чергу веде до кращого задоволення клієнті. Але не достатньо просто зробити схему, треба написати код та провести якісь тестування. Краще для візуалізації була зроблена схема (Рис. 2.10).

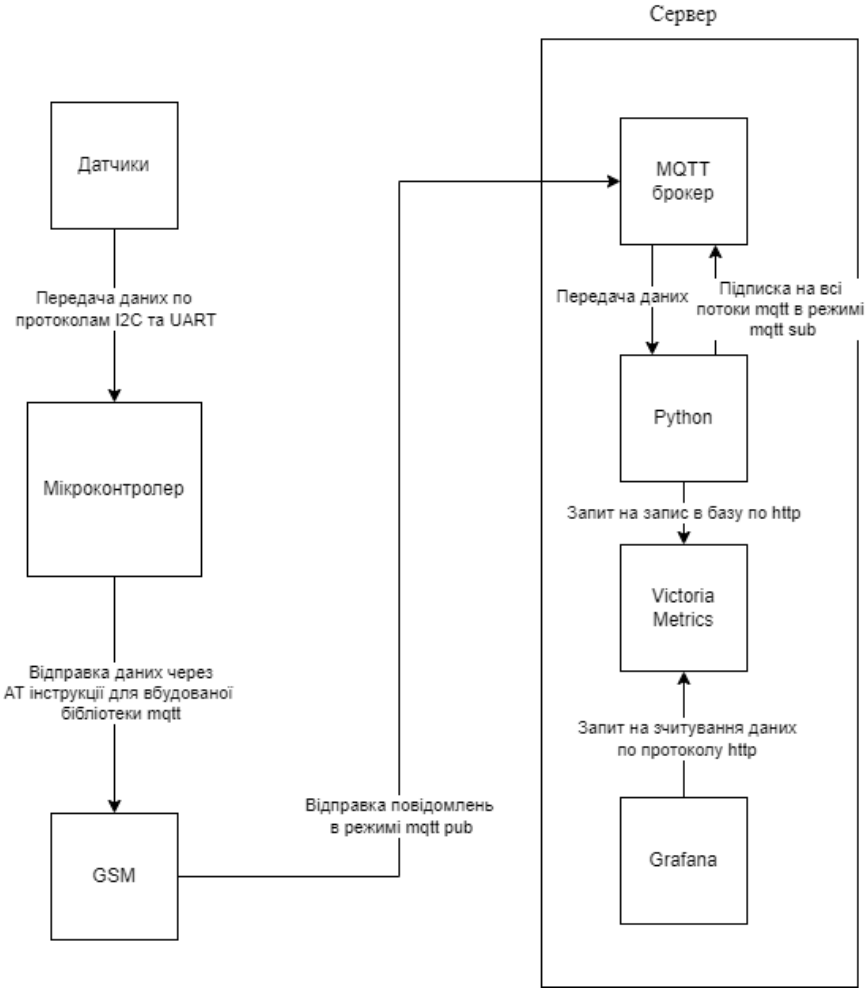


Рис. 2.10 Схема прикладу роботи пристрою

2.4 Розробка програмного забезпечення для IoT пристрою

Через те що в схемі є різні компоненти, треба підключити їх бібліотеки для роботи з ними. Для цього треба ці самі бібліотеки завантажити. Я буду використовувати для цього Arduino IDE де є менеджер бібліотек (Рис. 2.11).
Взагалі знадобится декілька бібліотек. Ось які саме:

- OneWire: Вона необхідна для комунікації між датчиком температури, який використовує OneWire протокол. Дозволяє Arduino встановлювати зв'язок з датчиком через один цифровий пін та читати з нього дані.
- DallasTemperature: Розроблена для роботи безпосередньо з такими датчиками температури як DS18B20, DS18S20, DS1822, DS1820.
- TinyGPS: Бібліотека призначена для роботи з GPS модулем, забезпечуючи функції для парсингу даних NMEA, які видаються GPS модулем.
- TinyGSM: Бібліотека використовується для зв'язку GPRS-GSM модуля і дозволяє відправляти SMS, здійснювати дзвінки та використовувати інтернет через GPRS.
- LiquidCrystal I2C: Бібліотека необхідна для спрощення роботи з символьними LCD дисплеєм, який буде підключатися через I2C інтерфейс.

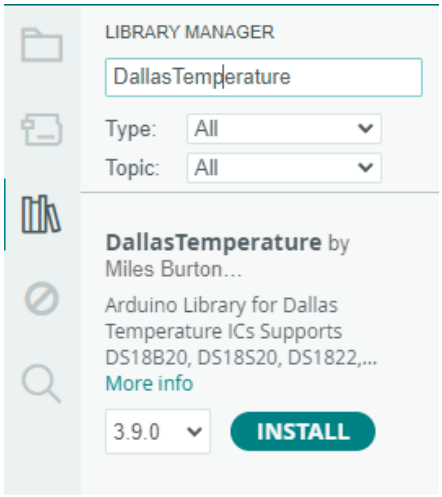


Рис. 2.11 Менеджер бібліотек Arduino IDE

Щоб gsm модем працював, треба на початку коду написати модель модема та можна підключити всі бібліотеки які були згадані вище.

Лістинг коду:

```
#define TINY_GSM_MODEM_SIM800 //для визначення що за модуль буде
//Бібліотеки для датчику температури
#include <DallasTemperature.h>
#include <OneWire.h>
//Бібліотеки для GPS, GSM модулів
#include <TinyGPS++.h>
#include <TinyGsmClient.h>
#include <PubSubClient.h>
//Бібліотеки для дисплею
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
```

Прописую до яких пінів Arduino Uno підключаю компоненти зі схеми, такі як дисплей, елемент Пельтьє, трекер та датчик температури.

Лістинг коду:

```
//ініціалізація пристроїв
#define DS_PIN 4 //DS18B20
#define TEC_PIN 6 //елемент Пельтьє
#define GPS_RX 11 //пін приймача GPS
#define GPS_TX 10 // пін передавача GPS
#define SIM_TXD 2 //пін приймача GSM
#define SIN_RXD 3 // пін передавача GSM
#define SDA_PIN A4 // SDA пін для LCD
#define SCL_PIN A5 // SCL пін для LCD
```

Просисую параметри підключення до MQTT брокера (його ip) та створення топіків на які в майбутньому буде підписаний брокер та будуть туди надсилатися дані.

Лістинг коду:

```
//Параметри підключення до MQTT broker
const char* broker = "192.168.0.147"; //IP брокера
const char* topicTemp = "temperature";
const char* topicLoc = "location";
```

Наступний фрагмент коду закладає основу для з'єднання з GSM-модемом і використання протоколу MQTT. Він створює об'єкти для полегшення зв'язку, а також встановлює змінні для відстеження останньої мітки часу повідомлення та

буфер для зберігання повідомлень. Ці компоненти необхідні для успішного надсилання даних через мережу GSM/GPRS до брокера MQTT.

Лістинг коду:

```
TinyGsm modem(Serial);
TinyGsmClient client(modem);
PubSubClient mqtt(client);
unsigned long lastMsg = 0;
#define MSG_BUFFER_SIZE (50)
char msg[MSG_BUFFER_SIZE];
int value = 0;
```

Функція “void setup()” широко використовується в програмуванні Arduino і виконується лише один раз при увімкненні плати або перезапуску мікроконтролера. Її основне призначення – підготувати апаратні компоненти та встановити різні конфігурації для оптимальної роботи. Тут я починаю робити ініціалізацію та підключення до мережі. Для виводу інформації буде використовуватися “Serial Monitor” зі швидкістю 9600 біт/сек. Для підключення до мережі потрібно в модем вставити sim-картку оператора та зробити налаштування к коді. В строці “modem.gprsConnect()” треба визначити APN (Access Point Name), це робиться в налаштуваннях sim-картки.

Лістинг коду:

```
void setup() {
  // Ініціалізація GSM модуля
  Serial.begin(9600);
  modem.begin();
  // Підключення до мережі
  modem.gprsConnect("internet");
```

Далі, за допомогою AT-команд я буду робити підключення до брокера. Спочатку код надсилає команду AT для конфігурації URL-адреси брокера за допомогою змінної broker, а потім чекає на успішну відповідь від модему. Далі він надсилає команду AT для встановлення ідентифікатора клієнта як «ArduinoClient» і знову чекає на відповідь. Нарешті, він надсилає команду для встановлення з’єднання з брокером, одночасно очікуючи підтвердження від модему.

Лістинг коду:

```
  modem.sendAT(GF("+SMCONF=\"URL\",\""), broker, GF("\""));
  modem.waitResponse();
  modem.sendAT(GF("+SMCONF=\"CLIENTID\",\"ArduinoClient\""));
  modem.waitResponse();
  modem.sendAT(GF("+SMCONN"));
  modem.waitResponse();
```

Наступним кроком буде налаштування піну елемента Пельтьє як вихідний та його встановлення в стан «вимкнуто». Також буде виконана ініціалізація датчика температури та дисплею. На самому дисплеї буде при увімкненні писати «Hello».

Лістинг коду:

```
pinMode(TEC_PIN, OUTPUT);
  digitalWrite(TEC_PIN, LOW);

  sensors.begin();
  lcd.init();
  lcd.backlight();
  lcd.setCursor(1, 0);
  lcd.print("Hello");
  delay(2000); // Затримка 2 секунди
  lcd.clear(); // Очищення дисплею
}
```

Тепер починається функція «void loop()» яка відповідає за логіку пристрою. Все починається з запиту даних про температуру з датчика DS18B20. Отримане значення температури у градусах Цельсія зберігається у змінній «tempC». Потім це значення виводиться на РК-дисплей. Спочатку курсор підводиться до другого рядка першого стовпчика, після чого на дисплеї з’являється рядок «Temp:», за яким слідує поточне значення температури і символ «C», що позначає градуси Цельсія.

Лістинг коду:

```
void loop() {
  sensors.requestTemperatures();
  float tempC = sensors.getTempCByIndex(0);
  lcd.setCursor(0, 1);
  lcd.print("Temp: ");
  lcd.print(tempC);
  lcd.print(" C");
```

Далі перевіряється наявність даних від GPS-модуля в послідовному буфері свідчить про його доступність. Ці дані зчитуються і передаються до бібліотеки GPS для декодування, коли вона стає доступною. Широта і довгота поточного місцезнаходження зберігаються, якщо інформація про місцезнаходження оновлюється. Потім генерується повідомлення, що містить ці координати та поточну температуру, і надсилається брокеру MQTT через GSM-модем.

Лістинг коду:

```
  while (Serial.available() > 0) {
    gps.encode(Serial.read());
    if (gps.location.isUpdated()) {
      // Збереження широти та довготи
      float lat = gps.location.lat();
      float lon = gps.location.lng();

      // Відправлення даних про місцезнаходження на MQTT broker
      char msg[50];
      snprintf(msg, sizeof(msg), "Location: %f,%f Temp: %f", lat, lon, tempC);
      modem.sendAT(GF("+SMPUB=\""), topicLoc, GF("\","), strlen(msg),
GF(",1,0,\""), msg, GF("\""));
      modem.waitResponse();
    }
  }
```

Функція loop() щосекунди надсилає дані про температуру до брокера MQTT. Для цього вона використовує функцію millis() для визначення часу, що минув з моменту відправлення останнього повідомлення. Якщо пройшла одна секунда або більше, температура форматується в рядок і AT-командами надсилається брокеру MQTT у вигляді повідомлення.

Лістинг коду:

```
  unsigned long now = millis();
  if (now - lastMsg > 1000) {
    lastMsg = now;
    ++value;
    snprintf (msg, MSG_BUFFER_SIZE, "%ld", tempC);
    Serial.print("Pub. message: ");
    Serial.println(msg);
    modem.sendAT(GF("+SMPUB=\""), topicTemp, GF("\","), strlen(msg),
GF(",1,0,\""), msg, GF("\""));
    modem.waitResponse();
  }
```

І останнім виконується «if else». Якщо датчик температури реєструє температуру більше 14 градусів по Цельсію то вмикається елемент Пельтьє.

Лістинг коду:

```
if (tempC > 14.0) {
  digitalWrite(TEC_PIN, HIGH);
} else {
  digitalWrite(TEC_PIN, LOW);
}
```

Висновок до розділу 2:

Розробка IoT пристрою на основі мікроконтролера ATmega являє собою важливий етап у створенні сучасних інтелектуальних систем для моніторингу та управління різноманітними процесами. Протягом цього розділу було виконано ряд завдань, які забезпечили реалізацію функціонального та надійного пристрою, здатного задовольнити поставлені вимоги. На початковому етапі було проведено ретельний аналіз вимог до IoT пристрою. Основними параметрами, які враховувалися, були точність і надійність збирання даних, можливість передачі інформації в режимі реального часу, енергоефективність, а також зручність інтеграції з існуючими системами. Вимоги включали необхідність моніторингу температури та місцезнаходження, що визначило вибір відповідних сенсорів та модулів для передачі даних.

На основі визначених вимог було здійснено вибір компонентів для розробки IoT пристрою. Мікроконтролер Arduino Uno був обраний через його високу продуктивність, енергоефективність і наявність широкого спектру бібліотек для розробки. Для забезпечення функціональності пристрою були обрані такі компоненти, як модуль SIM800C для GSM зв'язку, модуль NEO-6M GPS для визначення місцезнаходження, а також різні сенсори для моніторингу температури. Кожен компонент був ретельно обґрунтований з точки зору його відповідності вимогам та здатності забезпечити надійне функціонування системи.

Етап проектування включав розробку схеми з'єднання всіх компонентів, створення прототипу та тестування функціональності. Було розроблено детальну електричну схему, яка враховувала всі необхідні з'єднання та підключення.

Особлива увага була приділена забезпеченню стабільного живлення всіх компонентів, включаючи використання стабілізаторів напруги та конденсаторів для зменшення шумів. Також було розроблено програмне забезпечення, що дозволяє ефективно керувати всіма модулями та обробляти отримані дані.

Одним з ключових етапів стала розробка програмного забезпечення для мікроконтролера. Код був написаний в програмі Arduino IDE з використанням стандартних бібліотек для роботи з модулями GSM, GPS та сенсорами. Основні функції включали збирання даних, їх обробку та передачу на сервер. Особлива увага була приділена оптимізації коду для забезпечення швидкодії та енергоефективності пристрою. Також було реалізовано алгоритми обробки помилок та захисту від збоїв, що підвищило надійність роботи системи.

3 РОЗРОБКА СИСТЕМИ МОНІТОРИНГУ СТАНУ ЯКОСТІ ХАРЧОВИЧ ПРОДУКТІВ

3.1 Використання баз даних часових рядів для зберігання та аналізу даних

У цьому підрозділі буде проводитись аналіз tsdb та rdbms. Також будуть розглянуті види tsdb та їх порівняння між собою.

Time series database (TSDB) – це спеціалізована база даних, оптимізована для зберігання та обробки даних з часовими рядами. Дані часових рядів включають вимірювання або події, які відстежуються, контролюються, записуються та агрегуються протягом часу. До таких даних належать показники серверів, моніторинг продуктивності додатків, мережеві дані, дані датчиків, події, кліки, ринкові транзакції та багато інших видів аналітичних даних. TSDB розроблена спеціально для роботи з метриками та подіями, які мають часову мітку, і оптимізована для вимірювання змін у часі. Властивості, які відрізняють дані часових рядів від інших видів даних, включають управління життєвим циклом даних, узагальнення та сканування великого діапазону записів. Ці бази даних дозволяють ефективно обробляти і аналізувати великі обсяги даних, забезпечуючи високу продуктивність та точність, що робить їх незамінними для різноманітних аналітичних та моніторингових застосувань.

Relational Database (RDBMS) - це тип бази даних, яка зберігає та надає доступ до пов'язаних між собою точок даних. Вона базується на реляційній моделі, яка представляє дані у вигляді таблиць, що є інтуїтивно зрозумілим і простим способом організації інформації. У реляційній базі даних кожен рядок таблиці являє собою запис з унікальним ідентифікатором, відомим як ключ. Стовпці таблиці містять атрибути даних, і кожен запис має значення для кожного атрибуту, що полегшує встановлення зв'язків між даними.

Якщо порівнювати TSDB та RDBMS, то я дав перевагу саме першому через наступне: TSDB оптимізовані для роботи з часовими рядами, вони можуть швидко записувати, зчитувати та агрегувати такі дані, що робить їх ідеальними для моніторингу систем, аналізу даних датчиків, фінансового аналізу та інших завдань, пов'язаних з часовими рядами. Вони здатні зберігати дані з високою роздільною здатністю, тобто з невеликими інтервалами між точками даних, що дозволяє фіксувати детальні зміни в часі, важливі для багатьох аналітичних застосувань. Крім того, TSDB можуть масштабуватися горизонтально, додаючи нові сервери до кластера, що дозволяє їм обробляти все більші обсяги даних без втрати продуктивності. Також TSDB мають спеціальні мови запитів, оптимізовані для роботи з даними часових рядів, що дозволяє користувачам легко запитувати, фільтрувати та агрегувати дані без необхідності писати складні SQL-запити.

Для порівняння я обрав три види tsdb. Почну я з однієї з популярних tsdb, а саме Prometheus. Prometheus - це інструментарій для моніторингу та оповіщення систем з відкритим вихідним кодом, спочатку створений на SoundCloud. Від свого заснування у 2012 році Prometheus здобув широку популярність серед компаній та організацій по всьому світу завдяки своїй надійності та гнучкості. Проект активно підтримується великою спільнотою розробників та користувачів, що сприяє його постійному розвитку та вдосконаленню.

У 2016 році, щоб підкреслити свою незалежність та прозору структуру управління, Prometheus приєднався до Cloud Native Computing Foundation (CNCF) як другий проект після Kubernetes. Це дозволило проекту отримати додаткову підтримку та визнання в спільноті розробників.

Особливості в prometheus наступні:

- Модель на основі витягування

Prometheus використовує модель збору метрик, яка базується на активному витягуванні даних з цільових систем. Це дозволяє ефективно виявляти та моніторити динамічні середовища, автоматично адаптуючись до змін у них.

- PromQL

Prometheus надає потужну мову запитів під назвою PromQL, яка дозволяє створювати виразні та гнучкі запити до даних часових рядів. Завдяки PromQL користувачі можуть легко виконувати складний аналіз даних, отримуючи точну та релевантну інформацію.

- Сповіщення

Prometheus підтримує функцію сповіщень, які можуть бути налаштовані на основі визначених користувачем правил. Це забезпечує своєчасне оповіщення про критичні події та інтегрується з різними системами управління оповіщеннями, забезпечуючи надійний контроль та реагування на виникаючі проблеми.

Приклади використання Prometheus

- Моніторинг інфраструктури

Prometheus широко застосовується для моніторингу стану та продуктивності контейнерних інфраструктур та інфраструктур, заснованих на мікросервісах, таких як Kubernetes та Docker. Завдяки своїй здатності збирати та аналізувати дані з різних компонентів системи, Prometheus надає повну картину стану інфраструктури, що дозволяє операційним командам ефективно управляти ресурсами та забезпечувати стабільну роботу системи.

- Моніторинг продуктивності додатків (APM)

Prometheus може збирати власні метрики додатків за допомогою клієнтських бібліотек, що інтегруються безпосередньо в код додатків. Це дозволяє відстежувати продуктивність додатків у режимі реального часу, аналізувати затримки, помилки та інші критичні показники. Такий підхід допомагає розробникам швидко виявляти та виправляти проблеми, забезпечуючи високу якість та надійність програмного забезпечення.

- Сповіщення та виявлення аномалій

Prometheus дозволяє організаціям налаштовувати сповіщення на основі певних порогових значень або умов. Завдяки гнучкій системі сповіщень, можна швидко виявляти та реагувати на потенційні проблеми або аномалії в роботі системи. Це забезпечує проактивний підхід до управління інфраструктурою та додатками, знижуючи ризик простоїв та інших непередбачуваних проблем.

Якщо підсумувати, Prometheus – це інструмент для моніторингу та оповіщення з відкритим вихідним кодом, який став популярним завдяки своїй надійності та гнучкості. Його особливості включають модель витягування даних, мову запитів PromQL та налаштовувані сповіщення. Він використовується для моніторингу контейнерних середовищ, таких як Kubernetes і Docker, та продуктивності додатків у реальному часі. Це допомагає швидко виявляти та вирішувати проблеми, забезпечуючи стабільність і високу продуктивність систем.

Далі буде йти про OpenTSDB. OpenTSDB (Open Time Series Database) — це відкрита, розподілена та масштабована база даних часових рядів, розроблена на основі Apache HBase, NoSQL бази даних. Вона була створена для задоволення зростаючої потреби у зберіганні та обробці великих обсягів даних часових рядів, які генеруються різними джерелами, такими як пристрої Інтернету речей, датчики та системи моніторингу. Спочатку розроблена компанією StumbleUpon у 2010 році, OpenTSDB згодом стала незалежним проектом з активною спільнотою розробників та користувачів.

OpenTSDB складається з демона часових рядів (TSD), а також набору утиліт командного рядка. Взаємодія з OpenTSDB досягається в першу чергу шляхом запуску одного або декількох TSD. Кожен TSD є незалежним від інших, не маючи головного або спільного стану. Це означає, що ви можете легко масштабувати систему, запускаючи стільки TSD, скільки необхідно для обробки будь-якого обсягу навантаження. Схема даних в OpenTSDB максимально оптимізована для швидкої агрегації подібних часових рядів, що дозволяє мінімізувати простір для зберігання. Користувачам не потрібно безпосередньо звертатися до базового сховища даних; вони можуть взаємодіяти з TSD за допомогою простого протоколу telnet, HTTP API або вбудованого графічного інтерфейсу. Колектор використовується для отримання та передачі даних до TSD, а всі комунікації відбуваються через один порт, де TSD визначає протокол клієнта на основі перших кількох байтів, які він отримує.

OpenTSDB пропонує низку важливих можливостей:

- Масштабованість: Розподілена архітектура OpenTSDB дозволяє горизонтальне масштабування, забезпечуючи оптимальну обробку зростаючих обсягів даних часових рядів.
- Стиснення даних: OpenTSDB використовує різні методи стиснення, щоб зменшити обсяг збережених даних часових рядів, оптимізуючи ресурси.
- Мова запитів з підтримкою часових рядів: OpenTSDB має гнучку мову запитів, яка дозволяє виконувати агрегацію, вибірку, фільтрацію та інші операції для аналізу даних часових рядів, надаючи користувачам потужні інструменти для розуміння їхніх даних.

В цілому, OpenTSDB представляє собою потужну відкриту базу даних часових рядів, розроблену для ефективного зберігання та аналізу великих обсягів даних, що генеруються від різних джерел, включаючи пристрої IoT, датчики та системи моніторингу. Її розподілена архітектура, масштабованість та гнучкість управління дозволяють легко працювати з великими обсягами даних. За допомогою різноманітних інструментів, таких як мова запитів з підтримкою часових рядів та стиснення даних, OpenTSDB надає користувачам зручні та ефективні інструменти для аналізу та взаємодії з їхніми даними часових рядів.

Третьою та останньою tsdb я буду розглядати Victoria Metrics.

VictoriaMetrics — це tsdb з відкритим кодом, створена компанією VictoriaMetrics. Ця база даних покликана підтримувати приватних користувачів та організації у вирішенні завдань, пов'язаних з великими обсягами даних, пропонуючи передові технології для моніторингу та спостереження. VictoriaMetrics розроблена для забезпечення швидкого, економічно вигідного та масштабованого рішення для моніторингу та управління часовими рядами.

VictoriaMetrics представлений у двох варіантах: Single-server VictoriaMetrics та VictoriaMetrics Cluster. Single-server VictoriaMetrics являє собою універсальний бінарний файл, який вирізняється простотою використання та обслуговування. Цей варіант добре масштабується по вертикалі, що дозволяє йому обробляти мільйони метрик за секунду. З іншого боку, VictoriaMetrics Cluster складається з кількох компонентів, які дають змогу створювати горизонтально масштабовані кластери,

забезпечуючи високу доступність та масштабованість у вимогливих середовищах. Завдяки архітектурі VictoriaMetrics користувачі можуть обирати спосіб розгортання, що найкраще відповідає їхнім потребам, і за необхідності масштабувати інфраструктуру бази даних.

Victoria Metrics має такі особливості:

- Висока продуктивність

VictoriaMetrics розроблена для високопродуктивного зберігання та пошуку даних часових рядів. Вона здатна ефективно обробляти мільйони метрик за секунду, забезпечуючи швидке виконання запитів для аналізу в режимі реального часу.

- Масштабованість

Архітектура VictoriaMetrics підтримує як вертикальну, так і горизонтальну масштабованість. Це дає змогу користувачам адаптувати свою інфраструктуру моніторингу та бази даних часових рядів відповідно до зростання обсягів даних і попиту.

- Економічна ефективність

VictoriaMetrics пропонує економічно вигідне рішення для управління даними часових рядів. Ефективне зберігання та можливості запитів дозволяють мінімізувати операційні витрати при збереженні високої продуктивності.

Як підсумок, VictoriaMetrics є потужним та ефективним інструментом для роботи з даними часових рядів, що поєднує високу продуктивність, масштабованість та економічну ефективність. Завдяки своїй архітектурі, вона задовольняє вимоги як малих, так і великих організацій, забезпечуючи надійне зберігання та швидкий доступ до даних. VictoriaMetrics дозволяє користувачам адаптувати інфраструктуру відповідно до зростаючих потреб, забезпечуючи стабільну роботу в умовах великих навантажень та мінімізуючи витрати на експлуатацію.

Якщо порівнювати ці три tsdb то я дам перевагу саме Victoria Metrics. Хоч Prometheus популярний завдяки простоті налаштування та потужному механізму запитів, але обмежена масштабованість і проблеми з довготривалим зберіганням

даних можуть стати перешкодою для великих інфраструктур. OpenTSDB пропонує високу масштабованість, проте складний процес налаштування та підтримки може бути викликом для менш технічно підготовлених команд. Але VictoriaMetrics поєднує високу продуктивність, масштабованість та економічну ефективність, що робить її привабливим вибором для організацій, які потребують надійного та масштабованого рішення для роботи з часовими рядами. Завдяки можливості обробляти мільйони метрик за секунду та ефективному використанню ресурсів, VictoriaMetrics забезпечує оптимальне поєднання продуктивності та витрат, що є ключовою причиною для її вибору.

3.2 Використання протоколу MQTT для збору даних з датчиків IoT

Протокол MQTT, що стоїть за абревіатурою Message Queuing Telemetry Transport, є стандартом для обміну повідомленнями між різними пристроями, зокрема в Інтернеті речей. Його розробка була ініційована в кінці 20-го століття двома фахівцями: Енді Стенфорд-Кларком з компанії IBM та Арленом Ніппером з Cirrus Link, з метою створення ефективного способу комунікації між пристроями у мережах з обмеженими можливостями передачі даних.

Основна ідея MQTT полягає у мінімізації навантаження на мережу та забезпеченні стабільної роботи пристроїв з низьким енергоспоживанням. Це було особливо актуально для систем, де зв’язок здійснювався через дорогі супутникові канали. Такий підхід дозволив MQTT стати оптимальним рішенням для використання в сенсорних мережах, мобільних пристроях та інших елементах IoT, які працюють в умовах обмеженого живлення.

Завдяки своїй гнучкості та ефективності, MQTT швидко знайшов застосування у багатьох сферах, де потрібна надійна передача даних між пристроями з обмеженими ресурсами. Це стосується не тільки традиційних TCP/IP мереж, але й альтернативних, таких як Bluetooth, UDP, ZigBee, що робить його універсальним інструментом для різноманітних IoT-екосистем.

MQTT вирізняється серед інших протоколів реального часу, таких як Web Application Messaging Protocol, Streaming Text-Oriented Messaging Protocol та Alternative Message Queueing Protocol, своєю легкістю та високою продуктивністю, що робить його одним з найпопулярніших виборів для розробників IoT-рішень.

Я буду його встановлювати на Ubuntu Server, який в свою чергу буде на віртуальній машині. Взагалі я обрав цю ОС через кілька причин:

- Ubuntu Server відомий своєю простотою встановлення та налаштування. Це робить його ідеальним вибором для розробників, які хочуть швидко розгорнути та налаштувати свої сервери.
- Ubuntu має одну з найактивніших та найбільших спільнот серед всіх дистрибутивів Linux. Це означає, що я зможу знайти відповіді на більшість своїх питання та отримати допомогу в разі потреби.
- Ubuntu Server має сильні механізми безпеки, які допомагають захистити дані та систему. Це особливо важливо для серверів IoT, які часто стають мішенями для кібератак.
- Ubuntu Server підтримує MQTT “з коробки”, що робить його ідеальним вибором для проекту. Можна легко встановити та налаштувати MQTT брокер, такий як Mosquitto, використовуючи вбудовані пакети Ubuntu.
- Ubuntu Server може легко масштабуватися та налаштовуватися для потреб. Це означає, що він може рости разом з проектом та адаптуватися до змінюваних вимог.

Програму яку я буду використовувати для цього називається VirtualBox. Я її обрав через декілька переваг: відкритий код, широка підтримка платформ, контрольні точки. Після встановлення VirtualBox, розпочинаю додавання образу Ubuntu Server для подальшого встановлення. Для цього я йому даю 2 ядра та 3 Гб оперативної пам’яті.

Вже після встановлення Ubuntu Server починаю встановлювати MQTT брокер Mosquitto та Python. Для брокера потрібно написати наступну команду:

```
$ sudo apt install -y mosquitto
```

Після цього роблю перевірку чи почав працювати брокер (Рис. 3.1) та починаю встановлювати Python та paho-mqtt.

Рис. 3.1 Перевірка роботи брокера

Зазначу, що перед встановленням Python, для спрощення я додаю репозиторій від deadsnakes за допомогою наступної команди:

```
$ sudo add-apt-repository ppa:deadsnakes/ppa
```

Та вже після цього починаю встановлювати Python 3.12 та paho-mqtt. Взагалі, Python буде виступати в якості передавача інформації з пристрою на Victoria Metrics. Все буде відбуватися за допомогою наступних команд:

```
$ sudo apt install python3.12
$ sudo apt install python3-paho-mqtt
```

Перед тим як підключати пристрій, треба налаштувати конфігураційний файл, а потім перевірити чи працює pub sub на linux server. Для налаштування потрібно перейти до файлу “mosquito.conf”. Вже всередині прописую наступне:

```
listener 1883
allow_anonymous true
```

Після налашування можно розпочинати симуляцію відправки повідомлень. Для цього потрібно підключитися через якийсь додаток для віддаленого доступу на сервер. Для цього я буду використовувати PuTTy (Рис. 3.2). PuTTY – це безкоштовний емулятор терміналу з відкритим вихідним кодом, створений Саймоном Татемом для платформ Windows, Unix і Mac. Він функціонує як клієнт SSH і Telnet, дозволяючи користувачам отримувати доступ до інтерактивних сесій

на інших комп’ютерах. Програма підтримує зв’язок через послідовні порти і старі інтернет-протоколи, такі як Telnet. Основними особливостями програми є підтримка різних протоколів (SSH, Telnet, rlogin і TPC), портативність (не потребує інсталяції і може бути запущена з флешки) і відкритий код (доступний для вивчення або модифікації). Щоб правильно перевірити чи будуть йти дані на сервер треба під’єднатися через putty. Для цього потрібно знайти ip адресу. Щоб її знайти я буду використовувати команду ifconfig (Рис. 3.3):

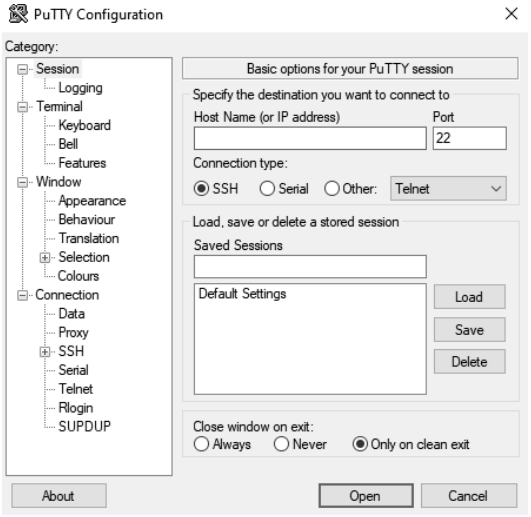


Рис. 3.2 Вигляд PuTTy

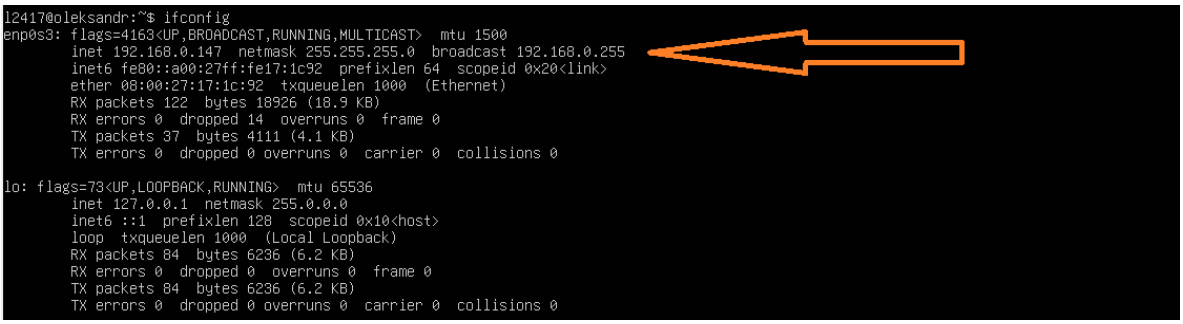


Рис. 3.3 Знаходження IP

Як можна побачити ми маємо ip-адресу (192.168.0.147). Тепер я записую цю адресу в PuTTy для підключення як pub клієнта, в той час як сервер буде sub клієнтом для отримання даних які буде публікувати юзер з PuTTy. Щоб зробити сервер sub клієнтом треба використовувати команду:

```
mosquitto_sub -t “temperature”
```

Роблю теж саме для pub клієнта який буде через PuTTy:

```
mosquitto_pub -t “temperature” -m “10.0”
```

Як можна побачити сервер прийняв повідомлення від pub клієнта (Рис. 3.4).

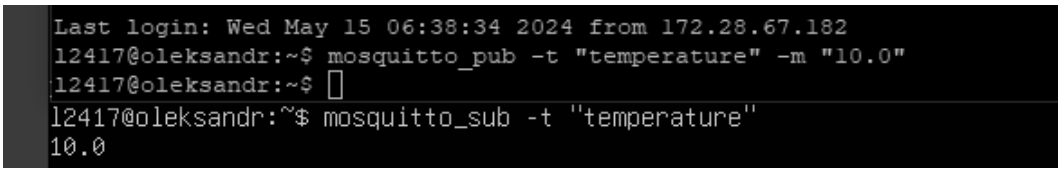


Рис. 3.4 Прийняте повідомлення

Тепер треба встановити на сервер Victoria Metrics. Victoria Metrics – це високоефективна і розширювана система управління базами даних, призначена для зберігання і обробки даних часових рядів, які складаються з набору значень, що відповідають певному часу і періодично збираються з датчиків або інших джерел. Вона оптимізована для швидкої та надійної роботи. Щоб почати працювати з нею, треба її встановити на сервер але для того щоб її можна було встановити, треба спочатку мати snapd за допомогою цієї команди:

```
$ sudo apt install snapd
```

І вже після встановлення snapd можна встановити Victoria Metrics за допомогою цієї команди:

```
$ sudo snap install victoriametrics
```

Після встановлення треба перевірити чи працює Victoria за допомогою команди:

```
$ sudo service snap.victoriametrics.victoriametrics status
```

І бачу що Victoria вже встановлена та працює (Рис. 3.5):

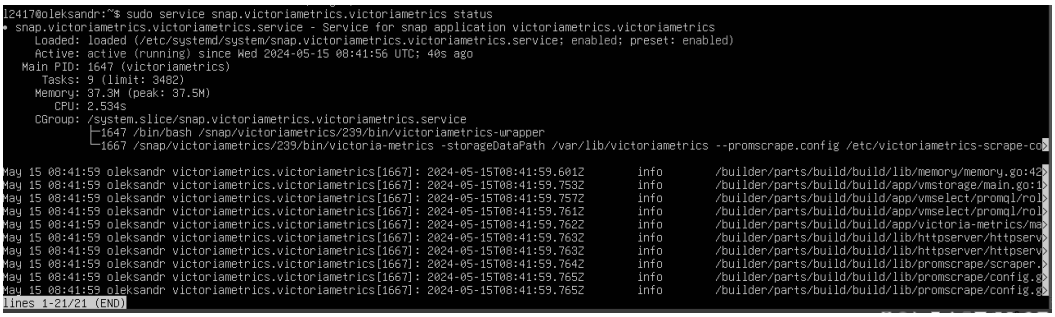


Рис. 3.5 Перевірка статусу Victoria Metrics

Тепер щоб перевірити чи все правильно зроблено, потрібно запустити любий браузер та вписати адресу «http://192.168.0.147:8428». Ось результат (Рис. 3.6).

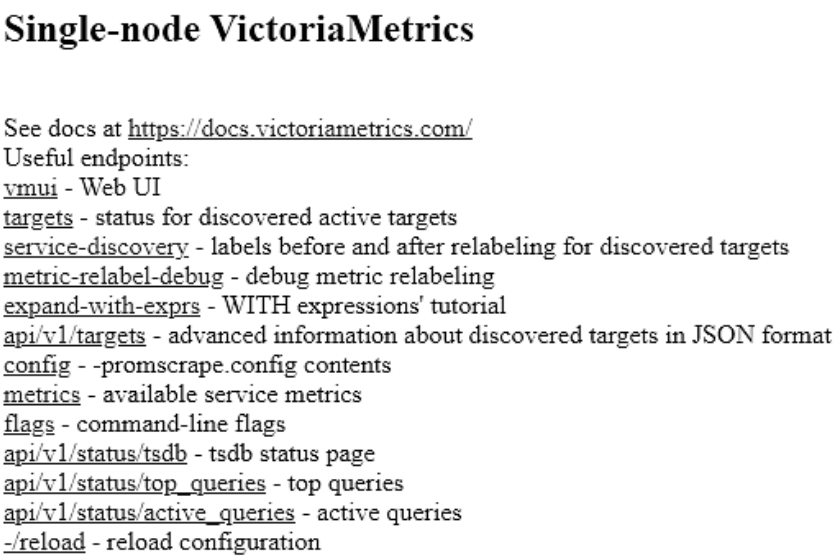


Рис. 3.6 Сторінка Victoria Metrics

3.3 Розробка системи для зберігання та відображення даних

Тепер треба написати код на python для автоматичного підпису на відповідні топіки для отримання даних та відправлення на Victoria Metrics. Для початку треба створити файл з розширенням «.py». Це робиться за допомогою команди:

```
$ nano sub.py
```

І починаю прописувати код. Спочатку треба зробити імпорт модулів. В мене будуть наступні модулі та бібліотека:

- paho.mqtt.subscribe – модуль з бібліотеки paho mqtt який використовується для підписки на топіки та отримання повідомлень від брокера.
- requests – бібліотека використовується для відправлення HTTP запитів. В коді вона буде відправляти дані на сервер Victoria Metrics.
- time – модуль буде використаний для роботи з часом.

Лістинг коду:

```
import paho.mqtt.subscribe as subscribe
import requests
import time
```

Далі роблю підключення до серверу Victoria Metrics. Я вписую «localhost» разом із портом 8428, який вказується за замовчуванням.

Лістинг коду:

```
dburl = "http://localhost:8428"
```

Наступним кроком буде визначення функції за допомогою команди «def ()». Там буде викликатися функція «on_message_print», коли буде приходити нове повідомлення.

Лістинг коду:

```
def on_message_print(client, userdata, message):
    ts = int(time.time()*1000)
    print("%s %s" % (message.topic, message.payload))
    if "temperature" in message.topic:
        jsonl = '{"metric": {"__name__": "temperature", "dev_name": "test1", "dev_type": "ds18b20"}, "values": ['+message.payload.decode('utf-8')+'], "timestamps": ['+str(ts)+']}\n'
        response = requests.post(dburl + '/api/v1/import', data = jsonl)
        print(message.payload.decode('utf-8'))
```

```
    if "loc" in message.topic:
        loc = message.payload.decode('utf-8').split(";")
        jsonl = '{"metric": {"__name__": "location", "position": "geo",
"dev_type": "neo-6m"}, "values": {'longitude': '"+loc[0]+"', 'latitude':
'"+loc[1]+"'}, "timestamp": ["'+str(ts)+"']}\n'


response = requests.post(dburl +  '"/api/v1/import"', data = jsonl)
        print(message.payload.decode('utf-8'))
```

Останнім кроком буде використання функції «subscribe.callback» для підписки на топік та встановлення функції зворотнього виклику, яка буде викликатися при отриманні повідомлення.

Лістинг коду:

```
subscribe.callback(on_message_print, "#", hostname="localhost")
```

Тепер роблю перевірку чи все буде правильно працювати. Для цього запускаю файл «sub.py» за допомогою команди:

```
$ python3 sub.py
```

І починаю перевірку. Для цього пишу команду в іншій сесії:

```
$ mosquito_pub -t “temperature” -m “10”
```

Там де був запущений файл виводить наступне: назву що саме прийшло (температура), назву його (test1), модель датчику та значення. Теж саме роблю з даними про розташування. Для цього я взяв випадкове місце з Google Maps. Пишу команду:

```
$ mosquito_pub -t “location” -m “51.51;30.51”
```

Виведення показників температури та місцезнаходження буде йти через Grafana. Grafana – це інструмент моніторингу та візуалізації даних з відкритим вихідним кодом, який дозволяє користувачам створювати красиві та інтерактивні графіки, діаграми та інші візуалізації для аналізу даних з різних джерел. Він широко

використовується для відображення інформації у вигляді легких для розуміння графіків і дашбордів, з основною метою спрощення процесу моніторингу, аналізу та візуалізації даних для оперативного прийняття рішень. Grafana підтримує інтеграцію з численними джерелами даних, такими як бази даних, метричні системи, веб-API та інші сервіси, що дозволяє користувачам отримувати дані з різних джерел і комбінувати їх для аналізу. Крім того, Grafana пропонує широкі можливості налаштування візуалізації, включаючи різні типи графіків, кольори, шрифти та стилі. Вона також підтримує різні способи організації даних на панелях, дозволяючи користувачам створювати зручні та інформативні дашборди для моніторингу та аналізу важливих метрик і показників.

Завдяки своїй гнучкості та простоті використання, Grafana стала популярним інструментом у різних сферах, включаючи системне адміністрування, розробку програмного забезпечення, інженерію надійності, моніторинг мереж та багато інших сфер, де візуалізація та аналіз даних мають вирішальне значення.

Спочатку для роботи з Grafana треба завантажити та встановити її. Це робиться за допомогою декількох команд:

```
$ wget https://dl.grafana.com/enterprise/release/grafana-enterprise_11.0.0_amd64.deb
$ sudo dpkg -i grafana-enterprise_11.0.0_amd64.deb
```

Після встановлення для перевірки можна зайти на його сторінку в браузері «http://192.168.0.147:3000» і побачити сторінку для входу (Рис. 3.7). Я вписав наступне username: admin, password: admin. Після входу запропонує щоб змінити пароль, я змінив на свій.

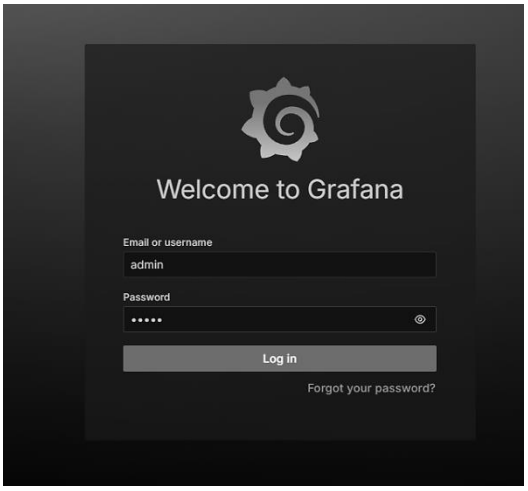


Рис. 3.7 Сторінка Grafana

Після цього можна побачити головну сторінку Grafana. Далі потрібно під’єднати в вкладці «Connection» Prometheus та вставити посилання на Victoria Metrics. Далі перехожу на вкладку «Dashboards» та створюю доску з якоюсь назвою. Наступним кроком буде створення «Visualization» для прикладу як буде відображатися температура та розташування. Для температури обирається візуалізація «Time series». В вкладці «Query» в «Data source» обираю «prometheus». В «metrics browser» вписую «query» «temperature{}». Після певного часу можна побачити, що дані які приймав сервер відображаються вже на графіку (Рис. 3.8).

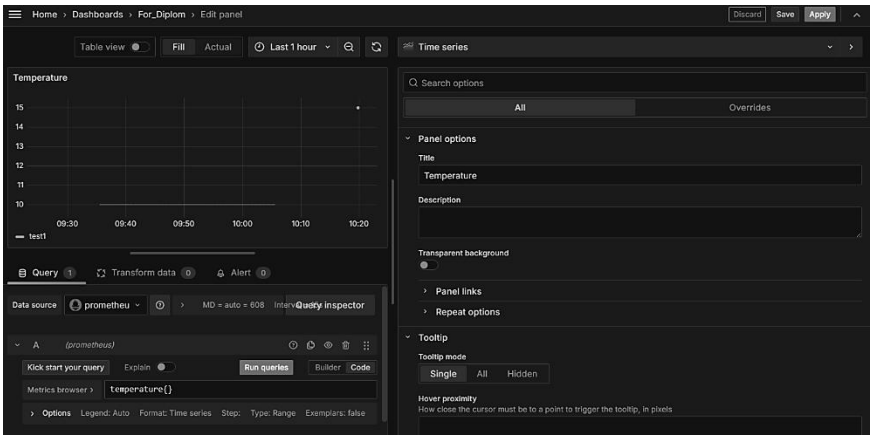


Рис. 3.8 Приклад виводу температури в пристрої

Для відображення розташування роблю знову кроки по створенню доски, тільки замість «Time series» обираю «Geomap». В вкладці «Query» в «Data source» знову обираю «Prometheus». В «metrics browser» вписую «query» «location{}» та обираю Після певного часу можна побачити, що дані які приймав сервер відображаються вже на графіку. Після певного часу можна побачити, що дані які приймав сервер відображаються вже на графіку (Рис. 3.9).

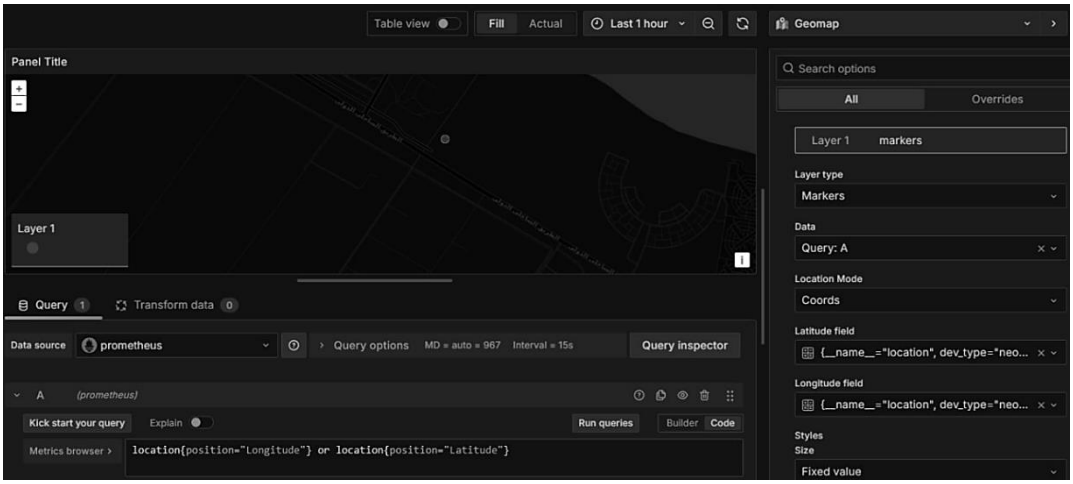


Рис. 3.9 Приклад виводу розташування пристрою

Висновок для розділу 3:

У цьому розділі було розроблено комплексну систему моніторингу параметрів, яка ефективно збирає та обробляє дані про температуру та місцезнаходження в режимі реального часу. Впровадження сучасних технологій Інтернету речей (IoT) забезпечує високий ступінь точності та надійності, що має вирішальне значення для численних застосувань, таких як управління рухомими об'єктами, моніторинг навколишнього середовища та інші інтелектуальні рішення. Простий та ефективний спосіб передачі даних між пристроями забезпечує протокол MQTT. MQTT був обраний через його здатність працювати в середовищах з обмеженими ресурсами, мінімальне енергоспоживання і високу швидкість передачі даних, а також через його здатність працювати в середовищах з обмеженими ресурсами. Безперервний потік даних про температуру та місцезнаходження був забезпечений установкою брокера Mosquito MQTT. Збір та обробка даних здійснювалася за допомогою мови програмування Python. Механізм підписки на MQTT-топи та обробки отриманих повідомлень був реалізований завдяки бібліотеці Paho-MQTT. Можливість швидкого розширення функціоналу системи дозволяє легко адаптувати її до різних умов експлуатації.

Зібрана інформація зберігалася в базі даних Victoria Metrics, яка відома своєю швидкістю та адаптивністю. Надійне зберігання великих обсягів даних з мінімальними затримками було забезпечено завдяки використанню Victoria

Metrics. Доступність інформації про температуру та місцезнаходження для подальшого дослідження та обробки відкриває численні можливості для їх застосування в різноманітних наукових дослідженнях.

Для візуалізації зібраних даних було інтегровано Grafana для створення інтерактивних та інформативних дашбордів. Зміни температури та місцезнаходження можна візуально відображати в режимі реального часу, що значно покращує зручність використання системи та її інформативність.

ВИСНОВКИ

Впровадження IoT може допомогти в контролі якості продуктів харчування, ліків ті ін. під час транспортування та зберігання.

У ході роботи було проведено аналіз вже існуючих систем для моніторингу та контролю якості харчових продуктів, розроблено схему пристрою. Були підібрані всі необхідні компоненти, включаючи мікроконтролер Arduino Uno, дисплей, датчик температури DS18B20, GPS та GSM модулі та транзистор IRL540N. Переваги пристрою на основі модуля Пельтьє, керованого мікроконтролером Arduino Uno, включають в себе високу точність і стабільність, завдяки використанню модулю Пельтьє і програмному управлінню. Це особливо важливо для деяких дослідницьких або промислових застосувань. Мікроконтролер Arduino Uno також надає можливість легко налаштовувати параметри і режими роботи температурного стенду, забезпечуючи гнучкість в налаштуванні і управлінні процесом. Крім того, було показано приклад як повинен в майбутньому працювати вже готовий пристрій.

Роботу представленого пристрою можна покращити, додавши більше модулів Пельтьє, але це збільшить споживання струму та вплине на інші параметри, які необхідно враховувати.

ПЕРЕЛІК ПОСИЛАНЬ

1. A secure food supply chain solution: blockchain and IoT-enabled container to enhance the efficiency of shipment for strawberry supply chain. *Frontiers*. URL: https://www.frontiersin.org/articles/10.3389/fsufs.2023.1294829/full#:~:text=IoT%20can%20enable%20container%20supply,starting%20to%20the%20ending%20point. (date of access: 22.05.2024).
2. A study on IoT application in the Food Industry using Keywords Analysis. *ScienceDirect*. URL: https://www.sciencedirect.com/science/article/pii/S1877050922003921 (date of access: 22.05.2024).
3. Cold Chain 101: What is the Cold Chain? | Nordic Cold Chain Solutions. *Nordic Cold Chain Solutions*. URL: https://www.nordiccoldchain.com/cold-chain-101-cold-chain/ (date of access: 22.05.2024).
4. Cold Chain Logistics - Cold Chain Management. *Maersk*. URL: https://www.maersk.com/ru-ru/supply-chain-logistics/cold-chain-logistics/cold-chain-management (date of access: 22.05.2024).
5. Compare OpenTSDB vs VictoriaMetrics. *InfluxDB Time Series Data Platform | InfluxData*. URL: https://www.influxdata.com/comparison/tsdb-vs-victoria/ (date of access: 22.05.2024).
6. Compare Prometheus vs VictoriaMetrics. *InfluxDB Time Series Data Platform | InfluxData*. URL: https://www.influxdata.com/comparison/prometheus-vs-victoria/ (date of access: 22.05.2024).
7. GEMAN O. An Intelligent IoT-Based Food Quality Monitoring Approach Using Low-Cost Sensors. *Academia.edu - Share research*. URL: https://www.academia.edu/83017869/An_Intelligent_IoT_Based_Food_Quality_Monitoring_Approach_Using_Low_Cost_Sensors (date of access: 22.05.2024).

8. Geomap | Grafana documentation. *Grafana Labs*. URL: https://grafana.com/docs/grafana/latest/panels-visualizations/visualizations/geomap/#location (date of access: 22.05.2024).
9. How Internet of Things Is Transforming the Food Industry. *Compare the Cloud*. URL: https://www.comparethecloud.net/articles/how-internet-of-things-transforming-food-industry/ (date of access: 22.05.2024).
10. How IoT is Revolutionizing the Food Industry - TagoIO. *IoT Cloud Platform | TagoIO*. URL: https://tago.io/blog/iot-revolutionizing-food-industry (date of access: 22.05.2024).
11. How the IoT is supporting food processing businesses | Rentokil. *The global experts in pest control | Rentokil*. URL: https://www.rentokil.com/blog/food-safety/iot-food-processing (date of access: 22.05.2024).
12. Install and configure VictoriaMetrics on Debian | Vultr Docs. *Vultr Docs | Start Building on The Everywhere Cloud*. URL: https://docs.vultr.com/install-and-configure-victoriametrics-on-debian (date of access: 22.05.2024).
13. Internet of Things enabled real time cold chain monitoring in a container port - Journal of Shipping and Trade. *SpringerOpen*. URL: https://jshippingandtrade.springeropen.com/articles/10.1186/s41072-022-00110-z (date of access: 22.05.2024).
14. Locke J. How IoT Temperature Sensors Are Revolutionizing the Cold Chain and Retail. *IIoT Devices and Services for M2M Networking | Digi International*. URL: https://www.digi.com/blog/post/iot-temperature-sensors-revolutionize-cold-chain (date of access: 22.05.2024).
15. Master the time-series database (TSDB). *QuestDB | High performance time series database*. URL: https://questdb.io/glossary/time-series-database/ (date of access: 22.05.2024).
16. Ndungu F. Install Mosquitto MQTT Broker On Ubuntu 20.04 Server | Vultr Docs. *Vultr Docs | Start Building on The Everywhere Cloud*. URL: https://docs.vultr.com/install-mosquitto-mqtt-broker-on-ubuntu-20-04-server (date of access: 22.05.2024).

17. Secure Transport of COVID-19 Vaccines with IoT Cold Chain Monitoring. *Innovate*. URL: https://innovate.ieee.org/innovation-spotlight/cold-chain-monitoring/ (date of access: 22.05.2024).
18. SIM800 Series MQTT Application Note. *Microchip 2G Documentation*. URL: https://microchip.ua/simcom/2G/Application%20Notes/SIM800%20Series_MQTT_Application%20Note_V1.03.pdf (date of access: 22.05.2024).
19. What is a relational database?. *Oracle | Cloud Applications and Cloud Platform*. URL: https://www.oracle.com/ua/database/what-is-a-relational-database/ (date of access: 22.05.2024).
20. Як встановити Python 3.10 на Ubuntu 20.04 | Zomro. *Zomro.com | Reliable Hosting | VPS/VDS server at an affordable price*. URL: https://zomro.com/ua/blog/faq/299-kak-ustanovit-python-310-na-ubuntu-2004 (дата звернення: 22.05.2024).

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО - КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення автоматизованих систем

КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Застосування IoT для моніторингу та контролю якості харчових продуктів під час транспортування та зберігання»

на здобуття освітнього ступеня бакалавра
зі спеціальності 126 Інформаційні системи та технології
освітньо-професійної програми Інформаційні системи та технології

Виконав: Олександр Бовкун, ІСД-42
Науковий керівник роботи:
Хоменчук В.О.

Київ - 2024

Актуальність теми: Найважливішим завданням у сучасному світі є забезпечення безпечності та якості харчових продуктів. Зростання чисельності населення планети та збільшення обсягів світової торгівлі призвели до збільшення обсягів перевезень харчових продуктів на великі відстані, а отже, до ризику їх псування та втрати якості.

Об'єкт дослідження: процес моніторингу та контролю якості харчових продуктів під час транспортування та зберігання.

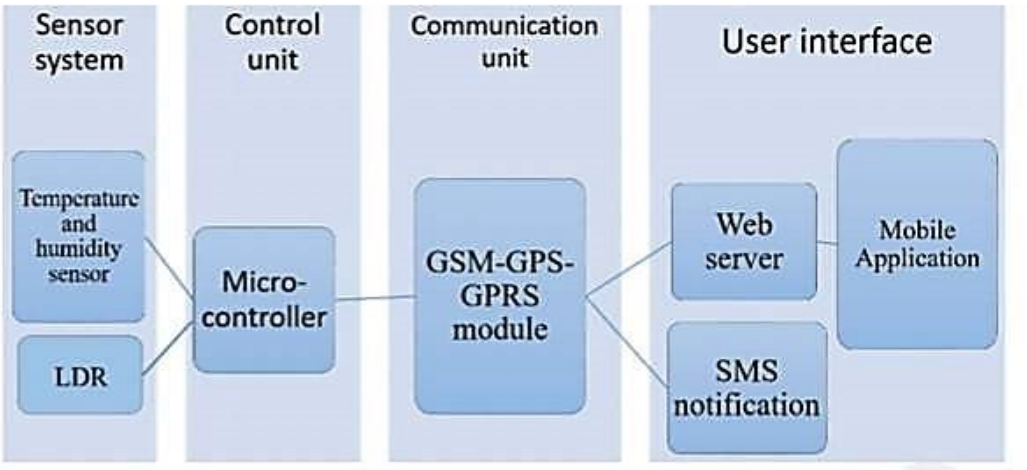
Предмет дослідження : пристрій для моніторингу та контролю якості харчових продуктів.

Мета дослідження : розробка пристрою, який буде запрогромований для моніторингу та забезпечення свіжості та безпечності продуктів харчування протягом транспортування та зберігання.

Завдання дослідження :

- аналіз існуючих систем моніторингу;
- обрати компоненти для пристрою та серверу та обґрунтувати цей вибір;
- розробити серверну частину, обрати компоненти для отримання даних;
- підключити пристрій до сервера та вивести отримані дані.

Схема системи моніторингу



3

Компоненти для пристрою

Для пристрою були обрані наступні компоненти:
1) Датчик температури DS18B20;
2) Елемент Пельтьє TEC1-12706;
3) Мікроконтролер Arduino Uno;
4) GPS-модуль NEO-6M;
5) GSM-модуль SIM800C;
6) LDC дисплей I2C QC1602A.


DS18B20


Arduino UNO


LCD I2C QC1602A


TEC1-12706


GPS NEO-6M


GSM SIM800C

4

SIM800C V2 для відправки даних на брокер MQTT Mosquitto

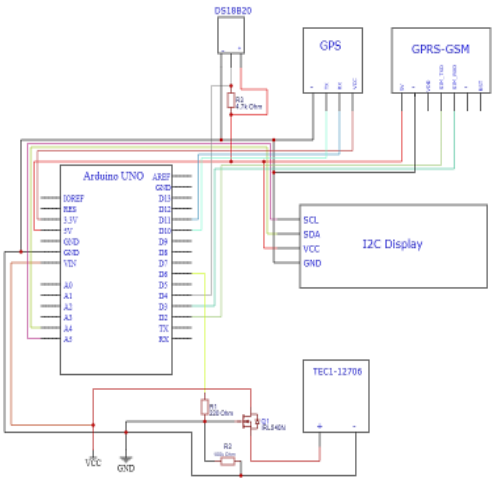
Модем SIM800C V2 – це універсальний GSM/GPRS модуль, що став невід'ємною складовою різноманітних IoT проектів для бездротового зв'язку. Він відзначається наступними особливостями:

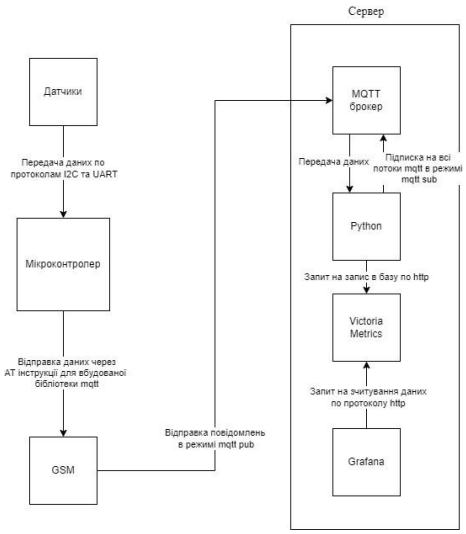
SIM800C V2

1. Вбудована MQTT підтримка: Завдяки цьому, використання модуля для передачі даних на MQTT брокери, такі як Mosquitto, стає набагато простішим. Всі необхідні функції для роботи з MQTT вже інтегровані в модем, що полегшує його використання.
2. Прості AT команди: Вони забезпечують налаштування параметрів з'єднання, підключення до мережі, підписку на топіки та публікацію повідомлень. Це означає, що можна використовувати ці команди для підписки на конкретні топіки в протоколі MQTT, щоб отримувати повідомлення, що надходять на ці топіки. Цей процес робиться максимально зручним та доступним завдяки простоті команд та їхній легкій зрозумілості.

5

Принципіальна схема пристрою та архітектура системи





6

Написання коду для пристрою за допомогою Arduino IDE

У «void setup()» здійснюються початкові налаштування для компонентів системи і включає наступні кроки:

1. Ініціалізація GSM модуля та підключення до мережі та налаштування MQTT брокера за допомогою AT команд;
2. Налаштування елемента Пельтьє;
3. Ініціазізація температурного датчика та LCD дисплея.

```
void setup() {
 Serial.begin(9600);
 modem.begin();
 modem.gprsConnect("internet");
 // Відправка AT команди до GSM модуля для налаштування адреси брокера
 modem.sendAT(GF("+SMCONF=\"URL\",\""), 192.168.0.147, GF("\""));
 // Налаштування та підключення до MQTT брокера
 modem.waitResponse();
 modem.sendAT(GF("+SMCONF=\"CLIENTID\",\"ArduinoClient\""));
 modem.waitResponse();
 modem.sendAT(GF("+SMCONN"));
 modem.waitResponse();

 pinMode(TEC_PIN, OUTPUT);
 digitalWrite(TEC_PIN, LOW);

 sensors.begin();
 lcd.init();
 lcd.backlight();
 lcd.setCursor(1, 0);
 lcd.print("Hello");
 delay(2000);
 lcd.clear();
}
```

7

Написання коду для пристрою за допомогою Arduino IDE

Основний цикл «void loop()» виконується постійно і включає такі кроки:

1. Відображення на дислпеї температури;
2. Читання температури з датчика DS18B20 та відображення її на дисплеї;
3. Перевірка даних від GPS модуля і відправка координат та температури на MQTT брокер;
4. Публікація температури на MQTT брокер з інтервалом в 1 секунду;

```
void loop() {
 sensors.requestTemperatures();
 float tempC = sensors.getTempCByIndex(0);
 lcd.setCursor(0, 1);
 lcd.print("Temp: ");
 lcd.print(tempC);
 lcd.print(" C");
 // Обробка даних з GPS та відправка на брокер
 while (Serial.available() > 0) {
  gps.encode(Serial.read());
  if (gps.location.isUpdated()) {
   float lat = gps.location.lat();
   float lon = gps.location.lng();
   char msg[50];
   snprintf(msg, sizeof(msg), "Location: %f,%fTemp: %f",lat, lon, tempC);
   modem.sendAT(GF("+SMPUB=\""), topicLoc, GF("\","), strlen(msg), GF(",1,0,\""), msg, GF("\""));
   modem.waitResponse();
  }
 }
 // Публікація температури на брокер кожну секунду
 unsigned long now = millis();
 if (now - lastMsg > 1000) {
  lastMsg = now;
  ++value;
  snprintf (msg, MSG_BUFFER_SIZE, "%ld", tempC);
  Serial.print("Pub. message: ");
  Serial.println(msg);
  modem.sendAT(GF("+SMPUB=\""), topicTemp, GF("\","), strlen(msg), GF(",1,0,\""), msg, GF("\""));
  modem.waitResponse();
 }
```

8

Налаштування системи для отримання даних

Для того щоб сервер збирав дані про температуру та розташування пристрою, треба написати код на мові програмування Python Для цього створюється файл «sub.py» щоб виконувалося наступне:

1. Визначається база даних (dburl);
2. Обробка повідомлень за допомогою функції «def»;
3. Обробка повідомлень за топіками «temperature» та «loc».

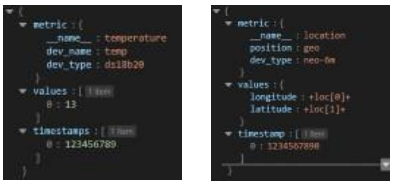
```
import paho.mqtt.subscribe as subscribe
import requests
import time

dburl= "http://localhost:8428"

def on_message_print(client, userdata, message):
    ts = int(time.time()*1000)
    print("%s %s" % (message.topic, message.payload))
    if "temperature" in message.topic:
        jsonl = '{"metric": {"__name__": "temperature", "dev_name": "test1", "dev_type": "ds18b20"}, "values": ["'+message.payload.decode('utf-8')+'"], "timestamps": ["'+str(ts)+'"]}\n'
        response =requests.post(dburl+ "/api/v1/import", data =jsonl)
        print(message.payload.decode('utf-8'))
    if "loc" in message.topic:
        loc = message.payload.decode('utf-8').split(";")
        jsonl = '{"metric": {"__name__": "location", "position": "geo", "dev_type": "neo-6m"}, "values": {'longitude': "'+loc[0]+'", 'latitude': "'+loc[1]+'"}, "timestamp": ["'+str(ts)+'"]}\n'

response =requests.post(dburl + "/api/v1/import", data =jsonl)
        print(message.payload.decode('utf-8'))

subscribe.callback(on_message_print, "#", hostname="localhost")
```

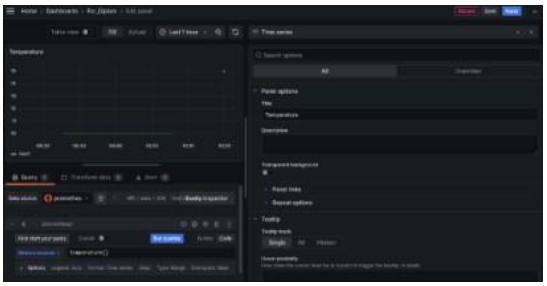


Jsonl файл з отриманими даними

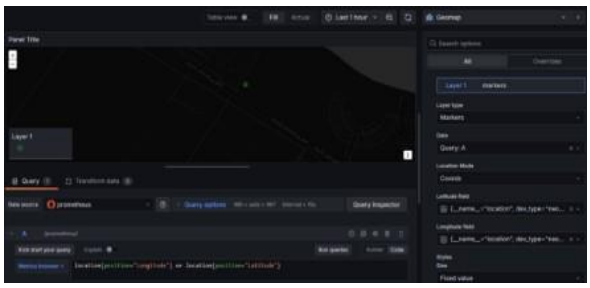
Відображення отриманих даних

Для відображення даних використовується на сервері встановлений веб-додаток з відкритим кодом для аналітики та інтерактивної візуалізації Grafana. Вона бере дані з time series data base Victoria Metrics. Victoria Metrics - це швидке, економічно ефективне і масштабоване рішення для моніторингу та база даних часових рядів, яка зазвичай використовується для обробки великих обсягів даних і довгострокового зберігання. На двох фото можна побачити як будуть відображатися отримані дані температури та розташування .

Температура:



Розташування:



Висновки

Впровадження IoT технологій може значно покращити контроль якості продуктів харчування та ліків під час їх транспортування і зберігання. У ході роботі було проведено аналіз існуючих систем моніторингу та розроблено пристрій на основі мікроконтролера Arduino Uno з використанням дисплея, датчика температури DS18B20, GPS та GSM модулів, а також транзистора IRL540N. Пристрій забезпечує високу точність і стабільність завдяки модулю Пельтьє, що є ключовим для промислових застосувань. Arduino Uno дозволяє легко налаштовувати параметри та режими роботи, що додає гнучкості у управлінні процесом.

Представлено приклад роботи готового пристрою, з можливістю подальшого вдосконалення шляхом додавання додаткових модулів Пельтьє, що вплине на споживання струму та інші параметри.

11

Апробація результатів дослідження

V Міжнародна науково-технічна конференція «Сучасний стан та перспективи IoT», яка проходила 18 квітня 2024 року. Теза на тему «Застосування IoT для моніторингу та контролю якості харчових продуктів під час транспортування та зберігання» було опубліковано у збірнику, присвяченому цій конференції .

Дякую за увагу!

12