МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет

імені Петра Могили

Факультет комп'ютерних наук

Кафедра комп'ютерної інженерії

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри,
д-р техн. наук, проф.

_____ І. М. Журавська

« __ » _____ 2024 р.

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

Система сповіщення змін кліматичних умов теплиці на базі AWS IoT

Спеціальність 123 Комп'ютерна інженерія

123 – КБР.01 – 405.22010604

Студент _____ М. С. Дмитришин
підпис
«__» _____ 202__ р.

Керівник ст. викладач _____ Є. С. Дарнапук
підпис
«__» _____ 202__ р.

Миколаїв – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет імені Петра Могили
Факультет комп'ютерних наук
Кафедра комп'ютерної інженерії

ЗАТВЕРДЖУЮ

Зав. кафедри _____ І. М. Журавська

«_____» _____ 2024 р.

ЗАВДАННЯ
на виконання кваліфікаційної бакалаврської роботи

Видано студенту групи 405 факультету комп'ютерних наук

_____ Дмитришин Максим Сергійович _____
(прізвище, ім'я, по батькові студента)

1. Тема кваліфікаційної роботи

__ Система сповіщення змін кліматичних умов теплиці на базі AWS IoT ___

Затверджена наказом по ЧНУ ім. Петра Могили від 30.01.2024 № 17.

2. Строк представлення кваліфікаційної роботи «____» _____ 20__ р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні
Розроблений проттотип система сповіщення змін кліматичних умов теплиці на базі AWS IoT _____
_____
_____
_____
_____

4. Перелік питань, що підлягають розробці
Аналіз існуючих рішень у сфері моніторингу та управління мікрокліматом у теплицях._____
Розробка алгоритмів для збору та обробки даних з датчиків кліматичних умов.
Проектування системи сповіщення на базі AWS IoT._____
Тестування системи та аналіз її ефективності та надійності._____
_____

5. Перелік графічних матеріалів
Схема підключення датчиків до контролера. _____
Діаграма архітектури системи моніторингу та сповіщення на базі AWS IoT__
_____
_____

6. Завдання до спеціальної частини
Проведення оцінки умов праці фахівців підприємства ТОВ «Глабллоджик Україна»_____

7. Консультанти:

Консультант	Кафедра (організація)	Частина роботи
Алексєєва Анна Олександрівна	Кафедра екології	Спеціальна частина з охорони праці

Керівник роботи
____ старший викладач Дарнапук Євгеній Сергійович _____
(посада, прізвище, ім'я, по батькові)

_____
(підпис)

Завдання прийнято до виконання

_____Дмитришин Максим Сергійович_____
(прізвище, ім'я, по батькові студента)

_____
(підпис)

Дата видачі завдання «____» _____ 20_____ р.

КАЛЕНДАРНИЙ ПЛАН

виконання кваліфікаційної роботи

Тема: Система сповіщення змін кліматичних умов теплиці на базі AWS IoT _ _____

№	Найменування роботи	Початок	Закінчення	Примітки
1	Розробка та затвердження завдання на виконання КБР	01.02.2024	10.02.2024	Виконав
2	Огляд літератури за темою роботи	11.02.2024	27.02.2024	Виконав
3	Складання календарного плану КБР	25.02.2024	01.03.2024	Виконав
4	Аналіз предметної області	05.03.2024	30.03.2024	Виконав
5	Розробка проєктних рішень	01.04.2024	30.04.2024	Виконав
6	Перевірка працездатності, тестування та апробація розробленого АПЗ, аналіз результатів тестування	01.04.2024	28.05.2024	Виконав
7	Відгук керівника КБР	06.06.2024	08.06.2024	Виконав
8	Оформлення КБР та презентації	15.05.2024	05.06.2024	Виконав
9	Попередній захист	28.05.2024	05.06.2024	Виконав
10	Рецензування	10.06.2024	14.06.2024	Виконав
11	Завершення оформлення КБР та презентації	15.06.2024	17.06.2024	Виконав
12	Захист бакалаврської кваліфікаційної роботи	26.06.2024	28.06.2024	Виконав
13				

Розробив здобувач ВО ____ Дмитришин Максим Сергійович ____

(прізвище, ім'я, по батькові) *(підпис)*

«____» _____ 20__ р.

Керівник роботи _ старший викладач кафедри Дарнапук Євген Сергійович _ _____

(посада, прізвище, ім'я, по батькові) *(підпис)*

«____» _____ 20__ р.

АНОТАЦІЯ

до кваліфікаційної бакалаврської роботи

«Система сповіщення змін кліматичних умов теплиці на базі AWS IoT»

Студент 405гр.: Дмитришин Максим Сергійович

Керівник: ст. викладач Дарнапук Євген Сергійович

У роботі розглянуто розробку системи сповіщення змін кліматичних умов теплиці на базі AWS IoT. Актуальність використання IoT технологій у сільському господарстві обумовлена зростаючою потребою в автоматизації та підвищенні ефективності аграрних процесів. Мета роботи - створення надійної системи моніторингу та сповіщення про зміни кліматичних умов у теплиці з використанням AWS IoT.

Розділ 1 містить теоретичний огляд IoT технологій, аналіз платформи AWS IoT та середовище розробки. У розділі 2 формуються вимоги до апаратної і програмної частини системи, включаючи функціональні та нефункціональні вимоги. Розділ 3 присвячений розробці системи, де детально описуються архітектура, вибір компонентів, підключення та конфігурація датчиків, програмування ESP32, налаштування AWS IoT та інтеграція з хмарними сервісами. Також описується інтерфейс для отримання даних з датчиків. У висновках підсумовується проведена робота та її результати.

Робота містить 77 сторінки, 22 рисунки, 4 таблиці, перелік з 18 джерел посилання та 3 додатки.

Ключові слова: *AWS IoT, система сповіщення, зміни кліматичних умов, теплиця, IoT технології, ESP32, хмарні сервіси, сільське господарство.*

ABSTRACT

of the Bachelor’s Thesis

"Greenhouse Climate Change Notification System Based on AWS IoT"

Student: Maksym Dmytryshyn Serhiyovych

Supervisor: sr. lecturer Yevhen Darnapuk Serhiyovych

The paper examines the development of a greenhouse climate change notification system based on AWS IoT. The relevance of using IoT technologies in agriculture is due to the growing need for automation and increased efficiency of agricultural processes. The goal of the work is to create a reliable system for monitoring and notifying climate changes in the greenhouse using AWS IoT.

Section 1 contains a theoretical overview of IoT technologies, an analysis of the AWS IoT platform, and the development environment. In Section 2, the requirements for the hardware and software components of the system are formed, including functional and non-functional requirements. Section 3 is devoted to the system development, where the architecture, component selection, sensor connection and configuration, ESP32 programming, AWS IoT setup, and integration with cloud services are described in detail. The interface for receiving data from the sensors is also described. The conclusions summarize the work done and its results.

The work contains 77 pages, 22 figures, 4 tables, a list of 18 references, and 3 appendices.

Keywords: *AWS IoT, notification system, climate change, greenhouse, IoT technologies, ESP32, cloud services, agriculture.*

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ 3
ВСТУП 4
1 Теоретічній огляд 6
1.1 Огляд IoT технологій у сільському господарстві 6
1.2 Аналіз платформи AWS IoT та її компонентів 10
1.3 Середа розробки 14
Висновки до розділу 15
2 ФОРМУВАННЯ ВИМОГ 18
2.1 Апаратні вимоги 18
2.2 Програмні вимоги 25
2.3 Функціональні вимоги 29
2.4 Нефункціональні вимоги 31
Висновки до розділу 35
3 РОЗРОБКА 36
3.1 Опис архітектури системи 36
3.2 Вибір компонентів 39
3.3 Підключення та конфігурація датчиків 40
3.4 Програмування ESP32 56
3.5 Налаштування AWS IoT та інтеграція з хмарними сервісами 61
3.6 Інтерфейс для отримання даних з датчиків 63
Висновки до розділу 65
ВИСНОВКИ 67
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ 70
ДОДАТОК А Звіт з антиплагіату 72

ПЕРЕЛІК СКОРОЧЕНЬ

ШІ – штучний інтелект

API – Application Programming Interface
AWS – Amazon Web Services
BMP – Barometric Pressure Sensor
DHT – Digital Humidity and Temperature
ESP – Espressif Systems
HTTP – HyperText Transfer Protocol
I2C – Inter-Integrated Circuit
IoT – Internet of Things
MQTT – Message Queuing Telemetry Transport
RH – Relative Humidity
SES – Simple Email Service
SNS – Simple Notification Service
SPI – Serial Peripheral Interface
UART – Universal Asynchronous Receiver-Transmitter
Wi-Fi – Wireless Fidelity

ВСТУП

У сучасному світі значення прецизійного сільського господарства невпинно зростає, особливо в контексті глобальних змін клімату та зменшення природних ресурсів. Ефективне управління теплицями є ключовим аспектом забезпечення сталого виробництва продукції. Створення точної і автоматизованої системи сповіщення, яка відстежує та реагує на зміни кліматичних умов, може суттєво підвищити ефективність управління мікрокліматом у теплицях.

Ця кваліфікаційна робота фокусується на розробці та аналізі автоматизованої системи сповіщення змін кліматичних умов у теплицях, що базується на технології Amazon Web Services (AWS) Internet of Things (IoT). Основна мета проєкту полягає в створенні надійної системи, здатної забезпечувати оперативне спостереження та реагування на коливання кліматичних параметрів внутрішнього середовища, що дозволить оптимізувати управління умовами зростання агрокультур. Завдання включають детальний аналіз існуючих систем, розробку алгоритмів для обробки даних, проєктування інтегрованої системи сповіщень, а також її тестування та налагодження для гарантування ефективності та надійності.

Предметом дослідження є алгоритми збору, обробки та аналізу даних з сенсорів, які моніторять температуру, вологість, рівень CO_2 та інші критичні параметри. Також у рамках роботи буде вивчено можливості платформи AWS IoT для інтеграції цих даних та їхнього використання у прийнятті рішень. [6, 18]

Об'єктом дослідження є автоматизована система сповіщення змін кліматичних умов у теплицях на базі технологій AWS IoT, зокрема алгоритми збору, обробки та аналізу даних з сенсорів для моніторингу температури, вологості, рівня CO2 та інших критичних параметрів, а також можливості інтеграції цих даних для оперативного управління мікрокліматом у теплицях.

Основні завдання, які ставляться перед дипломною роботою:

– аналіз існуючих рішень у сфері моніторингу та управління мікрокліматом у теплицях;
– розробка алгоритмів для збору та обробки даних з датчиків кліматичних умов;
– проєктування системи сповіщення на базі AWS IoT, забезпечення її інтеграції з обладнанням теплиці;
– тестування системи та аналіз її ефективності та надійності.

У результаті цієї дипломної роботи планується розробити та впровадити автоматизовану систему моніторингу та сповіщення, яка забезпечить точне та ефективне управління кліматичними умовами в теплиці. Система буде інтегрована з AWS IoT для забезпечення безперервного збору даних, аналізу та реагування на зміни умов у реальному часі. Це дозволить не тільки трекінг параметрів, таких як температура, вологість та рівень CO_2, але й автоматично адаптувати умови для оптимізації здоров'я та продуктивності рослин. [15] Такий підхід забезпечить вищу точність у моніторингу та контролі тепличного середовища, значно підвищивши його ефективність.

1 ТЕОРЕТІЧНІЙ ОГЛЯД

Використання IoT технологій у сільському господарстві є потужним фундаментом для підвищення ефективності аграрних підприємств через автоматизацію та точне керування сільськогосподарськими процесами. Завдяки можливостям збору даних в реальному часі, аграрії можуть отримати доступ до важливої інформації про стан ґрунтів, вологість, температуру, хімічний склад та інші критичні параметри, що впливають на зростання рослин. Інтеграція IoT не лише дозволяє збільшити врожайність та зменшити втрати, але й сприяє сталому використанню ресурсів за рахунок оптимізації водного та поживного балансу.

1.1 Огляд IoT технологій у сільському господарстві

В рамках цього розділу буде розглянуто ключові аспекти та переваги використання IoT технологій у сільському господарстві, аналізуючи приклади успішних імплементацій та досліджуючи потенціал для подальшого розвитку. Зокрема, зосередимо увагу на системах автоматизації поливу, моніторингу здоров'я тварин, а також управлінні мікрокліматом у теплицях, де IoT відіграє ключову роль у підтримці оптимальних умов для росту рослин. [2]

1.1.1 Приклади використання

Одним з прикладів використання IoT технологій у сільському господарстві є системи точного землеробства (англ. Precision Farming). Цей підхід передбачає використання передових технологій та інтелектуальних датчиків для моніторингу та управління землеробськими полями з високою точністю. Системи точного землеробства дозволяють аграріям збирати дані про стан ґрунту, вміст поживних речовин, вологість, температуру, а також інші важливі параметри в реальному часі.[7]

Ці дані використовуються для точного дозування води, добрив та інших ресурсів, що не тільки забезпечує більш ефективне використання входів, але й

значно знижує витрати, покращує якість та кількість врожаю, а також мінімізує вплив на довкілля. Крім того, за допомогою IoT можна реалізувати автоматизацію процесів, таких як полив, обробка та збір врожаю, що додатково знижує необхідність людської праці та підвищує загальну продуктивність операцій.

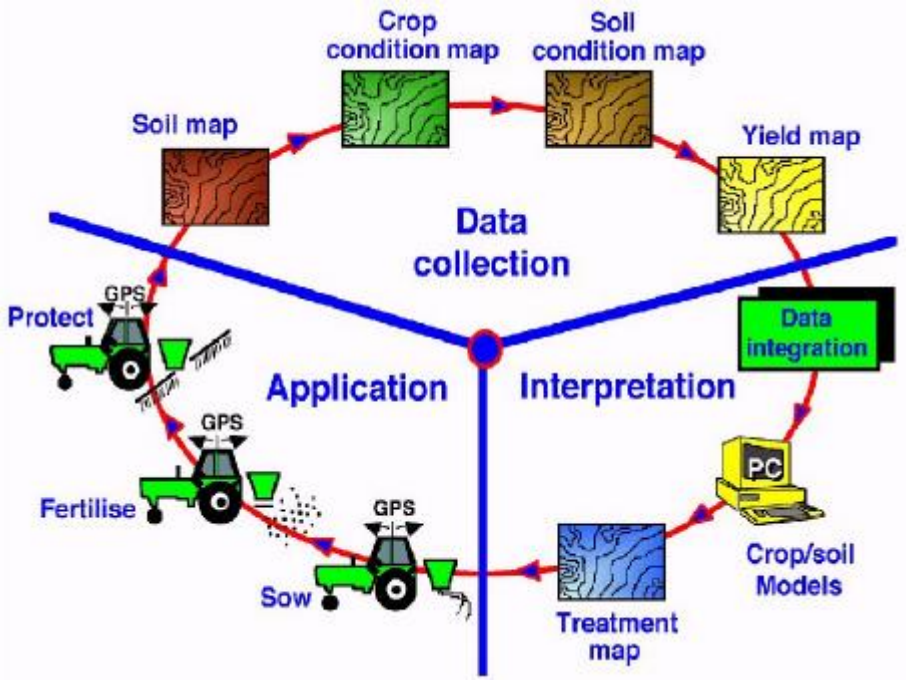


Рисунок 1.1 – Діаграма системи точного землеробства

Іншим прикладом використання таких технологій може слугувати автоматизоване управління поливом. [17] Ця система інтегрує датчики вологості ґрунту та погодні станції для точного моніторингу умов на полі, дозволяючи здійснювати полив рослин виключно за потреби. Автоматизовані системи поливу не тільки забезпечують рослини необхідною кількістю води, але й сприяють економії водних ресурсів шляхом оптимізації їх використання [10].

Такий підхід відіграє ключову роль у зниженні витрат та підвищенні ефективності аграрного виробництва. Сучасні IoT рішення дозволяють аграріям налаштовувати полив на основі даних про погоду та вологість, а також забезпечують можливість віддаленого керування та моніторингу через мобільні додатки або комп'ютери. Це відкриває широкі можливості для

прогресивного управління ресурсами на фермі, покращуючи усталеність та продуктивність сільськогосподарських операцій.

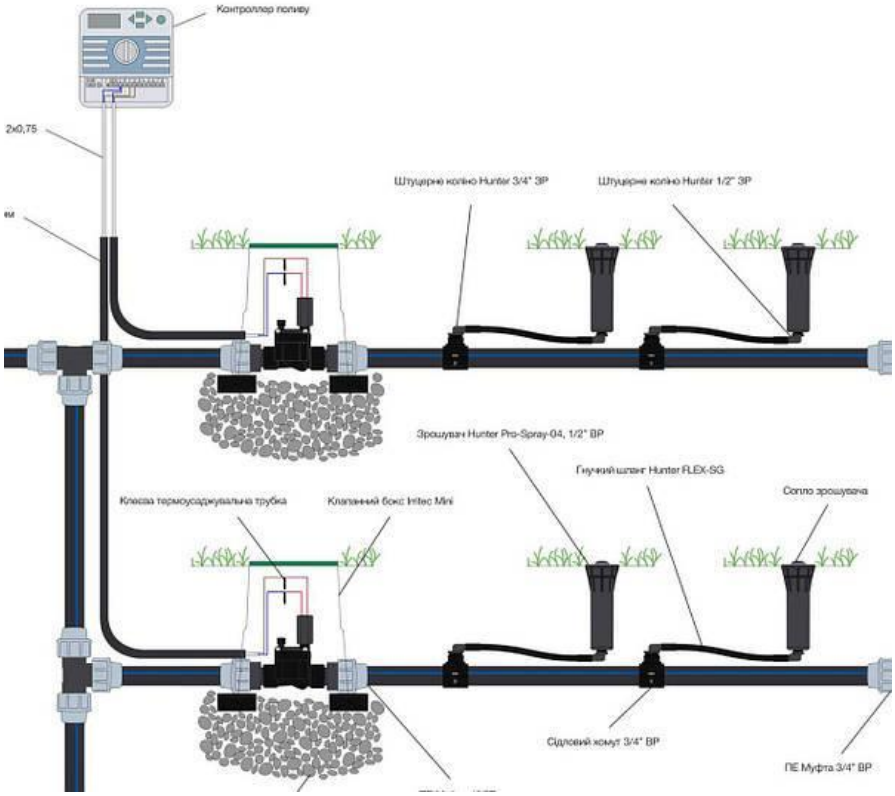


Рисунок 1.2 – Система автоматичного поливу

Також існують технології інтегрованого управління відходами, що є важливою складовою сучасного сільського господарства, де IoT відіграє ключову роль у ефективній утилізації та переробці аграрних відходів. Ці системи використовують датчики для моніторингу стану відходів на фермах, у теплицях або на переробних заводах, забезпечуючи оптимізацію процесів збору, сортування, переробки та ліквідації відходів. [7, 11]

Інтегроване управління відходами за допомогою IoT може включати наступні аспекти:

– моніторинг обсягів відходів: Автоматизовані системи можуть точно вимірювати кількість відходів, які генеруються в певних зонах, дозволяючи планувати їх вивезення або переробку;

– оптимізація маршрутів збору відходів: За допомогою аналізу даних можна визначити найбільш ефективні маршрути для вивезення відходів, знижуючи витрати на транспортування та паливо;
– сенсори якості відходів: спеціальні датчики можуть визначати тип відходів та їх придатність для переробки, сприяючи більш ефективному роздільному збору. [16]

Автоматизоване сортування: Розумні конвеєри та сортувальні механізми можуть автоматично сортувати відходи за типом, полегшуючи процес їх подальшої переробки.

Контроль умов зберігання: IoT датчики можуть моніторити умови зберігання, наприклад, температуру та вологість на складах відходів, щоб запобігти їх псуванню чи зниженню якості перероблених матеріалів.

1.1.2 Існуючі рішення

На ринку вже давно існують рішення для автоматизації аграрної промисловості, що включають різноманітні технології та сервіси. Наприклад, компанія «СпецПоливСервіс», яка спеціалізується на продажі, встановленні та обслуговуванні систем автоматичного поливу. Їх продукція дозволяє аграріям значно підвищити ефективність водного ресурсу, забезпечуючи рівномірне та точне зрошення рослин в залежності від потреб. Системи автоматичного поливу, які вони пропонують, можуть включати датчики вологості, температури та інші сенсори для моніторингу стану ґрунту та кліматичних умов, що дозволяє оптимізувати процес поливу та знизити витрати на воду.

Більш відповідним прикладом для теми роботи є компанія IoTji, яка пропонує рішення для автоматизації теплиць. Ця компанія спеціалізується на інтеграції різноманітних IoT технологій, спрямованих на оптимізацію процесів ведення тепличного господарства. У переліку послуг IoTji можна знайти системи для моніторингу та контролю кліматичних умов всередині теплиць, що включають автоматичне регулювання температури, вологості, освітлення та концентрації CO_2. [12]

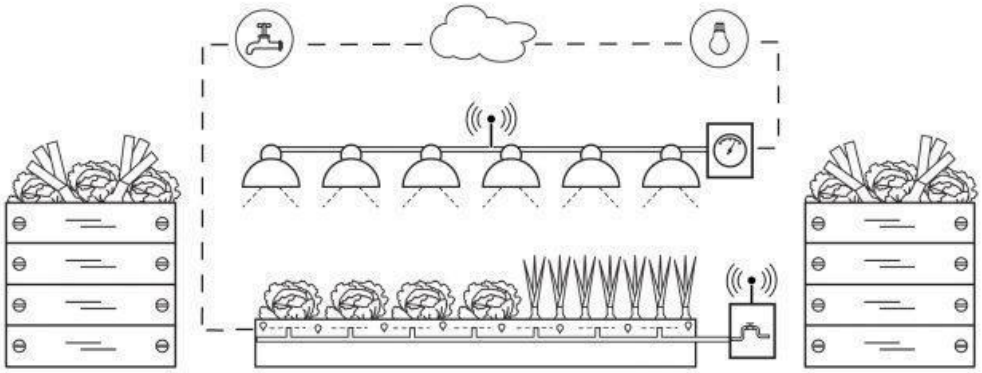


Рисунок 1.3 – Схема розумної теплиці iotji

1.2 Аналіз платформи AWS IoT та її компонентів

Amazon Web Services IoT є потужною платформою, яка надає різноманітні сервіси для підключення, керування та моніторингу IoT пристроїв. AWS IoT дозволяє з'єднувати мільйони пристроїв та керувати ними, обробляти потоки даних в реальному часі та інтегрувати отриману інформацію з іншими сервісами AWS для подальшої аналітики та обробки. Нижче буде розглянуто основні компоненти платформи AWS IoT та їх функціональні можливості. [3, 7]

1.2.1 AWS IoT компоненти

AWS IoT Core є основним сервісом платформи, який забезпечує безпечне підключення IoT пристроїв до хмари. Основні функції AWS IoT Core включають:

– безпечне підключення,
– обробка повідомлень,
– правила обробки даних,
– диспетчер підключень.

AWS IoT Device Management надає інструменти для реєстрації, організації, моніторингу та дистанційного керування IoT пристроями на великих масштабах. Основні можливості включають:

– реєстрацію пристроїв: автоматизує процес реєстрації та управління сертифікатами пристроїв;
– групування пристроїв: дозволяє створювати логічні групи пристроїв для зручного управління;
– моніторинг та оновлення прошивки: забезпечує віддалене оновлення прошивки пристроїв та моніторинг їхнього стану в реальному часі;
– журнали аудиту: надає функцію ведення журналів дій пристроїв для забезпечення безпеки та відповідності вимогам.

AWS IoT Analytics допомагає збирати, зберігати та аналізувати великі обсяги даних, що генеруються IoT пристроями. Основні компоненти включають:

– пайплайни даних: автоматизують процес збирання, обробки та зберігання даних;
– сховище даних: забезпечують довготривале зберігання великих обсягів даних;
– аналітичні запити: дозволяють виконувати складні запити та аналізувати дані за допомогою мов подібних до SQL (Structured Query Language);
– візуалізація даних: інтеграція з Amazon QuickSight для створення інтерактивних візуалізацій даних.

Цей сервіс надає можливість виконувати комплексну обробку даних для отримання цінної аналітичної інформації. Також дозволяють візуалізувати дані для їх подальшого аналізу.

1.2.2 AWS IoT Greengrass

AWS IoT Greengrass дозволяє переносити обчислювальні потужності AWS на крайову інфраструктуру, забезпечуючи локальне виконання функцій, обробку даних та інтеграцію з хмарними сервісами. Основні можливості включають:

– локальне виконання функцій Lambda: забезпечує виконання AWS Lambda функцій на локальних пристроях без підключення до хмари;

– обробка даних на краю: дозволяє обробляти дані на місці їх генерації, знижуючи затримки та зменшуючи обсяги даних, що передаються в хмару;
– локальні повідомлення: підтримує локальне обмін повідомленнями між пристроями без необхідності підключення до інтернету.

Загалом цей сервіс дозволить використовувати свої сценарії виконання, що зможе розширити можливості апаратно-програмного комплексу, за рахунок перенесення обробки на хмару.

1.2.3 Інтеграція з іншими сервісами AWS

Однією з ключових переваг AWS IoT є тісна інтеграція з іншими сервісами AWS, що дозволяє створювати комплексні рішення для обробки та аналізу даних. Це включає інтеграцію з такими сервісами, як:

– AWS Lambda: для виконання серверлесс функцій на основі подій з IoT пристроїв;
– Amazon S3: для зберігання великих обсягів даних;
– Amazon DynamoDB: для зберігання та управління даними у реальному часі;
– Amazon Kinesis: для обробки потокових даних в реальному часі;
– Amazon SageMaker: для створення та розгортання моделей машинного навчання.

Таким чином, платформа AWS IoT надає широкий спектр інструментів та сервісів для створення та управління IoT рішеннями, що дозволяє аграріям ефективно збирати, обробляти та аналізувати дані для підвищення ефективності їх діяльності.

1.2.4 Безпека у AWS IoT

Безпека є одним із найважливіших аспектів при впровадженні IoT рішень, і AWS IoT пропонує комплексний підхід до захисту даних та пристроїв. Враховуючи можливі загрози, платформа AWS IoT забезпечує багаторівневу безпеку, яка включає аутентифікацію пристроїв, шифрування

даних, контроль доступу та моніторинг безпеки. Нижче розглянемо основні елементи безпеки AWS IoT.

Аутентифікація визначає, чи дійсно пристрій чи користувач є тим, за кого себе видає. AWS IoT використовує сертифікати X.509 для аутентифікації пристроїв. Кожен пристрій отримує унікальний сертифікат, який підтверджує його ідентичність.

Авторизація забезпечує контроль доступу, дозволяючи тільки авторизованим пристроям та користувачам доступ до певних ресурсів або виконання конкретних дій. AWS IoT використовує політики AWS IAM (Identity and Access Management) для управління доступом, що дозволяє детально налаштовувати правила доступу до ресурсів.

Всі дані, що передаються між пристроями та AWS IoT Core, шифруються за допомогою протоколів TLS (Transport Layer Security), забезпечуючи захист від перехоплення та несанкціонованого доступу.

Дані, що зберігаються у AWS, можуть бути зашифровані за допомогою AWS KMS (Key Management Servic), який забезпечує управління ключами шифрування та їх ротацію.

AWS IoT надає інструменти для моніторингу та аудиту безпеки, що допомагає виявляти потенційні загрози та забезпечувати відповідність вимогам безпеки:

– AWS CloudTrail: веде журнали дій у AWS IoT, що дозволяє відстежувати всі запити до сервісу та виявляти підозрілу активність;
– AWS IoT Device Defender: забезпечує безперервний моніторинг пристроїв, виявляючи аномалії та потенційні загрози;
– AWS IoT Device Management: надає можливість моніторингу стану пристроїв, зокрема відстеження їх підключень та виявлення підозрілої поведінки.

AWS IoT дозволяє налаштовувати політики безпеки на рівні пристроїв та користувачів, що забезпечує гнучкий контроль доступу та дій. Політики можна налаштовувати за допомогою AWS IAM.

Для забезпечення високого рівня безпеки в IoT системах на базі AWS, рекомендується дотримуватися наступних практичних рекомендацій:

– використовувати унікальні сертифікати для кожного пристрою: це дозволить уникнути компрометації всієї системи в разі зламу одного пристрою;
– регулярно ротацію ключів та сертифікатів: забезпечення регулярної ротації ключів та сертифікатів допомагає підтримувати актуальний рівень безпеки;
– моніторинг безпеки в режимі реального часу: використання сервісів AWS IoT Device Defender та AWS CloudTrail для безперервного моніторингу та аудиту безпеки;
– обмеження доступу за принципом найменших привілеїв: налаштування політик безпеки таким чином, щоб пристрої та користувачі мали доступ лише до тих ресурсів та дій, які необхідні для виконання їх завдань.

AWS IoT використовує сертифікати для аутентифікації пристроїв та забезпечення безпечного з'єднання. AWS IoT Core надає можливість автоматизованого управління сертифікатами, включаючи їх створення, активацію, ротацію та від Revoke. AWS IoT також підтримує використання сертифікатів від сторонніх центрів сертифікації (CA), що дозволяє інтегруватися з існуючими інфраструктурами безпеки.

1.3 Середа розробки

Для розробки буде використана апаратна платформа Arduino, яка забезпечує простоту інтеграції різноманітних сенсорів і модулів, необхідних для збору кліматичних даних у теплиці. Мікроконтролери Arduino дозволяють легко програмувати та налаштовувати систему за допомогою середовища розробки Arduino integrated development environment (IDE).

Середовище розробки має можливості підключення різних бібліотек для роботи з різноманітними сенсорами та модулями зв'язку. Це дозволяє значно спростити процес розробки та інтеграції апаратних компонентів. Arduino IDE

підтримує велику кількість бібліотек, які можна легко додати до проєкту для забезпечення взаємодії з датчиками температури, вологості, освітленості, CO2 та іншими кліматичними параметрами.

Використання бібліотек, таких як DHT (Digital Humidity and Temperature) для датчиків температури та вологості, Adafruit для різних типів сенсорів, а також PubSubClient для роботи з протоколом MQTT (Message Queuing Telemetry Transport), дозволяє швидко та ефективно реалізувати необхідні функції збору та передачі даних. Крім того, наявність великої кількості документації та прикладів використання цих бібліотек спрощує процес їх інтеграції та налаштування.

Для використання Arduino IDE разом з ESP32 у вашому проєкті, необхідно встановити відповідні бібліотеки та налаштування. ESP32 є потужним мікроконтролером з вбудованими модулями Wi-Fi і Bluetooth, що робить його ідеальним вибором для системи сповіщення змін кліматичних умов теплиці на базі AWS IoT.

Висновки до розділу

Впровадження IoT технологій у сільському господарстві має значний потенціал для підвищення ефективності та стійкості аграрних підприємств. Розглянуті приклади та аналіз платформи AWS IoT демонструють широкі можливості для автоматизації, моніторингу та оптимізації аграрних процесів. Основні висновки можна підсумувати наступним чином.

1) Переваги IoT у сільському господарстві:

1) підвищення врожайності та якості продукції за рахунок точного управління ресурсами, такими як вода, добрива та енергія;

2) зменшення витрат на виробництво завдяки автоматизації процесів, зниженню потреби в ручній праці та оптимізації використання ресурсів;

3) поліпшення екологічної стійкості шляхом зменшення впливу на довкілля через оптимізацію використання води та хімічних речовин;

2) Безпека у AWS IoT:

1) AWS IoT забезпечує багаторівневу безпеку, включаючи аутентифікацію та авторизацію пристроїв, шифрування даних під час передачі та зберігання, а також моніторинг та аудит безпеки;

2) Використання унікальних сертифікатів для пристроїв, регулярна ротація ключів та сертифікатів, а також дотримання принципу найменших привілеїв дозволяє мінімізувати ризики безпеки;

3) реалізація системи сповіщення змін кліматичних умов теплиці на базі AWS IoT:

1) використання AWS IoT для моніторингу та керування мікрокліматом у теплицях дозволяє автоматизувати процеси контролю температури, вологості, освітлення та концентрації CO_2, що сприяє створенню оптимальних умов для росту рослин;

2) інтеграція датчиків та автоматизованих систем поливу та вентиляції забезпечує точне реагування на зміни кліматичних умов, що сприяє підвищенню врожайності та зниженню витрат.

IoT технології, зокрема платформа AWS IoT, надають аграріям потужні інструменти для підвищення ефективності та стійкості їх діяльності. Завдяки можливостям точного моніторингу та управління ресурсами, а також забезпечення високого рівня безпеки, впровадження IoT рішень сприяє підвищенню врожайності, зниженню витрат та покращенню екологічної стійкості сільськогосподарських підприємств. Важливою складовою успіху є ретельне планування та налаштування IoT систем, з урахуванням специфічних потреб та умов господарства, що дозволить максимально використати потенціал новітніх технологій.

Відкрите AWS application programming interface (API) дозволить інтегрувати різноманітні кліматичні сенсори та мікроконтролери безпосередньо з хмарною платформою, забезпечуючи таким чином швидку та ефективну обробку даних в реальному часі. Це також сприятиме автоматизації процесів управління теплицею, дозволяючи користувачам отримувати

актуальну інформацію про стан кліматичних умов і реагувати на них оперативно. Більше того, використання відкритого API надає можливість розширення функціональності системи шляхом додавання нових модулів та служб, що робить розробку більш гнучкою та адаптованою до специфічних потреб користувача.

Використання Arduino IDE забезпечить простоту та ефективність розробки програмного забезпечення для мікроконтролера ESP32, що дозволить інтегрувати різноманітні сенсори та модулі для збору кліматичних даних у теплиці. Arduino IDE надає зручний інтерфейс та широкий набір інструментів для написання, компіляції та завантаження коду, що значно спрощує процес розробки.

2 ФОРМУВАННЯ ВИМОГ

Формування вимог є важливим етапом, що визначає функціональність, продуктивність та безпеку майбутнього рішення. У цьому розділі ми розглянемо ключові технічні, функціональні та нефункціональні вимоги, які забезпечать ефективну роботу системи.

А дотримування вимог на проєктах, завжди забезпечує високу якість розробки та стабільність роботи створеного апаратно-програмного комплексу. Це включає в себе ретельне дотримання технічних специфікацій, стандартів безпеки та протоколів тестування, що дозволяє мінімізувати можливі помилки та несправності в роботі системи. Крім того, відповідність вимогам проєкту сприяє поліпшенню сумісності між різними компонентами системи, забезпечуючи їх безперебійне функціонування та інтеграцію. В результаті, користувачі отримують надійне і ефективне рішення для моніторингу та управління кліматичними умовами в теплиці.

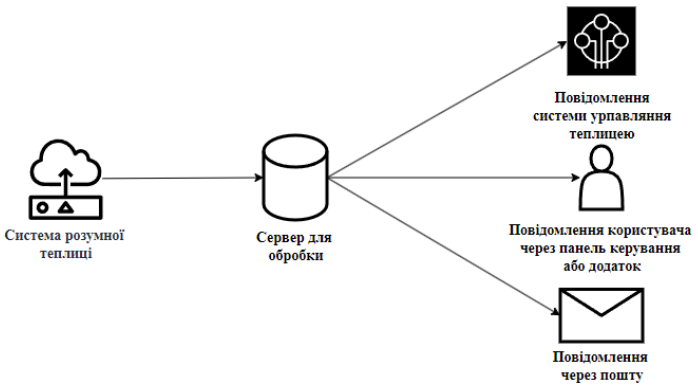


Рисунок 2.1 – Діаграмма роботи апаратно-програмного комплексу

2.1 Апаратні вимоги

Формування апаратних вимог забезпечить правильність підбору необхідних компонентів для стабільного та ефективного функціонування системи сповіщення змін кліматичних умов теплиці. Вимоги до контролерів, датчиків та мережевого обладнання гарантують, що всі компоненти системи

будуть сумісні між собою, відповідатимуть технічним стандартам і зможуть працювати в специфічних умовах теплиці.

Контролери повинні підтримувати підключення різних типів датчиків, мати достатню обчислювальну потужність, функції віддаленого оновлення прошивки, надійність та енергозбереження. Датчики вологості, температури, освітленості, вологості ґрунту, тиску та CO2 повинні мати високу точність вимірювань, підтримувати стандартні інтерфейси та бути стійкими до зовнішніх впливів. Мережеве обладнання, включаючи маршрутизатори, точки доступу та мережеві кабелі, має забезпечувати стабільне та безпечне з'єднання між компонентами системи та хмарною платформою. [1]

Ретельне дотримання цих апаратних вимог дозволить створити надійну інфраструктуру для збору, обробки та передачі даних, що є основою для ефективного управління кліматичними умовами у теплиці. Це, у свою чергу, сприятиме підвищенню врожайності та оптимізації використання ресурсів..

2.1.1 Контролер

Контролери є центральними елементами системи автоматизації теплиці, які відповідають за збір даних з датчиків, прийняття рішень та виконання команд для підтримки оптимальних кліматичних умов. Основні функції контролерів включають управління поливом, вентиляцією, освітленням та іншими системами теплиці. Нижче описано основні вимоги до контролерів:

– сумісність з датчиками: контролери повинні підтримувати підключення різних типів датчиків (температури, вологості, освітлення, CO2) через стандартні інтерфейси (наприклад, I2C, SPI, UART);

– інтерфейси зв'язку: підтримка протоколів зв'язку, таких як MQTT, HTTP (Hypertext transfer protocol), Zigbee, LoRaWAN;

– обчислювальна потужність: контролери повинні мати достатню обчислювальну потужність для обробки даних з датчиків, виконання локальних алгоритмів та прийняття рішень у режимі реального часу;

– можливість оновлення прошивки: Підтримка віддаленого оновлення прошивки для впровадження нових функцій та виправлення помилок.

– надійність та відмовостійкість: контролери повинні бути стійкими до збоїв та забезпечувати безперебійне функціонування системи навіть у випадку відмови окремих компонентів;

– енергозбереження: контролери повинні мати функції енергозбереження для мінімізації споживання енергії, що особливо важливо для автономних систем.

Вимоги до контролеру передбачають його спроможність підтримувати підключення різних типів датчиків (температури, вологості, освітлення, CO2) через стандартні інтерфейси, такі як I2C, SPI та UART. Контролери повинні підтримувати сучасні протоколи зв'язку (MQTT, HTTP, Zigbee, LoRaWAN), що забезпечує їх взаємодію з хмарними платформами та іншими пристроями. Достатня обчислювальна потужність контролерів необхідна для обробки даних з датчиків, виконання локальних алгоритмів та прийняття рішень у режимі реального часу.

Сформовані вимоги надають великий простір для підбору компонентів, адже їм можуть відповідати як контролери на базі Arduino NANO з використанням шилда, так і мікрокомп'ютери Raspberry Pi. Вимоги до сумісності з різними типами датчиків, підтримки сучасних протоколів зв'язку, достатньої обчислювальної потужності, можливості віддаленого оновлення прошивки, надійності, відмовостійкості та енергозбереження дозволяють обирати найрізноманітніші рішення для реалізації системи автоматизації теплиці.

2.1.2 Датчики

Датчики є невід'ємною частиною системи моніторингу та автоматизації кліматичних умов теплиці. Вони забезпечують збір необхідної інформації про

різні параметри навколишнього середовища, що впливає на зростання та розвиток рослин. У цьому розділі розглянемо вимоги до датчиків:

1) датчики вологості:

1) функція: вимірювання рівня вологості повітря в теплиці;

2) точність: висока точність вимірювань (похибка не більше ±3% RH);

3) діапазон: діапазон вимірювання від 0% до 100% RH;

4) інтерфейс: підтримка стандартних інтерфейсів (I2C, UART);

5) робочі умови: стійкість до конденсату та запиленості;

2) датчики температури:

1) функція: вимірювання температури повітря та ґрунту;

2) точність: висока точність вимірювань (похибка не більше ±0.5°C);

3) діапазон: діапазон вимірювання від -40°C до +85°C;

4) інтерфейс: підтримка стандартних інтерфейсів (I2C, 1-Wire, UART);

5) робочі умови: стійкість до впливу вологості та механічних пошкоджень;

3) датчики освітленості:

1) функція: вимірювання інтенсивності освітлення в теплиці;

2) точність: висока точність вимірювань (похибка не більше ±5%);

3) діапазон: широкий діапазон вимірювань від 0 до 100,000 люкс;

4) інтерфейс: підтримка стандартних інтерфейсів (I2C, SPI);

5) робочі умови: захист від впливу ультрафіолетового випромінювання та пилу;

4) датчики вологості ґрунту:

1) функція: вимірювання вологості ґрунту для оптимізації системи поливу;

2) точність: висока точність вимірювань (похибка не більше ±3% VWC);

3) діапазон: діапазон вимірювання від 0% до 100% VWC;

4) інтерфейс: підтримка стандартних інтерфейсів (I2C, Analog);

5) робочі умови: стійкість до корозії та механічних пошкоджень;

5) датчики тиску:

1) функція: вимірювання атмосферного тиску всередині теплиці;

2) точність: висока точність вимірювань (похибка не більше ±1 hPa);

3) діапазон: діапазон вимірювання від 300 до 1100 hPa;

4) інтерфейс: підтримка стандартних інтерфейсів (I2C, SPI);

5) робочі умови: стійкість до змін температури та вологості;

6) датчики CO2:

1) функція: вимірювання рівня концентрації CO2 в повітрі теплиці;

2) точність: висока точність вимірювань (похибка не більше ±50 ppm);

3) діапазон: діапазон вимірювання від 0 до 5000 ppm;

4) інтерфейс: підтримка стандартних інтерфейсів (UART або аналоговий);

5) робочі умови: стійкість до змін температури та вологості;

7) загальні вимоги:

1) калібрування: датчики повинні бути легко калібруваними для забезпечення точності вимірювань;

2) монтаж: датчики повинні мати зручні кріплення для швидкого та безпечного встановлення у різних частинах теплиці;

3) захист: датчики повинні бути захищені від зовнішніх впливів, таких як пил, волога та механічні пошкодження;

4) обслуговування: датчики повинні мати можливість легкого обслуговування та заміни без значних перерв у роботі системи.

Датчики вологості, температури, освітленості, вологості ґрунту, тиску та CO_2 є критично важливими компонентами системи моніторингу та автоматизації кліматичних умов теплиці. Вони забезпечують точне вимірювання різноманітних параметрів навколишнього середовища, що дозволяє підтримувати оптимальні умови для зростання рослин. Основні

вимоги до датчиків включають високу точність вимірювань, широкий діапазон вимірювання, підтримку стандартних інтерфейсів (I2C, UART, SPI, Analog), а також стійкість до впливу конденсату, пилу, вологи та механічних пошкоджень.

Загалом, використання відповідних датчиків з високою точністю та надійністю дозволяє ефективно автоматизувати процеси в теплиці, оптимізувати використання ресурсів, підвищити врожайність та забезпечити сталий розвиток аграрного виробництва. Це забезпечує стабільні та передбачувані умови для зростання рослин, що є ключовим фактором для успішного ведення сучасного сільського господарства.

2.1.3 Мережеве обладнання

Мережеве обладнання є критично важливим компонентом системи сповіщення. Воно забезпечує стабільне та безпечне з'єднання між датчиками, контролерами та хмарною платформою AWS IoT. У цьому розділі розглянемо вимоги до мережевого обладнання, необхідного для забезпечення надійного функціонування системи:

1) мережеві маршрутизатори:

1) функція: забезпечення з'єднання між локальною мережею теплиці та Інтернетом;

2) продуктивність: підтримка високошвидкісного з'єднання для передачі великих обсягів даних;

3) безпека: вбудовані функції безпеки, такі як шифрування трафіку, брандмауер та virtu VPN (virtual private network) підтримка;

4) надійність: стійкість до збоїв та можливість автоматичного перемикання на резервне з'єднання у випадку втрати основного з'єднання;

2) точки доступу Wi-Fi:

1) функція: забезпечення бездротового з'єднання для датчиків та контролерів у теплиці;

2) продуктивність: підтримка стандартів Wi-Fi 5 (802.11ac) або Wi-Fi 6 (802.11ax) для забезпечення високої швидкості та стабільності з'єднання;

3) покриття: широке покриття для забезпечення стабільного з'єднання у всіх частинах теплиці;

4) безпека: підтримка сучасних стандартів безпеки, таких як WPA3, для захисту бездротового з'єднання;

5) надійність: можливість роботи в складних умовах, таких як підвищена вологість та температура;

3) мережеві кабелі:

1) функція: забезпечення провідного з'єднання між мережевими пристроями;

2) якість: використання високоякісних кабелів категорії Cat5e, Cat6 або Cat6a для забезпечення стабільного та швидкого з'єднання;

3) надійність: захист від впливу зовнішніх факторів, таких як волога та механічні пошкодження.

Мережеве обладнання є критично важливим компонентом системи сповіщення змін кліматичних умов теплиці. Воно забезпечує стабільне та безпечне з'єднання між датчиками, контролерами та хмарною платформою, що є основою для надійного функціонування всієї системи. Вибір та налаштування мережевого обладнання мають вирішальне значення для забезпечення високої продуктивності та безпеки системи.

Загалом, маршрутизатори, кабелі та точки доступу є критично важливими компонентами мережевої інфраструктури системи сповіщення. Маршрутизатори забезпечують надійне з'єднання між локальною мережею теплиці та Інтернетом, підтримуючи високошвидкісну передачу даних, функції безпеки та стійкість до збоїв. Точки доступу (Wi-Fi) забезпечують бездротове з'єднання для датчиків та контролерів, підтримуючи сучасні стандарти Wi-Fi для високої швидкості та стабільності з'єднання, широке покриття та безпеку. Мережеві кабелі категорій Cat5e, Cat6 або Cat6a гарантують стабільне та швидке провідне з'єднання між пристроями, надійно

захищене від зовнішніх впливів, таких як волога та механічні пошкодження. Використання відповідного мережевого обладнання забезпечує стабільне функціонування системи, сприяючи ефективному управлінню та оптимізації умов для зростання рослин.

2.2 Програмні вимоги

Програмні вимоги визначають основні необхідні характеристики IoT сервісу та прошивки контролера, забезпечуючи їх взаємодію та ефективну роботу всієї системи.

2.2.1 Сервіси IoT

Основним компонентом в роботі є IoT сервіси, за допомогою яких забезпечується підключення, керування та моніторинг пристроїв, а також обробка даних у реальному часі. Програмні вимоги охоплюють декілька ключових аспектів, необхідних для забезпечення надійного та ефективного функціонування системи:

1) підключення та керування пристроями:

1) **широка підтримка протоколів:** сервіси повинні підтримувати різні протоколи зв'язку (MQTT, HTTP, CoAP, AMQP) для забезпечення сумісності з різноманітними IoT пристроями;

2) **масштабованість:** можливість підключення та керування великою кількістю пристроїв, зокрема мільйонами одночасних з'єднань;

1) **стабільність з'єднання:** забезпечення стабільного та надійного з'єднання між пристроями та хмарною платформою;

2) управління пристроями:

1) **реєстрація та ідентифікація:** можливість автоматизованої реєстрації нових пристроїв та призначення унікальних ідентифікаторів;

2) **моніторинг:** забезпечення функцій моніторингу стану пристроїв у реальному часі, включаючи відстеження підключень, стану акумулятора, сигналу тощо;

3) **оновлення прошивки:** підтримка віддаленого оновлення прошивки для впровадження нових функцій та виправлення помилок;

3) обробка даних:

1) **збір даних:** безперервний збір даних з підключених пристроїв для подальшої обробки та аналізу;

2) **обробка в реальному часі:** можливість обробки та аналізу даних у реальному часі для забезпечення швидкого реагування на події;

3) **зберігання даних:** Надійне зберігання великих обсягів даних з можливістю доступу до історичних даних;

2) автоматизація та управління:

1) **автоматичні дії:** ьожливість створення правил та тригерів для автоматизації дій на основі зібраних даних (наприклад, автоматичне увімкнення системи поливу при низькому рівні вологості ґрунту);

2) **інтеграція з іншими сервісами:** підтримка інтеграції з іншими сервісами та системами для розширення функціональності (наприклад, інтеграція з системами управління фермою або ERP-системами);

3) візуалізація та аналітика:

1) **інтерфейс для користувачів:** забезпечення веб-інтерфейсу та мобільного додатку для моніторингу та управління пристроями;

2) **аналітичні інструменти:** надання інструментів для аналізу даних, створення звітів та візуалізації інформації для прийняття рішень;

4) безпека та автентифікація:

1) **аутентифікація та авторизація:** використання надійних методів аутентифікації та авторизації для захисту доступу до системи;

2) **шифрування:** шифрування даних під час передачі та зберігання для захисту від несанкціонованого доступу;

3) **журнали безпеки:** ведення детальних журналів безпеки для відстеження дій та виявлення підозрілої активності;

5) масштабованість та продуктивність:

1) **горизонтальна масштабованість:** можливість додавання нових ресурсів для збільшення обчислювальної потужності та зберігання даних;

2) **висока доступність:** забезпечення високої доступності сервісу з мінімальними простоями;

3) **низькі затримки:** оптимізація для мінімальних затримок у передачі та обробці даних;

6) надійність та відмовостійкість:

1) **відмовостійкість:** можливість безперебійної роботи навіть у випадку відмови окремих компонентів системи;

2) **резервне копіювання:** регулярне резервне копіювання даних для запобігання втратам інформації;

7) легкість інтеграції та використання:

1) **документація та API:** надання детальної документації та зручного API для розробників для полегшення інтеграції та використання сервісів IoT;

2) **підтримка стандартів:** використання загальноприйнятих стандартів протоколів та форматів даних для забезпечення сумісності з іншими системами;

Загалом, ці вимоги слугують фундаментом для вибору IoT платформи, яка забезпечить стабільну, безпечну та ефективну роботу системи. Підтримка різних протоколів зв'язку, масштабованість, можливість стабільного з'єднання та ефективного управління пристроями є критично важливими для побудови надійної IoT інфраструктури. Автоматизація дій, інтеграція з іншими сервісами, зручні інтерфейси для користувачів та потужні аналітичні інструменти дозволяють оптимізувати процеси та забезпечити високу продуктивність системи.

Також важливо враховувати аспекти безпеки, такі як аутентифікація, шифрування та ведення журналів безпеки, для захисту даних і забезпечення довіри до системи. Масштабованість та продуктивність сервісів дозволяють системі рости разом із зростанням кількості підключених пристроїв та обсягу

даних. Надійність та відмовостійкість гарантують безперебійну роботу, а легкість інтеграції та використання спрощують процес впровадження та управління системою. Усе це разом робить IoT платформу потужним інструментом для забезпечення ефективного функціонування системи сповіщення змін кліматичних умов теплиці та інших IoT проєктів.

2.2.2 Прошивка мікроконтреллеру

Прошивка мікроконтролера є однією з основних частин роботи, яка забезпечує функціональність та інтеграцію з іншими компонентами системи автоматизації теплиці. Вона відповідає за збір даних з датчиків, обробку цих даних, прийняття рішень та виконання команд для підтримки оптимальних кліматичних умов.

Прошивка буде реалізована з використанням Arduino IDE, яка використовує мову програмування подібну до C/C++. Середа розробки надає зручне середовище для розробки, компіляції та завантаження коду на мікроконтролери. Її використання дозволяє ефективно управляти апаратними ресурсами, забезпечуючи високу продуктивність та надійність роботи прошивки. Прошивка буде відповідати вимогам проєкту, включаючи збирання даних від сенсорів, обробку та передачу цих даних до хмарної платформи AWS IoT, а також забезпечення необхідної логіки для налаштування сповіщень та інших функціональних можливостей системи.

Прошивка повинна підтримувати підключення різних типів датчиків (температури, вологості, освітлення, CO2) через стандартні інтерфейси, такі як I2C, SPI та UART. Вона також повинна забезпечувати підтримку сучасних протоколів зв'язку (MQTT, HTTP, Zigbee, LoRaWAN), що дозволяє контролеру взаємодіяти з хмарними платформами та іншими пристроями.

Функції енергозбереження в прошивці дозволяють мінімізувати споживання енергії, що є особливо важливим для автономних систем. Загалом, правильна прошивка мікроконтролера є ключовим елементом для забезпечення стабільної та ефективної роботи системи автоматизації теплиці,

відповідаючи за інтеграцію, обробку даних та управління різними компонентами системи.

2.3 Функціональні вимоги

Функціональні вимоги визначають основні можливості та функції, які повинна забезпечувати система автоматизації теплиці. Ці вимоги охоплюють аспекти моніторингу, сповіщень та автоматизації, що дозволяють ефективно керувати кліматичними умовами в теплиці.

У цьому проєкті функціональні вимоги визначають конкретні можливості та операції, які повинна виконувати система для досягнення своїх цілей. Вони описують найважливіші аспекти, такі як збір даних від кліматичних сенсорів, обробка та аналіз отриманих даних, передача інформації до хмарної платформи AWS IoT, а також надання сповіщень користувачам про зміни кліматичних умов у теплиці. Іншими важливими функціональними вимогами є можливість інтеграції системи з різними типами сенсорів, забезпечення безперебійної роботи в режимі реального часу та автоматичне реагування на критичні зміни кліматичних параметрів. Функціональні вимоги також охоплюють аспекти налаштування параметрів системи, доступу до інформації через зручний інтерфейс та можливість віддаленого управління. Дотримання цих вимог забезпечить ефективну роботу системи та досягнення її основних цілей.

2.3.1 Моніторинг

Моніторинг є ключовою функцією системи, яка забезпечує збір та аналіз даних з різних датчиків у режимі реального часу.

– здійснення моніторингу кліматичних умов у теплиці (температура, вологість, освітлення, рівень CO2, вологість ґрунту, тиск) у режимі реального часу;

– надання зручних інструментів для візуалізації даних у вигляді графіків, таблиць та інших візуальних елементів;

– зберігання історичних даних для аналізу тенденцій та прийняття обґрунтованих рішень;

Завдяки моніторингу у реальному часі з’являється можливість відслідковувати на аналізувати дані отримані з теплиці, що відкриває можливості для своєчасного регулювання умов

2.3.1 Сповіщення

Функція сповіщень забезпечує оперативне інформування користувачів про критичні зміни в кліматичних умовах теплиці, що дозволяє вчасно реагувати на потенційні проблеми.

– Можливість налаштування порогових значень для кожного параметра, при перевищенні яких буде надсилатися сповіщення;

– Відправка сповіщень через електронну пошту, SMS та пуш-повідомлення в мобільному додатку;

– Збереження історії сповіщень для подальшого аналізу та перевірки ефективності реагування;

– Встановлення пріоритетів для різних типів сповіщень, щоб критичні сповіщення мали вищий пріоритет.

Сповіщення можуть бути подані у вільному форматі, залежно від потреб та вподобань кінцевих користувачів. Це включає можливість налаштування різних типів сповіщень, таких як текстові повідомлення, електронні листи або push-сповіщення на мобільні пристрої.

Налаштування сповіщень та збереження історії не є обов'язковими компонентами, адже вони не впливають на основну функціональність системи. Основні функції системи зосереджені на зборі, обробці та передачі кліматичних даних для моніторингу умов у теплиці. Проте, наявність цих додаткових компонентів може значно покращити користувацький досвід, надаючи можливість гнучкого налаштування системи відповідно до індивідуальних потреб користувачів та забезпечуючи доступ до історичних даних для аналізу тенденцій та прийняття обґрунтованих рішень. Таким

чином, хоча налаштування сповіщень та збереження історії не є критичними для базової роботи системи, вони можуть стати важливими елементами для підвищення її зручності та ефективності в повсякденному використанні.

2.3.2 Автоматизація

Автоматизація процесів дозволяє ефективно керувати системою теплиці без постійного втручання людини, оптимізуючи використання ресурсів та підтримуючи оптимальні умови для зростання рослин.

– Можливість створення правил та тригерів для автоматизації дій на основі зібраних даних (наприклад, автоматичне увімкнення поливу при зниженні вологості ґрунту нижче заданого рівня);

– Підтримка інтеграції з іншими системами управління фермою або ERP-системами для розширення можливостей автоматизації;

– Можливість користувачів налаштовувати автоматизовані процеси відповідно до специфічних вимог та умов теплиці;

– Підтримка резервних сценаріїв для забезпечення стабільної роботи системи у випадку збою основних автоматизованих процесів.

Функціональні вимоги до системи автоматизації теплиці забезпечують ефективне керування кліматичними умовами, оперативне реагування на зміни та оптимізацію використання ресурсів, [4] що в кінцевому результаті підвищує продуктивність аграрного виробництва та покращує умови для зростання рослин.

2.4 Нефункціональні вимоги

Нефункціональні вимоги визначають характеристики системи, які впливають на її продуктивність, безпеку, зручність використання та здатність до масштабування. Вони є критичними для забезпечення надійної та ефективної роботи системи автоматизації теплиці.

Так і у цьому проєкті нефункціональні вимоги будуть описувати продуктивність, безпеку, гнучкість використання та масштабованість.

Продуктивність системи включатиме здатність швидко та ефективно обробляти дані від численних сенсорів, забезпечуючи своєчасне сповіщення про зміни кліматичних умов у теплиці. Безпека охоплюватиме заходи для захисту даних від несанкціонованого доступу та забезпечення конфіденційності інформації. Гнучкість використання передбачатиме можливість налаштування системи під різні сценарії використання та адаптацію до потреб користувача. Масштабованість забезпечить здатність системи розширюватися для обробки збільшеної кількості сенсорів та обсягів даних без втрати продуктивності та надійності.

2.4.1 Продуктивність

Продуктивність системи визначає, наскільки швидко і ефективно вона може обробляти дані та виконувати свої функції:

– час відгуку: система повинна забезпечувати швидкий відгук на запити користувачів та виконання автоматичних дій. час відгуку не повинен перевищувати 1 секунди для основних операцій;

– обробка даних: система повинна бути здатна обробляти великий обсяг даних у режимі реального часу, зокрема дані від датчиків, команди від контролерів та аналітичні обчислення;

– продуктивність мережі: забезпечення стабільного та високошвидкісного з'єднання між пристроями та хмарною платформою для передачі даних без затримок.

Крім цього, система повинна мати здатність масштабуватися для підтримки зростаючого обсягу даних і кількості підключених пристроїв. Надійність системи також є критично важливою для безперебійного функціонування та запобігання втраті даних.

2.4.2 Безпека

Безпека є критичним аспектом системи, яка повинна захищати дані та забезпечувати надійний доступ до системи.

– Аутентифікація та авторизація: використання надійних методів аутентифікації (наприклад, двофакторної аутентифікації) та авторизації для контролю доступу до системи.

– Шифрування даних: шифрування даних під час передачі та зберігання для захисту від несанкціонованого доступу.

– Журнали безпеки: ведення детальних журналів безпеки для відстеження дій користувачів та виявлення підозрілої активності.

– Регулярні оновлення: забезпечення регулярного оновлення програмного забезпечення для виправлення вразливостей та впровадження нових функцій безпеки.

Безпека проєкту здебільшого буде забезпечена завдяки механізмам AWS IoT, які включають шифрування даних, автентифікацію пристроїв та контроль доступу. AWS IoT надає можливість шифрування даних як під час передачі, так і на зберіганні, що запобігає несанкціонованому доступу до інформації. Автентифікація пристроїв гарантує, що тільки авторизовані пристрої можуть підключатися до системи та передавати дані. Контроль доступу дозволяє визначати, хто і до яких ресурсів має доступ, що знижує ризик несанкціонованого використання або модифікації даних. Крім того, регулярний моніторинг та оновлення безпекових політик забезпечують захист від нових загроз та вразливостей, роблячи систему надійною і стійкою до потенційних атак.

2.4.3 Зручність використання

Зручність використання визначає, наскільки легко та інтуїтивно зрозуміло користувачі можуть взаємодіяти із системою.

– Надання зручного та зрозумілого інтерфейсу для користувачів, що дозволяє легко отримувати доступ до основних функцій системи.

– Забезпечення детальної документації та інструкцій для користувачів, а також доступ до технічної підтримки у разі виникнення проблем.

– Підтримка різних мов інтерфейсу для зручності користувачів з різних країн.

Зручність використання загалом є суб'єктивною характеристикою, яка може варіюватися залежно від потреб та очікувань кінцевих користувачів. У контексті цього проєкту зручність використання визначатиметься такими факторами, як інтуїтивно зрозумілий інтерфейс користувача, легкість налаштування та управління системою, а також доступність інструкцій та підтримки. Система повинна бути розроблена таким чином, щоб користувачі з різним рівнем технічної підготовки могли легко зрозуміти, як нею користуватися, і швидко освоїти основні функції. Важливими аспектами також є мінімізація необхідності втручання користувача у повсякденну роботу системи та забезпечення можливості віддаленого моніторингу та управління.

2.4.4 Масштабованість

Масштабованість системи визначає її здатність розширюватися та адаптуватися до збільшення кількості користувачів та пристроїв без втрати продуктивності.

– Можливість додавання нових пристроїв та ресурсів без значних змін у архітектурі системи.

– Система повинна підтримувати підключення та управління великою кількістю пристроїв одночасно.

– Автоматичне балансування навантаження: Забезпечення автоматичного розподілу навантаження між серверами та пристроями для оптимізації роботи системи.

– Можливість легко додавати нові функціональні модулі та сервіси для розширення можливостей системи.

Загалом термін масштабованість може бути зрозумілим по-різному залежно від контексту. У даному проєкті масштабованість стосується здатності системи адаптуватися до зростання навантаження без втрати продуктивності та стабільності. Це включає можливість додавання нових

сенсорів та мікроконтролерів, збільшення обсягів зібраних та оброблюваних даних, а також підтримку додаткових функцій без необхідності суттєвих змін у існуючій архітектурі. Масштабованість забезпечує довготривалу життєздатність системи, дозволяючи їй ефективно працювати як у малих, так і у великих теплицях, а також адаптуватися до змін у вимогах та умовах експлуатації.

Висновки до розділу

Апаратні вимоги визначають необхідні компоненти для забезпечення стабільної та ефективної роботи системи автоматизації теплиці. Вони включають датчики, контролери та мережеве обладнання, які повинні бути сумісними між собою та відповідати технічним стандартам.

Програмні вимоги визначають основні характеристики IoT сервісів та прошивки контролера. IoT сервіси повинні підтримувати різноманітні протоколи зв'язку, забезпечувати масштабованість та стабільне з'єднання. Прошивка контролера повинна забезпечувати підтримку підключення датчиків, обробку даних у реальному часі, можливість віддаленого оновлення та надійність.

Функціональні вимоги охоплюють основні можливості системи, такі як моніторинг, сповіщення та автоматизація. Моніторинг забезпечує збір та аналіз даних з датчиків у реальному часі, візуалізацію та зберігання даних для подальшого аналізу. Сповіщення інформують користувачів про критичні зміни в кліматичних умовах, дозволяючи швидко реагувати на потенційні проблеми. Автоматизація процесів дозволяє ефективно керувати системою без постійного втручання людини, оптимізуючи використання ресурсів та підтримуючи оптимальні умови для зростання рослин.

Виконання цих вимог забезпечить створення надійної, ефективної та безпечної системи автоматизації теплиці, яка відповідає високим стандартам якості та задовольняє потреби користувачів.

3 РОЗРОБКА

У процессі розробки планується розглянути основні етапи створення апаратно-програмнорго комплексу сповіщення змін кліматичних умов теплиці на базі AWS IoT будуть детально описані такі етапи, як вибір і налаштування сенсорів для збору кліматичних даних, інтеграція сенсорів з мікроконтролером, а також налаштування програмного забезпечення для збору, обробки та передачі даних до хмарної платформи AWS IoT. Особлива увага буде приділена вибору відповідних протоколів зв'язку для забезпечення надійної та безперебійної передачі даних, а також розробці алгоритмів аналізу та візуалізації зібраної інформації для ефективного моніторингу та управління кліматичними умовами в теплиці.

3.1 Опис архітектури системи

Архітектура системи буде складатися з декількох основних компонентів, кожен з яких відіграватиме важливу роль у забезпеченні функціональності, надійності та масштабованості системи сповіщення змін кліматичних умов теплиці на базі AWS IoT. [5] Основні компоненти архітектури включають декілька компонентів.

1) Кліматичні сенсори та мікроконтролер ESP32:

1) сенсори: вимірюють різні параметри клімату, такі як температура, вологість, освітленість, рівень CO2 тощо;

2) мікроконтролер ESP32: збирає дані з сенсорів, обробляє їх та передає до хмарної платформи AWS IoT за допомогою Wi-Fi та протоколу MQTT.

2) Хмарна платформа AWS IoT:

1) AWS IoT Core: приймає дані від ESP32, забезпечує їх обробку та управління пристроями;

2) Amazon API Gateway: забезпечує безпечний доступ до API для взаємодії з іншими сервісами AWS та веб-додатками;

3) AWS Lambda: Обробляє вхідні дані, запускає необхідні функції для аналізу та сповіщення;

4) Amazon DynamoDB: зберігає зібрані дані для подальшого аналізу та історичного зберігання;

5) Amazon SNS (Simple Notification Service): відправляє сповіщення користувачам через SMS, електронну пошту або push-повідомлення.

У цій архітектурі сенсори збирають дані та передають їх на мікроконтролер ESP32. ESP32 обробляє дані та передає їх до AWS IoT через MQTT. AWS IoT Core приймає дані та передає їх для подальшої обробки до AWS Lambda та зберігання у DynamoDB. Amazon SNS забезпечує сповіщення користувачів про критичні зміни кліматичних умов, надсилая повідомлення про зміни на електрону пошту користувача.

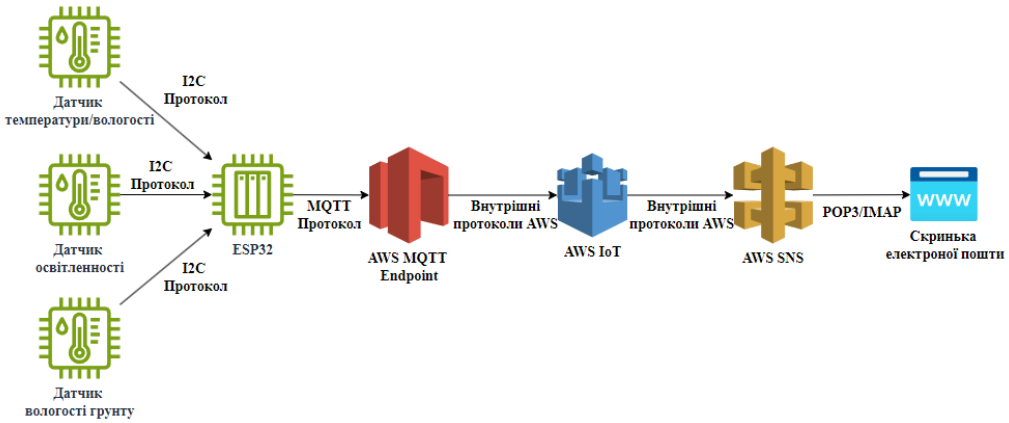


Рисунок 3.1 – Діаграма архітектури системи

На діаграмі зображено архітектуру системи сповіщення змін кліматичних умов теплиці на базі AWS IoT. Основні компоненти архітектури включають:

1) датчики кліматичних параметрів:
   1) датчик температури/вологості,
   2) Датчик освітленості,
   3) Датчик вологості ґрунту;
2) мікроконтролер ESP32:
   1) підключений до датчиків через протокол I2C;

2) збирає дані від датчиків і передає їх до хмарної платформи AWS IoT за допомогою протоколу MQTT;

3) AWS MQTT Endpoint приймає дані, передані ESP32 через протокол MQTT та передає їх до AWS IoT для подальшої обробки;

3) AWS IoT:

1) використовує внутрішні протоколи AWS для обробки отриманих даних;

2) забезпечує управління пристроями та обробку даних;

4) AWS SNS:

1) забезпечує можливість підписки;

2) виступає як шлюз для отримання даних;

5) скринька електронної пошти:

1) отримує дані через протоколи POP3, IMAP або інші;

2) забезпечує інтерфейс користувача для моніторингу кліматичних умов у теплиці.

Така архітектура може зустрітися у багатьох проєктах, пов'язаних з Інтернетом речей (IoT), де потрібно збирати, обробляти та аналізувати дані з різноманітних сенсорів для моніторингу навколишнього середовища. Вона забезпечує гнучкість та масштабованість, що дозволяє легко адаптувати систему до різних сценаріїв використання та додавати нові функції за необхідності. Використання хмарних сервісів AWS також дозволяє забезпечити високу надійність та безпеку даних, а також знижує витрати на інфраструктуру.

Тому використання цієї архітектури у цьому проєкті є доцільним і обґрунтованим рішенням, яке забезпечує кілька ключових переваг, таких як: масштабованість, надійність, безпека. Ефективність її використання підтверджується іншими проєктами, що використовують подібні підходи для управління та моніторингу кліматичних умов у різних середовищах. Приклади таких проєктів є у першому розділі роботи.

3.2 Вибір компонентів

Вибір компонентів апаратно-програмного комплексу буде включати в себе аналіз та порівняння різних моделей ESP, зокрема ESP8266 і ESP32. Цей аналіз дозволить вибрати найбільш підходящу модель мікроконтролера для забезпечення ефективної роботи системи сповіщення змін кліматичних умов у теплиці.

Для порівняння створимо таблицю, в якій зазначимо основні характеристики моделей ESP8266 та ESP32. Це допоможе наочно оцінити їх можливості та зробити обґрунтований вибір.

Таблиця 3.1 – Порівняльна таблиця характеристик ESP8266 та ESP32

Характеристика	ESP8266	ESP32
Процесор	Одноядерний 32-бітний Tensilica L106	Двоядерний 32-бітний Tensilica Xtensa LX6
Частота процесора	80/160 МГц	До 240 МГц
Оперативна пам'ять	32 КБ інструкцій, 80 КБ SRAM	520 КБ SRAM
Флеш-пам'ять	До 4 МБ	До 16 МБ (зазвичай 4 МБ)
Wi-Fi	802.11 b/g/n, STA/AP/STA+AP	802.11 b/g/n, STA/AP/STA+AP
Bluetooth	Немає	Bluetooth 4.2 та BLE
GPIO	До 17	До 36
Аналогові входи (ADC)	До 1	До 18
Цифрові входи/виходи	До 17	До 36
Інтерфейси	SPI, I2C, UART	SPI, I2C, UART, DAC, ADC, сенсорний дотик
Протоколи	TCP/IP, MQTT	TCP/IP, MQTT, HTTP/HTTPS
Ціна	Низька	Вища, але з більшою функціональністю

Виходячи з порівняльної таблиці, можна зробити наступні висновки:

– ESP32 має двоядерний процесор з вищою частотою, що забезпечує кращу продуктивність;

– ESP32 має значно більший обсяг оперативної пам'яті, що дозволяє обробляти більше даних та виконувати складніші завдання;

– обидва мікроконтролери підтримують достатню флеш-пам'ять для більшості IoT проєктів, але ESP32 має потенціал для більшого обсягу;

– обидві моделі мають вбудований модуль Wi-Fi з підтримкою основних режимів;

– Тільки ESP32 підтримує Bluetooth, що може бути критично важливим для деяких проєктів;

– ESP32 має більше загальновживаних входів/виходів, що забезпечує більшу гнучкість у підключенні додаткових компонентів;

– ESP32 підтримує більше інтерфейсів та протоколів, що робить його більш універсальним;

– ESP8266 має нижчу ціну, що може бути важливим для проєктів з обмеженим бюджетом.

У контексті проєкту системи сповіщення змін кліматичних умов теплиці на базі AWS IoT, вибір між ESP32 та ESP8266 буде залежати від конкретних вимог до системи та бюджету проєкту. Якщо проєкт передбачає високу продуктивність, підтримку Bluetooth та багатофункціональність, ESP32 буде найкращим вибором. Якщо ж основними критеріями є економічність та простота, ESP8266 стане більш доцільним варіантом.

Але, враховуючи архітектуру мікроконтролерів та вимоги проєкту, розробка може проходити як на базі ESP8266, так і ESP32. Обидва мікроконтролери здатні забезпечити базову функціональність системи сповіщення змін кліматичних умов теплиці на базі AWS IoT. В моєму випадку буде використовуватись – ESP32, як варіант, що є в мене наявності.

3.3 Підключення та конфігурація датчиків

Підключення датчиків до контролера не є складним завданням. Правильність підключення зазвичай можна перевірити, використовуючи інформацію з технічного документа певного компоненту системи. Зазвичай у таких документах позначається:

– розташування виводів (пінів);

– нумерація виводів;

– назви та функції виводів;

– схема підключення;

– електричні схеми;

– примітки щодо підключення;

– характеристики електричних сигналів;

– напруга живлення;

– споживання струму;

– рівні сигналів;

– програмування та налаштування;

– приклади коду;

– рекомендації щодо налаштувань.

Умовні позначення на датчиках та платах мікроконтролерів є стандартними маркуваннями, що допомагають ідентифікувати функції різних виводів (пінів) і правильно підключати їх між собою. Ось перелік з найпоширеніших умовних позначень:

– VCC – напруга живлення (іноді позначається як 3V3 або 5V, залежно від робочої напруги датчика);

– GND – земля (нульовий потенціал);

– DATA – цифровий сигнальний вихід (використовується для передачі даних, наприклад, у датчику DHT22);

– CLK або SCL – лінія тактового сигналу (Clock) для інтерфейсів I2C або SPI;

– SDA – лінія даних для інтерфейсу I2C (Serial Data);

– MISO – master In Slave Out (лінія даних для SPI);

– MOSI – master Out Slave In (лінія даних для SPI);

– CS або SS – chip Select або Slave Select (використовується для вибору пристрою на шині SPI);

– IRQ – лінія переривання (Interrupt Request);

– AO або AOUT – фналоговий вихід (використовується для передачі аналогових сигналів, наприклад, у датчику вологості ґрунту);

– 3V3 або 3.3V – вихід напруги 3.3V;

– 5V – вихід напруги 5V (часто використовується для живлення зовнішніх компонентів);

– GND – земля (нульовий потенціал);

– GPIOx – загальновживані входи/виходи, де "x" – номер піну (наприклад, GPIO14);

– RX – прийом даних для UART (Serial Receive);

– TX – передача даних для UART (Serial Transmit);

– SDA – лінія даних для інтерфейсу I2C (Serial Data);

– SCL – лінія тактового сигналу для інтерфейсу I2C (Serial Clock);

– MISO – лінія даних для SPI (Master In Slave Out);

– MOSI – лінія даних для SPI (Master Out Slave In);

– SCK – лінія тактового сигналу для SPI (Serial Clock);

– CS або SS – лінія вибору пристрою на шині SPI (Chip Select або Slave Select);

– A0, A1, A2, ...: AN – аналогові входи (використовуються для зчитування аналогових сигналів);

– DAC1, DAC2 – цифро-аналогові перетворювачі (виходи DAC).

Правильне підключення датчиків до мікроконтролера залежить від розуміння умовних позначень на датчиках та платах. Вивчення технічних документів та схем підключення допомагає забезпечити надійність та стабільність роботи системи. Використання стандартних маркувань спрощує процес розробки та налаштування системи, знижуючи ризик помилок при підключенні компонентів.

Зазвичай на платах присутні умовні позначки, які допомагають ідентифікувати функції різних виводів (пінів) та правильно підключати їх між собою. Ці позначки можуть включати маркування для живлення (VCC, GND), цифрових сигналів (DATA, CLK, SCL, SDA, MISO, MOSI, CS), аналогових сигналів (AO, AOUT), та інших важливих функцій. Правильне підключення датчиків до мікроконтролера залежить від розуміння цих умовних позначень на датчиках та платах. Наприклад, VCC позначає напругу живлення, GND – землю, а SDA і SCL використовуються для ліній даних та тактового сигналу у протоколі I2C.

Вивчення технічних документів та схем підключення допомагає забезпечити надійність та стабільність роботи системи. Документація зазвичай містить детальні схеми підключення, описи пінів та рекомендації щодо використання. Використання стандартних маркувань спрощує процес розробки та налаштування системи, знижуючи ризик помилок при підключенні компонентів. Завдяки цьому розробники можуть швидше та ефективніше налаштувати систему, забезпечуючи її коректну роботу і мінімізуючи можливі збої.

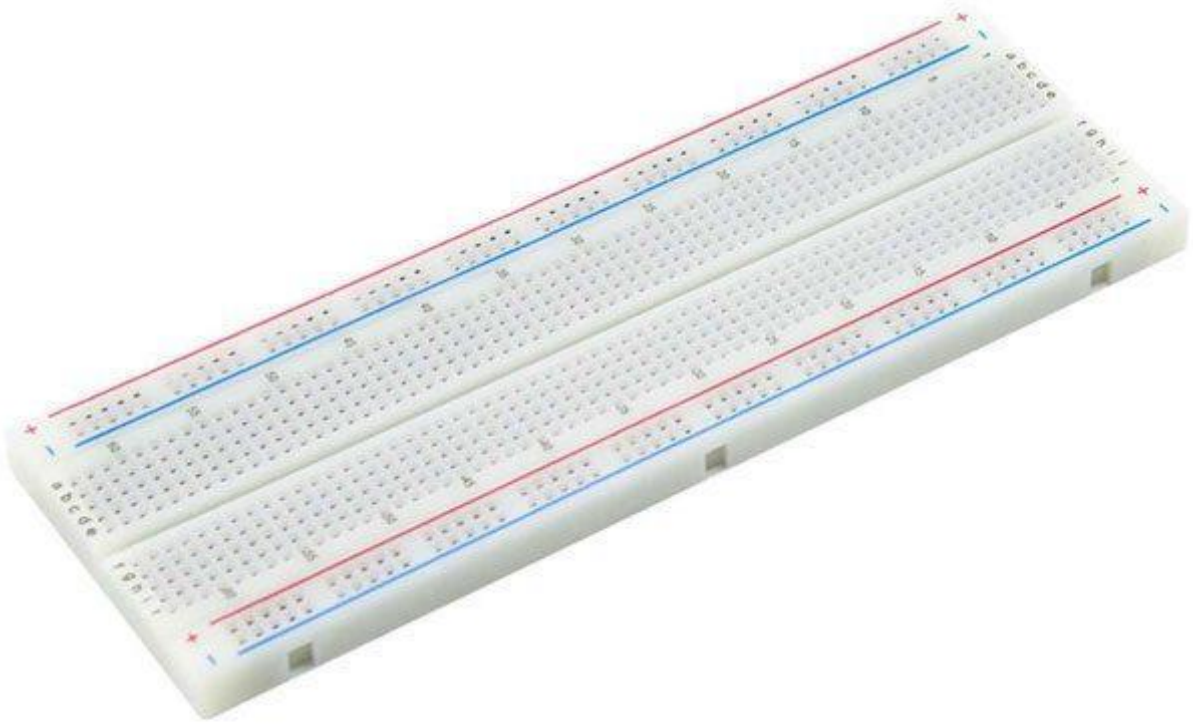


Рисунок 3.2 – Макетна плата на 830 точок

Підключення датчиків буде виконано з використанням макетної плати (англ. breadboard), що дозволяє легко і швидко з'єднувати компоненти без необхідності пайки. Макетна плата забезпечує гнучкість у створенні прототипів і дозволяє легко змінювати схему за потреби.

Для цього проєкту було обрано макетну плату на 830 контактів, що забезпечує достатньо місця для підключення всіх необхідних компонентів. Макетна плата дозволяє легко і швидко створювати електричні з'єднання між компонентами без пайки, що робить її ідеальним інструментом для розробки та тестування прототипів.

3.3.1 Підключення датчику температури та вологості DHT22

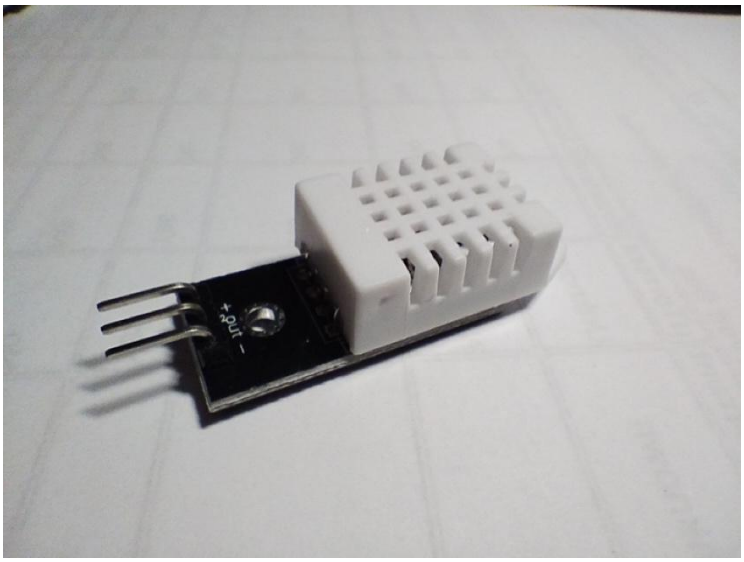


Рисунок 3.3 – Датчик DHT22 з палатою розширення

Датчик DHT22 , також відомий як AM2302, є популярним сенсором для вимірювання температури та вологості. Він використовує цифровий сигнал для передачі даних, що забезпечує високу точність і надійність вимірювань. Основні компоненти підключення включають чотири виводи: VCC, GND, DATA та N/C (не підключений). Завдяки чітким умовним позначкам і простій схемі підключення, DHT22 є зручним у використанні навіть для новачків у електроніці. [13]

Таблиця 3.2 – Таблиця характеристик датчика DHT22

Характеристика	**Значення**
Тип сенсора	Температура і вологість
Діапазон вимірювання температури	-40°C до +80°C
Точність вимірювання температури	±0.5°C
Діапазон вимірювання вологості	0% до 100% RH
Точність вимірювання вологості	±2-5% RH
Робоча напруга	3.3V до 5V
Споживання струму	1.5 мА (вимірювання), 0.3 мА (режим очікування)
Інтерфейс передачі даних	Цифровий (1-Wire протокол)
Час відгуку	2 секунди
Розміри	15.1 мм x 25 мм x 7.7 мм
Тип корпусу	Пластиковий корпус

Переваги використання DHT22:

– ваисока точність;

– цифровий інтерфейс;

– підходить для різних середовищ і умов експлуатації, від -40°C до +80°C і від 0% до 100% RH;

– низьке енергоспоживання.

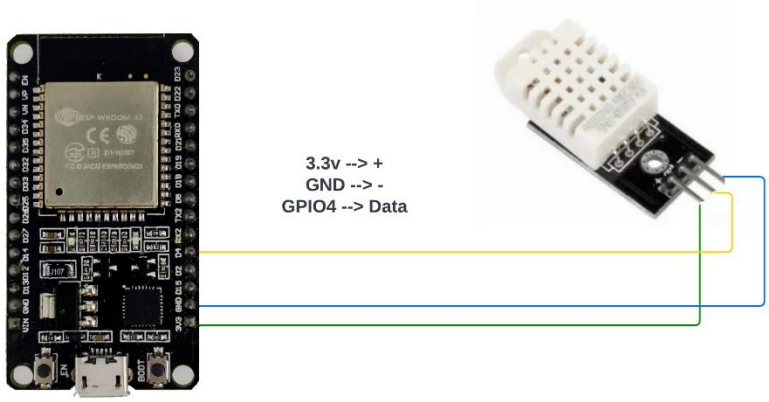


Рисунок 3.4 – Схема підключення DHT22 до ESP32

Схема підключення DHT22 до мікроконтролера включає такі кроки: підключити VCC до 3.3V або 5V джерела живлення, GND підключити до землі (GND), DATA підключити до цифрового входу мікроконтролера (наприклад, GPIO14), з підключенням 10kΩ резистора між VCC та DATA (якщо без платі розширення), а контакт N/C не використовується, або він не присутній на платі. [9]

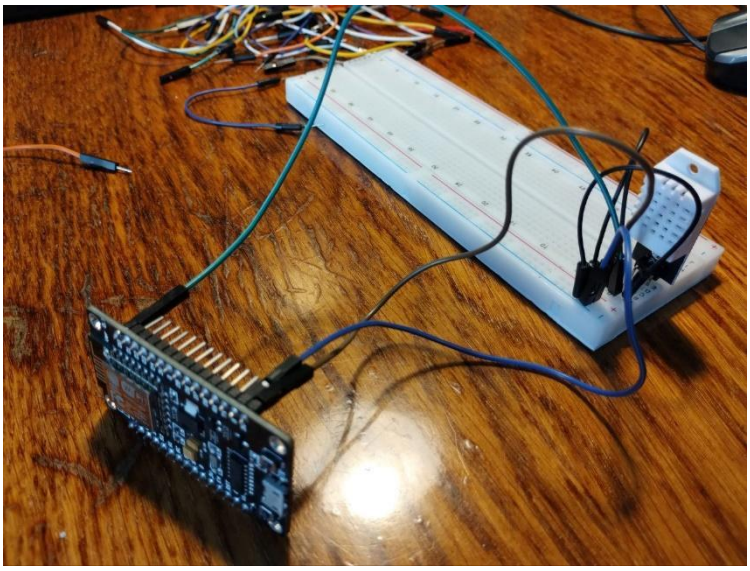


Рисунок 3.5 – Підключення датчику наживо

Загалом підключення DHT22 до мікроконтролера не є складним. Таке підключення забезпечить стабільну роботу датчика та коректну передачу даних до мікроконтролера. Відсутність підключення контакту N/C зумовлена тим, що він не має функціонального призначення в цій схемі, оскільки відповідає за заземлення як і GND.

3.3.2 Підключення датчику вологості ґрунту

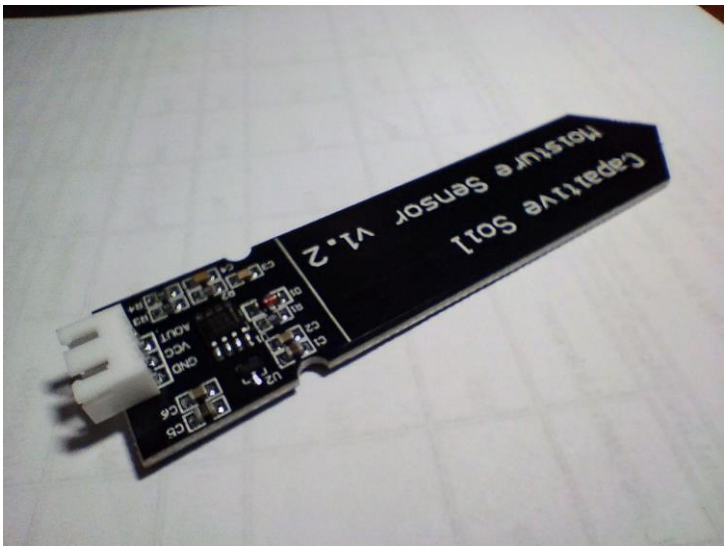


Рисунок 3.6 – Датчик вологості ґрунту

Для визначення вологості ґрунту будемо використовувати Capacitive Soil Moisture Sensor V1.2, який відзначається високою точністю і надійністю.

Цей сенсор використовує принцип ємнісного вимірювання, що дозволяє уникнути корозії та зносу, характерних для резистивних сенсорів. Завдяки цьому, сенсор має триваліший термін експлуатації і підходить для тривалого моніторингу вологості ґрунту в теплицях.

Принцип ємнісного вимірювання ґрунту полягає у визначенні вологості шляхом вимірювання зміни ємності між двома електродами, розташованими в сенсорі. Коли ґрунт сухий, його діелектрична проникність низька, що означає меншу ємність. З підвищенням вологості ґрунту, діелектрична проникність зростає через присутність води, що збільшує ємність сенсора. Ця зміна ємності використовується для визначення рівня вологості ґрунту.

Накопичувальний сенсор складається з двох електродів, які утворюють конденсатор. Ці електроди зазвичай покриті захисним ізоляційним шаром для запобігання корозії. Коли сенсор вставляється в ґрунт, утворюється електричне поле між електродами. Вода в ґрунті впливає на діелектричні властивості ґрунту, змінюючи його здатність зберігати електричний заряд.

Електрична ємність сенсора змінюється залежно від кількості води в ґрунті. Ця зміна вимірюється електронною схемою, яка перетворює ємність у відповідний аналоговий або цифровий сигнал. Чим вища вологість ґрунту, тим більша ємність сенсора. Мікроконтролер зчитує цей сигнал і обробляє його для визначення рівня вологості.

Таблиця 3.3 – Загальні характеристики датчика вологості ґрунту

Характеристика	**Значення**
Тип сенсора	Накопичувальний сенсор вологості ґрунту
Робоча напруга	3.3V до 5V
Вихідний сигнал	Аналоговий
Діапазон вимірювання	0% до 100% вологості
Розміри	98 мм x 23 мм
Інтерфейс	Аналоговий вихід (AO)
Матеріал	Непроникний для води корпус
Робоча температура	-40°C до +85°C
Довжина кабелю	200 мм

Із таблиці можна зробити висновок, що Capacitive Soil Moisture Sensor V1.2 є надійним і точним інструментом для вимірювання вологості ґрунту.

Цей сенсор використовує накопичувальний принцип вимірювання, що робить його менш вразливим до корозії та зносу, порівняно з резистивними сенсорами. Робоча напруга сенсора становить від 3.3V до 5V, що забезпечує гнучкість у виборі джерела живлення. Сенсор має аналоговий вихідний сигнал, що дозволяє легко інтегрувати його з різними мікроконтролерами, такими як ESP32.

Діапазон вимірювання сенсора становить від 0% до 100% вологості, що дозволяє точно контролювати стан ґрунту в широкому спектрі умов. Сенсор має компактні розміри (98 мм x 23 мм) та водонепроникний корпус, що забезпечує його довговічність і стабільну роботу в різних середовищах. Робоча температура сенсора охоплює діапазон від -40°C до +85°C, що робить його придатним для використання в екстремальних кліматичних умовах. Довжина кабелю приблизно 200 мм забезпечує достатню гнучкість при встановленні сенсора у потрібному місці.

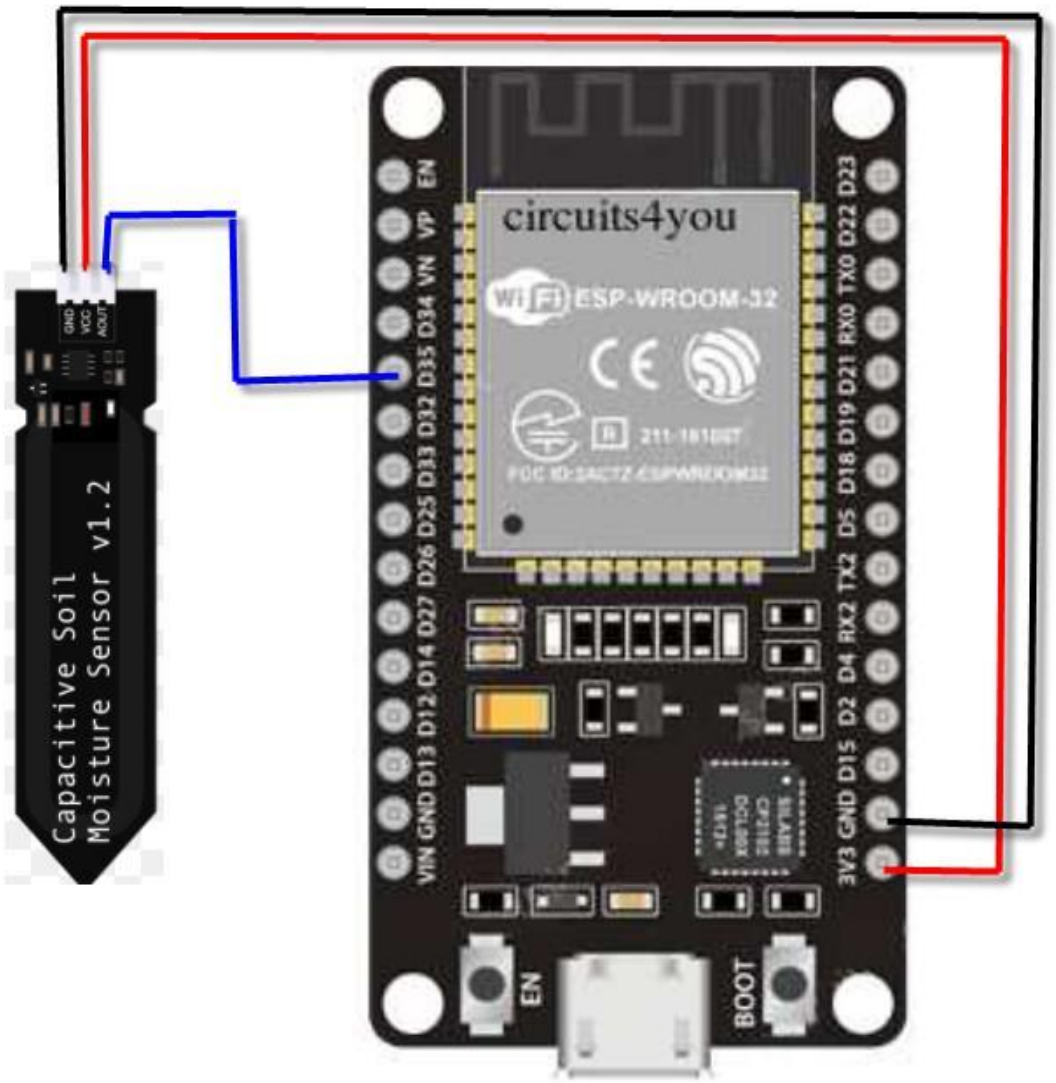


Рисунок 3.7 – Схема підключення датчику вологості грунту до ESP32

Схема підключення Capacitive Soil Moisture Sensor V1.2 до мікроконтролера включає три основні з'єднання. Вивід VCC сенсора потрібно підключити до джерела живлення з напругою 3.3V або 5V. Вивід GND підключається до землі (GND) мікроконтролера, забезпечуючи з'єднання з нульовим потенціалом. Вивід AO, який передає аналоговий сигнал, необхідно підключити до аналогового входу мікроконтролера, наприклад, до GPIO35 на ESP32.

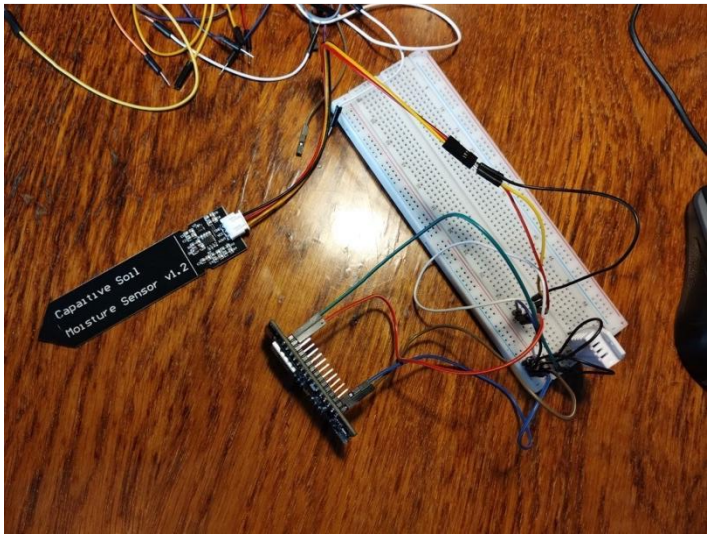


Рисунок 3.8 – Підключення датчику вологості ґрунту наживо

Завдяки своїм характеристикам та простоті використання, Capacitive Soil Moisture Sensor V1.2 є надійним вибором для моніторингу вологості ґрунту. Він допоможе забезпечити оптимальні умови для росту рослин, що особливо важливо в теплицях та інших контрольованих середовищах.

3.3.3 Підключення датчику освітленності

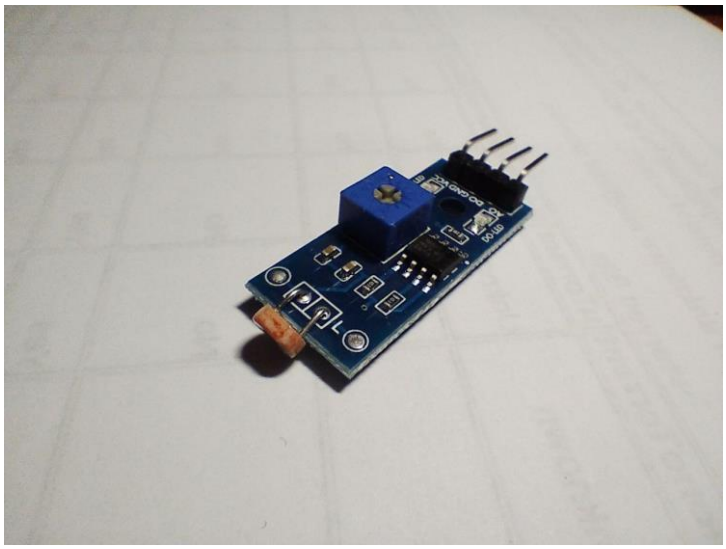


Рисунок 3.9 – Датчик освітленості MH Photoresistor Light Sensor

В якості датчика освітленості буде використовуватись MH Photoresistor Light Sensor. Цей сенсор використовує фоторезистор для вимірювання рівня освітленості, що дозволяє точно визначати кількість світла, яка потрапляє на

його поверхню. Фоторезистор змінює свій опір залежно від інтенсивності освітлення: чим більше світла, тим менший опір. Ця властивість дозволяє використовувати MH Photoresistor Light Sensor для моніторингу освітленості в різних умовах.

Таблиця 3.4 – Загальні характеристики фоторезистора MH Photoresistor Light Sensor

Характеристика	Значення
Тип сенсора	Фоторезистор
Робоча напруга	3.3V до 5V
Вихідний сигнал	Аналоговий/цифровий
Діапазон вимірювання	Широкий діапазон освітленості
Інтерфейс	Аналоговий вихід, цифровий вихід
Робоча температура	-30°C до +70°C
Довжина кабелю	-

Датчик освітленості MH Photoresistor Light Sensor є простим у використанні і надійним інструментом для вимірювання рівня освітлення. Завдяки своїй здатності змінювати опір залежно від інтенсивності світла, він забезпечує точні вимірювання і може бути легко інтегрований з різними мікроконтролерами, такими як ESP32 або ESP8266. Робоча напруга сенсора від 3.3V до 5V дозволяє підключати його до різноманітних джерел живлення. Аналоговий вихідний сигнал дозволяє легко зчитувати дані за допомогою аналогових входів мікроконтролерів.

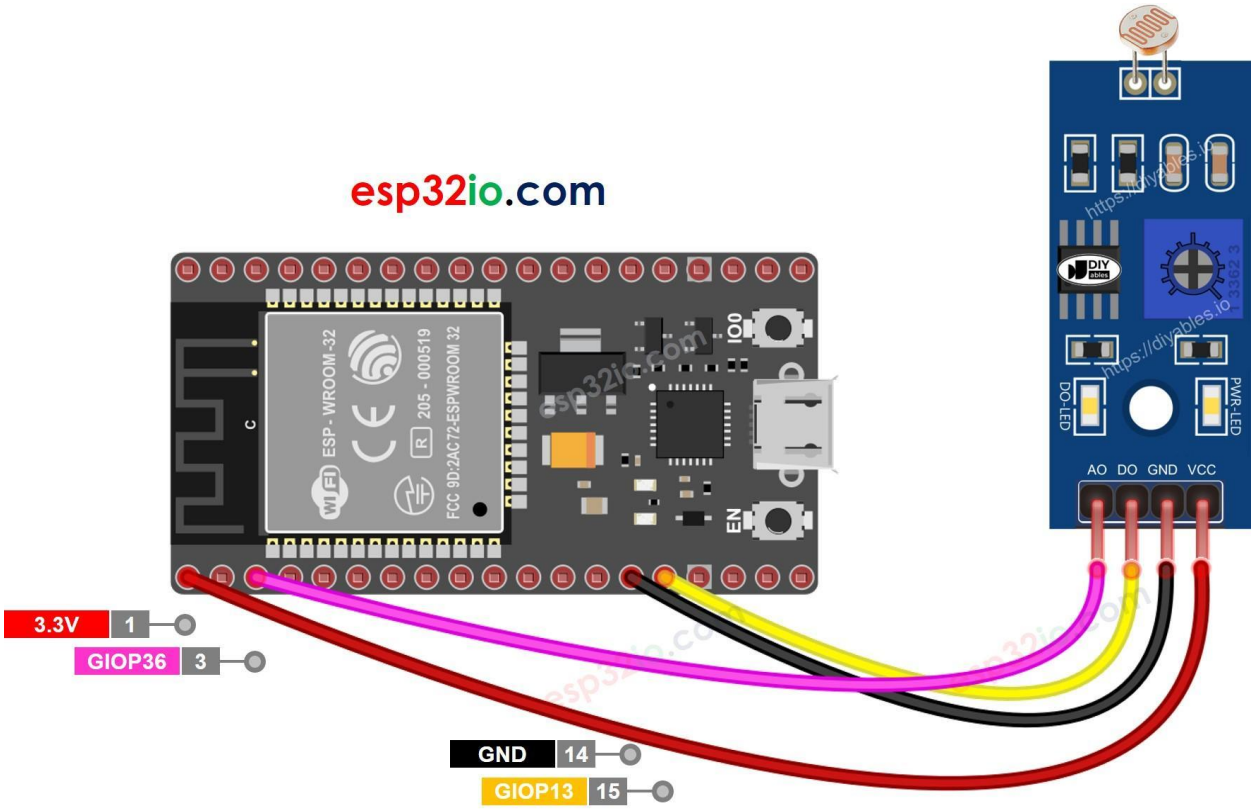


Рисунок 3.10 – Схема підключення фоторезистору до ESP32

Для підключення MH Photoresistor Light Sensor до мікроконтролера необхідно підключити VCC до 3.3V або 5V джерела живлення, GND до землі (GND), а DO до цифрового входу мікроконтролера (наприклад, GPIO13 на ESP32).

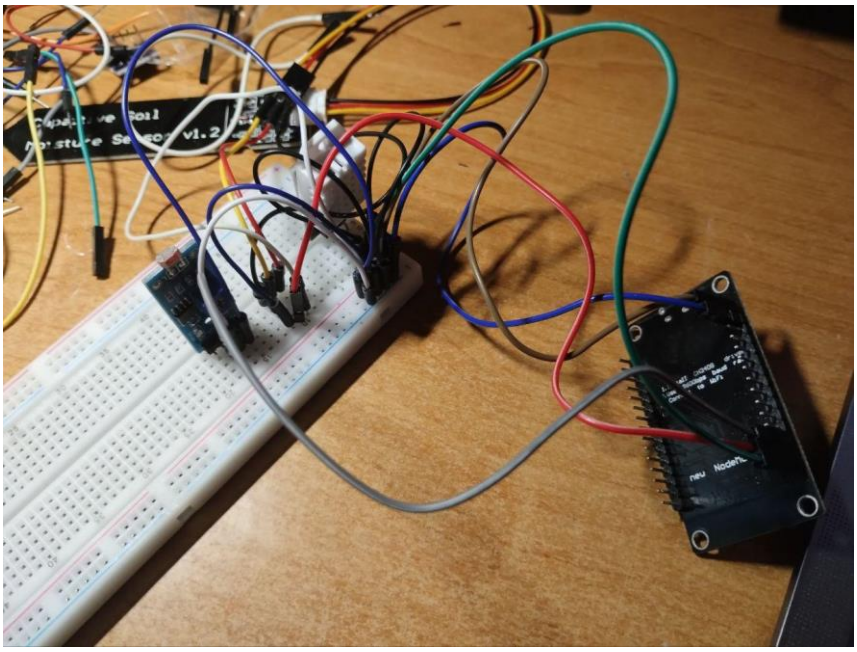


Рисунок 3.11 – Підключення датчику освітленості наживо

Використання MH Photoresistor Light Sensor для моніторингу освітленості у теплиці дозволяє точно контролювати рівень світла, забезпечуючи оптимальні умови для росту рослин. Це сприяє ефективному управлінню освітленням і підвищенню продуктивності вирощування.

3.3.4 Підключення датчику атмосферного тиску

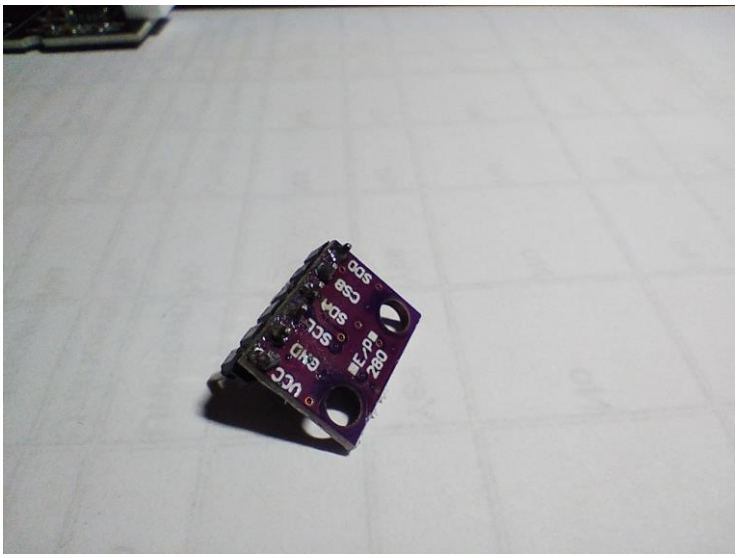


Рисунок 3.12 – Датчик тиску BMP280

Останнім датчиком, який буде використано в проєкті, є барометр BMP280. Цей сенсор відзначається високою точністю і надійністю у

вимірюванні атмосферного тиску та температури, що дозволяє ефективно контролювати кліматичні умови в теплиці.

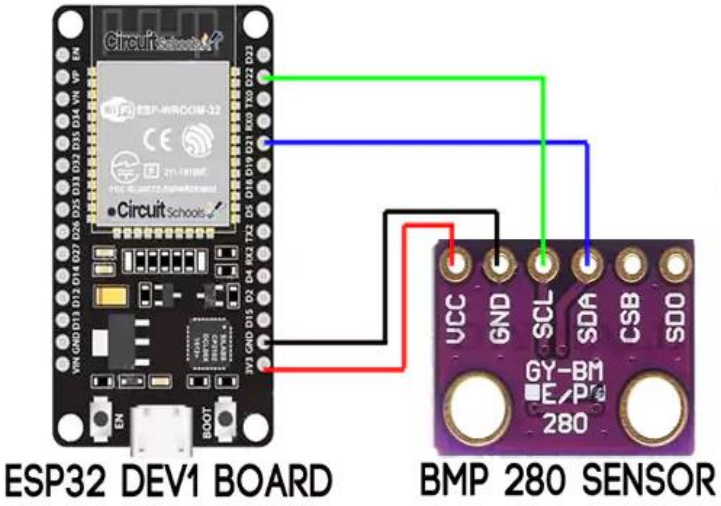


Рисунок 3.13 – Схема підключення датчику BMP280 до ESP32

Барометр BMP280 є високоточним сенсором, який дозволяє вимірювати атмосферний тиск і температуру з високою точністю. Використання інтерфейсів I2C або SPI забезпечує гнучкість у підключенні до різних мікроконтролерів, таких як ESP32 або ESP8266. Завдяки своїм компактним розмірам і низькому енергоспоживанню, BMP280 є ідеальним вибором для проєктів, де важлива точність вимірювань і економія енергії.

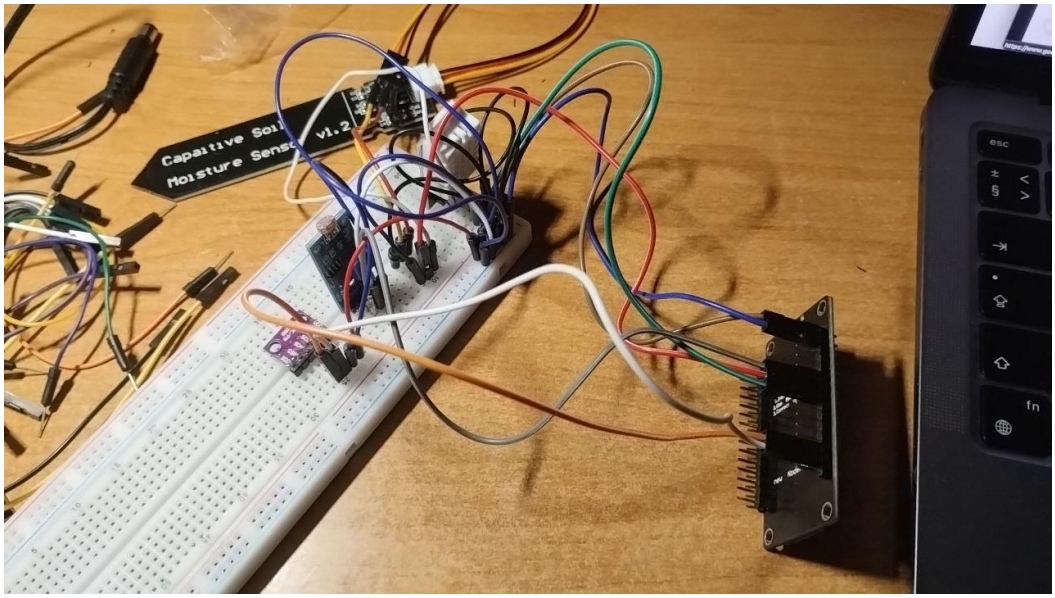


Рисунок 3.14 – Підключення датчику BMP280 наживо

Для підключення BMP280 до мікроконтролера ESP32 використовуватимемо інтерфейс I2C. Підключіть VCC до 3.3V джерела живлення на макетній платі, GND до шини землі (GND) на макетній платі, SCL до лінії тактового сигналу I2C на мікроконтролері, наприклад, GPIO22, а SDA до лінії даних I2C на мікроконтролері, наприклад, GPIO21.

3.3.5 Загальна схема підключення

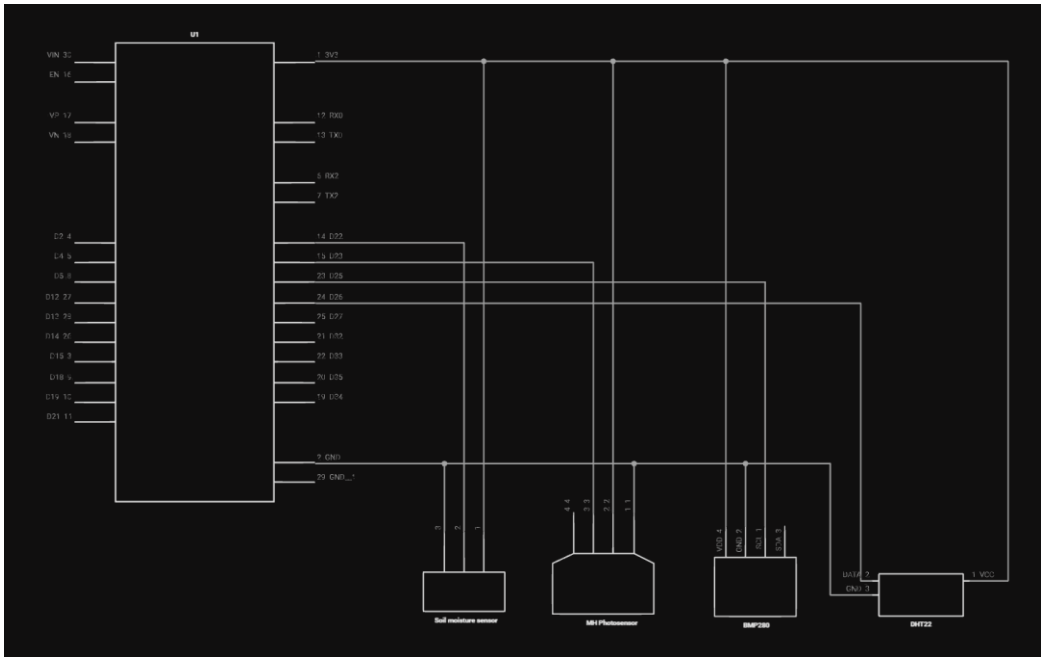


Рисунок 3.15 – Загальна схема підключення датчиків

Загальна схема апаратної частини проєкту виглядає наступним чином:

1) мікроконтролер ESP32:
    1) вивід 3V3 підключений до живлення датчиків;
    2) виводи GND підключені до загальної землі для всіх датчиків;
2) підключення DHT22:
    1) вивід VCC підключений до 3V3 на макетній платі;
    2) вивід GND підключений до GND на макетній платі;
    3) вивід DATA підключений до GPIO15 мікроконтролера;
3) підключення Capacitive Soil Moisture Sensor:

1) вивід VCC підключений до 3V3 на макетній платі;
2) вивід GND підключений до GND на макетній платі;
3) вивід AO підключений до GPIO35 мікроконтролера;

4) підключення MH Photoresistor Light Sensor:
   1) вивід VCC підключений до 3V3 на макетній платі;
   2) вивід GND підключений до GND на макетній платі;
   3) вивід AO підключений до GPIO34 мікроконтролера;

5) підключення BMP280:
   1) вивід VCC підключений до 3V3 на макетній платі;
   2) вивід GND підключений до GND на макетній платі;
   3) вивід SCL підключений до GPIO22 мікроконтролера;
   4) вивід SDA підключений до GPIO21 мікроконтролера.

Ця схема забезпечує надійне і стабільне підключення всіх датчиків до мікроконтролера ESP32, використовуючи макетну плату для спрощення з'єднань. Завдяки цьому підходу можна легко змінювати і налаштовувати конфігурацію компонентів, забезпечуючи гнучкість у процесі розробки і тестування системи.

Ця схема була використана для реалізації системи моніторингу кліматичних умов у теплиці. Вона включає підключення декількох сенсорів до мікроконтролера ESP32, що забезпечує збір даних про різні параметри навколишнього середовища. Конкретні підключення виконуються за допомогою макетної плати на 830 контактів, що дозволяє легко модифікувати схему та налаштовувати її відповідно до потреб проєкту. Повністю зібрана аппаратна частина була приведена на рисунку 3.11.

3.4 Програмування ESP32

Для отримання даних з датчиків необхідно запрограмувати мікроконтролер ESP32 відповідно до специфікацій кожного сенсора. Код буде реалізовано в середовищі розробки Arduino IDE, що дозволить зручно

управляти всіма підключеними датчиками і збирати необхідну інформацію про кліматичні умови в теплиці.

3.4.1 Отримання даних з датчику DHT22

Для роботи з датчиком BMP280 рекомендовано використовувати бібліотеку Adafruit_BMP280.h, яка забезпечує простий та зручний інтерфейс для взаємодії з цим сенсором. Ця бібліотека підтримує як I2C, так і SPI інтерфейси, що дозволяє гнучко налаштовувати підключення сенсора до мікроконтролера

Для установки бібліотеки Adafruit BMP280 в Arduino IDE потрібно відкрити програму Arduino IDE. У меню «Sketch», потрібно обрати «Include Library» і далі «Manage Libraries...». У вікні, що відкрилося, введіть «Adafruit BMP280» у поле пошуку. Після цього виберіть бібліотеку Adafruit BMP280 і натисніть «Install».

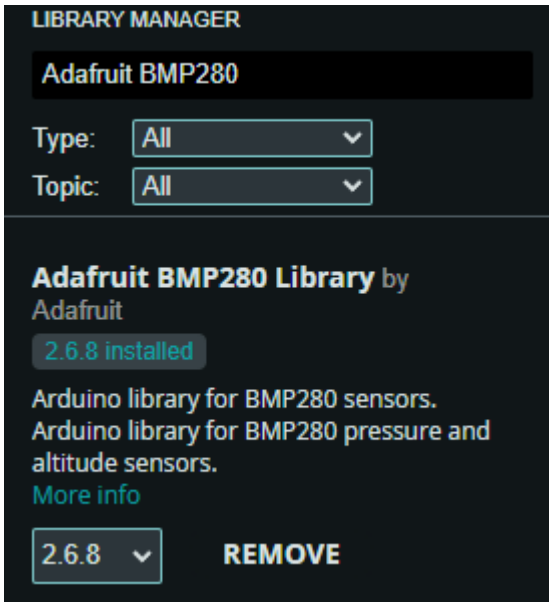


Рисунок 3.16 – Підключення бібліотеки Adafruit BMP280

Код призначений для зчитування даних з датчика BMP280 (температури, тиску та висоти) та виведення цих даних на серійний монітор. Він використовує бібліотеки Wire та Adafruit_BMP280.

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BMP280.h>

Adafruit_BMP280 bmp; // створення об'єкта для датчика

void setup() {
  Serial.begin(9600); // ініціалізація серійного порту для виводу даних
  if (!bmp.begin()) {
    Serial.println("Не вдалося знайти датчик BMP280. Перевірте підключення!");
    while (1);
  }
}

void loop() {
  Serial.print("Температура = ");
  Serial.print(bmp.readTemperature());
  Serial.println(" *C");

  Serial.print("Тиск = ");
  Serial.print(bmp.readPressure() / 100.0F);
  Serial.println(" hPa");

  Serial.println();
  delay(2000); // затримка між вимірами
}
```

Рисунок 3.17 – Код для роботи з датчиком BMP280

Оголошуються два піни, які використовуються для підключення I2C. Пін 21 призначений для SDA (лінія даних), а пін 22 для SCL (лінія годинника). Створюється об'єкт bmp класу Adafruit_BMP280, який буде використовуватися для взаємодії з датчиком BMP280 через I2C.

Спочатку налаштуємо контролер, для цього ініціалізується серійне з'єднання зі швидкістю 115200 біт/с. Це дозволяє виводити дані на серійний монітор для їх перегляду. Далі ініціалізується I2C-зв'язок, використовуючи піни 21 для SDA (лінія даних) і 22 для SCL (лінія годинника), що дозволяє взаємодіяти з датчиком. Потім ініціалізується сам BMP280 за адресою 0x76. Якщо датчик не знайдено, виводиться повідомлення про помилку на серійний монітор, і програма зациклюється, чекаючи на втручання користувача.

На кожному колі циклу виконання спочатку зчитується температура за допомогою методу readTemperature() об'єкта bmp. Отримане значення температури виводиться на серійний монітор з підписом "Temperature" і одиницями вимірювання в градусах Цельсія за допомогою функцій Serial.print() і Serial.println().

Потім зчитується тиск за допомогою методу readPressure() об'єкта bmp. Отримане значення тиску виводиться на серійний монітор з підписом "Pressure" і одиницями вимірювання в паскалях аналогічно до температури, використовуючи Serial.print() і Serial.println().

Далі обчислюється висота над рівнем моря за допомогою методу readAltitude(1013.25) об'єкта bmp. Значення 1013.25 є стандартним атмосферним тиском на рівні моря, який можна налаштувати відповідно до місцевих умов. Отримане значення висоти виводиться на серійний монітор з підписом "Approx. Altitude" і одиницями вимірювання в метрах. Після цього виконується затримка на 2000 мілісекунд (2 секунди) перед наступним вимірюванням за допомогою функції delay(2000), щоб зменшити частоту зчитувань і надавати оновлену інформацію через регулярні проміжки часу.

3.4.2 Зчитування данних з інших датчиків

Поєднуючи зчитування даних з усіх датчиків (DHT22, Capacitive Soil Moisture Sensor, MH Photoresistor Light Sensor та BMP280), можна створити комплексну систему моніторингу кліматичних умов теплиці. Ось як виглядає повний код для ESP32 з усіма підключеними датчиками.

```
// Створення об'єктів для датчиків
DHT dht(DHTPIN, DHTTYPE);
Adafruit_BMP280 bmp; // I2C

void setup() {
  Serial.begin(115200);
  dht.begin();
  Wire.begin(BMP_SDA, BMP_SCL);

  if (!bmp.begin(0x76)) {
    Serial.println("Could not find a valid BMP280 sensor, check wiring!");
    while (1);
  }
}

void loop() {
  // Зчитування даних з DHT22
  float humidity = dht.readHumidity();
  float temperatureDHT = dht.readTemperature();

  // Зчитування даних з Soil Moisture Sensor
  int soilMoistureValue = analogRead(SOIL_MOISTURE_PIN);

  // Зчитування даних з Light Sensor
  int lightLevel = analogRead(LIGHT_SENSOR_PIN);

  // Зчитування даних з BMP280
  float temperatureBMP = bmp.readTemperature();
  float pressure = bmp.readPressure();
  float altitude = bmp.readAltitude(1013.25); // Стандартний атмосферний тиск на рівні моря в hPa

  // Виведення даних на серійний монітор
  Serial.print("Humidity: ");
  Serial.println(humidity);
  Serial.print("C°: ");
  Serial.println(temperatureDHT);
  Serial.print("Temperature (DHT22): ");
  Serial.println(temperatureDHT);
  Serial.print("Temperature (BMP280): ");
  Serial.println(temperatureBMP);
  Serial.print("Pressure: ");
  Serial.println(pressure / 100.0);
  Serial.print("Altitude: ");
  Serial.println(altitude);

  delay(2000); // затримка між вимірами
}
```

Рисунок 3.18 – Код для зчитування даних з інших датчиків

У функції loop() здійснюється зчитування даних з кількох датчиків і виведення цих даних на серійний монітор. Спочатку зчитуються дані з датчика DHT22, який вимірює вологість і температуру. Метод readHumidity() об'єкта dht зчитує значення вологості, а метод readTemperature() — температуру. Отримані значення зберігаються у змінних humidity та temperatureDHT відповідно.

Після цього зчитуються дані з датчика вологості ґрунту, підключеного до аналогового піна 35. Метод analogRead(SOIL_MOISTURE_PIN) повертає значення вологості ґрунту, яке зберігається у змінній soilMoistureValue. Аналогічно, зчитується рівень освітленості за допомогою датчика світла, підключеного до аналогового піна 34, і зберігається у змінній lightLevel.

Далі здійснюється зчитування даних з датчика BMP280. Метод readTemperature() об'єкта bmp зчитує температуру, метод readPressure() —

тиск, а метод readAltitude(1013.25) обчислює висоту над рівнем моря, використовуючи стандартний атмосферний тиск на рівні моря (1013.25 гПа). Значення зберігаються у змінних temperatureBMP, pressure та altitude відповідно.

Отримані значення з усіх датчиків виводяться на серійний монітор. Для кожного вимірювання використовується функція Serial.print() для виведення підписів та значень, і Serial.println() для завершення рядка. Після виведення всіх даних, функція delay(2000) робить паузу на 2000 мілісекунд (2 секунди) перед наступним циклом зчитування, що дозволяє зменшити частоту оновлення даних і зробити виведення більш зручним для користувача.

3.5 Налаштування AWS IoT та інтеграція з хмарними сервісами

Зазвичай комунікація ESP32 з сервісами Amazon здійснюється за допомогою протоколу MQTT (Message Queuing Telemetry Transport). MQTT є легким протоколом обміну повідомленнями, який ідеально підходить для роботи з IoT-пристроями завдяки його низькому енергоспоживанню та ефективності передачі даних.

MQTT на ESP32 реалізується за допомогою бібліотеки PubSubClient, яка надає простий та ефективний інтерфейс для роботи з протоколом MQTT. Для успішної реалізації комунікації з AWS IoT необхідно налаштувати Wi-Fi-з'єднання, встановити сертифікати безпеки та налаштувати параметри MQTT-з'єднання.

Для інтеграції отриманої системи з AWS IoT необхідно налаштувати комунікацію між мікроконтролером ESP32 і сервісами AWS IoT. Це включає налаштування AWS IoT, створення необхідних ресурсів, встановлення сертифікатів безпеки і написання коду для ESP32, який буде відправляти дані до AWS IoT.

```
23  void setup() {
24    Serial.begin(115200);
25    Wire.begin(BMP_SDA, BMP_SCL);
26
27    // Підключення до Wi-Fi
28    connectToWiFi();
29
30    // Підключення до MQTT
31    connectToMQTT();
32  }
33
34  void connectToWiFi() {
35    WiFi.begin(ssid, password);
36    Serial.print("Connecting to Wi-Fi...");
37    while (WiFi.status() != WL_CONNECTED) {
38      delay(500);
39      Serial.print(".");
40    }
41    Serial.println(" connected!");
42  }
43
44  void connectToMQTT() {
45    wifiClient.setCACert(root_ca);
46    wifiClient.setCertificate(client_cert);
47    wifiClient.setPrivateKey(private_key);
48
49    Serial.print("Connecting to MQTT...");
50    while (!client.connected()) {
51      if (client.connect(deviceID)) {
52        Serial.println(" connected!");
53      } else {
54        Serial.print(" failed, rc=");
55        Serial.print(client.state());
56        delay(2000);
57      }
58    }
59  }
60
```

Рисунок 3.19 – Код для інтеграції ESP32 з AWS IoT

На початку коду підключаються необхідні бібліотеки: WiFi.h, WiFiClientSecure.h, PubSubClient.h, Wire.h, Adafruit_Sensor.h, Adafruit_BMP280.h, та DHT.h. Визначаються константи для налаштування Wi-Fi, AWS IoT, ідентифікатор пристрою, та піни для підключення датчиків.

Оголошуються об'єкти для датчиків: dht для датчика вологості і температури DHT22, bmp для датчика BMP280, а також wifiClient для захищеного з'єднання Wi-Fi і client для з'єднання з AWS IoT через MQTT протокол.

Функція connectToWiFi() ініціалізує підключення до Wi-Fi, використовуючи задані SSID і пароль. Вона постійно перевіряє статус підключення і виводить відповідні повідомлення на серійний монітор.

Функція connectToAWS() налаштовує сертифікати та приватний ключ для захищеного з'єднання з AWS IoT. Вона намагається підключитися до AWS IoT, використовуючи заданий ідентифікатор пристрою, і виводить відповідні повідомлення на серійний монітор.

Функція setup() виконує початкові налаштування, включаючи серійне з'єднання, підключення до Wi-Fi, ініціалізацію I2C-шини та датчика BMP280, ініціалізацію датчика DHT22, і підключення до AWS IoT. Якщо датчик BMP280 не вдається знайти, програма виводить повідомлення про помилку і зациклюється.

Функція loop() виконується безперервно і містить основну логіку роботи. Спочатку перевіряється підключення до AWS IoT, і якщо з'єднання втрачено, функція повторно підключається. Потім зчитуються дані з датчиків DHT22, вологості ґрунту, рівня освітленості та BMP280. Отримані значення формуються в JSON-об'єкт і відправляються на AWS IoT за допомогою методу client.publish(). Успішність відправлення виводиться на серійний монітор. Після цього виконується затримка на 2000 мілісекунд перед наступним циклом вимірювання.

3.6 Інтерфейс для отримання даних з датчиків

Для отримання даних з датчиків та їх відправки на електронну пошту можна використовувати сервіс AWS Simple Email Service (SES) разом з AWS Lambda. Ця інтеграція дозволить автоматично відправляти електронні листи з даними, отриманими з датчиків, на задану адресу електронної пошти. Нижче наведено кроки для налаштування цієї системи.

Для налаштування AWS IoT Rules Engine і SNS для відправки електронних листів спочатку необхідно створити IoT Topic Rule. У консолі AWS IoT відкривається розділ "Act" і вибирається опція "Create a rule". Вводиться ім'я для нового правила і SQL-запит для фільтрації повідомлень. Наприклад, SQL-запит може виглядати як SELECT * FROM 'your/topic', що означає вибір усіх повідомлень з вказаної теми. Далі додається дія "Send a message to an SNS topic" і конфігурується ця дія, вибираючи існуючу тему SNS або створюючи нову.

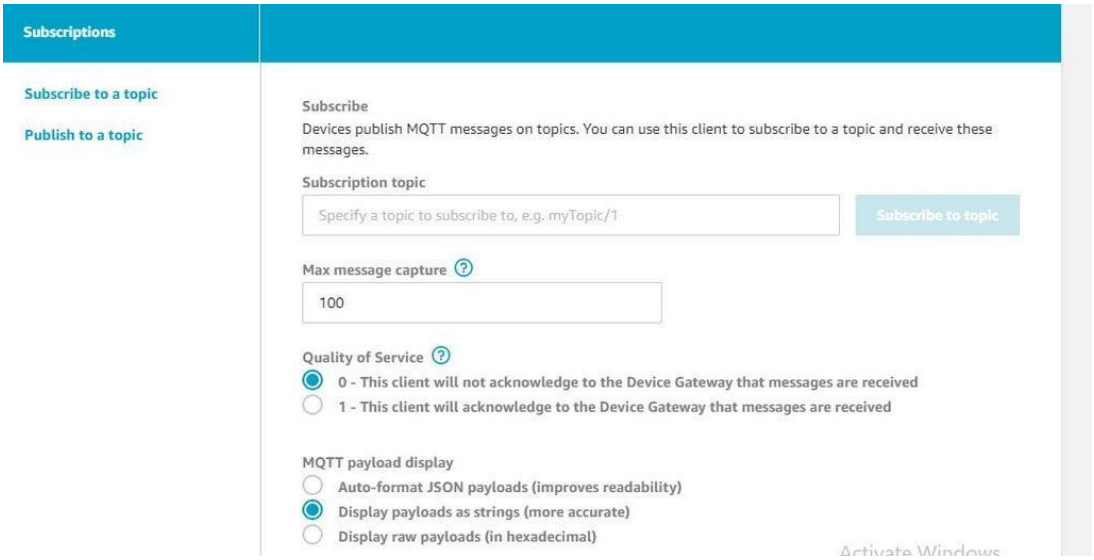


Рисунок 3.20 – Створення підписки для SNS

Наступним кроком є створення теми SNS. У консолі Amazon SNS вибирається опція "Topics", а потім "Create topic". Вибирається тип теми (стандартна або FIFO) і вводиться ім'я для нової теми. Після створення теми відкривається її сторінка і вибирається опція "Create subscription". У полі "Protocol" вибирається "Email", а в полі "Endpoint" вводиться електронна адреса, на яку будуть надсилатися повідомлення. Після цього натискається "Create subscription".

Підписка на тему SNS повинна бути підтверджена. Це робиться шляхом переходу за посиланням у листі, який надсилається на вказану електронну адресу після створення підписки. Після підтвердження підписки система готова до відправки повідомлень на електронну адресу через тему SNS.

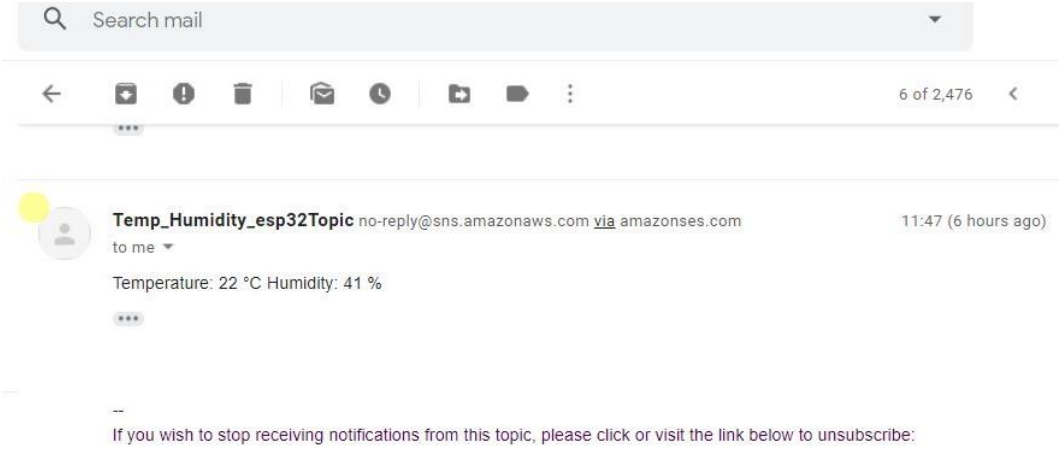


Рисунок 3.21 – Відправка листів на пошту

Управління сертифікатами та ключами для AWS IoT виконується у функції connectToAWS(). Для безпечного з'єднання з AWS IoT клієнтський об'єкт wifiClient налаштовується з використанням кореневого сертифіката, клієнтського сертифіката і приватного ключа. Підключення до AWS IoT здійснюється за допомогою методу client.connect(), який використовує унікальний ідентифікатор пристрою. Якщо з'єднання не вдалося встановити, виводяться відповідні повідомлення про помилку і повторюються спроби підключення.

Основний цикл програми забезпечує безперервне зчитування даних з датчиків і відправку їх на AWS IoT. Якщо з'єднання з AWS IoT втрачається, виконується повторне підключення. Дані з датчиків зчитуються та формуються у JSON-об'єкт, який відправляється на задану тему AWS IoT за допомогою методу client.publish(). Відправлені дані включають вологість, температуру (зчитану з двох різних датчиків), рівень вологості ґрунту, рівень освітленості, тиск і висоту над рівнем моря. Виконується затримка у 2000 мілісекунд перед наступним циклом вимірювання для зменшення частоти оновлень.

Висновки до розділу

В процесі розробки системи була реалізована комплексна архітектура, що включає кілька компонентів для зчитування, обробки та передачі даних з датчиків у хмарні сервіси. Опис архітектури системи включає детальне пояснення структури та взаємодії між компонентами, що дозволяє забезпечити надійність та ефективність роботи системи.

На етапі вибору компонентів було ретельно підібрано апаратне та програмне забезпечення, що оптимально відповідає вимогам проєкту, зокрема, використання датчиків DHT22, BMP280, датчиків вологості ґрунту та освітленості, а також контролера ESP32, який забезпечує необхідні функції та продуктивність.

Підключення та конфігурація датчиків було здійснено відповідно до технічних характеристик і специфікацій, що включає налаштування відповідних пінів та початкове налаштування для забезпечення коректного зчитування даних. Програмування ESP32 включало розробку коду для зчитування даних з датчиків, обробки цих даних та їх передачі до хмарного сервісу AWS IoT, що забезпечує інтеграцію з іншими системами та сервісами.

Налаштування AWS IoT та інтеграція з хмарними сервісами дозволила створити надійний канал для передачі даних з датчиків до хмари, що включає налаштування правил IoT та використання SNS для відправки повідомлень, таких як електронні листи, що дозволяє отримувати дані в режимі реального часу та оперативно реагувати на них.

Інтерфейс для отримання даних з датчиків забезпечує зручний спосіб моніторингу та аналізу отриманих даних, що включає як локальний вивід на серійний монітор, так і відправку даних до хмарних сервісів для подальшого аналізу та використання в різних додатках.

ВИСНОВКИ

Було виконано поставлену мету вивчити та реалізувати IoT технології для сільського господарства, зокрема з використанням платформи AWS IoT. В ході роботи були успішно виконані наступні завдання:

– проведення всебічного аналізу IoT технологій, їх застосування у сільському господарстві та можливостей платформи AWS IoT;

– визначення переваг використання AWS IoT для моніторингу та управління аграрними процесами;

– огляд основних середовищ розробки для створення IoT-рішень і вибір найкращих інструментів для проєкту;

– формування вимог до апаратних та програмних компонентів системи;

– вибір і конфігурація апаратних компонентів, включаючи датчики та контролери;

– розробка програмного коду для збирання даних з датчиків і передачі їх до AWS IoT;

– налаштування інтеграції з хмарними сервісами AWS IoT для ефективного моніторингу.

На етапі формування вимог були визначені необхідні апаратні та програмні компоненти для розробки системи. Визначення апаратних вимог включало вибір відповідних датчиків, контролера та інших компонентів, що забезпечують збирання та передачу даних. Програмні вимоги описували необхідне програмне забезпечення для інтеграції та обробки даних, зокрема, використання платформ та бібліотек для роботи з AWS IoT. Функціональні вимоги охоплювали необхідні функції системи, такі як збирання, обробка та передача даних, а нефункціональні вимоги описували очікувані характеристики, такі як надійність, безпека та масштабованість системи. Висновки цього розділу підсумували визначені вимоги та підготовку до етапу розробки.

Розробка включала кілька ключових етапів, починаючи з опису архітектури системи, що охоплює всі компоненти та їх взаємодію. Було проведено ретельний вибір апаратних компонентів, включаючи датчики для збирання даних про температуру, вологість, тиск, освітленість та вологість ґрунту. Після вибору компонентів було здійснено їх підключення та конфігурацію, що забезпечує коректну роботу датчиків і збирання даних. Програмування ESP32 включало розробку коду для збирання даних з датчиків та їх передачу до AWS IoT. Наступним етапом було налаштування AWS IoT для забезпечення інтеграції з хмарними сервісами та відправки даних, що дозволило створити надійну і ефективну систему для моніторингу. Інтерфейс для отримання даних з датчиків забезпечує зручність у використанні та моніторингу отриманих даних. [14]

Загалом, виконана робота продемонструвала можливості та ефективність використання IoT технологій у сільському господарстві для моніторингу та управління різними процесами. Реалізація проєкту показала, що інтеграція апаратних компонентів із хмарними сервісами AWS IoT дозволяє створювати потужні рішення для збирання, обробки та аналізу даних, що сприяє підвищенню ефективності та продуктивності у сільськогосподарському секторі.

Всі завдання було виконано в повному обсязі, і в результаті роботи було розроблено апаратно-програмний комплекс для системи сповіщення змін кліматичних умов теплиці на базі AWS IoT.

У майбутньому, інтеграція штучного інтелекту (ШІ) у розроблений апаратно-програмний комплекс для системи сповіщення змін кліматичних умов теплиці відкриває нові можливості для підвищення ефективності та автоматизації управління теплицями. Використання алгоритмів машинного навчання та інших технологій ШІ може значно покращити функціональність та інтелектуальність системи.

Застосування методів машинного навчання дозволить створити моделі для прогнозування змін кліматичних умов всередині теплиці. На основі

історичних даних про температуру, вологість, тиск та освітленість можна навчити модель передбачати майбутні зміни та виявляти потенційні аномалії. Це допоможе вчасно вживати заходів для запобігання несприятливим умовам для рослин.

ШІ може забезпечити адаптивне управління мікрокліматом у теплиці, автоматично налаштовуючи параметри, такі як температура, вологість, вентиляція та освітлення, залежно від потреб рослин у різні періоди їхнього розвитку. Це забезпечить оптимальні умови для росту та розвитку рослин, підвищуючи врожайність.

ШІ може допомогти оптимізувати використання ресурсів, таких як вода, світло та добрива, на основі аналізу даних з датчиків та зовнішніх джерел. Наприклад, система може автоматично регулювати полив залежно від рівня вологості ґрунту та прогнозованої погоди, що знижує витрати ресурсів та підвищує ефективність.

Інтеграція ШІ дозволить створити більш гнучкі та адаптивні системи, які можуть взаємодіяти з іншими IoT-пристроями та сервісами. Наприклад, система може отримувати дані з метеорологічних станцій або інших джерел для більш точного прогнозування та управління. Також можлива інтеграція з системами управління фермою для загальної координації роботи.

Таким чином, інтеграція штучного інтелекту в систему сповіщення змін кліматичних умов теплиці має великі перспективи для підвищення автоматизації, ефективності та інтелектуальності управління, що сприятиме оптимальному росту рослин і підвищенню врожайності.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Kumaraperumal R., Sellaperumal P., Kaliaperumal R. Smart Farming: Internet of Things (IoT)-Based Sustainable Agriculture. Agriculture. 2022. Т. 12, № 10. С. 1745.

2. Javaid M., Haleem A., Singh R.P. Role of IoT Technology in Agriculture: A Systematic Literature Review. MDPI. 2020. Т. 10, № 9. С. 3111.

3. Chaudhary V., Agarwal R., Sharma A. Internet of Things for the Future of Smart Agriculture: A Comprehensive Review. IEEE Xplore. 2017.

4. Turukmane A. V., Pradeepa M., Reddy K. S. S., Suganthi R., Riyazuddin Y. Md., Tallapragada V. V. S. Smart farming using cloud-based IoT data analytics // Measurement: Sensors. 2023. Т. 27. С. 100806.

5. Manchanda G., Papnai B., Lochab A., Badhani S. IoT-Based Smart Farming for Sustainable Agriculture // Advances in IoT and Security with Computational Intelligence: зб. доп. на конференції ICAISA 2023. LNNS, Vol. 756. 2023. С. 27–37.

6. Digitising Agriculture. Developing Digital Technologies. Precision Farming. URL: https://ec.europa.eu/eip/agriculture/en/digitising-agriculture/developing-digital-technologies/precision-farming-0.html (дата звернення: 03.06.2024).

7. Розумна система поливу Eve Aqua Smart Water Controller Apple HomeKit. URL: https://avtopoliv.com.ua/rozumna-systema-palyva-eve-aqua-smart-water-controller-apple-homekit-10ecc8101/ (дата звернення: 03.06.2024).

8. Розумні теплиці. URL: https://iotji.io/solutions/rozumni-teplytsi/ (дата звернення: 03.06.2024).

9. 3 категорії датчиків температури. URL: https://iotji.io/3-kategorii-datchykiv-tempeatury/ (дата звернення: 03.06.2024).

10. Components of Precision Farming: A Breakdown of the Essential Elements. URL: https://semantictech.in/blogs/components-of-precision-farming-a-breakdown-of-the-essential-elements/ (дата звернення: 03.06.2024).

11. Схема підключення системи поливу. URL: https://poliv-service.kiev.ua/ua/a287397-shema-podklyucheniya-sistemy.html (дата звернення: 03.06.2024).

12. IoTJI. URL: https://iotji.io/ (дата звернення: 03.06.2024).

13. Datasheet DHT22. URL: https://arduino.ua/docs/DHT22.pdf (дата звернення: 03.06.2024).

14. Greenhouse Environment Control System: пат. 10615412 США : A01G 9/00. № US 10615412 ; заявл. 27.02.2018 ; опубл. 07.04.2020, Бюл. № 14. 15 с.

15. Automated Greenhouse Management System Using IoT: пат. 2021104408 Австралія : H04L 12/28. № AU 2021104408 ; заявл. 12.04.2021 ; опубл. 20.10.2021, Бюл. № 41. 13 с.

16. Temperature Monitoring and Airflow Control System for Greenhouse: пат. 11027440 Китай : H04W 4/00. № CN 11027440 ; заявл. 14.03.2018 ; опубл. 10.09.2019, Бюл. № 37. 11 с.

17. IoT-Based Smart Greenhouse Control System: пат. 2852321 Канада : A01G 7/04. № CA 2852321 ; заявл. 11.12.2013 ; опубл. 09.06.2015, Бюл. № 23. 10 с.

18. Intelligent Greenhouse Monitoring and Control System: пат. 3021218 ЄС : G01W 1/10. № EP 3021218 ; заявл. 19.11.2015 ; опубл. 24.05.2017, Бюл. № 21. 12 с.

ДОДАТОК А

ЗВІТ З АНТИПЛАГІАТУ

бакалаврської кваліфікаційної роботи на тему:
«Система попередження ДТП шляхом виявлення засинання водія»

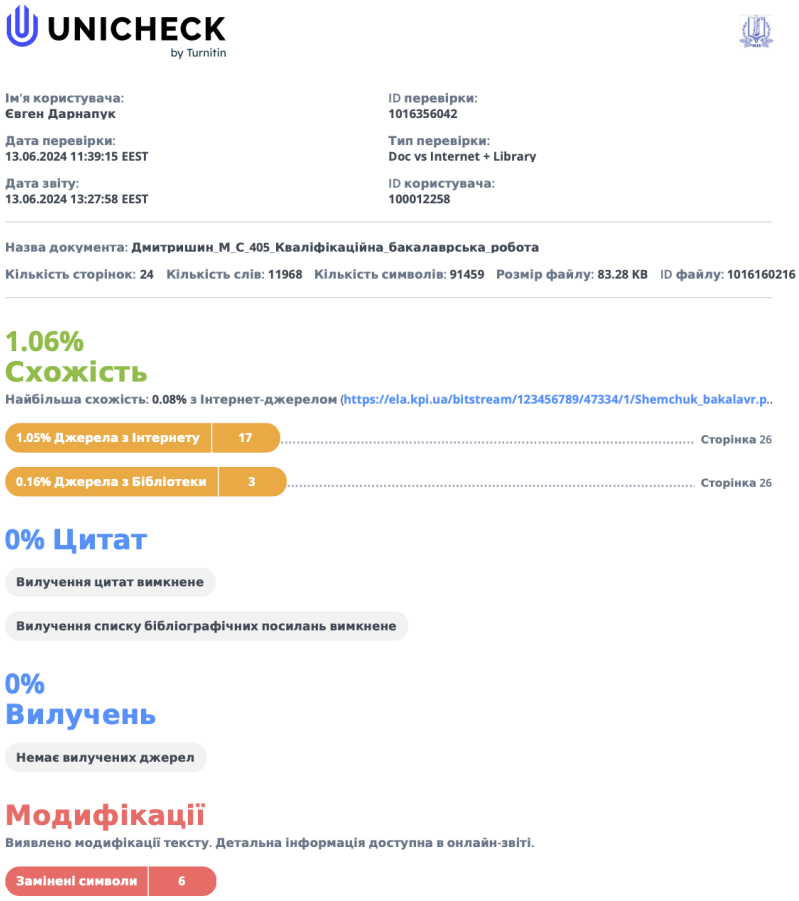
студента спеціальності 123 «Комп'ютерна інженерія», 405 групи

Дмитришин Максим Сергійович

(прізвище, ім'я, по-батькові)

Перевірку тексту здійснено сервісом: онлайн-сервіс Unicheck

Результат перевірки тексту бакалаврської кваліфікаційної роботи: схожість складає 2,29 %.


UNICHECK
by Turnitin

Ім'я користувача: Євген Дарнапук
ID перевірки: 1016356042
Дата перевірки: 13.06.2024 11:39:15 EEST
Тип перевірки: Doc vs Internet + Library
Дата звіту: 13.06.2024 13:27:58 EEST
ID користувача: 100012258

Назва документа: Дмитришин_М_С_405_Кваліфікаційна_бакалаврська_робота
Кількість сторінок: 24 Кількість слів: 11968 Кількість символів: 91459 Розмір файлу: 83.28 KB ID файлу: 1016160216

1.06%
Схожість
Найбільша схожість: 0.08% з Інтернет-джерелом (https://ela.kpi.ua/bitstream/123456789/47334/1/Shemchuk_bakalavr.p..

1.05% Джерела з Інтернету 17 Сторінка 26
0.16% Джерела з Бібліотеки 3 Сторінка 26

0% Цитат
Вилучення цитат вимкнене
Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень
Немає вилучених джерел

Модифікації
Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.
Замінені символи 6

Здобувач:

_____М. С. Дмитришин_____
підпис ініціали, прізвище

Керівник:
ст. викладач
_____Є. С. Дарнапук
підпис ініціали, прізвище

Дата: «___» _____ 2024 р.

ДОДАТОК Б

Лістинг коду прошивки

```
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include <PubSubClient.h>
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BMP280.h>
#include "DHT.h"

// Налаштування Wi-Fi
const char* ssid = " SSID";
const char* password = " PASSWORD";

// Налаштування AWS IoT
const char* awsEndpoint = " AWS_IOT_ENDPOINT";
const char* rootCA = "-----BEGIN CERTIFICATE-----
\nROOT_CA_CERTIFICATE\n-----END CERTIFICATE-----";
const char* clientCert = "-----BEGIN CERTIFICATE-----
\nCLIENT_CERTIFICATE\n-----END CERTIFICATE-----";
const char* privateKey = "-----BEGIN RSA PRIVATE KEY-----
\nPRIVATE_KEY\n-----END RSA PRIVATE KEY-----";

// Ідентифікатор пристрою
const char* deviceID = "DEVICE_ID";
// Піни для датчиків
#define DHTPIN 15
#define DHTTYPE DHT22
#define SOIL_MOISTURE_PIN 35
#define LIGHT_SENSOR_PIN 34
#define BMP_SDA 21
#define BMP_SCL 22
// Створення об'єктів для датчиків
```

```
DHT dht(DHTPIN, DHTTYPE);
Adafruit_BMP280 bmp; // I2C
WiFiClientSecure wifiClient;
PubSubClient client(awsEndpoint, 8883, wifiClient);

void connectToWiFi() {
  Serial.print("Connecting to WiFi...");
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }
  Serial.println(" connected");
}

void connectToAWS() {
  wifiClient.setCACert(rootCA);
  wifiClient.setCertificate(clientCert);
  wifiClient.setPrivateKey(privateKey);

  Serial.print("Connecting to AWS IoT...");
  while (!client.connected()) {
    if (client.connect(deviceID)) {
      Serial.println(" connected");
    } else {
      Serial.print(" failed, rc=");
      Serial.print(client.state());
      delay(2000);
    }
  }
}

void setup() {
  Serial.begin(115200);
```

```
    connectToWiFi();

    Wire.begin(BMP_SDA, BMP_SCL);
    if (!bmp.begin(0x76)) {
      Serial.println("Could not find a valid BMP280 sensor, check wiring!");
      while (1);
    }

    dht.begin();
    connectToAWS();
  }

  void loop() {
    if (!client.connected()) {
      connectToAWS();
    }
    client.loop();

    // Зчитування даних з DHT22
    float humidity = dht.readHumidity();
    float temperatureDHT = dht.readTemperature();

    // Зчитування даних з Soil Moisture Sensor
    int soilMoistureValue = analogRead(SOIL_MOISTURE_PIN);
    // Зчитування даних з Light Sensor
    int lightLevel = analogRead(LIGHT_SENSOR_PIN);
    // Зчитування даних з BMP280
    float temperatureBMP = bmp.readTemperature();
    float pressure = bmp.readPressure();
    float altitude = bmp.readAltitude(1013.25);
    // Формування JSON-об'єкта для відправки
    String payload = "{";
    payload += "\"humidity\": " + String(humidity) + ",";
```

```
    payload += "\"temperatureDHT\": " + String(temperatureDHT) + ",";
    payload += "\"soilMoisture\": " + String(soilMoistureValue) + ",";
    payload += "\"lightLevel\": " + String(lightLevel) + ",";
    payload += "\"temperatureBMP\": " + String(temperatureBMP) + ",";
    payload += "\"pressure\": " + String(pressure) + ",";
    payload += "\"altitude\": " + String(altitude);
    payload += "}";

    // Відправка даних на AWS IoT
    if (client.publish("your/topic", payload.c_str())) {
      Serial.println("Publish succeeded");
    } else {
      Serial.println("Publish failed");
    }

    delay(2000); // Затримка між вимірюваннями
  }
```