Міністерство освіти і науки України
Національний університет “Львівська політехніка”
Інститут комп’ютерних наук та інформаційних технологій

Кафедра САПР

Лабораторна робота №4

з дисципліни: “Технології та стандарти інтернету речей”
на тему:
“Розроблення архітектури системи інтернету речей”

Виконав:
Ст. групи ПП-44
Верещак Б. О.
Прийняв:
асис. Гавран В. Б.

Львів - 2025

Мета роботи

Ознайомитись з основними принципами та етапами створення IoT-систем, навчитися проектувати та реалізовувати архітектуру для збору, обробки та передачі даних між пристроями, а також інтеграцією цих даних з хмарними або серверними рішеннями.

Теоретичні відомості

Інтернет речей (Internet of Things, IoT) - концепція обчислювальної мережі фізичних предметів («речей»), оснащених вбудованими технологіями для взаємодії один з одним або з зовнішнім середовищем, яка розглядає організацію таких мереж як явище, здатне перебудувати економічні та суспільні процеси, що виключає з частини дій і операцій необхідність участі людини.

У найзагальнішому вигляді з інфокомунікаційної точки зору Інтернет речей можна записати у вигляді наступної символічної формули:

IoT = Сенсори (датчики) + Дані + Мережі + Послуги.

У загальному випадку під Інтернетом речей розуміється сукупність різноманітних приладів, сенсорів, пристроїв, об'єднаних у мережу за допомогою будь-яких доступних каналів зв'язку, що використовують різні протоколи взаємодії між собою та єдиний протокол доступу до глобальної мережі.

Зараз в якості глобальної мережі для Інтернету речей використовується Інтернет, де основним протоколом є IP. Іншими словами, Інтернет речей – це світова мережа комп'ютерів, сенсорів і виконавчих пристроїв, що взаємодіють між собою за допомогою протоколу IP (Internet Protocol). Завдяки Інтернету речей можуть бути впроваджені різноманітні «розумні» (smart) рішення в різних сферах діяльності та повсякденного життя людини.

Лабораторне завдання

- Ознайомитися з теоретичними відомостями.
- Розробити архітектуру системи інтернету речей згідно з варіантом
  - Розумні парковки: Система моніторингу та резервування місць;
  - Інтелектуальна система керування шторами;
  - Інтелектуальна система керування електроспоживанням будинку;
  - Інтелектуальна система керування та оптимізації домашніх генераторів;
  - Застосунок для керування LED-стрічкою та освітленням;
  - Розумний інкубатор;
  - Система моніторингу периметру і виявлення загрози;
  - Інтелектуальна система догляду за садом;
  - Розумний термостат для опалення;
  - Розумні двері для домашніх тварин.
- Оформити звіт до лабораторної роботи.

Результати виконання завдання:

На цій лабораторній роботі, потрібно розробити архітектуру системи інтернету речей згідно з моїм варіантом, для якого тема є «Розробка інформаційної системи моніторингу погодних умов у реальному часі». Архітектура системи складається з пристроїв (сенсорів, датчиків), мережі для передачі даних, хмарних сховищ, аналітичних платформ та інтерфейсу користувача. Давайте розглянемо кожен з пунктів для цієї системи.

Пристрої

- **ESP32**
  - Швидкий, перевірений та надійний пристрій;
  - Має вбудований Wifi і Bluetooth модуль;
  - Мав попередній досвід користування;
  - Має власну пам’ять, до якої розробник може зчитувати та записувати дані;
  - Має можливість ввімкнути deep sleep режим, який змушує пристрій заснути, що тратить в рази менше електроенергії.

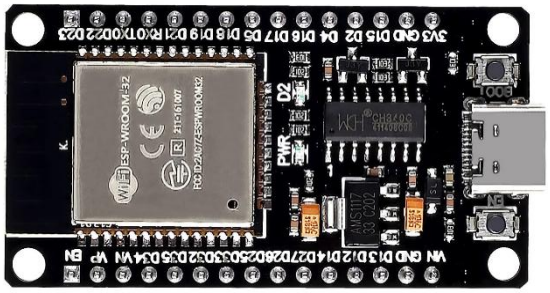

Рис. 1. ESP32 DevKit v1 30 Pins

- **BME280**
  - Складається з датчика BMP280, який відповідає за показ температури та атмосферного тиску;
  - Вміщає датчик DH11 для вимірювання вологості повітря;
  - Дешева ціна та попередній досвід використання.

Рис. 2. Датчик BME280

- **ENS160**
  - Надає інформацію про вуглекислий газ CO2;

- Загальну кількість органічних сполук в повітрі TVOC;
- Загальну оцінку якості повітря AQI;
- Відносно невелика ціна;
- Працює як окремий пристрій, який сам нагрівається та обчислює змінні якості повітря.

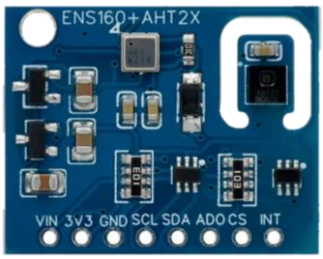

Рис. 3. Датчик ENS160

- **HALL-сенсор**
  - Потрібен для автоматизації дощового вимірювання.

Рис. 4. Сенсор Холла (HALL Sensor)

Підключення

Вся метеостанція, разом з перерахованими пристроями підключенні на breadboard, також планується виготовити спеціальне PCB плату для проекту. Схему підключення можна бачити на Рис. 5.

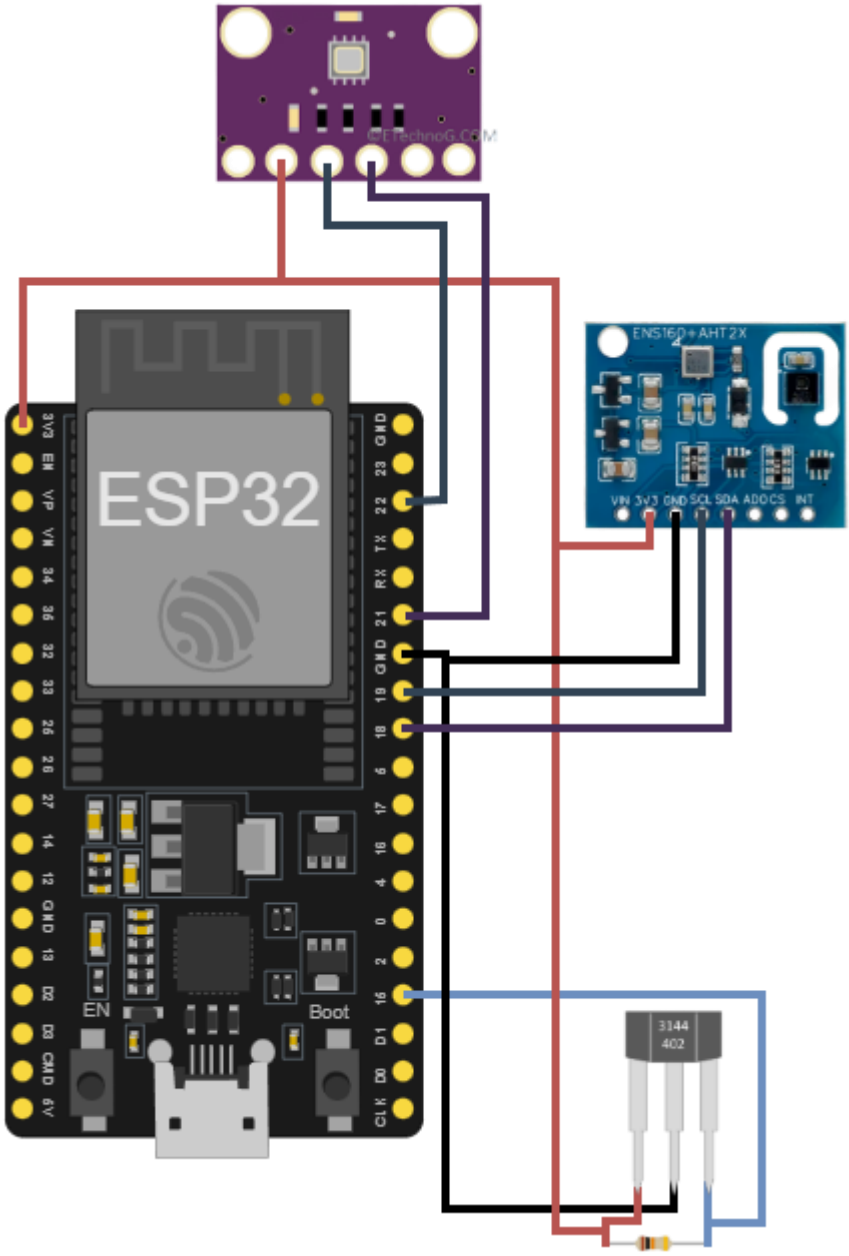

Рис. 5. Схема підключення метеостанції

Налаштування

1. Мережеве обладнання

Wi-Fi-роутер або точка доступу для забезпечення зв’язку між метеостанцією та сервером. У разі використання LoRa-модуля необхідний LoRa-шлюз для прийому даних і їх передачі до сервера через Інтернет. Для забезпечення стабільності зв’язку в сільській місцевості можуть використовуватися підсилювачі сигналу або 4G-модеми.

2. Відправка даних:

Коли всі датчики зчитали інформацію (температура, тиск, вологість, якість повітря, кількість дощу), система формує JSON-пакет – структурований текстовий формат, який легко обробляти на сервері.

Потім цей JSON відправляється HTTP POST-запитом на сервер (наприклад, http://192.168.0.102:3000/api/data). Якщо Wi-Fi підключений, дані йдуть відразу. Якщо ні – можна було б зберігати їх у пам’яті і відправити пізніше (але в нашому випадку ESP32 просто чекає поки з’явиться мережа).

3. Отримання конфігурації з сервера

Метеостанція не лише відправляє дані, але й може отримувати налаштування:

- Інтервал оновлення (update_interval_ms) – як часто відправляти дані (наприклад, кожні 30 секунд або 5 хвилин).
- Параметри Wi-Fi – якщо змінився пароль або назва мережі, станція може оновитися без перепрошивки.

Це відбувається через HTTP GET-запит на /api/config. Сервер повертає JSON з новими налаштуваннями, і ESP32 їх застосовує.

4. Синхронізація часу

Щоб час у даних був точним, метеостанція періодично (кожні 2 години) запитує поточний час у сервера через /time. Отримує Unix-час (кількість секунд з 1970 року) і встановлює його в системі.

Коли сервер не відповідає, ESP:

- Використовує внутрішній таймер (на основі esp_timer_get_time()).
- Пробує знову через 5 секунд (але не більше 3 спроб).

Хмарне сховище

Через нерівнозначність фінального вигляду даних, було вирішено використовувати нераляційну базу даних. Серед багатьох варіантів ми зійшлися на MongoDB, так як ми мали попередній досвід користування та там був безкоштовний план.

В MongoDB використовують систему колекцій, тобто кожна колекція це є окремий об’єкт, який переважно має свою json структуру. До колекцій належить:

Колекція	Опис
sensorData	Тут зберігається вся інформація яка надходить до нас від embedded системи, тобто від esp32. Серед них дані про навколишнє середовище, якість повітря, дощ і час запису.
config	Ця колекція слугує для налаштування різноманітних об’єктів системи, таких як інтервал оновлення та пороги значень температури і вологості.
time	Використовується для оновлення значення часу, яку esp32 запитує у сервера та повторює запит кожні 2 години.

Аналітична платформа

Вся аналітика відбувається безпосередньо на серверній частині системи. Після отримання даних від метеостанції сервер проводить попередню обробку та аналіз у режимі реального часу.

Основні етапи роботи аналітичної платформи:

1. **Обробка даних:**
   Сервер перевіряє отримані значення на коректність (наприклад, виключає аномальні показники, що виходять за фізично можливі межі). Також здійснюється нормалізація даних — перетворення у стандартний формат для подальшого аналізу.

2. **Збереження та агрегація:**
   Дані зберігаються у базі MongoDB, де вони групуються за часом і типом сенсорів (температура, тиск, вологість, якість повітря). Для кожного параметра формуються статистичні вибірки (середнє, мінімальне, максимальне значення за день, тиждень, місяць).

3. **Аналітичні алгоритми:**
   На основі накопичених даних система:
   - Виявляє тренди зміни погодних умов;
   - Аналізує динаміку температури й вологості протягом доби;
   - Може прогнозувати погоду на короткий період (за допомогою лінійної регресії чи ковзного середнього);
   - Визначає аномальні ситуації — наприклад, різке падіння тиску чи перевищення рівня CO_2.

4. **Система сповіщень:**
   Якщо значення перевищують задані користувачем пороги (наприклад, температура > 30 °C або AQI > 150), сервер автоматично генерує повідомлення та надсилає його користувачу через email або push-сповіщення.

5. **API доступ:**
   Результати аналізу доступні через REST API, який надає запити типу:
   - /api/stats/daily — середні значення за день;
   - /api/stats/trends — побудова трендів зміни погодних параметрів;
   - /api/alerts — історія сповіщень та порушень порогів.

Таким чином, аналітична платформа забезпечує автоматичну обробку, виявлення аномалій і підготовку зручних для користувача звітів, що допомагає швидко приймати рішення на основі поточних погодних умов.

Інтерфейс користувача

Клієнтський веб-інтерфейс створений для зручної взаємодії користувача з метеостанцією та візуалізації даних у реальному часі.

Основні функції:

- Візуалізація даних у вигляді графіків (температура, вологість, тиск, опади) за вибраний період (день, тиждень, місяць).
- Налаштування порогових значень для сповіщень (наприклад, температура <10°C або >30°C).
- Відображення історії вимірювань у табличному вигляді.
- Адаптивний дизайн, що підтримує використання на смартфонах, планшетах і комп’ютерах.
- Для побудови графіків використовується бібліотека Chart.js, яка забезпечує інтерактивні та візуально привабливі діаграми.

Висновки

На даній лабораторній роботі я ознайомився з основними принципами та етапами створення IoT-систем, навчився проектувати та реалізовувати архітектуру для збору, обробки та передачі даних між пристроями, а також інтеграцією цих даних з хмарними або серверними рішеннями. У процесі роботи було спроектовано повноцінну структуру системи, яка включає сенсорний рівень (ESP32 з датчиками BME280, ENS160 та HALL), мережеву частину для передачі даних через Wi-Fi, серверний рівень на базі Flask та MongoDB, а також клієнтську частину з інтерактивним веб-інтерфейсом.