ЛАБОРАТОРНА РОБОТА №4

.РОЗРОБКА IoT КЛІЄНТА (БІЗНЕС-ЛОГІКИ ТА ФУНКЦІЙ НАЛАШТУВАННЯ)

Система моніторингу опалення у домогосподарствах

Версія 1.0, затверджена

Підготовлено Макогоном Б.О.

ПЗПІ-22-6

30.12.2024

Інтеграція цієї системи з Arduino дозволить використовувати апаратні сенсори для збору реальних даних, які передаватимуться в REST API системи для подальшої обробки. Ось покроковий опис, як це можна реалізувати:

1. Апаратна частина: Arduino

Для збору даних будуть використовуватися фізичні сенсори, підключені до мікроконтролера Arduino. Наприклад:

- Сенсори температури: DS18B20 або DHT11/DHT22.

- Сенсори вологості: DHT11/DHT22.

- Інші сенсори: наприклад, барометри BMP180.

Arduino зчитує дані із сенсорів за допомогою відповідних бібліотек:

- Для DHT11/DHT22 використовується бібліотека `DHT.h`.

- Для DS18B20 — бібліотека `OneWire` і `DallasTemperature`.

2. Передача даних через Arduino

Для передачі даних із Arduino в систему REST API можна використовувати Ethernet Shield або Wi-Fi модулі (наприклад, ESP8266 чи ESP32). Вибраний модуль дозволить відправляти HTTP-запити до серверного API.

Приклад програмного забезпечення для Arduino:

Код для збору та передачі даних:

```cpp
#include <ESP8266WiFi.h>

#include <DHT.h>

#include <ESP8266HTTPClient.h>

#define DHTPIN 2 // Пін для підключення сенсора

#define DHTTYPE DHT22 // Тип сенсора DHT11 або DHT22

const char* ssid = "YourWiFiSSID";

const char* password = "YourWiFiPassword";
```

```
const char* serverUrl = "http://your-server-address/api/sensors/{sensorId}/data";
DHT dht(DHTPIN, DHTTYPE);
void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  // Очікуємо з'єднання з Wi-Fi
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
  Serial.println("Connected to WiFi!");
  dht.begin();
}
void loop() {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    http.begin(serverUrl);
    http.addHeader("Content-Type", "application/json");
    // Зчитуємо дані з сенсора
    float temperature = dht.readTemperature();
    float humidity = dht.readHumidity();
    if (isnan(temperature) || isnan(humidity)) {
      Serial.println("Failed to read from DHT sensor!");
      return;
    }
    // Формуємо JSON
    String jsonData = "{\"temperature\": " + String(temperature) +
```

```
                    ", \"humidity\": " + String(humidity) + "}";

    //Відправляємо POST-запит
   int httpResponseCode = http.POST(jsonData);
   if (httpResponseCode > 0) {
     Serial.println("Data sent successfully: " + String(httpResponseCode));
   } else {
     Serial.println("Error sending data: " + String(httpResponseCode));
   }
   http.end();
  }
  delay(5000); // Затримка 5 секунд між запитами
}
```

3. REST API на стороні системи

Система вже має реалізований ендпоїнт для отримання даних від сенсора:

```java
@PostMapping("/{sensorId}/data")
public ResponseEntity<SensorData> addSensorData(
        @PathVariable Long sensorId,
        @RequestBody SensorDataDTO sensorDataDTO) {
   SensorData sensorData = sensorService.addSensorData(sensorId, sensorDataDTO);
   return ResponseEntity.ok(sensorData);
}
```

При інтеграції з Arduino API приймає POST-запити з даними про температуру та вологість, додаючи їх у базу даних.

4. Налаштування сервера для роботи з Arduino

Сервер має бути доступним для підключення з Arduino. Це може бути локальний сервер або хмарний сервіс із відкритим доступом через публічний IP чи доменне ім’я. Для захисту даних рекомендується:

1. Використовувати HTTPS.

2. Додати базову авторизацію або API-ключі для кожного пристрою.

Приклад заголовка для авторизації:

```cpp
http.addHeader("Authorization", "Bearer your-api-token");
```

5. Розширення функціональності

Інтеграція з Arduino може бути розширена наступними способами:

- Контроль сенсорів:

  REST API може надсилати конфігурацію на Arduino, наприклад, змінюючи інтервали вимірювань чи граничні значення.

  ```java
  @GetMapping("/{sensorId}/settings")
  public ResponseEntity<SensorSettings> getSensorSettings(@PathVariable Long sensorId) {
      SensorSettings settings = sensorService.getSettings(sensorId);
      return ResponseEntity.ok(settings);
  }
  ```

- Нотифікації:

Система може перевіряти отримані дані на перевищення порогових значень і надсилати повідомлення через email, SMS чи інші сервіси.

- Інтеграція з аналітикою:

Дані від Arduino зберігаються в базі даних і можуть використовуватися для побудови графіків або статистичних звітів через API аналітики.

6. Тестування

Для перевірки коректності роботи інтеграції:

1. Використовуйте емулятори Arduino, наприклад, PlatformIO або тестові програми.

2. Перевірте, чи дані правильно передаються в REST API.

3. Переконайтесь, що сервер відповідає на запити Arduino з коректними кодами HTTP.

7. Переваги інтеграції

- Реальний збір даних: Інтеграція з Arduino дозволяє отримувати актуальні дані в реальному часі.

- Гнучкість системи: Завдяки REST API, до системи можна підключати будь-які пристрої, що підтримують HTTP-запити.

- Масштабованість: Arduino може використовуватися як основа IoT-мережі для збору даних з десятків і навіть сотень сенсорів.

Ця інтеграція відкриває можливості для побудови масштабованої системи моніторингу й управління на базі сучасних апаратних і програмних рішень.

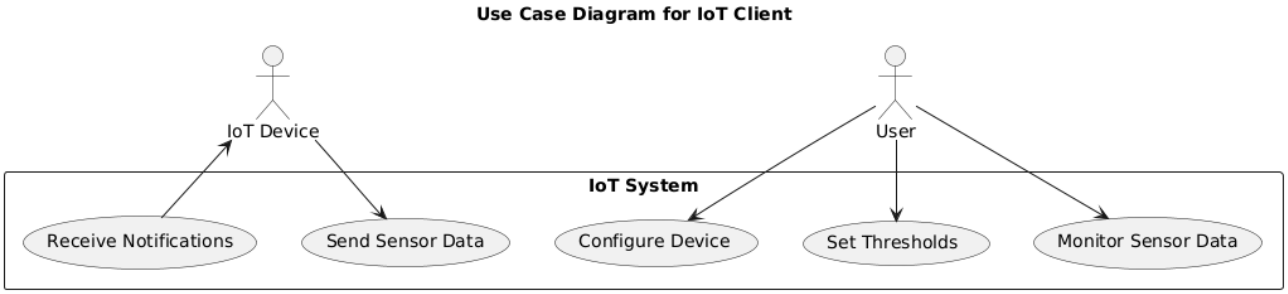

Рисунок 1. UML діаграма прецедентів IoT клієнта

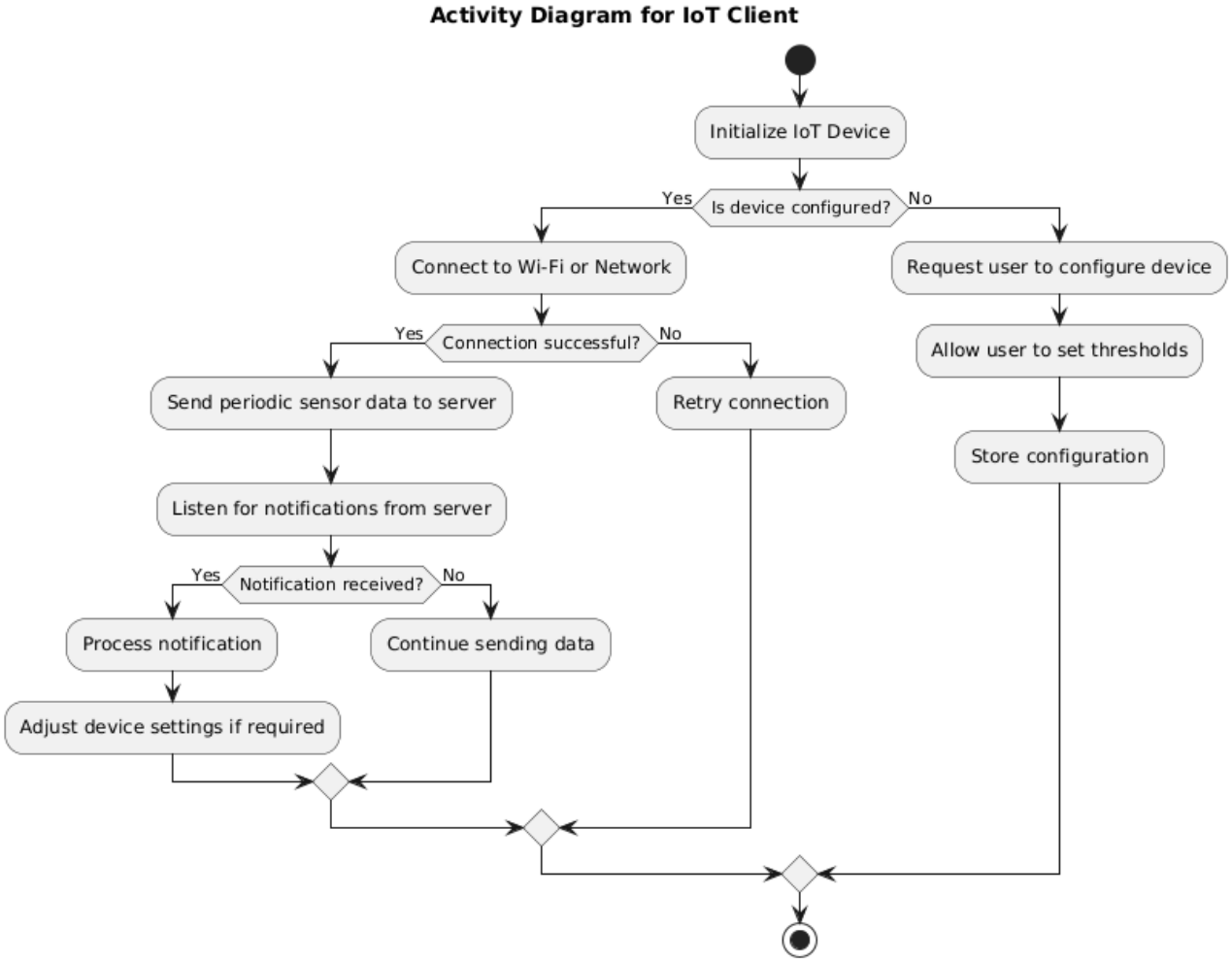

Рисунок 2 . UML діаграма діяльності IoT клієнта.