Міністерство освіти та науки України Харківський національний університет радіоелектроніки

Кафедра ПІ

Звіт

З лабораторної роботи 4

Тема роботи: «РОЗРОБКА IoT КЛІЄНТА

(БІЗНЕС-ЛОГІКИ ТА ФУНКЦІЙ НАЛАШТУВАННЯ)»

з дисципліни «Аналіз та рефакторинг коду»

Виконав: Перевірив:

ст. гр. ПЗПІ-22-10 ст. викладач Сокорчук І.П.

Клецов М.Д.

Харків 2024

**Мета роботи:** На лабораторній роботі №4 потрібно розробити програмне забезпечення для IoT або SmartDevice пристрою, створеного на базі будь-якої поширеної на сьогодні платформи, придатної для реалізації вбудованих систем (Embedded System).

**Хід роботи:**

Проект полягав у розробці програмного забезпечення для ІоТ клієнта, який збирає дані з датчиків і передає їх через MQTT на сервер. Спочатку була створена архітектура та UML діаграми для візуалізації процесів і взаємодії з системою. Потім розроблена бізнес-логіка для обробки даних, програмна реалізація функцій налаштування та обробки отриманих даних. Завершальним етапом було тестування ІоТ клієнта для перевірки його коректної роботи.

1. **Текстовий опис інженерних рішень**

Інженерні рішення в проекті полягали в розробці системи, яка здатна збирати дані і передавати їх на сервер через MQTT. Для цього була обрана архітектура з використанням ESP32, що забезпечує бездротову передачу даних. Дані з датчиків обробляються на мікроконтролері за допомогою алгоритмів, що забезпечують точність вимірювань і адаптацію під зміни умов навколишнього середовища.

Для забезпечення надійності передачі даних використовувався MQTT-брокер, що гарантує швидку та ефективну доставку інформації на сервер для подальшого зберігання та обробки.

Изображение выглядит как электроника, Электронная техника, Компонент схемы, Электронный компонент

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1 – Схема IoT пристрою

1. **Фрагменти програмного коду:**

* Бізнес логіки IoT клієнта (математичної обробки повʼязаних із предметною областю даних);

1. #include <WiFi.h>
2. #include <PubSubClient.h>
3. // Налаштування Wi-Fi
4. const char\* ssid = "Wokwi-GUEST";
5. const char\* password = "";
6. // Налаштування MQTT
7. const char\* mqttServer = "broker.hivemq.com"; // або IP-адреса вашого MQTT-брокера
8. const int mqttPort = 1883;
9. // Створюємо об'єкт WiFi та MQTT-клієнта
10. WiFiClient espClient;
11. PubSubClient client(espClient);
12. // Функція для підключення до Wi-Fi
13. void setupWiFi() {
14. delay(10);
15. Serial.println();
16. Serial.print("Підключення до Wi-Fi: ");
17. Serial.println(ssid);
19. WiFi.begin(ssid, password);
21. while (WiFi.status() != WL\_CONNECTED) {
22. delay(500);
23. Serial.print(".");
24. }
26. Serial.println("");
27. Serial.print("Підключено! IP адреса: ");
28. Serial.println(WiFi.localIP());
29. }
30. // Функція для підключення до MQTT-брокера
31. void connectMQTT() {
32. while (!client.connected()) {
33. Serial.print("Підключення до MQTT...");
34. // Створюємо унікальний клієнтський ID
35. String clientId = "ESP32Client-";
36. clientId += String(random(0xffff), HEX);
38. if (client.connect(clientId.c\_str())) {
39. Serial.println("підключено!");
40. } else {
41. Serial.print("Помилка підключення, код: ");
42. Serial.print(client.state());
43. Serial.println(" Повторна спроба через 5 секунд");
44. delay(5000);
45. }
46. }
47. }
48. void setup() {
49. Serial.begin(115200);
50. setupWiFi();
51. client.setServer(mqttServer, mqttPort);
52. }
53. void loop() {
54. if (!client.connected()) {
55. connectMQTT();
56. }
57. client.loop();
59. // Симулюємо зчитування даних (наприклад, кількість авто на різних ділянках)
60. int count1 = random(0, 2); // статус датчика (активний\неактивний)
61. int count2 = random(1, 4); // номер датчика (на в'їзді, всередені, на виїзді)
62. int count3 = random(0, 20); // кількість авто наприклад, від 0 до 20 авто
63. if (count1==0) {count3=0;}
64. String payload = "{\"Status\": " + String(count1) + ", \"Type\": " + String(count2) + ", \"Count\": " + String(count3) +"}";

67. // Публікуємо дані в топік "iot/count"
68. if (client.publish("iot/count", payload.c\_str())) {
69. Serial.print("Дані відправлено: ");
70. Serial.println(payload);
71. } else {
72. Serial.println("Не вдалося відправити дані");
73. }
75. delay(5000); // надсилання даних кожні 5 секунд
76. }

**Графічні ілюстрації (рисунки):**

* UML діаграма прецедентів ІоТ клієнта.

**Изображение выглядит как текст, линия, диаграмма, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.**

Рисунок 2 – UML діаграма прецедентів ІоТ клієнта

* UML діаграма діяльності ІоТ клієнта.

Изображение выглядит как текст, диаграмма, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3 – UML діаграма діяльності ІоТ клієнта

**Висновки:** У результаті виконаної роботи було розроблено ІоТ клієнта для збору та передачі даних з трьох датчиків. Створена архітектура системи, яка забезпечує збір даних та їх подальшу передачу на сервер через MQTT. Було реалізовано бізнес-логіку для обробки отриманих даних та налаштувань пристроїв. Програмна реалізація продемонструвала стабільну роботу з датчиками та надійну передачу даних, що підтвердило ефективність розробленого рішення для використання в ІоТ системах з датчиками дистанції.

**Посилання на відео:** https://youtu.be/0I1TwvRVFAk