A. Cara Kerja Alat

1. Pembacaan Suhu:
   - ESP32 membaca suhu air menggunakan sensor suhu DS18B20 yang terhubung ke salah satu pin digitalnya.
   - Sensor DS18B20 menggunakan protokol OneWire untuk berkomunikasi dengan mikrokontroler. ESP32 membaca data suhu dari sensor menggunakan protokol ini.
2. Tampilan Suhu pada LCD:
   - Setelah ESP32 mendapatkan data suhu dari sensor, nilai suhu tersebut ditampilkan pada LCD yang terhubung melalui interface I2C.
   - Program Arduino di ESP32 mengirimkan instruksi untuk menampilkan nilai suhu pada LCD.
3. Koneksi Internet:
   - ESP32 terhubung ke internet melalui koneksi Wi-Fi yang telah dikonfigurasi sebelumnya.
   - Koneksi internet ini diperlukan untuk mengirimkan pesan suhu ke bot Telegram.
4. Pengiriman Pesan Suhu ke Bot Telegram:
   - Setelah mendapatkan nilai suhu, ESP32 menggunakan koneksi internetnya untuk mengirimkan pesan suhu ke bot Telegram.
   - ESP32 mengirimkan permintaan HTTP POST ke API Telegram menggunakan token bot yang telah didaftarkan sebelumnya.
   - Pesan yang dikirim berisi informasi suhu air dalam akuarium.
5. Integrasi dengan Bot Telegram:
   - Bot Telegram menerima pesan dari ESP32 yang berisi nilai suhu.
   - Pengguna dapat menerima notifikasi suhu dari bot Telegram secara real-time.
   - Pengguna juga dapat mengirimkan perintah ke bot Telegram untuk meminta informasi suhu terkini atau melakukan tindakan lainnya pada alat.
6. Monitoring dan Interaksi:
   - Pengguna dapat memantau suhu akuarium dari jarak jauh melalui bot Telegram.
   - Bot Telegram memberikan kemampuan interaktif, seperti meminta informasi suhu terkini atau mengatur parameter suhu tertentu.

B. Pejelasan Code

```
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
```

1. #include <WiFi.h>: Library ini digunakan untuk mengakses dan mengontrol koneksi WiFi pada perangkat Arduino. Dengan menggunakan library ini, Anda dapat membuat perangkat Arduino terhubung ke jaringan WiFi.
2. #include <WiFiClientSecure.h>: Library ini mirip dengan WiFi.h, namun ini untuk koneksi WiFi yang aman dan menggunakan protokol HTTPS. Dengan menggunakan library ini, Anda dapat membuat koneksi aman ke server melalui jaringan WiFi.
3. #include <UniversalTelegramBot.h>: Library ini memungkinkan perangkat Arduino untuk berkomunikasi dengan platform Telegram menggunakan API bot. Dengan menggunakan

library ini, perangkat Arduino dapat menerima dan mengirim pesan ke pengguna melalui bot Telegram.

4. #include <OneWire.h>: Library ini digunakan untuk mengontrol perangkat yang terhubung menggunakan protokol OneWire. Protokol ini sering digunakan untuk menghubungkan sensor suhu atau perangkat lainnya ke perangkat Arduino dengan menggunakan satu jalur data.
5. #include <DallasTemperature.h>: Library ini bekerja bersama dengan OneWire.h untuk membaca data dari sensor suhu yang kompatibel dengan protokol OneWire, terutama sensor suhu dari Dallas Semiconductor. Dengan menggunakan library ini, perangkat Arduino dapat membaca data suhu dari sensor tersebut.
6. #include <Wire.h>: Library ini digunakan untuk mengakses dan mengontrol perangkat yang terhubung menggunakan protokol I2C (Inter-Integrated Circuit). Protokol ini digunakan untuk menghubungkan berbagai perangkat seperti sensor, display, dan lainnya dengan menggunakan hanya dua kabel.
7. #include <LiquidCrystal_I2C.h>: Library ini digunakan untuk mengontrol dan menampilkan teks pada LCD (Liquid Crystal Display) yang terhubung dengan perangkat Arduino menggunakan protokol I2C. Dengan menggunakan library ini, Anda dapat dengan mudah menampilkan teks atau karakter pada layar LCD.

```
const char* ssid = "hehe";
const char* password = "terserah hehe";
#define ONE_WIRE_BUS 2
#define BOTtoken "6618209994:AAHLfbeoVnwtfpg3y6WUShjk3IRly40kOWM"
#define CHAT_ID "5174254230"
```

1. const char* ssid = "hehe";: Variabel ini menyimpan nama (SSID) dari jaringan WiFi yang akan digunakan oleh perangkat. Di sini, nilai "hehe" adalah contoh dari nama jaringan WiFi yang sebenarnya. Anda perlu menggantinya dengan nama jaringan WiFi yang sesuai.
2. const char* password = "terserah hehe";: Variabel ini menyimpan kata sandi untuk jaringan WiFi yang telah didefinisikan sebelumnya. Nilai "terserah hehe" adalah contoh dari kata sandi yang sebenarnya. Anda harus menggantinya dengan kata sandi yang sesuai untuk jaringan WiFi tersebut.
3. #define ONE_WIRE_BUS 2: Mendefinisikan pin Arduino yang digunakan untuk komunikasi dengan perangkat yang terhubung menggunakan protokol OneWire. Di sini, pin 2 digunakan sebagai contoh. Anda dapat menggantinya dengan pin yang sesuai dengan konfigurasi perangkat Anda.
4. #define BOTtoken "6618209994:AAHLfbeoVnwtfpg3y6WUShjk3IRly40kOWM": Mendefinisikan token bot Telegram yang akan digunakan untuk mengakses API Telegram. Token ini diperlukan agar bot dapat berkomunikasi dengan Telegram. Nilai "6618209994:AAHLfbeoVnwtfpg3y6WUShjk3IRly40kOWM" adalah contoh token bot. Anda harus menggantinya dengan token bot yang sesuai yang diperoleh saat membuat bot baru di BotFather.
5. #define CHAT_ID "5174254230": Mendefinisikan ID obrolan (chat) di Telegram. ID ini menunjukkan ke obrolan atau pengguna mana pesan dari bot akan dikirim. Nilai "5174254230" adalah contoh ID obrolan. Anda harus menggantinya dengan ID obrolan yang sesuai yang ingin Anda tuju.

```
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

WiFiClientSecure client;
UniversalTelegramBot bot(BOTtoken, client);
```

1. OneWire oneWire(ONE_WIRE_BUS);: Membuat objek oneWire yang merupakan instance dari class OneWire. Objek ini digunakan untuk mengontrol perangkat yang terhubung menggunakan protokol OneWire. Saat membuat objek ini, parameter ONE_WIRE_BUS digunakan untuk menentukan pin Arduino yang digunakan untuk komunikasi dengan perangkat yang terhubung menggunakan protokol OneWire.
2. DallasTemperature sensors(&oneWire);: Membuat objek sensors yang merupakan instance dari class DallasTemperature. Objek ini digunakan untuk membaca data suhu dari sensor suhu yang terhubung menggunakan protokol OneWire. Saat membuat objek ini, parameter &oneWire digunakan untuk menentukan objek oneWire yang akan digunakan untuk komunikasi dengan sensor suhu.
3. WiFiClientSecure client;: Membuat objek client yang merupakan instance dari class WiFiClientSecure. Objek ini digunakan untuk membuat koneksi aman (HTTPS) ke server menggunakan koneksi WiFi. Objek ini akan digunakan untuk komunikasi dengan server Telegram karena Telegram API biasanya menggunakan protokol HTTPS untuk koneksi yang aman.
4. UniversalTelegramBot bot(BOTtoken, client);: Membuat objek bot yang merupakan instance dari class UniversalTelegramBot. Objek ini digunakan untuk berinteraksi dengan bot Telegram menggunakan Telegram API. Saat membuat objek ini, parameter BOTtoken digunakan untuk memberikan token bot yang akan digunakan untuk mengotentikasi ke server Telegram, dan parameter client digunakan untuk menentukan objek client yang akan digunakan untuk koneksi ke server Telegram.

```
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

1. Ubtuk Instalasi LCD

```
void setup() {
  Serial.begin(115200);
  pinMode (2, INPUT_PULLUP);
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  client.setCACert(TELEGRAM_CERTIFICATE_ROOT);

  lcd.init();
  lcd.backlight();

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  sensors.begin();
}
```

1. Serial.begin(115200);: Menginisialisasi komunikasi serial dengan kecepatan 115200 baud. Ini memungkinkan perangkat Arduino untuk berkomunikasi dengan komputer melalui kabel USB dan mencetak pesan debug atau informasi ke terminal serial.
2. pinMode(2, INPUT_PULLUP);: Mengatur pin 2 sebagai input dengan resistor pull-up internal diaktifkan. Ini sering digunakan untuk membaca sinyal dari perangkat yang terhubung ke pin tersebut.
3. WiFi.mode(WIFI_STA);: Mengatur mode WiFi perangkat Arduino sebagai mode Station (STA), yang berarti perangkat akan mencoba untuk terhubung ke jaringan WiFi sebagai klien.
4. WiFi.begin(ssid, password);: Memulai proses koneksi WiFi dengan menggunakan nama jaringan (SSID) dan kata sandi yang telah ditentukan sebelumnya. Proses ini akan mencoba untuk terhubung ke jaringan WiFi yang sesuai.
5. client.setCACert(TELEGRAM_CERTIFICATE_ROOT);: Mengatur sertifikat root (CA certificate) untuk koneksi aman ke server Telegram. Ini diperlukan karena koneksi ke server Telegram biasanya menggunakan protokol HTTPS.
6. lcd.init(); lcd.backlight();: Inisialisasi dan menghidupkan backlight pada LCD yang telah terhubung dengan perangkat Arduino. Ini mempersiapkan LCD untuk menampilkan teks atau informasi lainnya.
7. while (WiFi.status() != WL_CONNECTED) { delay(500); Serial.print("."); }: Loop ini menunggu hingga perangkat terhubung ke jaringan WiFi. Selama perangkat tidak terhubung (WiFi.status() != WL_CONNECTED), program akan menunggu 500 milidetik (delay(500)) dan mencetak titik (Serial.print(".")) untuk menunjukkan bahwa proses koneksi sedang berlangsung.
8. Setelah perangkat terhubung ke jaringan WiFi, pesan "WiFi connected" akan dicetak ke terminal serial (Serial.println("WiFi connected");), dan fungsi sensors.begin(); akan dipanggil untuk memulai penggunaan sensor suhu yang telah diinisialisasi sebelumnya.

```
void loop() {
  Serial.print("Requesting temperatures...");
  sensors.requestTemperatures();
  Serial.println("DONE");
  float tempC = sensors.getTempCByIndex(0);

  if(tempC != DEVICE_DISCONNECTED_C)
  {
    Serial.print("Temperature for the device 1 (index 0) is: ");
    Serial.println(tempC);
    lcd.clear(); // Membersihkan tampilan LCD sebelum menampilkan suhu yang baru
    lcd.setCursor(0, 0);
    lcd.print("Suhu Air: ");
    lcd.print(tempC);
    lcd.print(" C");

    if(tempC > 35 || tempC > 125) { // Mengirimkan pesan jika suhu kurang dari 30 atau lebih dari 125 derajat
      String message = "PERINGATAN!\n";
      message += "Suhu Air Aquarium Terlalu Tinggi: " + String(tempC, 2) + " °C";
      bot.sendMessage(CHAT_ID, message, "");
    }
  }
  else
  {
    Serial.println("Error: Could not read temperature data");
  }
  delay(5000);
}
```

1. Serial.print("Requesting temperatures...");: Mencetak pesan ke terminal serial untuk menandakan bahwa perangkat sedang meminta data suhu.
2. sensors.requestTemperatures();: Meminta data suhu dari sensor suhu yang telah diinisialisasi sebelumnya. Metode ini mengambil data suhu aktual dari sensor dan menyimpannya dalam buffer untuk diakses kemudian.
3. Serial.println("DONE");: Mencetak pesan ke terminal serial untuk menandakan bahwa proses meminta data suhu telah selesai.
4. float tempC = sensors.getTempCByIndex(0);: Mengambil nilai suhu dalam derajat Celsius dari sensor suhu yang telah diinisialisasi. Nilai ini disimpan dalam variabel tempC.
5. if(tempC != DEVICE_DISCONNECTED_C) { ... }: Memeriksa apakah nilai suhu yang diperoleh valid atau tidak terputus. Jika nilainya valid, blok kode di dalam if akan dieksekusi.
6. Serial.print("Temperature for the device 1 (index 0) is: ");: Mencetak pesan ke terminal serial untuk menunjukkan bahwa nilai suhu dari sensor telah diperoleh.
7. Serial.println(tempC);: Mencetak nilai suhu aktual ke terminal serial.
8. lcd.clear();: Menghapus layar LCD sebelum menampilkan nilai suhu yang baru.
9. lcd.setCursor(0, 0);: Mengatur kursor pada layar LCD ke posisi baris 1, kolom 1.
10. lcd.print("Suhu Air: ");: Mencetak teks "Suhu Air: " pada layar LCD.
11. lcd.print(tempC);: Mencetak nilai suhu aktual pada layar LCD.
12. lcd.print(" C");: Mencetak satuan derajat Celsius pada layar LCD.
13. if(tempC > 35 || tempC > 125) { ... }: Memeriksa apakah nilai suhu melebihi batas yang ditentukan (35 derajat Celsius atau 125 derajat Celsius). Jika melebihi batas, blok kode di dalam if akan dieksekusi.
14. String message = "PERINGATAN!\n"; ...: Membuat pesan yang akan dikirimkan ke bot Telegram jika suhu melebihi batas yang ditentukan. Pesan ini berisi informasi peringatan suhu yang tinggi.
15. bot.sendMessage(CHAT_ID, message, "");: Mengirim pesan peringatan ke bot Telegram dengan menggunakan metode sendMessage dari objek bot yang telah diinisialisasi sebelumnya.
16. Jika nilai suhu tidak valid (misalnya, sensor terputus), akan dicetak pesan kesalahan ke terminal serial (Serial.println("Error: Could not read temperature data");).
17. delay(5000);: Menunda eksekusi program selama 5000 milidetik (5 detik) sebelum melanjutkan ke iterasi berikutnya dari fungsi loop(). Ini dilakukan untuk menghindari pembacaan sensor yang terlalu cepat dan membebani sistem.