

Lembar Program Keseluruhan (Coding)

```
#include "thingProperties.h"
#include "HX711.h" // library untuk sensor load cell HX711

const int LOADCELL_DOUT_PIN = 14; // pin data HX711 dihubungkan
ke pin D4 pada NodeMCU
const int LOADCELL_SCK_PIN = 12; // pin clock HX711 dihubungkan
ke pin D5 pada NodeMCU
const int sensorPin = 4; // pin sensor tetesan dihubungkan ke
pin D2 pada NodeMCU

HX711 scale; // inisialisasi objek untuk sensor load cell
unsigned long previousMillis = 0; // menyimpan waktu sebelumnya
const long interval = 60000; // interval waktu dalam milidetik
(1 menit)
// int count = 0; // menyimpan jumlah tetesan sebelumnya
int flag = HIGH; // menyimpan status sensor tetesan

void setup() {
    // Initialize serial and wait for port to open:
    Serial.begin(9600);
    // This delay gives the chance to wait for a Serial Monitor
    without blocking if none is found
    delay(1500);

    // Defined in thingProperties.h
    initProperties();

    // Connect to Arduino IoT Cloud
    ArduinoCloud.begin(ArduinoIoTPreferredConnection);
    /*
        The following function allows you to obtain more
        information
        related to the state of network and IoT Cloud connection
        and errors
```

the higher number the more granular information you'll get.

The default is 0 (only errors).

Maximum is 4

*/

```
setDebugMessageLevel(2);  
ArduinoCloud.printDebugInfo();
```

pinMode(sensorPin, INPUT); // konfigurasi pin sensor tetesan sebagai input

scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN); // inisialisasi sensor load cell HX711

scale.set_scale(3300.0 / 1023.0); // faktor kalibrasi untuk sensor load cell 5kg

scale.tare(); // mengatur nol pada sensor load cell

}

void loop() {

ArduinoCloud.update();

// Your code here

unsigned long currentMillis = millis(); // mengambil waktu sekarang

int sensorValue = digitalRead(sensorPin); // membaca nilai sensor tetesan

if (sensorValue != flag) { // jika terjadi perubahan status sensor tetesan

flag = sensorValue; // update status sensor tetesan

if (flag == LOW) { // jika terjadi perubahan dari HIGH ke LOW

count++; // tambahkan jumlah tetesan

Serial.print("Tetesan ke-"); // tampilkan pesan

Serial.println(count);

}

}

if (currentMillis - previousMillis >= interval) { // jika waktu sudah mencapai interval

if (count > 0) { // jika ada tetesan

```

        Serial.print("Jumlah tetesan dalam 1 Menit: "); //
tampilkan pesan
        Serial.println(count);
        float weight = scale.get_units(); // ambil nilai berat
dari sensor load cell
        Serial.print("Persentase Sisa Infus: "); // tampilkan
pesan
        berat = (weight/142.52/5.97);
        Serial.println(berat);
    }
    else { // jika tidak ada tetesan
        Serial.println("Tidak ada tetesan dalam 1 Menit."); //
tampilkan pesan
    }
    count = 0; // reset jumlah tetesan
    previousMillis = currentMillis; // simpan waktu sekarang
sebagai waktu sebelumnya
    }
}

/*
    Since Berat is READ_WRITE variable, onBeratChange() is
    executed every time a new value is received from IoT Cloud.
*/
void onBeratChange() {
    // Add your code here to act upon Berat change
}

/*
    Since Count is READ_WRITE variable, onCountChange() is
    executed every time a new value is received from IoT Cloud.
*/
void onCountChange() {
    // Add your code here to act upon Count change
}

```