

USER MANUAL BOOK



STORAGE PINTAR BERBASIS IOT UNTUK KETERSEDIAAN MAKANAN TERNAK

**PROGRAM STUDI D3 TEKNOLOGI KOMPUTER
FAKULTAS ILMU TERAPAN
UNIVERSITAS TELKOM BANDUNG
2024**

Daftar Tabel

Daftar Tabel	1
A. Mengenai Storage Pintar.....	2
B. Blok Diagram.....	3
C. Flowchart	4
D. Penggunaan Storage Pintar	6
E. Source Code Storage Pintar	13
F. Source Code Kalibrasi Loadcell	16

A. Mengenai Storage Pintar



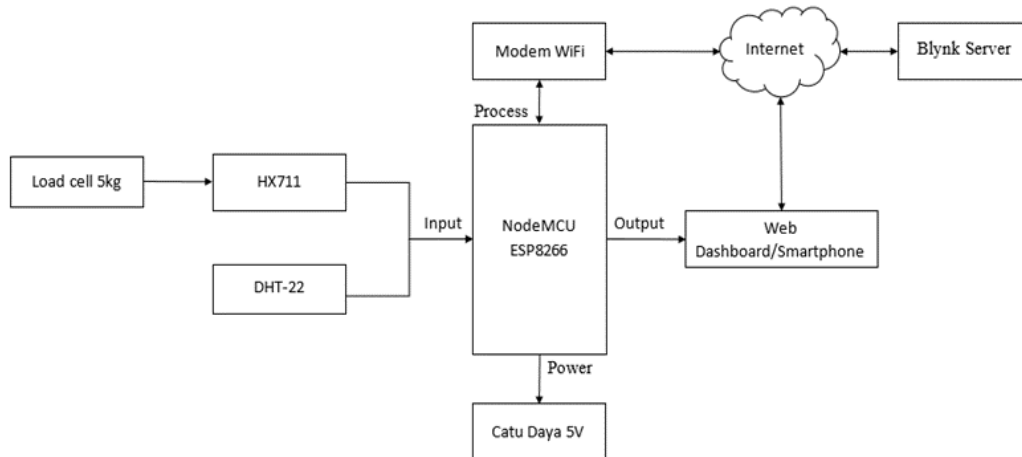
Storage pintar adalah sebuah prototipe sistem monitoring berbasis *internet of things* untuk ketersediaan pakan hijauan dan konsentrat pada *chamber* kandang kambing/domba. Penggunaan mikrokontroler NodeMCU ESP8266 dibutuhkan untuk memproses masukan data dari ketersediaan atau kondisi berat pakan dapat dideteksi menggunakan sensor *loadcell* 5kg, selain itu sensor DHT-22 digunakan juga pada sistem untuk mendeteksi suhu dan kelembaban di sekitar kandang kambing/domba.

Keadaan suhu dan kelembaban yang terus berubah di sekitar lingkungan kandang domba akan mempengaruhi perubahan pola nafsu makan pada hewan ternak. Maka berdasarkan kebutuhan peternak kambing/domba tersebut, implementasi teknologi dapat dilakukan agar membantu meringankan pekerjaan pengelolaan pakan domba dalam beternak ruminansia.

Bahwa *storage* yang sudah dipasangkan beberapa modul sensor pada prototype Selanjutnya NodeMCU ESP8266 membutuhkan sumber catu daya dengan adaptor 5V untuk dapat aktif dan bertugas melakukan pemrosesan data yang terdeteksi oleh sensor *loadcell* dan DHT-22. Kemudian NodeMCU ESP8266 akan mengirimkan kedua data pembacaan sensor secara *online* terhubung dengan koneksi internet menggunakan modem *WiFi*.

Kemudian Server blynk akan menerima data pembacaan sensor yang dikirim oleh NodeMCU ESP8266 sehingga dapat ditampilkan data pembacaan sensor pada aplikasi *smartphone* Blynk Android secara *real-time* pada saat pertukaran data antara pembacaan dari *hardware* sensor dengan *server* blynk. Selain mendapatkan notifikasi dari aplikasi blynk, penggunaan aplikasi pengirim pesan seperti telegram diterapkan pada sistem sehingga pengguna telegram dapat menerima hasil data pembacaan sensor juga.

B. Blok Diagram

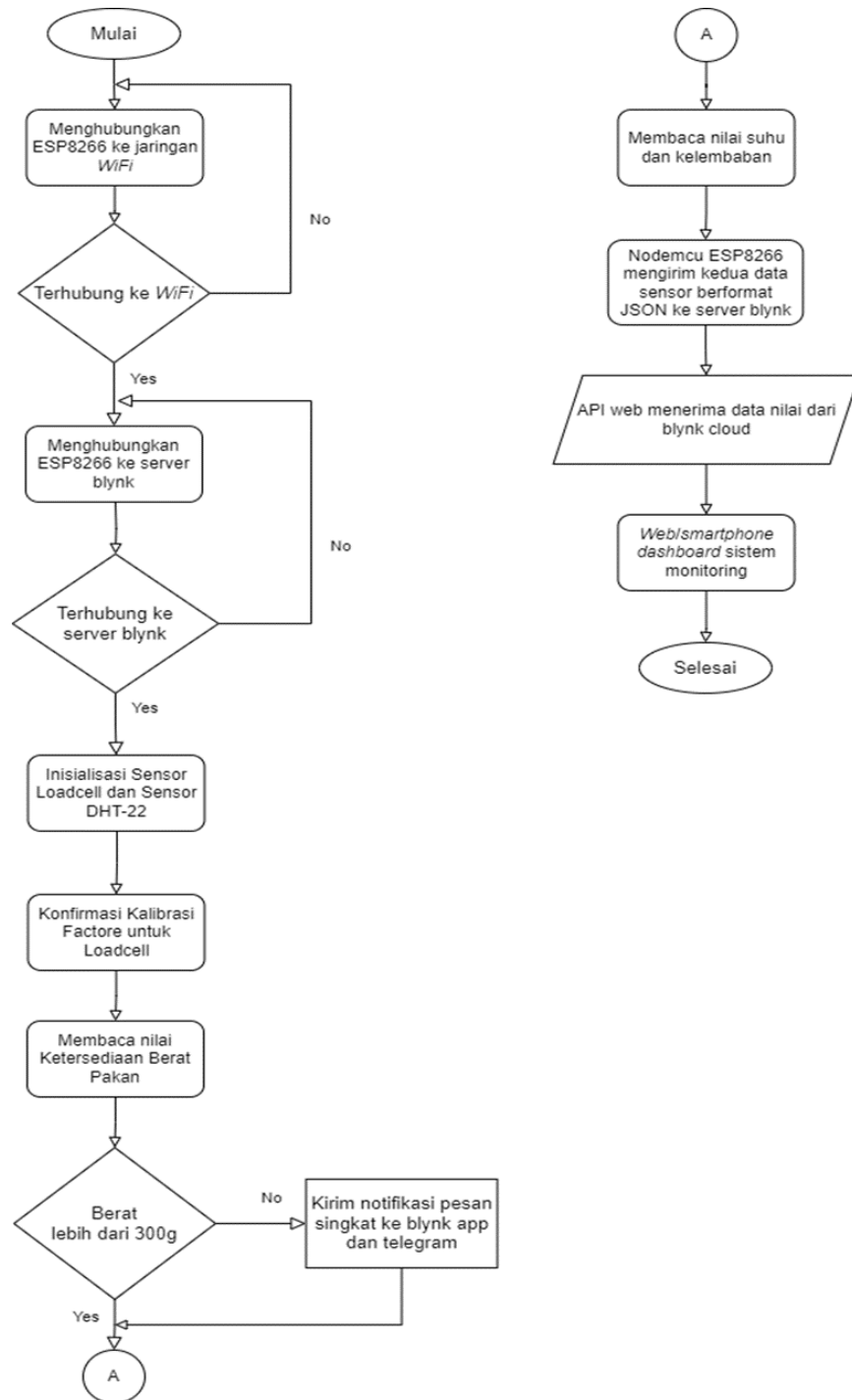


blok diagram sistem *storage* pintar yang akan dipasang pada *chamber* aslinya pada kandang, terdapat beberapa komponen *hardware* dan *software* yang dibutuhkan agar *prototype* dapat beroperasi. Untuk penjelasan dipaparkan sebagai berikut.

Pada saat peternak memberikan pakan hijauan berupa rumput pada *storage* pintar, terdapat sensor yang mendeteksi berat, suhu dan kelembaban. Modul sensor *loadcell* 5Kg berperan mendeteksi berat pakan yang diberikan pada *storage* untuk mengetahui ketersediaan pakan dengan batas maksimum deteksinya seberat 5Kg. Untuk modul sensor DHT-22 berperan dalam mendeteksi suhu dan kelembaban di sekitar kandang domba untuk mengetahui keaintdaan kandang serta pola perilaku nafsu makan domba terhadap suhu dan kelembaban, kedua modul sensor dijadikan sebagai input.

NodeMCU ESP8266 membutuhkan power supply sebesar 5V agar dapat berjalan untuk mengolah data yang didapat dari input sensor sekaligus mengirim nilai data melalui internet dari modem *WiFi*. Data pembacaan sensor yang dikirim oleh NodeMCU ESP8266 akan diterima server Blynk. Peternak dapat mengetahui kondisi pakan yang diakses dimanapun dan kapanpun melalui *smartphone*, *output* menggunakan tampilan aplikasi Blynk.

C. Flowchart



Pada proses pertama sistem melakukan inisialisasi sensor loadcell yang terhubung dengan HX711, kemudian melakukan inisialisasi DHT-22 juga jika kedua sensor telah terpasang dengan benar ke NodeMCU ESP8266. Selanjutnya NodeMCU ESP8266 akan mencoba menghubungkan koneksi internet yang akan diterima oleh modem *WiFi*. disuatu kondisi apabila koneksi internet tidak terhubung antara NodeMCU ESP8266 dengan modem *WiFi* maka proses akan mengulang kembali untuk mencari koneksi internet yang tersedia, sebaliknya jika koneksi telah terhubung maka akan melanjutkan ke proses berikutnya.

Sensor *loadcell* akan berjalan melakukan proses membaca nilai berat sementara sensor DHT22 akan melakukan proses membaca nilai suhu dan kelembaban, setelah kedua sensor dapat membaca dan menghasilkan data inputan nilai sensor. Kemudian NodeMCU ESP8266 akan mengirimkan hasil inputan data nilai sensor ke *server* Blynk.

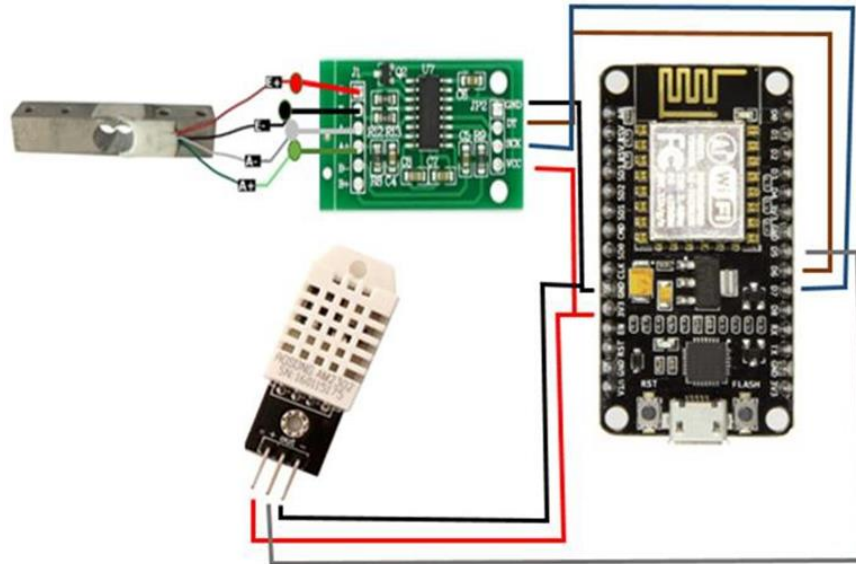
Proses selanjutnya server blynk menerima hasil inputan data nilai sensor sehingga hasil data pembacaan data sensor dapat ditampilkan ke *smartphone*. Beberapa informasi akan ditampilkan seperti informasi ketersediaan dari angka berat (Kg) pakan yang terisi pada *storage*, sementara data suhu dan kelembaban dideteksi dari keadaan kandang domba untuk pagi, siang, sore dan malam hari berupa angka dengan satuan celcius bagi suhu dan persen bagi kelembaban.

Suatu kondisi apabila berat sensor *loadcell* yang terbaca oleh loadcell kurang dari 300g maka pada tampilan layar *smartphone* akan mengirimkan informasi berupa notifikasi aplikasi Blynk, email dan *SMS* dengan pesan pakan hampir habis. Sebaliknya apabila berat sensor loadcell yang terbaca oleh loadcell lebih dari 300g maka akan menampilkan informasi *dashboard* status pembacaan kedua sensor angka berupa angka berat satuan kilogram dan angka suhu satuan celcius beserta kelembaban satuan persen.

D. Penggunaan Storage Pintar

1. Perangkat Keras

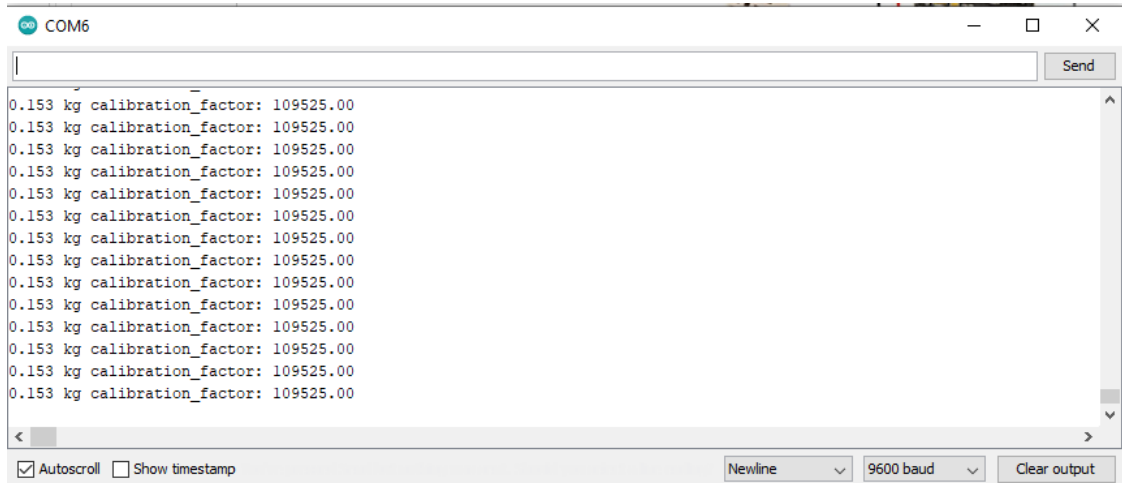
- Mengenai pinout sistem storage pintar, pinout NodeMCU ESP8266 yang terhubung dengan sensor loadcell 5Kg (DT ke D6 dan SCK ke D7), modul HX711 dan DHT22 (data output ke D5). Pada NodeMCU ESP8266 terdapat 4 pinout yang terhubung dengan pin HX711 (VCC 3.3V, Ground, DT, SCK). Pada loadcell terdapat 4 kabel jumper yang terhubung dengan pinout HX711 yaitu (A+, A-, E+, E-). Untuk pinout DHT22 (VCC, Ground, Data).



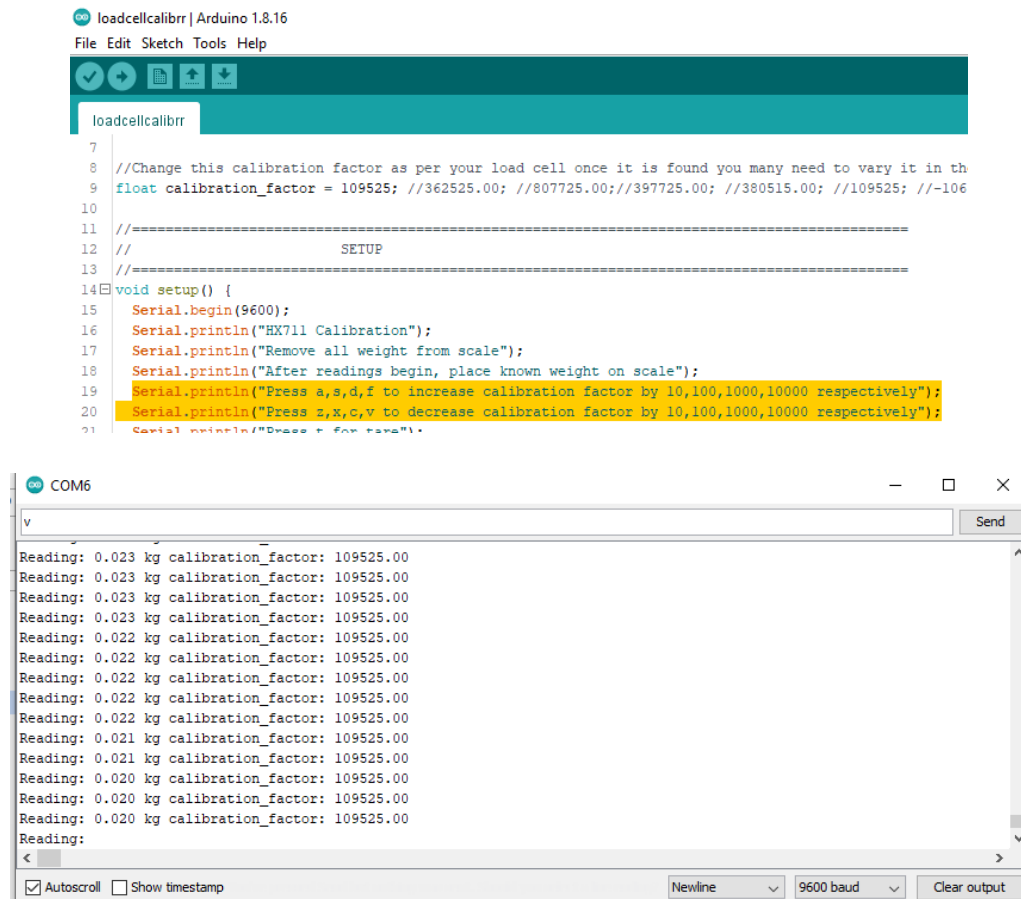
- Lakukan kalibrasi factor pada *loadcell* untuk menemukan nilai kalibrasi berat yang stabil dinilai 0 kg, *upload* skrip kodingan program kalibrasi *loadcell* melalui Arduino IDE. Untuk libraries yang diperlukan program seperti ESP8266wifi, blynk, arduinojson 5.13, ctbot, dhtsensor, hx711_adc, dan universaltelegrambot.

```
loadcellcalibr | Arduino 1.8.16
File Edit Sketch Tools Help
loadcellcalibr
7
8 //Change this calibration factor as per your load cell once i
9 float calibration_factor = 109525; //362525.00; //807725.00;
10
11 //=====
12 //                      SETUP
13 //=====
14 void setup() {
15   Serial.begin(9600);
16   Serial.println("HX711 Calibration");
17   Serial.println("Remove all weight from scale");
18   Serial.println("After readings begin, place known weight on
19   Serial.println("Press a,s,d,f to increase calibration facto
20   Serial.println("Press z,x,c,v to decrease calibration facto
21   Serial.println("Press t for tare");
22   scale.set_scale();
23   scale.tare(); //Reset the scale to 0
24
```

- Cari objek benda sebagai patokan berat loadcell, misal anak timbangan 100 gram, lalu simpan pada loadcell hingga terbaca nilai berat benda tersebut.



- Masukkan nilai kalibrasi angka dengan menambah dan mengurangi nilai kalibrasi pada serial monitor Arduino IDE, berikut penambahan nilai kalibrasi dengan input huruf a = 10, s = 100, d = 1000, f = 10000 sementara pengurangan nilai kalibrasi dengan input huruf z = 10, x = 100, c = 1000, v = 10000. Maka hasil nilai kalibrasi sudah berhasil didapat dari pencarian tersebut.



- Jika sudah menemukan berat benda yang sesuai dan stabil, sebelum memasukkan huruf “t” pada serial monitor. Ambil benda yang ditimbang tersebut. Kemudian masukkan huruf t untuk tare maka nilai awal tanpa objek berat benda adalah 0 kg.

- Kemudian masukkan nilai kalibrasi yang sudah didapat ke skrip kodingan program *storage pintar*, Lalu upload skrip kodingan tersebut. *Hardware* sudah siap digunakan.

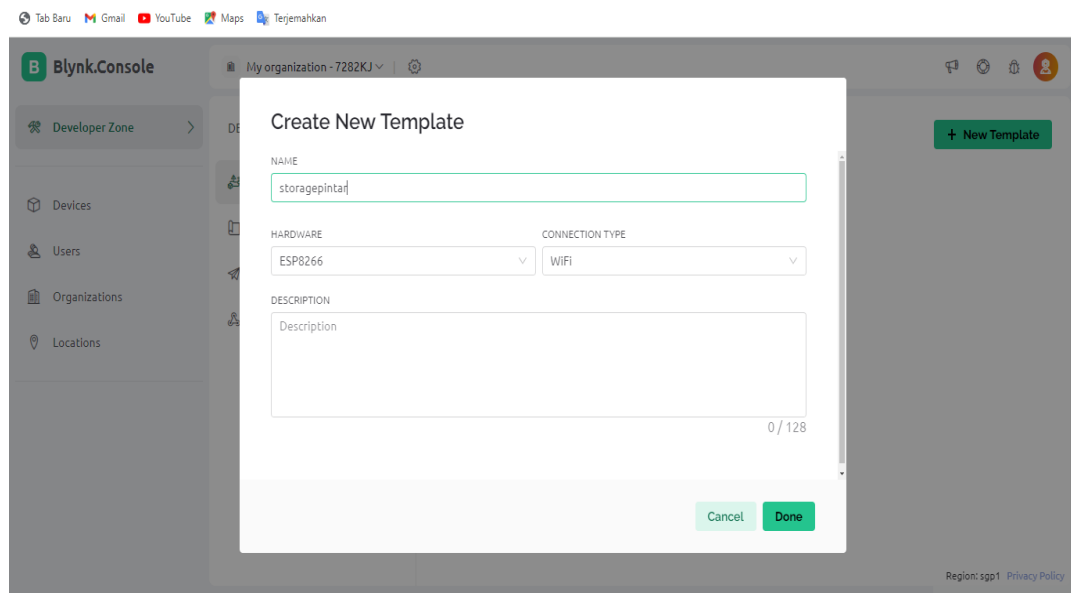
```

19 HX711 scale(D6, D7);
20 DHT dht(DHTPIN, DHTTYPE);
21 CTBot telebot;
22 String token ="6430863003:AAHKqGD0e6442FA-KVbft34WiSsolJkHj4...";
23 const int id = 1554564842;
24
25 float weight;
26 //float calibration_factor = -410825.00; // nilai kalibrasi
27 float calibration_factor = 362525.00; //374725.00;
28
29 void setup()
30 {
31   Serial.begin(115200);
32   Blynk.begin(auth, ssid, pass, "blynk.cloud", 80);
33   WiFi.begin(ssid, pass);
34   while(WiFi.status() != WL_CONNECTED)
35   {
36     delay(500);
37   }
38 }

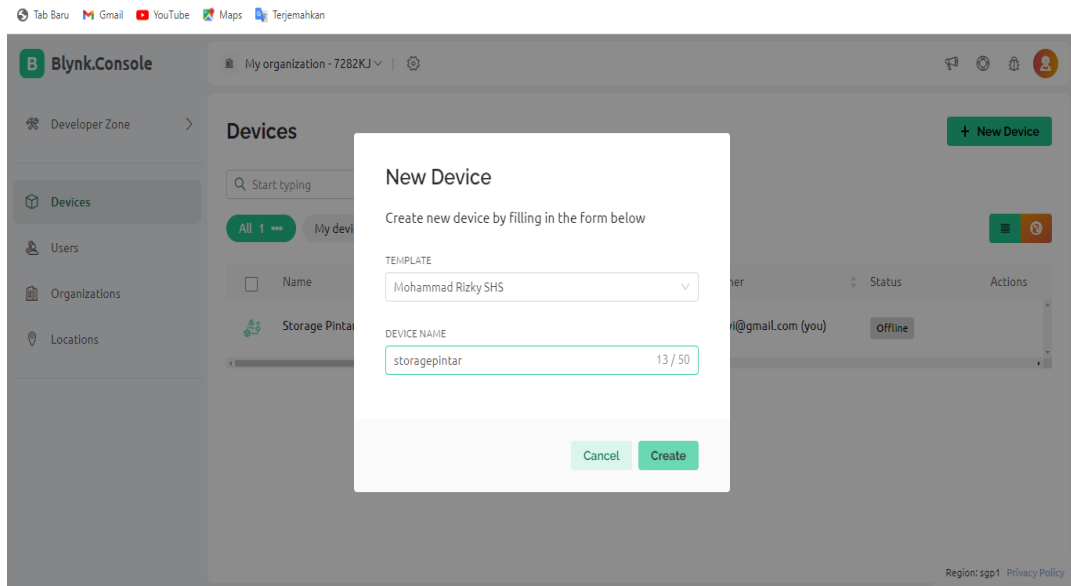
```


2. Perangkat Lunak

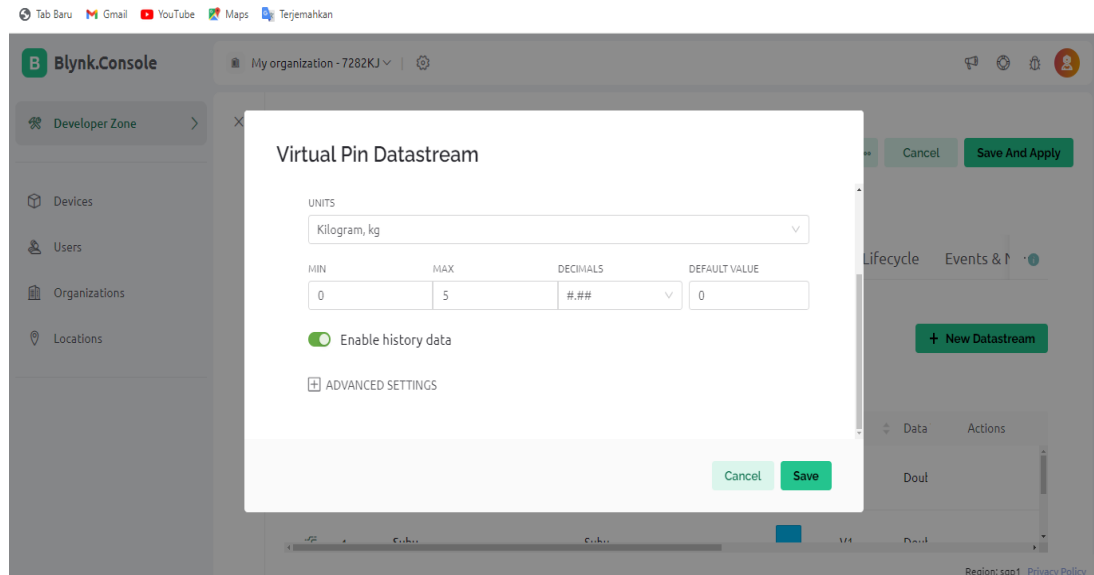
- Buka *browser* dan ketikkan blynk console.
- *Login* menggunakan akun blynk, jika belum lakukan *sign up*.
- Pada *dashboard* awal *developer zone*, jika belum membuat *template* pilih *new template*. Kemudian masukkan beberapa informasi seperti nama, nama *hardware*, *type connection* dan Deskripsi. Lalu pilih *done* jika sudah.




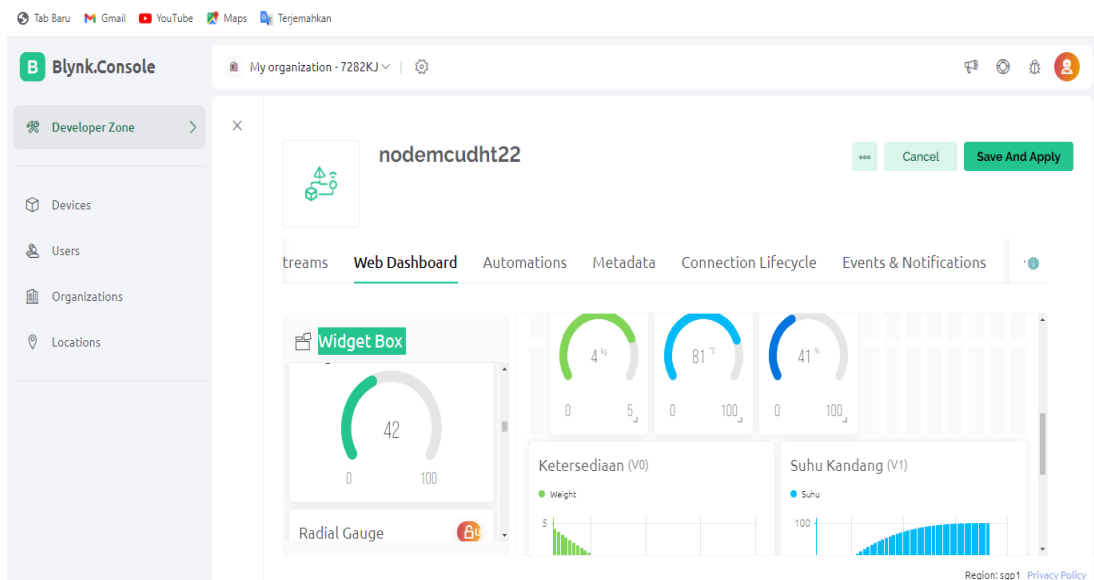
- Pada menu *devices* pilih *new device* > *from template* > choose template > storage pintar yang sudah dibuat sebelumnya > nama *device* > *create*. Jika sudah maka akan muncul *auth token* yang perlu dimasukkan pada skrip program.

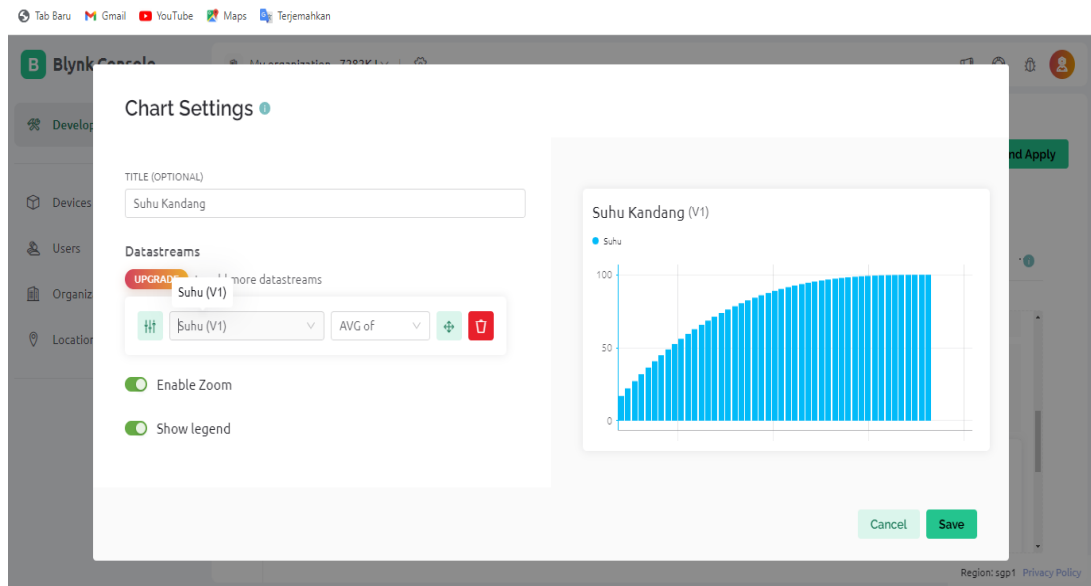


- Pilih  > edit dashboard > datastreams > new datastreams > *virtual pin* masukkan beberapa informasi seperti nama pin, tipe data, unit dan format angka > *save*. Lakukan hingga buat tiga datastream yaitu weight, suhu dan kelembaban.



- Pada *web dashboard*, masukkan beberapa *widget* seperti bar angka maupun gambar grafik. Kemudian pilih  isikan nama dan sesuaikan datastream pada masing masing *widget* > *save*.





- Pada *event* dan *notification* > *add new event* > *general*, isikan nama *event*, *event code*, tipe *event* dan deskripsi pesan. Menu *notifications* pilih untuk email dan *push notifications to* (blynk android). Kemudian *save* > *save and apply*.

weight_alerts

General Notifications

EVENT NAME: weight_alerts
EVENT CODE: weight_alerts

TYPE: Info **Warning** Critical Content

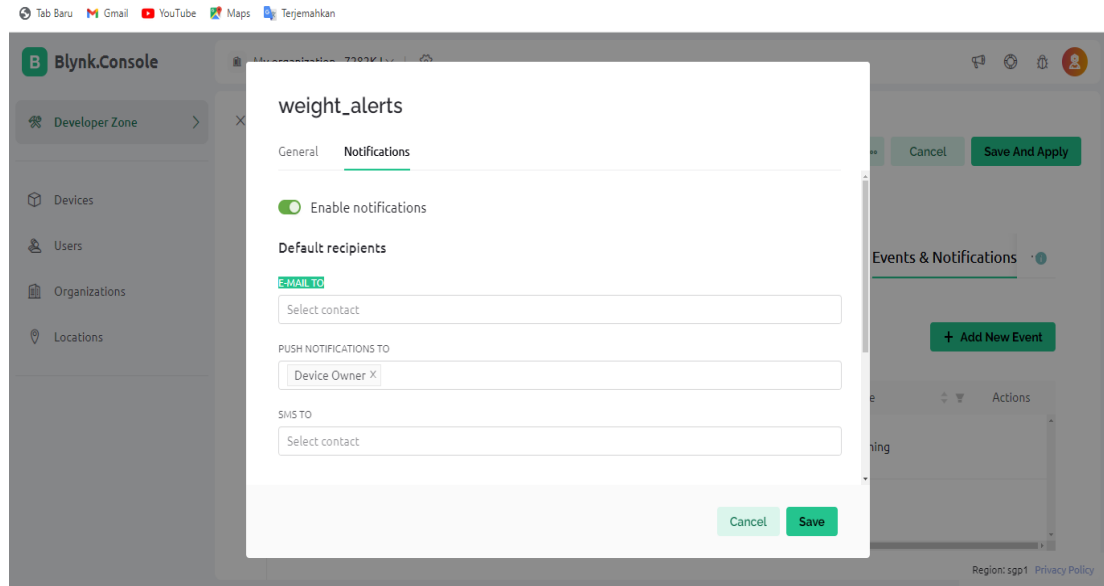
DESCRIPTION (OPTIONAL): Pakan Hampir Habis !!!

22 / 300

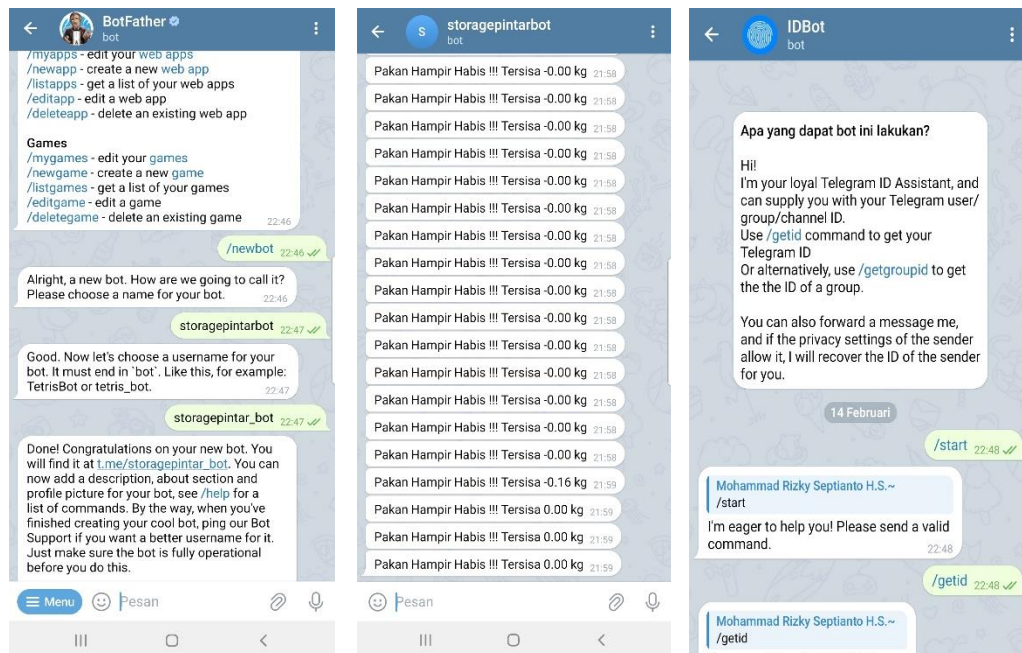
Cancel Save

Background: Events & Notifications, Add New Event

Region: sgp1 Privacy Policy



- *Web dashboard* blynk console telah selesai dikonfigurasi, maka sistem *storage* pintar sudah dapat dijalankan jika program kodingan sudah diupload dan NodeMCU ESP8266 ESP8266 menghubungkan ke sumber listrik.
- Konfigurasi untuk notifikasi telegram menggunakan fitur chatbot dan idbot. Sebelumnya gunakan libraries *ctbot* dan *arduinojson* pada Arduino IDE, pada telegram cari kontak bot bernama *botfather* > pada kolom chat ketikkan */start* > */newbot* > ketikkan nama bot untuk *storage* pintar > ketikkan juga username untuk *storage* pintar. Maka akan muncul akses token chat yang akan dimasukkan pada skrip program. Selanjutnya cari juga kontak *idbot* lalu ketikkan pada kolom chat */start* > ketikkan lagi */getid* maka akan melakukan *generate* id berupa angka yang akan dimasukkan pada skrip program.



E. Source Code Storage Pintar

```
#define BLYNK_TEMPLATE_ID "TMPL6NCn9MEbn"

#define BLYNK_TEMPLATE_NAME "nodemcuesp8266dht22"

#define BLYNK_AUTH_TOKEN "XzXgLdePUGkUX3EZDDkG0QnU62ZehsQQ"

#define BLYNK_PRINT Serial

#define DHTPIN 5 // pin 2 untuk koneksi sensor DHT22

#define DHTTYPE DHT22 // Type DHT-22


#include "HX711.h"

#include "DHT.h"

#include <ESP8266WiFi.h>

#include <BlynkSimpleEsp8266.h>

#include <CTBot.h>

char auth[] = BLYNK_AUTH_TOKEN;

char ssid[] = "SorcefourG"; // wifi ssid

char pass[] = "Ekahijiduatu321"; // wifi password

HX711 scale(D6, D7);

DHT dht(DHTPIN, DHTTYPE);

CTBot telebot;

String token = "6430863003:AAHKqGD0e6442FA-KVbft34WiSsolJkHj40";

const int id = 1554564842;

float weight;

//float calibration_factor = -410825.00; // nilai kalibrasi

float calibration_factor = 362525.00;//374725.00;


void setup()

{

    Serial.begin(115200);
```

```

Blynk.begin(auth, ssid, pass, "blynk.cloud", 80);
WiFi.begin(ssid, pass);
while(WiFi.status() != WL_CONNECTED)
{
    delay(500);
}
telebot.wifiConnect(ssid, pass);
telebot.setTelegramToken(token);
scale.set_scale();
scale.tare(); //Reset the scale to 0
long zero_factor = scale.read_average(); //Pembacaan loadcell
dht.begin();
}

void loop()
{
    scale.set_scale(calibration_factor); //menghubungkan dengan calibration factor
    weight = scale.get_units(5);
    Serial.print("Weight: ");
    Serial.print(weight);
    Serial.println(" KG");
    Serial.println();
    float Temperature = dht.readTemperature();
    float Humidity = dht.readHumidity();
    Serial.print("Suhu: ");
    Serial.print(Temperature);
    Serial.println(" °C");
    Serial.print("Kelembaban: ");
    Serial.print(Humidity);

```

```

Serial.println(" %");

Blynk.virtualWrite(V0, weight);

Blynk.virtualWrite(V1, Temperature);

Blynk.virtualWrite(V2, Humidity);

Blynk.run();

if (weight < 0.20){

  batasberat();

}

}

void batasberat(){

  back:

  Blynk.logEvent("weight_alerts", String("Pakan Hampir Habis !!! Tersisa ") + weight + String("
kg"));

  telebot.sendMessage(id, String("Pakan Hampir Habis !!! Tersisa ") + weight + String(" kg"));

  weight = scale.get_units(5);

  Serial.print("Weight: ");

  Serial.print(weight);

  Serial.println(" KG");

  Serial.println();

  float Temperature = dht.readTemperature();

  float Humidity = dht.readHumidity();

  Serial.print("Suhu: ");

  Serial.print(Temperature);

  Serial.println(" °C");

  Serial.print("Kelembaban: ");

  Serial.print(Humidity);

  Serial.println(" %");

```



```

Blynk.virtualWrite(V0, 0.00);

Blynk.virtualWrite(V1, Temperature);

Blynk.virtualWrite(V2, Humidity);

Blynk.run();

```

```

if (weight < 0.20){
    goto back;
}
else {
    loop();
}
}

```

F. Source Code Kalibrasi Loadcell

```

#include "HX711.h" //You must have this library in your arduino library folder

#define DOUT D6
#define CLK D7

HX711 scale(DOUT, CLK);

//Change this calibration factor as per your load cell once it is found you many need to vary it in
thousands

float calibration_factor = 362525.00; //807725.00; //397725.00; //380515.00; //109525; //-
106600 worked for my 40Kg max scale setup

//=====
=====

//          SETUP

```

```

//=====
=====

void setup() {
  Serial.begin(9600);
  Serial.println("HX711 Calibration");
  Serial.println("Remove all weight from scale");
  Serial.println("After readings begin, place known weight on scale");
  Serial.println("Press a,s,d,f to increase calibration factor by 10,100,1000,10000 respectively");
  Serial.println("Press z,x,c,v to decrease calibration factor by 10,100,1000,10000 respectively");
  Serial.println("Press t for tare");
  scale.set_scale();
  scale.tare(); //Reset the scale to 0

  long zero_factor = scale.read_average(); //Get a baseline reading
  Serial.print("Zero factor: "); //This can be used to remove the need to tare the scale. Useful in
  permanent scale projects.
  Serial.println(zero_factor);
}

//=====
=====

//          LOOP

//=====
=====

void loop() {

  scale.set_scale(calibration_factor); //Adjust to this calibration factor

  Serial.print("Reading: ");
  Serial.print(scale.get_units(), 3);

```

Serial.print(" kg"); //Change this to kg and re-adjust the calibration factor if you follow SI units like a sane person

```
Serial.print(" calibration_factor: ");
```

```
Serial.print(calibration_factor);
```

```
Serial.println();
```

```
if(Serial.available())
```

```
{
```

```
    char temp = Serial.read();
```

```
    if(temp == '+' || temp == 'a')
```

```
        calibration_factor += 10;
```

```
    else if(temp == '-' || temp == 'z')
```

```
        calibration_factor -= 10;
```

```
    else if(temp == 's')
```

```
        calibration_factor += 100;
```

```
    else if(temp == 'x')
```

```
        calibration_factor -= 100;
```

```
    else if(temp == 'd')
```

```
        calibration_factor += 1000;
```

```
    else if(temp == 'c')
```

```
        calibration_factor -= 1000;
```

```
    else if(temp == 'f')
```

```
        calibration_factor += 10000;
```

```
    else if(temp == 'v')
```

```
        calibration_factor -= 10000;
```

```
    else if(temp == 't')
```

```
        scale.tare(); //Reset the scale to zero
```

```
}
```

```
}
```