DAFTAR PUSTAKA

Andrianto, R., and F. Irawan. "Implementasi Metode Regresi Linear Berganda Pada Sistem Prediksi Jumlah Tonase Kelapa Sawit di PT. Paluta Inti Sawit." *Jurnal Pendidikan Tambusai* 7.1 (2023): 2926-2936.

Alamanda, Asri Elies. "Peran Pemerintah dalam Optimalisasi Pelaksanaan Peremajaan Tanaman Kelapa Sawit." *Wajah Hukum* 7.1 (2023): 73-83.

AS Muhammad. (2024). Sistem Pemantauan Kadar Saline dan Kesehatan Cerdas Berbasis IOT. Diperoleh dari [PDF]

Bhaskar, S., Bradley, S., Chattu, V. K., Adisesh, A., Nurtazina, A., Kyrykbayeva, S., . . . Thomas, P. (2020). Telemedicine across the globe-position paper from the COVID-19 pandemic health system resilience PROGRAM (REPROGRAM) international consortium (Part 1). Frontiers in public health, 8, 556720.

Coklat, C. dkk. (2020). "Aplikasi Timbang Elektronik pada Penimbangan Ritel, Laboratorium, dan Industri Menggunakan Sensor Load Cell."

Gunawan, R. (2020). "Integrasi Smart Sensor dalam Industri Perkebunan Kelapa Sawit". Jurnal Teknologi dan Industri Pertanian, 24(2), 223-234.

Haryono, R., & Suryadi, T. (2017). *Efisiensi Kerja dengan Sistem Otomatisasi pada Loading Ramp Pabrik Kelapa Sawit*. Jurnal Teknologi Hasil Pertanian Indonesia, 22(2), 123-134.

Juneau, Christopher M., dkk. "Keandalan dan Kegunaan Variabel Gaya-Waktu yang Diperoleh dari Sel Beban yang Dikumpulkan Selama Tugas Ekstensi Lutut Isometrik yang Dibatasi dan Tidak Dibatasi pada Plinth." *Jurnal Sains dalam Olahraga dan Latihan* 6.1 (2024): 81-89.

Massaad, Y. (2018). Akuisisi Data dan Pemrosesan Sinyal untuk Sensor Cerdas. Springer.

Mulyadi, D. (2016). "Penggunaan Sensor Load Cell dalam Proses Penimbangan Otomatis". Prosiding Seminar Nasional Teknologi Terapan, 5(1), 55-63.

Prasetyo, D. (2019). *Analisis Efektivitas Loading Ramp Terhadap Kapasitas Produksi Pabrik Kelapa Sawit*. Jurnal Keteknikan Pertanian, 25(1), 45-58.

Rahardjo, AS (2019). "Pengembangan Sistem Otomatisasi Pabrik Kelapa Sawit Menggunakan ICT". Jurnal Sistem dan Informatika Pertanian, 18(3), 98-112.

Rahman, H. (2018). Potret pertumbuhan Ekonomi, Kesenjangan dan Kemiskinan di Indonesia Dalam Tinjauan Ekonomi Politik Pembangunan. Jurnal Ilmu dan Budaya, 40(55).

Said, Andi Ali. *Pentingnya Kompetensi Pada Dunia Pendidikan Berbasis Information & Communication Technology (ICT)*. uwais inspirasi indonesia, 2023.

Santoso, B. (2017). "Implementasi Sensor Berat pada Sistem Produksi Kelapa Sawit". Jurnal Teknologi Pertanian Indonesia, 15(2), 157-168.

Sharma, A., Pruthi, M., & Sageena, G. (2022). Adoption of telehealth technologies: an approach to improving healthcare system. Translational Medicine Communications, 7(1), 20.

Smith, J., Jhonson, A,& Brown, C. (2018). Penerapan Sensor Load Cell pada Industri Otomotif. Jurnal Teknik Otomotif, 10(2), 45-58.

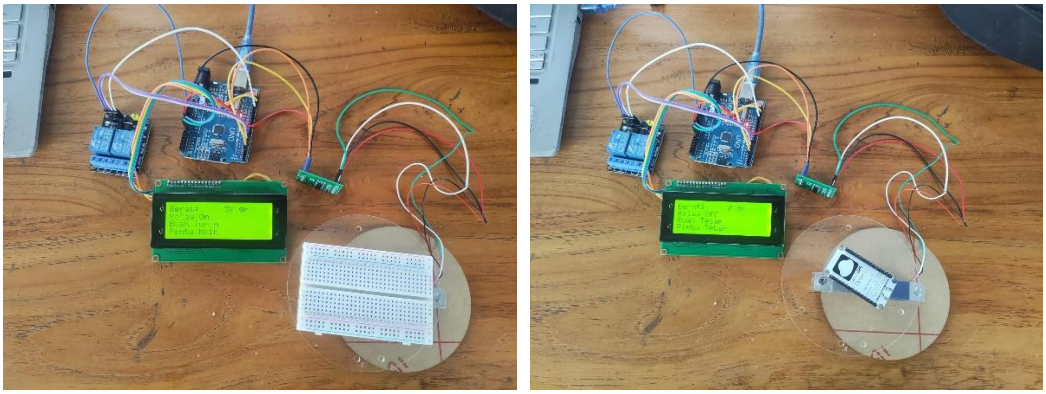
Wilson, D.dkk. (2021). "Penelitian dan Pengembangan Sensor Load Cell untuk Analisis Beban Struktural dan Pengujian Material."

Winarno, A., & Sutrisno, B. (2016). *Pengaruh Desain Loading Ramp Terhadap Keselamatan dan Kenyamanan Operator*. Jurnal Ergonomi Indonesia, 12(3), 67-80.

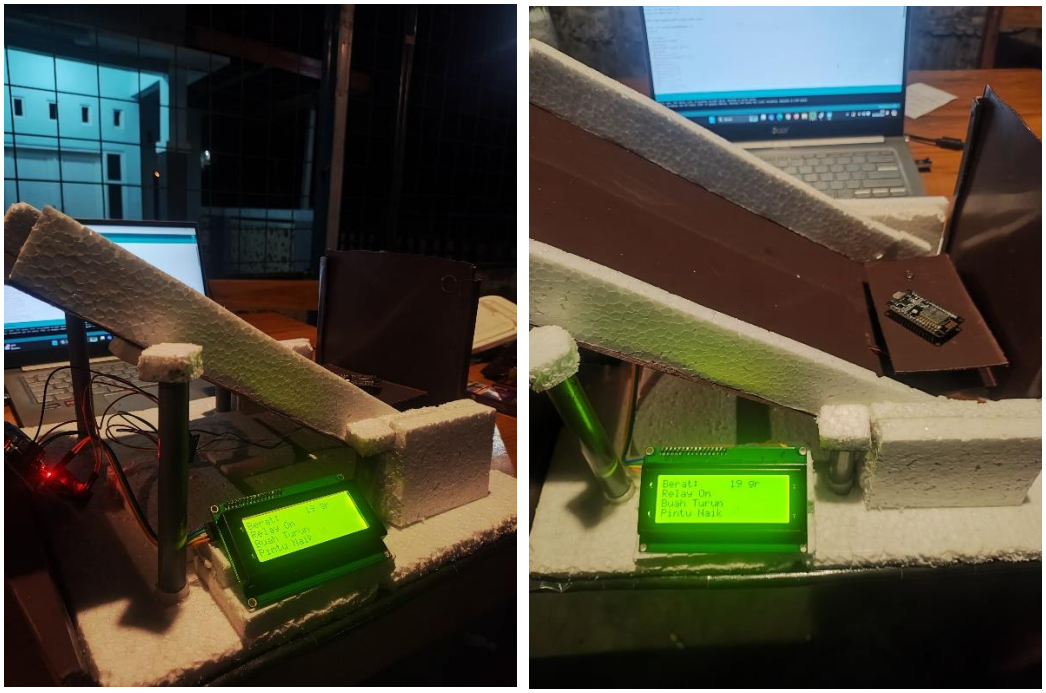
Yulianto, K. (2018). *Peningkatan Produktivitas Dengan Penggunaan Loading Ramp pada Pabrik Kelapa Sawit*. Jurnal Industri Perkebunan, 18(4), 98-110.

LAMPIRAN

Gamar Rangkaian Elektronika dengan Kotak Komponen Elektronika



Gambar Rangkaian Elektronika



Gambar Model Alat Prototype

Koding Kalibrasi

```
#include <HX711_ADC.h>
#if defined(ESP8266)|| defined(ESP32) || defined(AVR)
#include <EEPROM.h>
#endif
const int HX711_dout = 2;
const int HX711_sck = 3;
HX711_ADC LoadCell(HX711_dout, HX711_sck);
const int calVal_eepromAdress = 0;
 unsigned long t = 0;
void setup() {
```

```
  Serial.begin(57600); delay(10);
  Serial.println();
  Serial.println("Starting...");
  LoadCell.begin();
  unsigned long stabilizingtime = 2000;
  boolean _tare = true;
  LoadCell.start(stabilizingtime, _tare);
  If (LoadCell.getTareTimeoutFlag() ||
        LoadCell.getSignalTimeoutFlag()) {
    Serial.println("Timeout, check MCU>HX711 wiring and pin
        designations");
    while (1);
  }
  else {
    LoadCell.setCalFactor(1.0);
    Serial.println("Startup is complete");
  }
  while (!LoadCell.update());
  calibrate();
}
void loop() {
  static boolean newDataReady = 0;
  const int serialPrintInterval = 0;
```

```
  if (LoadCell.update()) newDataReady = true;
  if (newDataReady) {
    if (millis() > t + serialPrintInterval) {
      float i = LoadCell.getData();
      Serial.print("Load_cell output val: ");
      Serial.println(i);
      newDataReady = 0;
      t = millis();
    }
  }
  if (Serial.available() > 0) {
    char inByte = Serial.read();
    if (inByte == 't') LoadCell.tareNoDelay();
    else if (inByte == 'r') calibrate();
    else if (inByte == 'c') changeSavedCalFactor();
  }
  if (LoadCell.getTareStatus() == true) {
    Serial.println("Tare complete");
  }
}
void calibrate() {
  Serial.println("***");
  Serial.println("Start calibration:");
```

```
Serial.println("Place the load cell an a level stable surface.");
Serial.println("Remove any load applied to the load cell.");
Serial.println("Send 't' from serial monitor to set the tare offset.");
boolean _resume = false;
while (_resume == false) {
  LoadCell.update();
  if (Serial.available() > 0) {
    if (Serial.available() > 0) {
      char inByte = Serial.read();
      if (inByte == 't') LoadCell.tareNoDelay();
    }
  }
  if (LoadCell.getTareStatus() == true) {
    Serial.println("Tare complete");
    _resume = true;
  }
}
Serial.println("Now, place your known mass on the loadcell.");
Serial.println("Then send the weight of this mass (i.e. 100.0) from
        serial monitor.");
float known_mass = 0;
_resume = false;
while (_resume == false) {
```

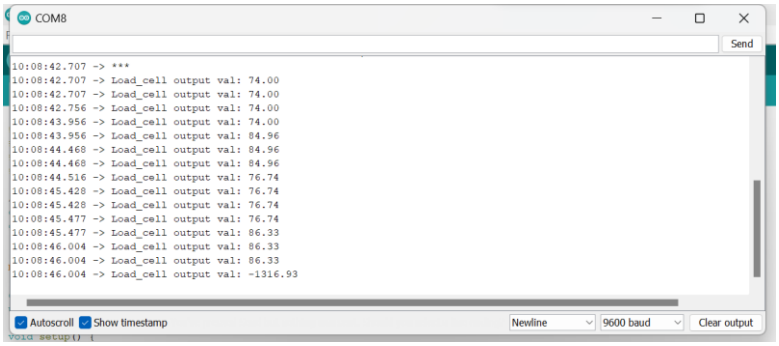
```
  LoadCell.update();
  if (Serial.available() > 0) {
    known_mass = Serial.parseFloat();
    if (known_mass != 0) {
      Serial.print("Known mass is: ");
      Serial.println(known_mass);
      _resume = true;
    }
  }
}
LoadCell.refreshDataSet();
float newCalibrationValue =
        LoadCell.getNewCalibration(known_mass);
Serial.print("New calibration value has been set to: ");
Serial.print(newCalibrationValue);
Serial.println(", use this as calibration value (calFactor) in your
        project sketch.");
Serial.print("Save this value to EEPROM adress ");
Serial.print(calVal_eepromAdress);
Serial.println("? y/n");
_resume = false;
while (_resume == false) {
  if (Serial.available() > 0) {
```

```
      char inByte = Serial.read();
      if (inByte == 'y') {
#if defined(AVR)|| defined(AVR)
#endif
        EEPROM.put(calVal_eepromAdress, newCalibrationValue);
#if defined(AVR)|| defined(AVR)
#endif
        EEPROM.get(calVal_eepromAdress, newCalibrationValue);
        Serial.print("Value ");
        Serial.print(newCalibrationValue);
        Serial.print(" saved to EEPROM address: ");
        Serial.println(calVal_eepromAdress);
        _resume = true;
      }
      else if (inByte == 'n') {
        Serial.println("Value not saved to EEPROM");
        _resume = true;
      }
    }
  }
  Serial.println("End calibration");
  Serial.println("***");
  Serial.println("To re-calibrate, send 'r' from serial monitor.");
```

```
  Serial.println("For manual edit of the calibration value, send 'c'
          from serial monitor.");
  Serial.println("***");
}
void changeSavedCalFactor() {
  float oldCalibrationValue = LoadCell.getCalFactor();
  boolean _resume = false;
  Serial.println("***");
  Serial.print("Current value is: ");
  Serial.println(oldCalibrationValue);
  Serial.println("Now, send the new value from serial monitor, i.e.
          696.0");
  float newCalibrationValue;
  while (_resume == false) {
   if (Serial.available() > 0) {
     newCalibrationValue = Serial.parseFloat();
     if (newCalibrationValue != 0) {
       Serial.print("New calibration value is: ");
       Serial.println(newCalibrationValue);
       LoadCell.setCalFactor(newCalibrationValue);
       _resume = true;
     }
   }
```

```
  }
  _resume = false;
  Serial.print("Save this value to EEPROM adress ");
  Serial.print(calVal_eepromAdress);
  Serial.println("? y/n");
  while (_resume == false) {
    if (Serial.available() > 0) {
      char inByte = Serial.read();
      if (inByte == 'y') {
#if defined(ESP8266)|| defined(ESP32)
        EEPROM.begin(512);
#endif
        EEPROM.put(calVal_eepromAdress, newCalibrationValue);
#if defined(ESP8266)|| defined(ESP32)
        EEPROM.commit();
#endif
        EEPROM.get(calVal_eepromAdress, newCalibrationValue);
        Serial.print("Value ");
        Serial.print(newCalibrationValue);
        Serial.print(" saved to EEPROM address: ");
        Serial.println(calVal_eepromAdress);
        _resume = true;
      }
```

```
      else if (inByte == 'n') {
        Serial.println("Value not saved to EEPROM");
        _resume = true;
      }
    }
  }
  Serial.println("End change calibration value");
  Serial.println("***");
}
```



Hasil Kalibrasi yang tepat

Koding pengukuran beban

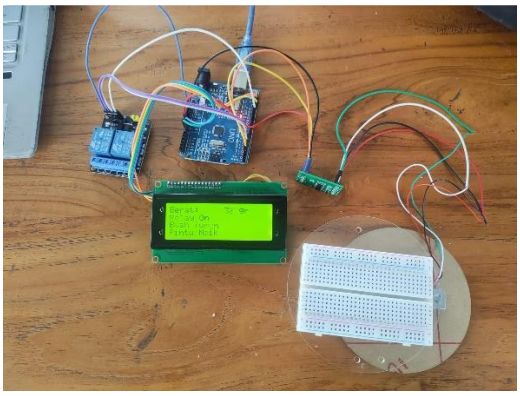
```
#include<HX711_ADC.h>
#include<EEPROM.h>
#include<LiquidCrystal_I2C.h>
#include<Relay_on/off.h>
LiquidCrystal_I2C lcd(0x27,20,4);
const int HX711_dout = 2;
const int HX711_sck = 3;
```

```
const int In1 = 4;
const int In2 = 5;
HX711_ADC LoadCell(HX711_dout, HX711_sck);
Relay_on/off(In1, In2)
const int calVal_eepromAdress = 0;
long t;
void setup() {
  Serial.begin(9600);
  lcd.begin();
  pinMode(HX711_dout, INPUT);
  delay(10);
  Serial.println();
  Serial.println("Memulai...");
  lcd.begin();
  LoadCell.begin();
  lcd.setCursor(0,0);
  lcd.print("Brt: ");
  lcd.setCursor(12,0);
  lcd.print("0");
  lcd.setCursor(14,0);
  lcd.print("gr");
  float calibrationValue;
  calibrationValue = 696.0;
```

```
  EEPROM.get(calVal_eepromAdress, calibrationValue);
  long stabilizingtime = 2000;
  boolean _tare = true;
  LoadCell.start(stabilizingtime, _tare);
  if (LoadCell.getTareTimeoutFlag()) {
    Serial.println("Timeout, cek kabel MCU>HX711 pastikan sudah
          tepat");
    while (1);
  }
  else {
    LoadCell.setCalFactor(calibrationValue);
    Serial.println("Startup selesai");
  }
}
void loop() {
  static boolean newDataReady = 0;
  const int serialPrintInterval = 0;
  if (LoadCell.update()) newDataReady = true;
  if (newDataReady) {
    if (millis() > t + serialPrintInterval) {
      int i = LoadCell.getData();
      if(i<0){
        i=0;
```

```
      }
      tampil(i);
      newDataReady = 0;
      t = millis();
    }
  }
  if(Serial.available() > 0){
    float i;
    char inByte = Serial.read();
    if (inByte == 't') LoadCell.tareNoDelay();
  }
  if (LoadCell.getTareStatus() == true) {
    Serial.println("Tara selesai");
  }
}
void tampil(int j){
  lcd.setCursor(4,0);
  lcd.print("          ");
  if(j<10){
    lcd.setCursor(12,0);
  }else if(j<100 && j>=10){
    lcd.setCursor(11,0);
  }else if(j<1000 && j>=100){
```

```
  lcd.setCursor(10,0);
 }else if(j<10000 && j>=1000){
  lcd.setCursor(9,0);
 }else if(j<100000 && j>=10000){
  lcd.setCursor(8,0);
 }else if(j<1000000 && j>=100000){
  lcd.setCursor(7,0);
 }else if(j<10000000 && j>=1000000){
  lcd.setCursor(6,0);
 }else if(j<100000000 && j>=10000000){
  lcd.setCursor(5,0);
 }else{
  lcd.setCursor(4,0);
 }
 lcd.print(j);
}
```



Hasil Pengukuran Beban