DAFTAR PUSTAKA

Abderrezzaq, Z., Mohammed, M., Ammar, N., Nordine, S., Rachid, D., & Ahmed, B. (2017, December). Impact of dust accumulation on PV panel performance in the Saharan region. In 2017 18th international conference on sciences and techniques of automatic control and computer engineering (STA) (pp. 471-475). IEEE.

Abhinawa, A. T. (2021). RANCANG BANGUN POLARIMETER BERBASIS MIKROKONTROLER ARDUINO UNO DAN SENSOR BH1750 (Doctoral dissertation, Universitas Diponegoro).

Adinoyi, M. J., & Said, S. A. (2013). Effect of dust accumulation on the power outputs of solar photovoltaic modules. Renewable energy, 60, 633-636.

Babiuch, M., Foltýnek, P., & Smutny, P. (2019). Using the ESP32 Microcontroller for Data Processing. 2019 20th International Carpathian Control Conference (ICCC), 1-6.

Breiman, L. (2001). Random forests. Machine learning, 45, 5-32.

Carbonell, J. G., Michalski, R. S., & Mitchell, T. M. (1983). An overview of machine learning. Machine learning, 3-23.

Cheddadi, Y., Cheddadi, H., Cheddadi, F., Errahimi, F., & Es-sbai, N. (2020). Design and implementation of an intelligent low-cost IoT solution for energy monitoring of photovoltaic stations. SN Applied Sciences, 2(7), 1165.

Çınar, Z. M., Abdussalam Nuhu, A., Zeeshan, Q., Korhan, O., Asmael, M., & Safaei, B. (2020). Machine learning in predictive maintenance towards sustainable smart manufacturing in industry 4.0. Sustainability, 12(19), 8211.

Çınar, Z. M., Abdussalam Nuhu, A., Zeeshan, Q., Korhan, O., Asmael, M., & Safaei, B. (2020). Machine learning in predictive maintenance towards sustainable smart manufacturing in industry 4.0. Sustainability, 12(19), 8211.

Direktorat Aneka Energi Baru dan Energi . 2020. Pedoman Teknis Kegiatan Pengembangan Sistem Perlindungan Hortikultura Tahun 2014. Direktorat Jenderal Energi Baru Terbarukan Dan Konservasi Energi Kementerian Sumber Daya Mineral, Jakarta. 84 hal.

Elamim, A., Hartiti, B., Barhdadi, A., Haibaoui, A., Lfakir, A., & Thevenin, P. (2018). Photovoltaic output power forecast using artificial neural networks. Journal of Theoretical and Applied Information Technology, 96(15), 5116-5126.

Hanna J dan Patricia. 2012. "Analisis Keekonomian Kompleks Perudddmahan Berbasis Energi Sel Surya (Studi Kasus: Perumahan Cyber Orchid Town Houses, Depok)". Depok: Universitas Indonesia.

Hodson, T. O. (2022). Root-mean-square error (RMSE) or mean absolute error (MAE): When to use them or not. Geoscientific Model Development, 15(14), 5481-5487.

Instruments, T. (2015). INA219 zero-drift, bidirectional current/power monitor with I2C interface. Dallas: Ti E2e.

Kazem, A. A., Chaichan, M. T., & Kazem, H. A. (2014). Dust effect on photovoltaic utilization in Iraq. Renewable and Sustainable energy reviews, 37, 734-749.

Li, Y., & He, J. (2017, October). Design of an intelligent indoor air quality monitoring and purification device. In 2017 IEEE 3rd Information Technology and Mechatronics Engineering Conference (ITOEC) (pp. 1147-1150). IEEE.

Liu, Y., Wang, Y., & Zhang, J. (2012). New machine learning algorithm: Random forest. In Information Computing and Applications: Third International Conference, ICICA 2012, Chengde, China, September 14-16, 2012. Proceedings 3 (pp. 246-252). Springer Berlin Heidelberg.

Loh, W. Y. (2011). Classification and regression trees. Wiley interdisciplinary reviews: data mining and knowledge discovery, 1(1), 14-23.

Louppe, G. (2014). Understanding random forests: From theory to practice. arXiv preprint arXiv:1407.7502.

Maier, A., Sharp, A., & Vagapov, Y. (2017). Comparative analysis and practical implementation of the ESP32 microcontroller module for the internet of things. 2017 Internet Technologies and Applications (ITA), 143-148.

Maleki, F., Ovens, K., Najafian, K., Forghani, B., Reinhold, C., & Forghani, R. (2020). Overview of machine learning part 1: fundamentals and classic approaches. Neuroimaging Clinics, 30(4), e17-e32.

Mohandoss, D. P., Shi, Y., & Suo, K. (2021, January). Outlier prediction using random forest classifier. In 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC) (pp. 0027-0033). IEEE.

Murphy A., & Moore C. (2019). Random forest (machine learning). Radiopaedia.org.

Osmani, K., Haddad, A., Lemenand, T., Castanier, B., & Ramadan, M. (2020). A review on maintenance strategies for PV systems. Science of the Total Environment, 746, 141753.

Pekel, E. (2020). Estimation of soil moisture using decision tree regression. Theoretical and Applied Climatology, 139(3-4), 1111-1119.

Rahaman, M. H., & Iqbal, T. (2020). A remote thermostat control and temperature monitoring system of a single-family house using openHAB and MQTT. European Journal of Electrical Engineering & Computer Science, 4(5).

Rumantri, R., Khakim, M. Y. N., & Iskandar, I. (2018). Design and characterization of low-cost sensors for air quality monitoring system. Jurnal Pendidikan IPA Indonesia, 7(3), 347-354.

Shaveta. (2023). A review on machine learning. International Journal of Science and Research Archive, 2023, 09(01), 281–285.

Song, Y. Y., & Ying, L. U. (2015). Decision tree methods: applications for classification and prediction. Shanghai archives of psychiatry, 27(2), 130.

Srivastava, D., Kesarwani, A., & Dubey, S. (2018). Measurement of Temperature and Humidity by using Arduino Tool and DHT11. International Research Journal of Engineering and Technology (IRJET), 5(12), 876-878.

Willmott, C. J., & Matsuura, K. (2005). Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. Climate research, 30(1), 79-82.

Wu, S. L., Chen, H. C., & Peng, K. J. (2023). Effect of Dust Deposition on Solar Panel in Solar Power Generation. Sensors & Materials, 35.

Yan, J., Zhang, Z., Xie, L., & Zhu, Z. (2019). A unified framework for decision tree on continuous attributes. IEEE Access, 7, 11924-11933.

Zhu, H., Li, X., Sun, Q., Nie, L., Yao, J., & Zhao, G. (2015). A power prediction method for photovoltaic power plant based on wavelet decomposition and artificial neural networks. Energies, 9(1), 11.

LAMPIRAN

Lampiran 1. Hasil Prediksi Model dan Data Aktual di Kondisi Lingkungan Minim Debu

Tanggal/Waktu	Data Aktual	Data Prediksi	
		DTR	RFR
2023-07-31 09.00.00	19,878	19,88194	19,88188
2023-07-31 09.10.00	19,882	19,88589	19,88576
2023-07-31 09.20.00	19,882	19,88783	19,88765
2023-07-31 09.30.00	19,88	19,88577	19,88553
2023-07-31 09.40.00	19,876	19,88771	19,88725
2023-07-31 09.50.00	19,868	19,88366	19,88298
2023-07-31 10.00.00	19,864	19,8816	19,8807
2023-07-31 10.10.00	19,848	19,87754	19,87643
2023-07-31 10.20.00	19,838	19,87749	19,87615
2023-07-31 10.30.00	19,832	19,87943	19,87788
2023-07-31 10.40.00	19,826	19,88137	19,8796
2023-07-31 10.50.00	19,822	19,88331	19,88133
2023-07-31 11.00.00	19,818	19,88726	19,88505
2023-07-31 11.10.00	19,802	19,8812	19,87878
2023-07-31 11.20.00	19,796	19,87714	19,8745
2023-07-31 11.30.00	19,786	19,87709	19,87423
2023-07-31 11.40.00	19,766	19,87303	19,86972
2023-07-31 11.50.00	19,74	19,86477	19,86145
2023-07-31 12.00.00	19,73	19,86252	19,85917
2023-07-31 12.10.00	19,736	19,85426	19,85089
2023-07-31 12.20.00	19,708	19,848	19,84461
2023-07-31 12.30.00	19,71	19,84775	19,84433
2023-07-31 12.40.00	19,698	19,83949	19,83606
2023-07-31 12.50.00	19,692	19,83923	19,83578
2023-07-31 13.00.00	19,688	19,83898	19,83629
2023-07-31 13.10.00	19,69	19,84072	19,83802
2023-07-31 13.20.00	19,686	19,84046	19,83777
2023-07-31 13.30.00	19,68	19,83621	19,83353
2023-07-31 13.40.00	19,68	19,84015	19,83604
2023-07-31 13.50.00	19,676	19,84409	19,83992
2023-07-31 14.00.00	19,67	19,84184	19,8378
2023-07-31 14.10.00	19,672	19,85178	19,84768

2023-07-31 14.20.00	19,672	19,85352	19,85086
2023-07-31 14.30.00	19,676	19,86147	19,85852
2023-07-31 14.40.00	19,676	19,86541	19,8624
2023-07-31 14.50.00	19,674	19,86935	19,8649
2023-07-31 15.00.00	19,676	19,8793	19,87479

Lampiran 2. Hasil Prediksi Model dan Data Aktual di Kondisi Lingkungan Sangat Berdebu

Tanggal/Waktu	Data Aktual	Data Prediksi	
		DTR	RFR
2023-10-15 09.00.00	103,676	103,7549	103,7516
2023-10-15 09.10.00	103,483	103,6509	103,6442
2023-10-15 09.20.00	103,305	103,5518	103,5419
2023-10-15 09.30.00	103,126	103,4587	103,4455
2023-10-15 09.40.00	102,936	103,3617	103,3451
2023-10-15 09.50.00	102,756	103,2686	103,2487
2023-10-15 10.00.00	102,567	103,1805	103,1573
2023-10-15 10.10.00	102,389	103,0975	103,0708
2023-10-15 10.20.00	102,219	103,0204	102,9909
2023-10-15 10.30.00	102,043	102,9434	102,9105
2023-10-15 10.40.00	101,833	102,8503	102,8151
2023-10-15 10.50.00	101,673	102,7912	102,7538
2023-10-15 11.00.00	101,523	102,7242	102,6844
2023-10-15 11.10.00	101,411	102,7091	102,6671
2023-10-15 11.20.00	101,319	102,71	102,6657
2023-10-15 11.30.00	101,221	102,709	102,6624
2023-10-15 11.40.00	101,127	102,7019	102,6531
2023-10-15 11.50.00	101,035	102,6968	102,6459
2023-10-15 12.00.00	100,954	102,6918	102,6387
2023-10-15 12.10.00	100,908	102,7107	102,6556
2023-10-15 12.20.00	100,8535	102,7176	102,6604
2023-10-15 12.30.00	100,8045	102,7266	102,6673
2023-10-15 12.40.00	100,761	102,7375	102,6762
2023-10-15 12.50.00	100,725	102,7525	102,689
2023-10-15 13.00.00	100,688	102,7664	102,7009
2023-10-15 13.10.00	100,646	102,7793	102,7118
2023-10-15 13.20.00	100,618	102,8123	102,7425
2023-10-15 13.30.00	100,578	102,8402	102,7683
2023-10-15 13.40.00	100,526	102,8631	102,7891
2023-10-15 13.50.00	100,472	102,8721	102,796
2023-10-15 14.00.00	100,396	102,863	102,7847
2023-10-15 14.10.00	100,326	102,8619	102,7815
2023-10-15 14.20.00	100,254	102,8579	102,7753
2023-10-15 14.30.00	100,18	102,8508	102,7662
2023-10-15 14.40.00	100,164	102,8917	102,8049

2023-10-15 14.50.00	100,111	102,8987	102,8097
2023-10-15 15.00.00	100,039	102,8896	102,7985

Lampiran 3. Source Code Pengambilan Data Arus Listrik

```
#include <esp_now.h>
#include <WiFi.h>
#include <Wire.h>
#include <Adafruit_INA219.h>  //memanggil library dari INA219

Adafruit_INA219 ina219; //memberikan nama pada modul sensor dengan nama "ina219"
//membuat variabel yang akan dipakai untuk perhitungan, tipe datanya adalah bilangan desimal
float shuntvoltage = 0;
float busvoltage = 0;
float current_mA = 0;
float loadvoltage = 0;
float power_mW = 0;
// REPLACE WITH THE RECEIVER'S MAC Address XX:XX:XX:XX:XX:XX
uint8_t broadcastAddress[] = {0xXX, 0xXX, 0xXX, 0xXX, 0xXX, 0xXX};
// Structure example to send data
// Must match the receiver structure
typedef struct struct_message {
    int id; // must be unique for each sender board
    float x;
    float y;
    float z;
    float a;
} struct_message;
// Create a struct_message called myData
```

```
struct_message myData;
// Create peer interface
esp_now_peer_info_t peerInfo;

// callback when data is sent
void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status) {
  Serial.print("\r\nLast Packet Send Status:\t");
  Serial.println(status == ESP_NOW_SEND_SUCCESS ? "Delivery Success" :
"Delivery Fail");
}
void setup() {
  // Init Serial Monitor
  Serial.begin(115200);
  Wire.begin();
  uint32_t currentFrequency; //mengatur current frequency
  if (! ina219.begin()) { //inisiasi untuk memulai modul sensor ina219
    Serial.println("Failed to find INA219 chip"); //jika gagal akan ada tulisan tsb
    while (1) { delay(10); } //jika gagal, program tidak akan lanjut
  }
  WiFi.begin(ssid, password);
  // Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);
  // Init ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }
  // Once ESPNow is successfully Init, we will register for Send CB to
  // get the status of Trasnmitted packet
```

```
  esp_now_register_send_cb(OnDataSent);
  // Register peer
  memcpy(peerInfo.peer_addr, broadcastAddress, 6);
  peerInfo.channel = 0;
  peerInfo.encrypt = false;
  // Add peer
  if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Failed to add peer");
    return;
  }
}
void loop() {
  // Set values to send
  myData.id = 2;
  shuntvoltage = ina219.getShuntVoltage_mV();
  busvoltage = ina219.getBusVoltage_V();
  current_mA = ina219.getCurrent_mA();
  power_mW = ina219.getPower_mW();
  loadvoltage = busvoltage + (shuntvoltage / 1000);
  //menampilkan hasil pengukuran
  Serial.print("Bus Voltage:   "); Serial.print(busvoltage); Serial.println(" V");
  Serial.print("Shunt Voltage: "); Serial.print(shuntvoltage); Serial.println(" mV");
  Serial.print("Load Voltage:  "); Serial.print(loadvoltage); Serial.println(" V");
  Serial.print("Current:       "); Serial.print(current_mA); Serial.println(" mA");
  Serial.print("Power:         "); Serial.print(power_mW); Serial.println(" mW");
  myData.x = current_mA;
  myData.y = loadvoltage;
  myData.z = power_mW;
```

```
  // Send message via ESP-NOW
  esp_err_t result = esp_now_send(broadcastAddress, (uint8_t *) &myData, sizeof(myData));
  if (result == ESP_OK) {
    Serial.println("Sent with success");
    Serial.print("Arus Listrik: ");
    Serial.print(myData.x);
    Serial.println("Tegangan Listrik: ");
    Serial.print(myData.y);
    Serial.println("Daya Listrik: ");
    Serial.print(myData.z);
  }
  else {
    Serial.println("Error sending the data");
  }
}
```

Lampiran 4. Source Code Pengambilan Data Suhu, Kelembaban, Intensitas Cahaya, dan Konsentrasi Debu

```
#include <esp_now.h>
#include <WiFi.h>
#include <DHT.h>
#include <Wire.h>
#include <BH1750.h>
#include <GP2Y1010AU0F.h>
int measurePin = 34;
int ledPin     = 5;
GP2Y1010AU0F dustSensor(ledPin, measurePin);
BH1750 lightMeter;
```

```
#define DHT_PIN 4
DHT dht(DHT_PIN,DHT11);
// REPLACE WITH THE RECEIVER'S MAC Address XX:XX:XX:XX:XX:XX
uint8_t broadcastAddress[] = {0xXX, 0xXX, 0xXX, 0xXX, 0xXX, 0xXX};
// Structure example to send data
// Must match the receiver structure
typedef struct struct_message {
    int id; // must be unique for each sender board
    float x;
    float y;
    float z;
    float a;
} struct_message;
// Create a struct_message called myData
struct_message myData;
// Create peer interface
esp_now_peer_info_t peerInfo;
// callback when data is sent
void OnDataSent(const uint8_t *mac_addr, esp_now_send_status_t status) {
  Serial.print("\r\nLast Packet Send Status:\t");
  Serial.println(status == ESP_NOW_SEND_SUCCESS ? "Delivery Success" :
"Delivery Fail");
}
void setup() {
  // Init Serial Monitor
  Serial.begin(115200);
  pinMode(DHT_PIN, INPUT_PULLUP);
  dht.begin();
  Wire.begin();
```

```
  lightMeter.begin();
  dustSensor.begin();
  pinMode(ledPin,OUTPUT);
  WiFi.begin(ssid, password);
  // Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);
  // Init ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }
  // Once ESPNow is successfully Init, we will register for Send CB to
  // get the status of Trasnmitted packet
  esp_now_register_send_cb(OnDataSent);
  // Register peer
  memcpy(peerInfo.peer_addr, broadcastAddress, 6);
  peerInfo.channel = 0;
  peerInfo.encrypt = false;
  // Add peer
  if (esp_now_add_peer(&peerInfo) != ESP_OK){
    Serial.println("Failed to add peer");
    return;
  }
}
void loop() {
  // Set values to send
  myData.id = 1;
  myData.x = dht.readTemperature();
  myData.y = dht.readHumidity();
```

```
  myData.z = lightMeter.readLightLevel();
  float dustDensity = dustSensor.read();
  if(dustDensity < 0){
    dustDensity = 0;
  }
  myData.a = dustDensity;
  // Send message via ESP-NOW
  esp_err_t result = esp_now_send(broadcastAddress, (uint8_t *) &myData, sizeof(myData));
  if (result == ESP_OK) {
    Serial.println("Sent with success");
    Serial.print("Temperature: ");
    Serial.print(myData.x);
    Serial.print("Humidity: ");
    Serial.print(myData.y);
    Serial.print("Intensitas Cahaya: ");
    Serial.print(myData.z);
    Serial.print("Dust Density = ");
    Serial.print(myData.a);
  }
  else {
    Serial.println("Error sending the data");
  }}
```

Lampiran 5. Source Code Pengambilan Data *Gateway*

```
#include "ThingSpeak.h"
#include <esp_now.h>
#include <WiFi.h>
//SSID dan Password WIFI
```

```
const char* ssid = "*****";
const char* password = "******";
WiFiClient client;
//Channel number dan APIKey Thingspeak
unsigned long channelNumber = *******;
// Service API Key
const char* apiKey = "****************";
// Structure example to receive data
// Must match the sender structure
typedef struct struct_message {
  int id;
  float x;
  float y;
  float z;
  float a;
}struct_message;
// Create a struct_message called myData
struct_message myData;
// Create a structure to hold the readings from each board
struct_message board1;
struct_message board2;
struct_message board3;
// Create an array with all the structures
struct_message boardsStruct[3] = {board1, board2, board3};
// callback function that will be executed when data is received
void OnDataRecv(const uint8_t * mac_addr, const uint8_t *incomingData, int
len) {
  char macStr[18];
  Serial.print("Packet received from: ");
```

```
  snprintf(macStr, sizeof(macStr), "%02x:%02x:%02x:%02x:%02x:%02x",
           mac_addr[0], mac_addr[1], mac_addr[2], mac_addr[3], mac_addr[4],
mac_addr[5]);
  Serial.println(macStr);
  memcpy(&myData, incomingData, sizeof(myData));
  Serial.printf("Board ID %u: %u bytes\n", myData.id, len);
  // Update the structures with the new incoming data
  boardsStruct[myData.id-1].x = myData.x;
  boardsStruct[myData.id-1].y = myData.y;
  boardsStruct[myData.id-1].z = myData.z;
  boardsStruct[myData.id-1].a = myData.a;
  Serial.printf("x value: %d \n", boardsStruct[myData.id-1].x);
  Serial.printf("y value: %d \n", boardsStruct[myData.id-1].y);
  Serial.printf("z value: %d \n", boardsStruct[myData.id-1].z);
  Serial.printf("a value: %d \n", boardsStruct[myData.id-1].a);
  Serial.println();
}
void setup() {
  //Initialize Serial Monitor
  Serial.begin(115200);
  //Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);
  //Init ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }
  // Once ESPNow is successfully Init, we will register for recv CB to
  // get recv packer info
```

```
  esp_now_register_recv_cb(OnDataRecv);
  WiFi.begin(ssid, password);
  Serial.println("Connecting");
  while(WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print("Connecting to WiFi...");
  }
  Serial.print("Connected to WiFi network with IP Address: ");
  Serial.println(WiFi.localIP());
  ThingSpeak.begin(client);
}
void loop() {
  // Acess the variables for each board
  float board1X = boardsStruct[0].x;
  float board1Y = boardsStruct[0].y;
  float board1Z = boardsStruct[0].z;
  float board1A = boardsStruct[0].a;
  float board2X = boardsStruct[1].x;
  float board2Y = boardsStruct[1].y;
  float board2Z = boardsStruct[1].z;
  ThingSpeak.setField(1, board1X);
  ThingSpeak.setField(2, board1Y);
  ThingSpeak.setField(3, board1Z);
  ThingSpeak.setField(4, board1A);
  ThingSpeak.setField(5, board2X);
  ThingSpeak.setField(6, board2Y);
  ThingSpeak.setField(7, board2Z);
  int x = ThingSpeak.writeFields(channelNumber, apiKey);
  if(x == 200){
```

```
    Serial.println("You've been connected to ThingSpeak");
    Serial.println(board1X);
    Serial.println(board1Y);
    Serial.println(board1Z);
    Serial.println(board1A);
    Serial.println(board2X);
    Serial.println(board2Y);
    Serial.println(board2Z);
    }
  if(x==500){
    Serial.println("server eror");
  }
  delay(60000);
}
```

Lampiran 6. Source Code *Preprocessing Data*

```
from google.colab import files
uploaded = files.upload()
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.preprocessing import MinMaxScaler
from sklearn.tree import DecisionTreeRegressor
```

```
from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean_squared_error

from sklearn.metrics import mean_absolute_error

from sklearn.metrics import r2_score

color_pal = sns.color_palette()

plt.style.use('fivethirtyeight')

# Membaca file Excel

df = pd.read_excel("Nama_File.xlsx")

print(df)

# Mengubah kolom Timestamp menjadi tipe data datetime

df['Tanggal/Waktu'] = pd.to_datetime(df['Tanggal/Waktu'])

# Membuat kondisi untuk memilih data antara jam 9 sampai jam 15

condition = (df['Tanggal/Waktu'].dt.hour >= 9) & (df['Tanggal/Waktu'].dt.hour
    <= 15)

# Menggunakan kondisi untuk memilih data yang sesuai

df = df[condition]

# Menentukan rentang jam yang akan dihapus (15:00 sampai 15:59)

start_time = pd.to_datetime('15:01').time()

end_time = pd.to_datetime('15:59').time()

# Menghapus baris-baris pada rentang waktu yang ditentukan

df = df[~df['Tanggal/Waktu'].apply(lambda x: start_time <= x.time() <=
    end_time)]

# Mengubah 'Tanggal/Waktu' menjadi tipe datetime

df['Tanggal/Waktu'] = pd.to_datetime(df['Tanggal/Waktu'])

# Mengubah format 'Tanggal/Waktu' menjadi 'YYYY-MM-DD HH:mm'

df['Tanggal/Waktu'] = df['Tanggal/Waktu'].dt.strftime('%Y-%m-%d %H:%M')
```

```
# Filter baris dengan menit kelipatan 10
df = df[df['Tanggal/Waktu'].dt.minute % 10 == 0]
print(df)
 # Mengganti nilai <= 0 dengan NaN
df.loc[df['Suhu'] <= 0, 'Suhu'] = None
df.loc[df['Kelembaban'] <= 0, 'Kelembaban'] = None
df.loc[df['Intensitas_Cahaya'] <= 0, 'Intensitas_Cahaya'] = None
df.loc[df['Konsentrasi_Debu'] <= 0, 'Konsentrasi_Debu'] = None
df.loc[df['Arus_Listrik'] <= 0, 'Arus_Listrik'] = None
# Melakukan interpolasi untuk mengisi nilai yang hilang
df['Suhu'] = df['Suhu'].interpolate()
df['Kelembaban'] = df['Kelembaban'].interpolate()
df['Intensitas_Cahaya'] = df['Intensitas_Cahaya'].interpolate()
df['Konsentrasi_Debu'] = df['Konsentrasi_Debu'].interpolate()
df['Arus_Listrik'] = df['Arus_Listrik'].interpolate()
print(df)
# Memisahkan variabel tanggal/waktu
df = df.set_index('Tanggal/Waktu')
df.index = pd.to_datetime(df.index)
def create_features(df):
  df = df.copy()
  df['hour'] = df.index.hour
  df['day'] = df.index.day
  df['minute'] = df.index.minute
  return df
df = create_features(df)
```

```
print(df)

#penentuan variabel independent dan dependent

prediktor = df[['Suhu', 'Kelembaban', 'Intensitas_Cahaya', 'Konsentrasi_Debu',
       'hour', 'day', 'minute', 'Tambahan_Debu']]

print(prediktor)

target = df['Arus_Listrik'].values.reshape(-1, 1)

print(target)

#normalisasi data

scaler=MinMaxScaler()

scaler.fit(prediktor)

X = scaler.transform(prediktor)

print("Normalisasi variabel prediktor :")

print(X)

scaler1 = MinMaxScaler()

scaler1.fit(target)

y = scaler1.transform(target)

print("Normalisasi variabel target :")

print(y)
```

Lampiran 7. Source Code Pembuatan model dan Prediksi

```
#Pembagian Data

X_train = X[:518]

X_test = X[518:]

y_train = y[:518]

y_test = y[518:]

print("Jumlah data training :")
```

```
print(X_train.shape, y_train.shape)
print("Jumlah data testing :")
print(X_test.shape, y_test.shape)
#Pembuatan model dengan hyperparameter default
Model_DTR = DecisionTreeRegressor()
Model_DTR.fit(X_train, y_train.ravel())
default_params_dt = Model_DTR.get_params()
# Cetak parameter default
print(default_params_dt)
Model_RFR = RandomForestRegressor()
Model_RFR.fit(X_train, y_train.ravel())
default_params_rf = Model_RFR.get_params()
# Cetak parameter default
print(default_params_rf)
#Prediksi data uji
y_pred_DTR_test = Model_DTR.predict(X_test)
y_pred_DTR_test.reshape(-1, 1)
y_pred_DTR_test = pd.DataFrame(y_pred_DTR_test)
y_pred_DTR_test
y_pred_RFR_test = Model_RFR.predict(X_test)
y_pred_RFR_test.reshape(-1, 1)
y_pred_RFR_test = pd.DataFrame(y_pred_RFR_test)
y_pred_RFR_test
#Evaluasi Hasil Prediksi
score_mse_dt = mean_squared_error(y_test, y_pred_DTR_test)
score_rmse_dt = np.sqrt(mean_squared_error(y_test, y_pred_DTR_test))
```

```
score_mae_dt = mean_absolute_error(y_test, y_pred_DTR_test)
print(f'MSE Score on Test set DECISION TREE: {score_mse_dt}')
print(f'RMSE Score on Test set DECISION TREE: {score_rmse_dt}')
print(f'MAE Score on Test set DECISION TREE: {score_mae_dt}')
score_mse_rf = mean_squared_error(y_test, y_pred_RFR_test)
score_rmse_rf = np.sqrt(mean_squared_error(y_test, y_pred_RFR_test))
score_mae_rf = mean_absolute_error(y_test, y_pred_RFR_test)
print(f'\nMSE Score on Test set RANDOM FOREST: {score_mse_rf}')
print(f'RMSE Score on Test set RANDOM FOREST: {score_rmse_rf}')
print(f'MAE Score on Test set RANDOM FOREST: {score_mae_rf}')
# Daftar hyperparameter yang ingin diuji beserta rentang nilainya
param_grid = {
    'max_depth': [None, 5, 10, 15, 20, 25, 30, 35, 40, 45],
    'min_samples_split': [2, 5, 10, 15, 20, 25, 30, 35, 40, 45],
    'min_samples_leaf': [1, 2, 4, 8, 12, 16, 20, 25, 30, 35],
}
# Inisialisasi Grid Search dengan model dan parameter grid
grid_search_dt = GridSearchCV(DecisionTreeRegressor(),
        param_grid=param_grid)
# Melakukan pencarian parameter terbaik
grid_search_dt.fit(X_train, y_train.ravel())
# Menampilkan parameter terbaik
best_params_dt = grid_search_dt.best_params_
print("Parameter Terbaik:", best_params_dt)
# Inisialisasi Grid Search dengan model dan parameter grid
```

```
grid_search_rf = GridSearchCV(RandomForestRegressor(), param_grid=param_grid)
# Melakukan pencarian parameter terbaik
grid_search_rf.fit(X_train, y_train.ravel())
# Menampilkan parameter terbaik
best_params_rf = grid_search_rf.best_params_
print("Parameter Terbaik:", best_params_rf)
# Inisialisasi model DecisionTreeRegressor
Model_DTR = DecisionTreeRegressor(
    max_depth=*,
    min_samples_split=*,
    min_samples_leaf=*
)
Model_DTR.fit(X_train, y_train.ravel())
# Inisialisasi model DecisionTreeRegressor
Model_RFR = RandomForestRegressor(
    max_depth=*,
    min_samples_split=*,
    min_samples_leaf=*
)
Model_RFR.fit(X_train, y_train.ravel())
#Prediksi data uji
y_pred_DTR_test = Model_DTR.predict(X_test)
y_pred_DTR_test.reshape(-1, 1)
y_pred_DTR_test = pd.DataFrame(y_pred_DTR_test)
y_pred_DTR_test
```

```
y_pred_RFR_test = Model_RFR.predict(X_test)
y_pred_RFR_test.reshape(-1, 1)
y_pred_RFR_test = pd.DataFrame(y_pred_RFR_test)
y_pred_RFR_test
#Evaluasi Hasil Prediksi
score_mse_dt = mean_squared_error(y_test, y_pred_DTR_test)
score_rmse_dt = np.sqrt(mean_squared_error(y_test, y_pred_DTR_test))
score_mae_dt = mean_absolute_error(y_test, y_pred_DTR_test)
print(f'MSE Score on Test set DECISION TREE: {score_mse_dt}')
print(f'RMSE Score on Test set DECISION TREE: {score_rmse_dt}')
print(f'MAE Score on Test set DECISION TREE: {score_mae_dt}')
score_mse_rf = mean_squared_error(y_test, y_pred_RFR_test)
score_rmse_rf = np.sqrt(mean_squared_error(y_test, y_pred_RFR_test))
score_mae_rf = mean_absolute_error(y_test, y_pred_RFR_test)
print(f'\nMSE Score on Test set RANDOM FOREST: {score_mse_rf}')
print(f'RMSE Score on Test set RANDOM FOREST: {score_rmse_rf}')
print(f'MAE Score on Test set RANDOM FOREST: {score_mae_rf}')
# denormalisasi data testing
prediksi_asli_DTR = scaler1.inverse_transform(y_pred_DTR_test)
prediksi_asli_RFR = scaler1.inverse_transform(y_pred_RFR_test)
aktual_asli = scaler1.inverse_transform(y_test)
#Visualisasi hasil prediksi
plt.plot(aktual_asli, 'r--', label='Aktual')
plt.plot(prediksi_asli_DTR, 'b', label='Prediksi DTR')
plt.plot(prediksi_asli_RFR, 'g', label='Prediksi RFR')
plt.grid(True)
```

```
plt.xlabel('Jumlah Data')
plt.ylabel('Arus Listrik (mA)')
plt.title('Grafik Data Aktual dan Prediksi')
plt.legend()
plt.show()
#Mencari tren data
df['Arus_Listrik'] = df['Arus_Listrik'].rolling(50).mean()
df['DTR'] = df['DTR'].rolling(50).mean()
df['RFR'] = df['RFR'].rolling(50).mean()
#Visualisasi Tren data aktual dan prediksi
import matplotlib.pyplot as plt
df = df.set_index('Tanggal/Waktu')
df.index = pd.to_datetime(df.index)
df['DTR'].plot(label='Prediksi DTR', style='-', color='blue')
df['RFR'].plot(label='Prediksi RFR', style='-', color='red')
df['Arus_Listrik'].plot(label='Data Aktual', style='-', color='yellow')
plt.xlabel('Waktu')
plt.ylabel('Arus Listrik (mA)')
plt.title('Grafik Tren Data Aktual dan Prediksi Arus Listrik Pada Kondisi
    Lingkungan')
plt.legend()
plt.show()
```

Lampiran 8. Dataset

https://github.com/abif47/Panel_Surya_TA

Lampiran 9. Source Code App

```
import pickle, pandas as pd
from flask import Flask, redirect, request, render_template, jsonify, session, url_for
from flask_sqlalchemy import SQLAlchemy
app = Flask(__name__)
app.secret_key = 'secret'
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///arus.db'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
db = SQLAlchemy(app)
app.app_context().push()
#Create db model
class Arus(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    prediksi_dtr = db.Column(db.String(20), unique=False, nullable=False)
    prediksi_rfr = db.Column(db.String(20), unique=False, nullable=False)
    aktual = db.Column(db.String(20), unique=False, nullable=False)
    def __repr__(self):
        return f'<User(id: {self.id}, prediksi_dtr: {self.prediksi_dtr}, prediksi_rfr: {self.prediksi_rfr}, aktual: {self.aktual})>'
with open("model/model_numpy.pkl", "rb") as model_file:
    model_numpy_RFR = pickle.load(model_file)
with open("model/model_numpy_DTR.pkl", "rb") as model_file:
    model_numpy_DTR = pickle.load(model_file)
@app.route("/", methods=['GET', 'POST'])
def index():
    if request.method == 'POST':

        # Ambil data dari formulir
        kode = request.form['kode']
        api = request.form['api']
        channel = request.form['channel']
```

```
        # Arahkan ke halaman berikutnya
        return redirect(url_for('predict_arus', api=api, channel=channel, kode=kode))
    else:
        return render_template('form.html')
@app.route("/predict", methods=['GET','POST'])
def predict_arus():
    if request.method == 'GET':
        kode = request.args.get('kode')
        api = request.args.get('api')
        channel = request.args.get('channel')
        return render_template('coba.html',kode=kode, api=api, channel=channel)
    elif request.method == 'POST':
        kode = request.args.get('kode')
        api = request.args.get('api')
        channel = request.args.get('channel')
        arus_aktual = request.form.get('arus')
        excluded_field = 'arus'  # Ganti dengan nama field yang ingin diabaikan
        filtered_values = []
        for key, value in request.form.items():
            if key != excluded_field:
                filtered_values.append(float(value))
        columns = ['Suhu', 'Kelembaban', 'Intensitas_Cahaya', 'Konsentrasi_Debu', 'hour', 'day', 'minute', 'arus']
        columns.remove(excluded_field)
        new_data = pd.DataFrame([filtered_values], columns=columns)
        result_RFR = model_numpy_RFR.predict(new_data)
        result_DTR = model_numpy_DTR.predict(new_data)
        Prediksi = Arus(prediksi_dtr=str(result_DTR[0]), prediksi_rfr=str(result_RFR[0]), aktual=str(arus_aktual))
        db.session.add(Prediksi)
        db.session.commit()
```

```
        return
render_template('coba.html',pred_RFR="{:.3f}".format(result_RFR[0]),
                        pred_DTR="{:.3f}".format(result_DTR[0]),kode=kode,
api=api, channel=channel)
@app.route('/get_data')
def get_data():
    data = Arus.query.limit(8).all()
    result = [{'id': item.id,
          'prediksi_dtr': str(item.prediksi_dtr).strip('[]'),
          'prediksi_rfr': str(item.prediksi_rfr).strip('[]'),
          'aktual': item.aktual
         } for item in data]
    return jsonify(result)
if __name__ == '__main__':
    app.run(debug=True)
```

Lampiran 10. Source Code Tampilan *Realtime Monitoring* dan Prediksi Dashboard

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Dashboard Monitoring Solar Panel</title>
    <link                                                    rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css">
    <link                                                    rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min
.js">
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
     <style>
       body {
          font-family: Arial, sans-serif;
```

```
            margin: 20px;
        }
        .chart-container {
            width: 30%; /* Lebar setiap grafik */
            margin: 10px;
            float: left; /* Setiap grafik akan berada di sebelah kiri yang memungkinkan 3 kolom */
        }
        .right {
            float: right;
        }
        .center {
            float: center;
        }
        .clearfix::after {
            content: "";
            clear: both;
            display: table;
        }
        .myform {
            display: none;
        }
        th, td {
            border: 1px solid #ddd; /* Tambahkan garis 1 piksel solid dengan warna abu-abu muda (#ddd) */
            padding: 8px;
            text-align: left;
        }
        th {
            background-color: #f2f2f2; /* Warna latar belakang untuk header */
        }
    </style>
```

```
</head>
<body>
<center><h1>Realtime Monitoring Solar Panel Area {{ kode }}</h1></center>
<div class="chart-container">
   <canvas id="chart1" width="400" height="200"></canvas>
   <center><h5>Suhu (°C)</h5></center>
</div>
<div class="chart-container center">
   <canvas id="chart2" width="400" height="200"></canvas>
   <center><h5>Kelembapan (%)</h5</center>
</div>
<div class="chart-container right">
   <canvas id="chart3" width="400" height="200"></canvas>
   <center><h5>Intensitas Cahaya (Lux)</h5</center>
</div>
<div class="chart-container">
   <canvas id="chart4" width="400" height="200"></canvas>
   <center><h5>Konsentrasi Debu (µg/m³)</h5</center>
</div>
<div class="chart-container right">
   <canvas id="chart5" width="400" height="200"></canvas>
   <center><h5>Arus Listrik (mA)</h5</center>
</div>
<div class="clearfix"></div>
<center>
   <h3>Data Terbaru</h3>
   <table border="1">
   <thead>
      <tr>
         <th>Timestamp</th>
         <th>Suhu</th>
         <th>Kelembapan</th>
```

```
            <th>Intensitas Cahaya</th>
            <th>Konsentrasi Debu</th>
            <th>Arus Listrik</th>
            <th>Prediksi DTR</th>
            <th>Prediksi RFR</th>
        </tr>
    </thead>
    <tbody id="data-container"></tbody>
    </table>
    <br>
    <div class="myform">
    <form id="myForm" action="/predict" method="post">
        <label for="suhu">Suhu:</label>
        <input type="text" id="suhu" name="suhu" required>
        <br>
        <label for="kelembapan">Kelembapan:</label>
        <input type="text" id="kelembapan" name="kelembapan" required>
        <br>
        <label for="cahaya">Intensitas Cahaya:</label>
        <input type="text" id="cahaya" name="cahaya" required>
        <br>
        <label for="debu">Konsentrasi Debu:</label>
        <input type="text" id="debu" name="debu" required>
        <br>
        <label for="jam">Jam:</label>
        <input type="text" id="jam" name="jam" required>
        <br>
        <label for="hari">Hari:</label>
        <input type="text" id="hari" name="hari" required>
        <br>
        <label for="menit">Menit:</label>
        <input type="text" id="menit" name="menit" required>
```

```
        <br>
        <label for="arus">arus:</label>
        <input type="text" id="arus" name="arus" required>
        <br>
        <button type="submit">Predict</button>
    </form>
    </div>
    <h3>Perbandingan Data Arus Listrik Aktual dan Prediksi (mA)</h3>
    <canvas id="myChart" width="200" height="100"></canvas>
</center>
<script>
// Ganti dengan API Key dan Channel ID ThingSpeak Anda
const apiKey = "{{ api }}";
const channelID = "{{ channel }}";
// URL untuk mengambil data dari ThingSpeak
const apiUrl = `https://api.thingspeak.com/channels/${channelID}/feeds.json?api_key=${apiKey}&results=10`;
// Fungsi untuk mengambil data dari ThingSpeak dan menampilkan grafik
function fetchData(chartId, field) {
    fetch(apiUrl)
        .then(response => response.json())
        .then(data => {
            // Ambil data dari respons JSON
            const feeds = data.feeds;
            // Pisahkan data ke dalam array untuk field tertentu dan waktu
            const fieldData = feeds.map(feed => feed[field]);
            const timeData = feeds.map(feed => {
                const time = new Date(feed.created_at);
                return `${time.getHours()}:${time.getMinutes()}`;
            });
            // Buat grafik menggunakan Chart.js
```

```
        var ctx = document.getElementById(chartId).getContext('2d');
        var myChart = new Chart(ctx, {
            type: 'line',
            data: {
                labels: timeData, // Label waktu pada sumbu x
                datasets: [
                    { label: field, data: fieldData, borderColor: 'rgba(75, 192, 192, 1)',
borderWidth: 1, fill: false }
                ]
            },
            options: {
                scales: {
                    x: { type: 'category', position: 'bottom' },
                    y: { beginAtZero: true }
                }
            }
        });
    })
    .catch(error => console.error("Error fetching data:", error));
}
// Panggil fetchData untuk setiap field dan setiap grafik saat halaman dimuat
pertama kali
fetchData('chart1', 'field1');
fetchData('chart2', 'field2');
fetchData('chart3', 'field3');
fetchData('chart4', 'field4');
fetchData('chart5', 'field5');
const                                api1Data                                =
`https://api.thingspeak.com/channels/${channelID}/feeds.json?api_key=${apiK
ey}&results=1`;
function UpdateData() {
  fetch(api1Data)
```

```
.then(response => response.json())
.then(data => {
    // Ambil data dari respons JSON
    const feed = data.feeds[0];
    // Ubah timestamp ke zona waktu setempat
    const localTimestamp = new Date(feed.created_at).toLocaleString();
    // Buat string HTML untuk menampilkan data dalam tabel
    let htmlString = `
        <tr>
            <td>${localTimestamp}</td>
            <td>${feed.field1} °C</td>
            <td>${feed.field2} %</td>
            <td>${feed.field3} Lux</td>
            <td>${feed.field4} µg/m³</td>
            <td>${feed.field5} mA</td>
            <td>{{pred_DTR}} mA</td>
            <td>{{pred_RFR}} mA</td>
        </tr>
    `;
    const nilaiTglWaktu = `${localTimestamp}`;
    // Membagi nilai menjadi tanggal dan waktu
    var [tanggal, waktu] = nilaiTglWaktu.split(', ');
    // Memisahkan nilai tanggal menjadi hari, bulan, dan tahun
    var [hari, bulan, tahun] = tanggal.split('/');
    // Memisahkan nilai waktu menjadi jam, menit, dan detik
    var [jam, menit, detik] = waktu.split('.');
    // Tampilkan data dalam elemen dengan ID "data-container"
    document.getElementById("data-container").innerHTML = htmlString;
    document.getElementById('suhu').value = `${feed.field1}`;
    document.getElementById('kelembapan').value = `${feed.field2}`;
    document.getElementById('cahaya').value = `${feed.field3}`;
    document.getElementById('debu').value = `${feed.field4}`;
```

```
            document.getElementById('arus').value = `${feed.field5}`;
            document.getElementById('jam').value = jam;
            document.getElementById('hari').value = hari;
            document.getElementById('menit').value = menit;
        })
        .catch(error => console.error("Error fetching data:", error));
}
// Panggil fetchData saat halaman dimuat pertama kali
UpdateData();
// Fungsi untuk mengirim formulir
function submitForm() {
     document.getElementById("myForm").submit();
   }
   // Set interval untuk menjalankan fungsi submitForm setiap 10 menit (600000
milidetik)
setInterval(submitForm, 610000);
      // Ambil data dari server menggunakan AJAX
      var xhr = new XMLHttpRequest();
       xhr.onreadystatechange = function () {
          if (xhr.readyState == 4 && xhr.status == 200) {
              var data = JSON.parse(xhr.responseText);
              // Buat array untuk label dan data
              var labels = [];
              var y1Values = [];
              var y2Values = [];
              var y3Values = [];
              // Ambil data dari array JSON
              data.forEach(function (item) {
                  labels.push(item.id);
                  y1Values.push(item.prediksi_dtr);
                  y2Values.push(item.prediksi_rfr);
                  y3Values.push(item.aktual);
```

```
});
// Gambar grafik menggunakan Chart.js
var ctx = document.getElementById('myChart').getContext('2d');
var myChart = new Chart(ctx, {
    type: 'line',
    data: {
        labels: labels,
        datasets: [
            {
                label: 'DTR',
                data: y1Values,
                borderColor: 'rgba(255, 0, 0, 1)',
                borderWidth: 1,
                fill: false
            },
            {
                label: 'RFR',
                data: y2Values,
                borderColor: 'rgba(0, 255, 0, 1)',
                borderWidth: 1,
                fill: false
            },
            {
                label: 'Aktual',
                data: y3Values,
                borderColor: 'rgba(0, 0, 255, 1)',
                borderWidth: 1,
                fill: false
            }
        ]
    },
    options: {
```

```
                scales: {
                    y: {
                        beginAtZero: true
                    }
                }
            }
        });
    }
};
xhr.open('GET', '/get_data', true);
xhr.send();
</script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.j
s"></script>
</body>
</html>
```

Lampiran 11. Source Code Tampilan utama *Dashboard*

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Dashboard Monitoring Solar Panel</title>
    <link                                                    rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css">
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <style>
        body {
            font-family: Arial, sans-serif;
```

```
    margin: 20px;
    background-color: #f8f9fa; /* Warna latar belakang keseluruhan */
}
.navbar {
    background-color: #007bff; /* Warna latar belakang navbar */
    padding: 10px 0;
}
.navbar-brand {
    color: #fff; /* Warna teks navbar */
    font-size: 36px;
    font-weight: bold;
    margin-left: auto;
    margin-right: auto;
    text-align: center; /* Teks ditengahkan secara horizontal */
}
table {
    margin-top: 20px;
    border-collapse: collapse;
    width: 100%;
    margin-left: auto;
    margin-right: auto;
}
th, td {
    border: 1px solid #dddddd;
    text-align: left;
    padding: 8px;
}
th {
    background-color: #f2f2f2;
}
tr:hover {
    background-color: #f5f5f5;
```

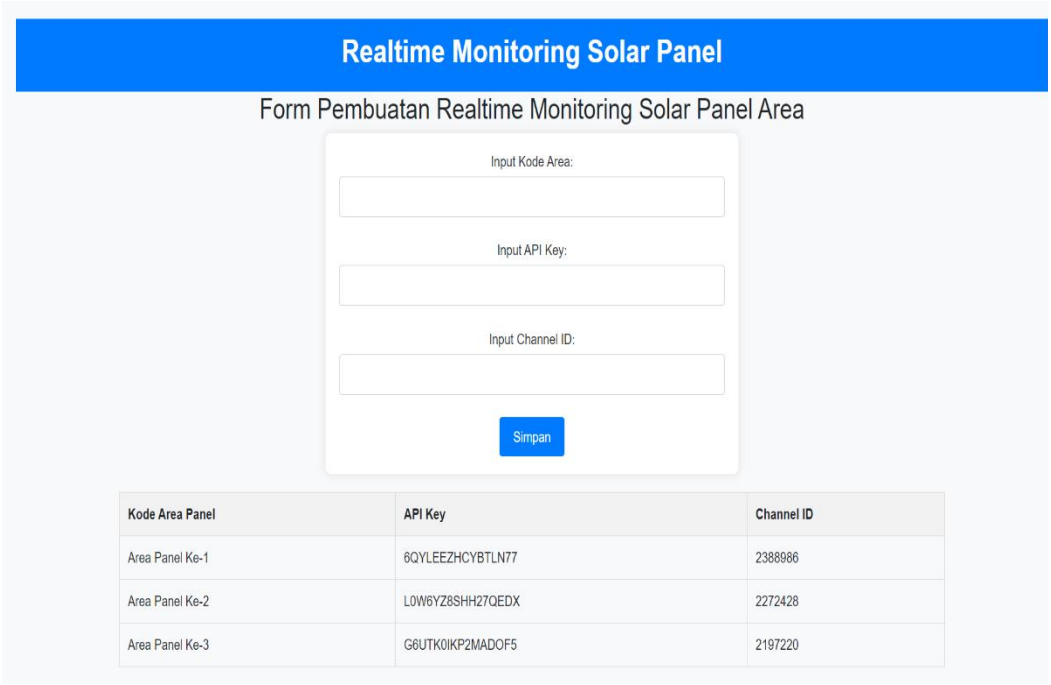
```
            cursor: pointer;
        }
        form {
            width: 50%;
        }
    </style>
</head>
<body>
    <nav class="navbar">
        <div class="container-fluid">
            <h1 class="navbar-brand">Realtime Monitoring Solar Panel</h1>
        </div>
    </nav>
    <center>
        <h2>Form Pembuatan Realtime Monitoring Solar Panel Area</h2>
        <form id="predictForm" style="width: 50%;">
            <div style="margin-bottom: 10px;">
                <label for="kode" style="display: inline-block; width: 150px;">Input
Kode Area:</label>
                <input type="text" id="kode" name="kode" required>
            </div>
            <div style="margin-bottom: 10px;">
                <label for="api" style="display: inline-block; width: 150px;">Input
API Key:</label>
                <input type="text" id="api" name="api" required>
            </div>
            <div style="margin-bottom: 10px;">
                <label for="channel" style="display: inline-block; width:
150px;">Input Channel ID:</label>
                <input type="text" id="channel" name="channel" required>
            </div>
            <button type="submit">Kirim</button>
```

```
    </form>
  </center>
  <table id="dataTable" style="display: none;">
    <tr>
      <th>Kode Area Panel</th>
      <th>API Key</th>
      <th>Channel ID</th>
    </tr>
  </table>
  <script>
    document.getElementById("predictForm").addEventListener("submit",
function(event) {
      event.preventDefault(); // Mencegah pengiriman formulir secara default
      const formData = new
FormData(document.getElementById("predictForm"));
      // Menambahkan data ke dalam tabel
      const kode = formData.get('kode');
      const api = formData.get('api');
      const channelID = formData.get('channel');
      const dataTable = document.getElementById("dataTable");
      const newRow = dataTable.insertRow(-1);
      const cell1 = newRow.insertCell(0);
      const cell2 = newRow.insertCell(1);
      const cell3 = newRow.insertCell(2);
      cell1.textContent = `Area Panel Ke-${kode}`;
      cell2.textContent = api;
      cell3.textContent = channelID;
      // Menambahkan event listener untuk setiap baris tabel
      newRow.addEventListener("click", function() {
        window.location.href =
`/predict?kode=${kode}&api=${api}&channel=${channelID}`;
      });
```

```
            // Mengosongkan nilai formulir setelah ditambahkan ke tabel
            document.getElementById("kode").value = "";
            document.getElementById("api").value = "";
            document.getElementById("channel").value = "";
            // Setel properti display tabel menjadi block setelah menambahkan data
            dataTable.style.display = "table";
        });
    </script>
</body>
</html>
```

Lampiran 12. Alat dan Bahan

Lampiran 13. *Dashboard Realtime Monitoring* dan Prediksi



Kode Area Panel	API Key	Channel ID
Area Panel Ke-1	6QYLEEZHCYBTLN77	2388986
Area Panel Ke-2	L0W6YZ8SHH27QEDX	2272428
Area Panel Ke-3	G6UTK0IKP2MADOF5	2197220

LEMBAR PERBAIKAN SKRIPSI

"PERBANDINGAN METODE DECISION TREE REGRESSION DAN RANDOM FOREST REGRESSION UNTUK PEMELIHARAAN PREDIKTIF PADA PEMBANGKIT LISTRIK TENAGA SURYA. (STUDI KASUS : PREDIKSI DEGRADASI DAYA AKIBAT DEBU PADA PANEL SURYA)"

OLEH:

MOH ABIB SAFAQDILLAH
D121191026

Skripsi ini telah dipertahankan pada Ujian Akhir Sarjana tanggal 03 Mei 2024.

Telah dilakukan perbaikan penulisan dan isi skripsi berdasarkan usulan dari penguji dan pembimbing skripsi.

Persetujuan perbaikan oleh tim penguji:

	Nama	Tanda Tangan
Ketua	Dr. Eng. Zulkifli Tahir, S.T., M.Sc.	
Sekretaris	Prof. Dr. Ir. Syafruddin Syarif, M.T.	
Anggota	Ir. Christoforus Yohannes, M.T.	
	Iqra Aswad, S.T., M.T	

Persetujuan Perbaikan oleh pembimbing:

Pembimbing	Nama	Tanda Tangan
I	Dr. Eng. Zulkifli Tahir, S.T., M.Sc..	
II	Prof. Dr. Ir. Syafruddin Syarif, M.T.	