DAFTAR PUSTAKA

[1] Y. Devireddy, M. Sanke, K. Ragi, H. Maganti and A. Thorlikonda, "Real Time Energy Monitoring And Controlling System using IoT," 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, 2021, pp. 157-161, doi: 10.1109/ICOSEC51865.2021.959 1867.

[2] A. -W. O. Muhammed et al., "Design and Implementation of an IoT Based Home Energy Monitoring System," 2022 5th Information Technology for Education and Development (ITED), Abuja, Nigeria, 2022, pp.1-7,doi:10.1109/ITED56637.2022.1005 1192.

[3] Ruly Naufaldi Kurniawan, Rendy Munadi, Imam Hedi Santoso. 2021. Sistem Monitoring kWh Meter Dengan Media Komunikasi *WhatsApp* Berbasis IoT. Universitas Telkom Bandung

[4] *Lenni, Isfan Achmad Fadhillah*.2021. Desain sistem monitoring kWh *meter* dengan media komunikasi ESP32 dan Blynk. Universitas Muhammadiyah Tangerang

[5] Rio Guntur Dany Rafi. 2023. Rancang Bangun kWh Meter Digital Menggunakan *Internet of Things.* Politeknik Negeri Cilacap.

[6] Putu Ardi Widyatmika, Ni Putu Ayu Widyanata Indrawati, Wayan Wahyu Adi Prastya, Ketut Darminta, Gde Nyomsn Sangka, Anak Agung Ngurah Gde Sapteka.2021. Perbandingan Kinerja Arduino Uno dengan ESP32 Terhadap Pengukuran Arus dan Tegangan. Politeknik Negeri Bali.

[7] Innovators Guru. "PZEM-004T V3.0: Overview, Specifications, and Usage." *Innovators Guru*. https://innovatorsguru.com/pzem-004t-v3/. Accessed 5 Agustus. 2024.

[8] Gandhi Yuda. Penampilan tulisan berjalan berbasis mikrokontroler. Fakultas Teknik Universitas Negeri Semarang.

[9] Elangsakti, “Pengertian, Fungsi, Prinsip, dan Cara Kerja Relay”, Online.Available:https://www.elangsakti.com/2013/03/pengertian-fungsi-prinsip-dan-cara.html#comment-form.

[10] Merdeka.com, “Mengenal Fungsi MCB pada Instalasi Listrik, Berikut Pengertian dan Jenisnya.” Online. Available: https: //www.merdeka.com/jabar/mengenal-fungsi-mcb-pada instalasi - listrik-berikut-pengertian-dan -jenisnya kln.html?Page =6.

[11] Circuit Digest. "AC to DC Converter Circuit Diagram." *Circuit Digest*,https://circuitdigest.com/electronic-circuits/ac-to-dc converter- circuit-diagram. Accessed 5 Agustus. 2024.

[12] A. K, "Pengertian HTTP: Fungsi, Cara Kerja, dan Manfaat."Online.Available:https://www.gramedia.com /literasi/ pengertian-http/.

[13] Maghfiroh, Muhammad Alfiatul - 173310006 (2020) "SISTEM PRESENSI DENGAN RFID BERBASIS NODEMCU DITERAPKAN UNTUK PONDOK PESANTREN PPMa NURBAITURRAHMAN". Diploma thesis, STMIK AKAKOM Yogyakarta.

[14] Biznetgio, "Mengenal Laravel." [Online]. Available: https://www.biznetgio.com/news/apa-itu-laravel

[15] C. Shah, "MySQL," *A Hands-On Introd. to Data Sci.*, pp. 187-206, 2020, doi: 10.1017/9781108560412.008.

[16] B. A. B. Ii, "Landasan Teori Visual Code," pp. 6–18, 2011.

[17] Muhammad Robith Adani, "Memahami Konsep Penggunaan Xampp untuk Kebutuhan Development." [Online]. Available: https://www.sekawanmedia.co.id/blog/apa-itu-xampp/.

[18] erintafifah, "Mengenal Perangkat Lunak Arduino IDE," 8 oktober2021.[Online].Available:https://www.kmtech.id/post/mengenal-perangkat-lunak-arduino-ide

LAMPIRAN A
Program Arduino IDE Esp32

```
#include <WiFi.h>
#include <PZEM004Tv30.h>
#include <ArduinoJson.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <HTTPClient.h>
#include <EEPROM.h>
#include <WiFiClientSecure.h>

#if !defined(PZEM_RX_PIN) &&
!defined(PZEM_TX_PIN)
#define PZEM_RX_PIN 16
#define PZEM_TX_PIN 17
#endif

#if !defined(PZEM_SERIAL)
#define PZEM_SERIAL Serial2
#endif

PZEM004Tv30 pzem(PZEM_SERIAL, PZEM_RX_PIN,
PZEM_TX_PIN);
LiquidCrystal_I2C lcd(0x27, 20, 4);  // Alamat
I2C, kolom, baris

const char *ssid = "Stevy";
const char *password = "11111111";

const char *host = "stevy.pantaulistrik.com";
// Alamat IP server Laravel
const int httpPort = 80;           // Port
Laravel

const char *relayEndpoint = "/kontrol-relay-
value";

const char *tokenEndpoint = "/get-token";
```

```
unsigned long wktsend = 0;
unsigned long interval_send = 7000;
unsigned long wktkontrol = 0;
unsigned long interval_kontrol = 1000;

float v, i, p;
long rp;
float dy_aktif, dy_reaktif, dy_semu, frekuensi,
energi, biaya, token, kwh_terakhir,
pulsa_listrik, kwh_terakhir_tersimpan,
pulsa_sisa;

float storeLog = 0;
float storePulsa = 0;
float rp_per_detik;
const int relayPin = 5;

#define EEPROM_SIZE 4
const float tarifPerKwh = 605.0;  // Tarif
listrik per kWh (contoh: Rp 605 per kWh)

void setup() {
  Serial.begin(115200);
  lcd.init();
  lcd.backlight();

  pinMode(relayPin, OUTPUT);
  digitalWrite(relayPin, LOW);

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  Serial.print("Connecting");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.print("Connected to ");
  Serial.println(ssid);
```

```
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  EEPROM.begin(EEPROM_SIZE);

  storeLog = EEPROM.read(7);  // Membaca kWh
  terakhir dari EEPROM
  storePulsa = EEPROM.read(11);

  Serial.print("kWh Terakhir= ");
  Serial.println(storeLog);
  Serial.print("Pulsa Terakhir= ");
  Serial.println(storePulsa);
}

void loop() {


  baca_pzem();

  tampillcd();

  unsigned long wktSekarang = millis();
  if (wktSekarang - wktsend > interval_send) {
    kirim();
    wktsend = wktSekarang;
  }

  kontrolRelay();
  getToken();

  useToken();

  delay(1000);
}

void baca_pzem() {
  // Membaca tegangan dari PZEM
  v = pzem.voltage();
  if (isnan(v)) {
```

```
    v = 0.0; // Handle error jika pembacaan
gagal
  }

  // Membaca arus dari PZEM
  i = pzem.current();
  if (isnan(i)) {
    i = 0.0; // Handle error jika pembacaan
gagal
  }

  // Membaca daya aktif dari PZEM
  p = pzem.power();
  if (isnan(p)) {
    p = 0.0; // Handle error jika pembacaan
gagal
  }

  // Menghitung biaya per detik
  rp_per_detik = (p / 1000.0) * (tarifPerKwh /
3600.0); // Biaya per detik

  // Update energi dan biaya total
  energi = pzem.energy();
  biaya += rp_per_detik;
}

void tampillcd() {
  lcd.clear();

  lcd.setCursor(0, 0);
  lcd.print("V:");
  lcd.print(v);
  lcd.print("V");

  lcd.setCursor(0, 1);
  lcd.print("I:");
  lcd.print(i);
  lcd.print("A");
```

```
  lcd.setCursor(0, 2);
  lcd.print("P:");
  lcd.print(p);
  lcd.print("W");
  lcd.setCursor(10,2);
  lcd.print("kWh:");
  lcd.print(energi, 3);  // Menampilkan energi
dalam kWh dengan 3 digit      desimal

  lcd.setCursor(0, 3);
  lcd.print("RP:");
  lcd.print(biaya, 2);  // Menampilkan biaya
dalam Rupiah dengan 2 digit desimal
}

//Kirim nilai sensor PZEM ke APlikasi lokal
//void kirim() {
//  WiFiClient client;
//
//  if (!client.connect(host, httpPort)) {
//    Serial.println("Connection failed");
//    return;
//  }
//
//  String url = String("/stevy123/") + v + "/"
+ i + "/" + dy_aktif + "/" + dy_reaktif + "/" +
dy_semu + "/" + frekuensi + "/" + energi + "/" +
biaya+ "/" + pulsa_listrik;
//
//  Serial.print("\nRequesting URL: ");
//  Serial.println(url);
//
//  client.print(String("GET ") + url + "
HTTP/1.1\r\n" +
// "Host: " + host + "\r\n" +
// "Connection: close\r\n\r\n");
//
//  while (client.connected() ||
client.available()) {
//    if (client.available()) {
```

```
//      String line =
client.readStringUntil('\n');
//      Serial.println(line);
//    }
//  }
//  client.stop();
//}

void kirim() {
  WiFiClientSecure client;

  client.setInsecure();  // Use this for testing
purposes

  if (!client.connect(host, 443)) {  // Use port
443 for SSL
    Serial.println("Connection failed");
    return;
  }

 String url = String("/stevy123/") + v + "/" + i
+ "/"   + p+ "/"+ energi + "/" + biaya + "/" +
pulsa_listrik;

  Serial.print("\nRequesting URL: ");
  Serial.println(url);

  client.print(String("GET ") + url + "
HTTP/1.1\r\n" +
               "Host: " + host + "\r\n" +
               "Connection: close\r\n\r\n");

  while (client.connected() ||
client.available()) {
    if (client.available()) {
      String line =
client.readStringUntil('\n');
      Serial.println(line);
    }
  }
```

```
  client.stop();
}

// Mengambil nilai state on/off relay dari
Aplikasi
void kontrolRelay() {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;

    // Request URL untuk kontrol relay
    String serverPath = String("https://") +
host + relayEndpoint;

    Serial.print("Requesting URL: ");
    Serial.println(serverPath);

    http.begin(serverPath);
    int httpResponseCode = http.GET();
    if (httpResponseCode > 0) {
      String response = http.getString();
      Serial.println("Relay Control Response: "
+ response);

      // Parsing JSON response
      DynamicJsonDocument doc(1024);
      DeserializationError error =
deserializeJson(doc, response);

      if (!error) {
        String relayStatus =
doc["status"].as<String>();

        if (relayStatus == "ON") {
          digitalWrite(relayPin, HIGH);  //
Menyalakan relay
          Serial.println("Relay ON");
        } else if (relayStatus == "OFF") {
          digitalWrite(relayPin, LOW);   //
Mematikan relay
          Serial.println("Relay OFF");
```

```
        } else {
          Serial.println("Invalid status in
response");
        }
      } else {
        Serial.print("Failed to parse JSON: ");
        Serial.println(error.c_str());
      }
    } else {
      Serial.print("Error on getting relay
control: ");
      Serial.println(httpResponseCode);
    }

    http.end();
  } else {
    Serial.println("WiFi not connected");
  }
}

//Mengambil nilai token dari Aplikasi
void getToken() {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http_token;

    // Request URL untuk kontrol relay
    String serverPath2 = String("https://") +
host + tokenEndpoint;

    Serial.print("Requesting URL: ");
    Serial.println(serverPath2);

    http_token.begin(serverPath2);
    int httpResponseCode = http_token.GET();

    if (httpResponseCode > 0) {
      String response = http_token.getString();
      Serial.println("Response Token: " +
response);
```

```
      // Parsing JSON response
      DynamicJsonDocument doc(1024);
      DeserializationError error =
deserializeJson(doc, response);

      if (!error) {
        token =
doc["history_token"].as<float>();
        Serial.print("Token received from
server: ");
        Serial.println(token);

      } else {
        Serial.print("Failed to parse JSON: ");
        Serial.println(error.c_str());
      }
    } else {
      Serial.print("Error on getting token: ");
      Serial.println(httpResponseCode);
    }

    http_token.end();
  } else {
    Serial.println("WiFi not connected");
  }
}


//logika penggunaan token
void useToken() {

 // Update the stored token with the received
value
  pulsa_listrik = token;

  // Update kWh terakhir dengan energi yang baru
  kwh_terakhir = storeLog + energi;

  // Hitung pulsa yang tersisa
  pulsa_listrik -= kwh_terakhir;
```

```
  // Simpan nilai terakhir ke EEPROM
  EEPROM.write(7, kwh_terakhir);
  EEPROM.write(11, pulsa_listrik);
  EEPROM.commit();


  Serial.print("Updated token: ");
  Serial.println(token);
  Serial.print("Updated kWh: ");
  Serial.println(kwh_terakhir);
  Serial.print("Remaining Pulsa: ");
  Serial.println(pulsa_listrik);

}
```

LAMPIRAN B
Program Tampilan *Website*

```
@extends('layouts.horizontal_dashboard.app')

@section('content')

<div class="container-xxl flex-grow-1 container-
p-y">
  <h4 class="py-3 mb-4"><span class="text-muted
fw-light"> </span> Dashboard Monitoring </h4>

  <!-- card mini banner -->
  <div class="row mb-4">
    <div class="col-sm-6 col-lg-3 mb-4">
      <div class="card card-border-shadow-info">
        <div class="card-body">
          <div class="d-flex align-items-center
mb-2 pb-1">
            <div class="avatar me-2">
              <span class="avatar-initial
rounded bg-label-info"><i class="ti ti-home-bolt
ti-md"></i></span>
            </div>
            <h4 class="ms-1 mb-0"
id="energiCard">NaN</h4>
          </div>
          <p class="mb-1">Penggunaan Energi
Listrik</p>
          <p class="mb-0">
            <small class="text-muted">Total
penggunaan realtime energi listrik</small>
          </p>
        </div>
      </div>
    </div>
    <div class="col-sm-6 col-lg-3 mb-4">
      <div class="card card-border-shadow-
warning">
        <div class="card-body">
```

```
          <div class="d-flex align-items-center
mb-2 pb-1">
            <div class="avatar me-2">
              <span class="avatar-initial
rounded bg-label-warning"><i class="ti ti-plug-
connected ti-md"></i></span>
            </div>
            <h4 class="ms-1 mb-0"
id="teganganCard">NaN</h4>
          </div>
          <p class="mb-1">Tegangan Listrik</p>
          <p class="mb-0">
            <small class="text-muted">Tegangan
realtime sistem</small>
          </p>
        </div>
      </div>
    </div>
    <div class="col-sm-6 col-lg-3 mb-4">
      <div class="card card-border-shadow-
danger">
        <div class="card-body">
          <div class="d-flex align-items-center
mb-2 pb-1">
            <div class="avatar me-2">
              <span class="avatar-initial
rounded bg-label-danger"><i class="ti ti-bolt
ti-md"></i></span>
            </div>
            <h4 class="ms-1 mb-0"
id="arusCard">NaN</h4>
          </div>
          <p class="mb-1">Arus Listrik</p>
          <p class="mb-0">
            <small class="text-muted">Arus
realtime sistem</small>
          </p>
        </div>
      </div>
    </div>
```

```
    <div class="col-sm-6 col-lg-3 mb-4">
      <div class="card card-border-shadow-
primary">
        <div class="card-body">
          <div class="d-flex align-items-center
mb-2 pb-1">
            <div class="avatar me-2">
              <span class="avatar-initial
rounded bg-label-primary"><i class="ti ti-
currency-dollar ti-md"></i></span>
            </div>
            <h4 class="ms-1 mb-0"
id="sisaCard">NaN</h4>
          </div>
          <p class="mb-1">Sisa Token</p>
          <p class="mb-0">
            <small class="text-muted">Realtime
token listrik</small>
          </p>
        </div>
      </div>
    </div>
  </div>
  <!-- end card mini banner  -->
  <div class="row">
    <!-- Line Area Charts -->
    <div class="col-12 col-xl-8 mb-5">
      <div class="card">
        <div class="card-header header-
elements">
          <h5 class="card-title mb-0">Penggunaan
Energi Listrik</h5>
        </div>
        <div class="card-body pt-2">
          <canvas id="energiListrikChart"
class="chartjs" data-height="450"></canvas>
        </div>
      </div>
    </div>
    <!-- /Line Area Charts -->
```

```
    <!-- table penggunaan daya  -->
    <div class="col-12 col-xl-4 mb-5">
      <div class="card h-100">
        <h5 class="card-header">Data Penggunaan
Energi Listrik</h5>
        <div class="table-responsive text-
nowrap">
          <table class="table">
            <thead>
              <tr>
                <th># </th>
                <th><i class="fa-solid fa-plug-
circle-bolt"></i> Energi Listrik</th>
                <th><i class="fa-solid fa-
clock"></i> Waktu</th>
              </tr>
            </thead>
            <tbody class="table-border-bottom-0"
id="showTabelEnergi">
              <!-- <tr style="font-size: 13px;">
                                <td>1</td>
                                <td>
                                    <span
class="fw-medium">0.0001 kWh</span>
                  </td>
                <td>5 minute ago</td>
              </tr> -->
            </tbody>
          </table>
        </div>
      </div>
    </div>
    <!-- end table penggunaan daya -->

    <!-- grafik tegangan dan arus -->
    <div class="col-12 col-xl-6 mb-5">
      <div class="card">
        <div class="card-header header-elements
d-flex justify-content-between align-items-
center">
```

```
          <h5 class="card-title mb-0">Grafik
Realtime Tegangan Listrik</h5>
          <h5 class="card-title mb-0"><span
style="text-decoration: underline;"
id="valueV">0</span> V</h5>
        </div>
        <div class="card-body pt-2">
          <canvas id="teganganListrikChart"
class="chartjs" data-height="450"></canvas>
        </div>
      </div>
    </div>
    <div class="col-12 col-xl-6 mb-5">
      <div class="card">
        <div class="card-header header-elements
d-flex justify-content-between align-items-
center">
          <h5 class="card-title mb-0">Grafik
Realtime Arus Listrik</h5>
          <h5 class="card-title mb-0"><span
style="text-decoration: underline;"
id="valueI">0</span> A</h5>
        </div>
        <div class="card-body pt-2">
          <canvas id="arusListrikChart"
class="chartjs" data-height="450"></canvas>
        </div>
      </div>
    </div>
    <div class="col-12 col-xl-6 mb-5">
      <div class="card">
        <div class="card-header header-elements
d-flex justify-content-between align-items-
center">
          <h5 class="card-title mb-0">Grafik
Realtime Daya</h5>
          <h5 class="card-title mb-0"><span
style="text-decoration: underline;"
id="valueP">0</span> Watt</h5>
        </div>
```

```
        <div class="card-body pt-2">
          <canvas id="dayaAktifChart"
class="chartjs" data-height="450"></canvas>
        </div>
      </div>
    </div>
  </div>
</div>

@endsection

@push('plugin-script')
<!-- <script
src="https://cdn.jsdelivr.net/npm/chart.js@2.9.4
/dist/Chart.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/chartjs-
plugin-datalabels@0.7.0"></script> -->
<script
src="https://cdn.jsdelivr.net/npm/chart.js@3.0.0
/dist/chart.min.js"></script>
<script
src="https://cdn.jsdelivr.net/npm/chartjs-
plugin-datalabels@2.0.0"></script>
@endpush

@push('script')
<script>
  'use strict';

  (function() {
    Chart.register(ChartDataLabels);
    // Color Variables
    const yellowColor = '#ffe800',
      cyanColor = '#28dac6',
      orangeColor = '#FF8132',
      orangeLightColor = '#FDAC34',
      oceanBlueColor = '#299AFF',
      greyColor = '#4F5D70',
      greyLightColor = '#EDF1F4';
```

```
    let cardColor, headingColor, labelColor,
borderColor, legendColor, blueColor,
blueLightColor;
    let redColor, redLightColor, amberColor,
amberLightColor, greenLightColor, greenColor,
purpleColor, purpleLightColor;

    if (isDarkStyle) {
      cardColor = '#2f3349';
      legendColor = '#b6bee3';
      headingColor = '#cfd3ec';
      labelColor = '#cbd5e1';
      borderColor = '#434968';
      blueColor = '#4d7cff';
      blueLightColor = '#96b9ff';
      redColor = '#f87171';
      redLightColor = '#fecaca';
      amberColor = '#f59e0b';
      amberLightColor = '#fed7aa';
      greenLightColor = '#7CC674';
      greenColor = '#38812F';
      purpleColor = '#8481DD';
      purpleLightColor = '#B2B0EA';

    } else {
      cardColor = '#fff';
      legendColor = '#6f6b7d';
      headingColor = '#5d596c';
      labelColor = '#a5a3ae';
      borderColor = '#dbdade';
      blueColor = '#007bff';
      blueLightColor = '#cce5ff';
      redColor = '#dc3545';
      redLightColor = '#f8d7da';
      amberColor = '#F0AB00';
      amberLightColor = '#F9E0A2';
      greenLightColor = '#BDE2B9';
      greenColor = '#38812F';
      purpleColor = '#5752D1';
```

```
      purpleLightColor = '#B2B0EA';
    }

    // Set height according to their data-height
    // ----------------------------------------
----------------------------
    const chartList =
document.querySelectorAll('.chartjs');
    chartList.forEach(function(chartListItem) {
      chartListItem.height =
chartListItem.dataset.height;
    });

    const energiListrikChart =
document.getElementById('energiListrikChart');
    if (energiListrikChart) {
      const energiListrikChartVar = new
Chart(energiListrikChart, {
        type: 'line',
        data: {
          labels: [],
          datasets: [{
            label: 'Energi Listrik ( kWh )',
            data: [],
            tension: 0,
            fill: 'start',
            backgroundColor: blueLightColor,
            pointStyle: 'circle',
            borderColor: blueColor,
            pointRadius: 0.5,
            pointHoverRadius: 5,
            pointHoverBorderWidth: 5,
            pointBorderColor: 'transparent',
            pointHoverBackgroundColor:
blueLightColor,
            pointHoverBorderColor: cardColor,
            datalabels: {
              align: 'start',
              anchor: 'start',
              color: '#000',
```

```
      backgroundColor: '#f5f5f5',
      borderColor: '#000',
      borderRadius: 32,
      borderWidth: 1,
      font: {
        weight: 'bold',
        size: 8
      },
      offset: 1,
      padding: 5,
      textAlign: 'center'
    }
  }, ]
},
// plugins: [ChartDataLabels],
options: {
  responsive: true,
  maintainAspectRatio: false,
  plugins: {
    legend: {
      position: 'top',
      rtl: isRtl,
      align: 'start',
      labels: {
        usePointStyle: true,
        padding: 35,
        boxWidth: 6,
        boxHeight: 6,
        color: legendColor
      }
    },
    tooltip: {
      // Updated default tooltip UI
      rtl: isRtl,
      backgroundColor: cardColor,
      titleColor: headingColor,
      bodyColor: legendColor,
      borderWidth: 1,
      borderColor: borderColor
    }
```

```
          },
          scales: {
            x: {
              grid: {
                color: 'transparent',
                borderColor: borderColor
              },
              ticks: {
                color: labelColor
              }
            },
            y: {
              grid: {
                color: 'transparent',
                borderColor: borderColor
              },
              ticks: {
                stepSize: 100,
                color: labelColor
              }
            }
          }
        }
      });

      let updateChartEnergiListrik = function()
{
        $.ajax({
          type: "GET",
          dataType: "json",
          url: "{{
route('collection.data.sensor') }}",
          headers: {
            'X-CSRF-TOKEN': $('meta[name="csrf-
token"]').attr('content')
          },
          success: function(data) {
            //  console.log(data);
            energiListrikChartVar.data.labels =
data.waktu;
```

```
            energiListrikChartVar.data.datasets[
0].data = data.energi;
            energiListrikChartVar.update();
          },
          error: function(data) {
            //  console.log(data);
          }
        });
      }
      updateChartEnergiListrik();
      setInterval(() => {
        updateChartEnergiListrik();
      }, 5000);
    }

    const teganganListrikChart =
document.getElementById('teganganListrikChart');
    if (teganganListrikChart) {
      const teganganListrikChartVar = new
Chart(teganganListrikChart, {
        type: 'line',
        data: {
          labels: [],
          datasets: [{
            label: 'Tegangan Listrik ( V )',
            labelColor: amberColor,
            data: [],
            tension: 0,
            fill: 'start',
            backgroundColor: amberLightColor,
            pointStyle: 'circle',
            borderColor: amberColor,
            pointRadius: 0.5,
            pointHoverRadius: 5,
            pointHoverBorderWidth: 5,
            pointBorderColor: 'transparent',
            pointHoverBackgroundColor:
blueColor,
            pointHoverBorderColor: cardColor,
            datalabels: {
```

```
        align: 'start',
        anchor: 'start',
        color: '#000',
        backgroundColor: '#f5f5f5',
        borderColor: '#000',
        borderRadius: 32,
        borderWidth: 1,
        font: {
          weight: 'bold',
          size: 8
        },
        offset: 1,
        padding: 5,
        textAlign: 'center'
      }
    }, ]
  },
  options: {
    responsive: true,
    maintainAspectRatio: false,
    plugins: {
      legend: {
        position: 'top',
        rtl: isRtl,
        align: 'start',
        labels: {
          usePointStyle: true,
          padding: 35,
          boxWidth: 6,
          boxHeight: 6,
          color: legendColor
        }
      },
      tooltip: {
        // Updated default tooltip UI
        rtl: isRtl,
        backgroundColor: cardColor,
        titleColor: headingColor,
        bodyColor: legendColor,
        borderWidth: 1,
```

```
            borderColor: borderColor
          }
        },
        scales: {
          x: {
            grid: {
              color: 'transparent',
              borderColor: borderColor
            },
            ticks: {
              color: labelColor
            }
          },
          y: {
            min: 150,
            max: 300,
            grid: {
              color: 'transparent',
              borderColor: borderColor
            },
            ticks: {
              color: labelColor
            }
          }
        }
      }
    });

    let updateChartTeganganListrik =
function() {
      $.ajax({
        type: "GET",
        dataType: "json",
        url: "{{
route('collection.data.sensor') }}",
        headers: {
          'X-CSRF-TOKEN': $('meta[name="csrf-
token"]').attr('content')
        },
        success: function(data) {
```

```
            //  console.log(data);
            teganganListrikChartVar.data.labels
= data.waktu;
            teganganListrikChartVar.data.dataset
s[0].data = data.tegangan;
            teganganListrikChartVar.update();
          },
          error: function(data) {
            //  console.log(data);
          }
        });
      }
      updateChartTeganganListrik();
      setInterval(() => {
        updateChartTeganganListrik();
      }, 5000);
    }

    const arusListrikChart =
document.getElementById('arusListrikChart');
    if (arusListrikChart) {
      const arusListrikChartVar = new
Chart(arusListrikChart, {
        type: 'line',
        data: {
          labels: [],
          datasets: [{
            label: 'Arus Listrik ( A )',
            data: [],
            tension: 0,
            fill: 'start',
            backgroundColor: redLightColor,
            pointStyle: 'circle',
            borderColor: redColor,
            pointRadius: 0.5,
            pointHoverRadius: 5,
            pointHoverBorderWidth: 5,
            pointBorderColor: 'transparent',
            pointHoverBackgroundColor:
greyLightColor,
```

```
    pointHoverBorderColor: cardColor,
    datalabels: {
      align: 'start',
      anchor: 'start',
      color: '#000',
      backgroundColor: '#f5f5f5',
      borderColor: '#000',
      borderRadius: 32,
      borderWidth: 1,
      font: {
        weight: 'bold',
        size: 8
      },
      offset: 1,
      padding: 5,
      textAlign: 'center'
    }
  }]
},
options: {
  responsive: true,
  maintainAspectRatio: false,
  plugins: {
    legend: {
      position: 'top',
      rtl: isRtl,
      align: 'start',
      labels: {
        usePointStyle: true,
        padding: 35,
        boxWidth: 6,
        boxHeight: 6,
        color: legendColor
      }
    },
    tooltip: {
      // Updated default tooltip UI
      rtl: isRtl,
      backgroundColor: cardColor,
      titleColor: headingColor,
```

```
            bodyColor: legendColor,
            borderWidth: 1,
            borderColor: borderColor
          }
        },
        scales: {
          x: {
            grid: {
              color: 'transparent',
              borderColor: borderColor
            },
            ticks: {
              color: labelColor
            }
          },
          y: {
            // min: 0,
            // max: 20,
            grid: {
              color: 'transparent',
              borderColor: borderColor
            },
            ticks: {
              color: labelColor
            }
          }
        }
      }
    });
    let updateChartArusListrik = function() {
      $.ajax({
        type: "GET",
        dataType: "json",
        url: "{{
route('collection.data.sensor') }}",
        headers: {
          'X-CSRF-TOKEN': $('meta[name="csrf-
token"]').attr('content')
        },
        success: function(data) {
```

```
            //  console.log(data);
            arusListrikChartVar.data.labels =
data.waktu;
            arusListrikChartVar.data.datasets[0]
.data = data.arus;
            arusListrikChartVar.update();
          },
          error: function(data) {
            //  console.log(data);
          }
        });
      }
      updateChartArusListrik();
      setInterval(() => {
        updateChartArusListrik();
      }, 5000);
    }

    //daya aktif
    const dayaAktifChart =
document.getElementById('dayaAktifChart');
    if (dayaAktifChart) {
      const dayaAktifChartVar = new
Chart(dayaAktifChart, {
        type: 'line',
        data: {
          labels: [],
          datasets: [{
            label: 'Daya Aktif ( Watt )',
            data: [],
            tension: 0,
            fill: 'start',
            backgroundColor: greenLightColor,
            pointStyle: 'circle',
            borderColor: greenColor,
            pointRadius: 0.5,
            pointHoverRadius: 5,
            pointHoverBorderWidth: 5,
            pointBorderColor: 'transparent',
```

```
            pointHoverBackgroundColor:
greyLightColor,
            pointHoverBorderColor: cardColor,
            datalabels: {
              align: 'start',
              anchor: 'start',
              color: '#000',
              backgroundColor: '#f5f5f5',
              borderColor: '#000',
              borderRadius: 32,
              borderWidth: 1,
              font: {
                weight: 'bold',
                size: 8
              },
              offset: 1,
              padding: 5,
              textAlign: 'center'
            }
          }]
        },
        options: {
          responsive: true,
          maintainAspectRatio: false,
          plugins: {
            legend: {
              position: 'top',
              rtl: isRtl,
              align: 'start',
              labels: {
                usePointStyle: true,
                padding: 35,
                boxWidth: 6,
                boxHeight: 6,
                color: legendColor
              }
            },
            tooltip: {
              // Updated default tooltip UI
              rtl: isRtl,
```

```
            backgroundColor: cardColor,
            titleColor: headingColor,
            bodyColor: legendColor,
            borderWidth: 1,
            borderColor: borderColor
          }
        },
        scales: {
          x: {
            grid: {
              color: 'transparent',
              borderColor: borderColor
            },
            ticks: {
              color: labelColor
            }
          },
          y: {
            // min: 0,
            // max: 20,
            grid: {
              color: 'transparent',
              borderColor: borderColor
            },
            ticks: {
              color: labelColor
            }
          }
        }
      }
    });
    let updateChartDayaAktif = function() {
      $.ajax({
        type: "GET",
        dataType: "json",
        url: "{{
route('collection.data.sensor') }}",
        headers: {
          'X-CSRF-TOKEN': $('meta[name="csrf-
token"]').attr('content')
```

```
          },
          success: function(data) {
            //  console.log(data);
            dayaAktifChartVar.data.labels =
data.waktu;
            dayaAktifChartVar.data.datasets[0].d
ata = data.dayaAktif;
            dayaAktifChartVar.update();
          },
          error: function(data) {
            //  console.log(data);
          }
        });
      }
      updateChartDayaAktif();
      setInterval(() => {
        updateChartDayaAktif();
      }, 5000);
    }

  })();
</script>

<!-- update data table dashboard -->
<script>
  // update table energi listrik
  let updateTableEnergiListrik = function() {
    $.ajax({
      type: "GET",
      dataType: "json",
      url: "{{ route('table.data.sensor') }}",
      headers: {
        'X-CSRF-TOKEN': $('meta[name="csrf-
token"]').attr('content')
      },
      success: function(response) {
        let html = '';
        let count = 1;
```

```
        if (response.datas != null &&
response.datas.length > 0) {

          for (let i = 0; i <
response.datas.length; i++) {
            let energi =
response.datas[i].energi;
            let waktu = response.datas[i].waktu;


            html += '<tr style="font-
size:13px;">' +
              '<td>' + count++ + '</td>' +
              '<td>' + energi + ' kWh' + '</td>'
+
              '<td>' + waktu + '</td>' +
              '</tr>';
          }
          $('#showTabelEnergi').html(html);
        } else {
          // Opsi untuk menangani jika
response.datas kosong atau null
          $('#showTabelEnergi').html('<tr><td
colspan="3" align="center">No data
available</td></tr>');
        }

        // console.log(html);
      },
      error: function(data) {
        console.log(data);
      }
    })
  }

  updateTableEnergiListrik();
  setInterval(() => {
    updateTableEnergiListrik();
  }, 5000);
```

```
  //Update Card Value sensor
  let updateCardValue = function() {
    $.ajax({
      type: "GET",
      dataType: "json",
      url: "{{ route('card.data.sensor') }}",
      headers: {
        'X-CSRF-TOKEN': $('meta[name="csrf-
token"]').attr('content')
      },
      success: function(response) {

        function formatNumber(value) {
        // Mengubah angka menjadi string dan
memotong trailing zeros
          let formattedValue =
parseFloat(value).toString();
          return formattedValue;
        }

        $('#energiCard').text(formatNumber(respo
nse.data[0].energi) + ' kWh');
        $('#sisaCard').text(formatNumber(respons
e.sisa_token) + ' kWh');
        $('#teganganCard').text(response.data[0]
.tegangan + ' Volt');
        $('#arusCard').text(response.data[0].aru
s + ' Ampere');
        $('#biayaCard').text('Rp. ' +
response.data[0].biaya);
        $('#valueV').text(response.data[0].tegan
gan);
        $('#valueI').text(response.data[0].arus)
;
        $('#valueP').text(response.data[0].dy_ak
tif);
        $('#valueF').text(response.data[0].freku
ensi);
      },
      error: function(error) {
```

```
            console.log(error);
          }
        })
    }

    updateCardValue();
    setInterval(() => {
      updateCardValue();
    }, 5000);
</script>
@endpush
```