```
// ---------------------------------------------
// Header Library
// ---------------------------------------------
#include <ESP8266WiFi.h>          // Library untuk koneksi WiFi di ESP8266
#include <WiFiClient.h>           // Library untuk membuat koneksi TCP/HTTP
#include <ESP8266HTTPClient.h>    // Library untuk mengirim permintaan HTTP (GET/POST)
#include <ArduinoJson.h>          // Library untuk memproses data JSON

// ---------------------------------------------
// Konfigurasi Serial Komunikasi
// ---------------------------------------------
// Menggunakan SoftwareSerial pada pin D2 (Rx) dan D1 (Tx)
// Komunikasi ini dipakai untuk pertukaran data antara ESP dan Arduino
#include<SoftwareSerial.h>
SoftwareSerial mySerial (D2, D1);         // D2 = RxESP, D1 = TxESP

// ---------------------------------------------
// Pengaturan WiFi dan Endpoint HTTP
// ---------------------------------------------
// SSID dan password jaringan WiFi yang digunakan ESP untuk koneksi internet
const char* ssid = "PublicWifi";          // Ganti nilai SSID
const char* password = "1234567890";      // Ganti nilai passwordnya

// Data awal untuk request, akan digunakan untuk menyusun URL permintaan HTTP
String urlJSON = "data";
// Digunakan untuk mengatur indeks permintaan
// IndexJSON akan bernilai 1 tingkat dibawah ID Meter pada kode Arduino
// Misal pada kode Arduino ID Meter = 10, maka indexJSON harus diisi dengan 9
// indexJSON akan berubah menyesuaikan ID Meter yang ada di kode Arduino
int indexJSON = 1;

// URL endpoint server untuk komunikasi data
// Server ini digunakan untuk:
// - Mengirimkan data air (urlData)
// - Mengirimkan status pintu (urlPintu)
const char *host = "http://lingindustri.com/pam/json/index.php";
String urlData   = "http://lingindustri.com/pam/json/add.php?data=";
String urlPintu  = "http://lingindustri.com/pam/json/pintu.php?data=";
// const char *host = "http://192.168.0.10/pwm-json/convert.php";
// String urlData   = "http://192.168.0.10/pwm-json/update.php?data=";

// Parameter tambahan untuk menunjukkan ID meter (ID pelanggan)
String urlPls    = "&num=";   // Index ID Pelanggan

// ---------------------------------------------
// Status Koneksi Internet
// ---------------------------------------------
// Digunakan sebagai flag untuk memeriksa apakah ESP sudah terkoneksi dengan WiFi
bool cekInet = false;
```

```
// ------------------------------------------------
// Fungsi setup()
// ------------------------------------------------
// Fungsi setup() dijalankan satu kali saat ESP menyala atau di-reset
// Tugasnya adalah:
// - Memulai komunikasi serial dengan PC dan Arduino
// - Menyiapkan sistem awal sebelum loop berjalan
// ------------------------------------------------
void setup() {
  Serial.begin(9600);       // Serial utama (USB) untuk debugging
  mySerial.begin(9600);     // Serial kedua (D2, D1) untuk komunikasi dengan Arduino
  delay(2000);              // Delay untuk memastikan semua modul siap
}

// ------------------------------------------------
// Fungsi loop()
// ------------------------------------------------
// Fungsi ini berjalan terus-menerus setelah `setup()`
// - Menerima data serial dari Arduino melalui `mySerial`
// - Melakukan aksi berdasarkan perintah (msg) yang diterima:
//    - "init": menghubungkan ESP ke internet
//    - "SendData": menerima data ID dan nomor meteran, lalu kirim ke server
//    - "SendPintu": menerima status pintu dan ID meteran, lalu kirim ke server
// ------------------------------------------------
void loop() {
//  if(cekInet == false){
    // Baca data dari komunikasi serial sampai karakter carriage return '\r'
    String msg = mySerial.readStringUntil('\r');
    msg.trim();      // Menghapus spasi/karakter tak perlu di awal/akhir string
    // ------------------------------------------------
    // Perintah untuk Inisialisasi Koneksi Internet
    // ------------------------------------------------
    if(msg.equals("init") && cekInet == false){
      sambungInternet();    // Panggil fungsi sambungInternet() untuk menyambungkan ke WiFi
      cekInet = true;       // Tandai bahwa sudah tersambung
    }
    // delay(5000);
//  }

    // ------------------------------------------------
    // Perintah untuk Mengirim Data Penggunaan Air
    // Format data masuk: "SendData" + data diikuti dengan ID#NOMOR+
    // ------------------------------------------------
    if(msg.equals("SendData")){
      String dataSend = mySerial.readString();  // Baca sisa data setelah perintah
      dataSend.replace("\n", "");               // Hapus karakter newline
      dataSend.trim();                          // Bersihkan whitespace
      Serial.println(dataSend);                 // Debug: tampilkan data masuk

       // Parsing ID dan Nomor dari format: ID#NOMOR+
      int inData1 = dataSend.indexOf("#");
      int inData2 = dataSend.indexOf("+");
      String dataIDku = dataSend.substring(0, inData1);
      String dataNOku = dataSend.substring(inData1+1,inData2);
```

```
        // Debug untuk memastikan parsing benar
        Serial.println(dataIDku);
        Serial.println(dataNOku);
        Serial.print("I received: ");
        Serial.println(dataSend);

        // Kirim data ke server menggunakan fungsi kirimData()
        kirimData(dataIDku, dataNOku);
      }

      // TambahanPintu
      // ---------------------------------------------
      // Perintah untuk Mengirim Status Pintu
      // Format data masuk: "SendPintu" + data dengan format: STATUS#ID+
      // Contoh: "terbuka#MTR123+"
      // ---------------------------------------------
      if(msg.equals("SendPintu")){
        String dataSend = mySerial.readString();  // Baca sisa data setelah perintah
        dataSend.replace("\n", "");
        dataSend.trim();
        Serial.println(dataSend);

        // Parsing data: status dan ID
        int inData1 = dataSend.indexOf("#");
        int inData2 = dataSend.indexOf("+");
        String dataPintuku = dataSend.substring(0, inData1);    // Status pintu
        String dataNOku = dataSend.substring(inData1+1,inData2);// ID Meter
        Serial.println(dataPintuku);
        Serial.println(dataNOku);
        Serial.print("I received: ");
        Serial.println(dataSend);

        // Kirim status pintu ke server
        kirimPintu(dataPintuku, dataNOku);
      }
}


// ---------------------------------------------
// Fungsi ini digunakan untuk menyambungkan ESP8266 ke jaringan WiFi.
// Proses:
// 1. Menampilkan SSID tujuan di serial monitor
// 2. Mengaktifkan mode WiFi Station (WIFI_STA)
// 3. Memulai koneksi menggunakan SSID dan password
// 4. Menunggu sampai status koneksi berhasil
// 5. Menampilkan alamat IP yang diperoleh
// 6. Memanggil fungsi ambilData() setelah koneksi berhasil
//
// Catatan:
// - Fungsi ambilData() dapat digunakan untuk mengambil data awal dari server
// ---------------------------------------------
void sambungInternet(){
// Set your Static IP address
```

```
//  IPAddress local_IP(192, 168, 0, 110);
//  IPAddress gateway(192, 168, 0, 1);
//  IPAddress subnet(255, 255, 255, 0);
//  IPAddress primaryDNS(8, 8, 8, 8);
//  IPAddress secondaryDNS(8, 8, 4, 4);

  Serial.print("Connecting to : ");
  Serial.println(ssid);     // Menampilkan SSID tujuan


  // Configures static IP address
  //  if (WiFi.config(local_IP, gateway, subnet, primaryDNS, secondaryDNS)) {
  //    Serial.println("Static IP Configured");
  //  }else {
  //    Serial.println("Static IP Configuration Failed");
  //  }

  // Mode Station (agar ESP bertindak sebagai klien WiFi, bukan hotspot)
  WiFi.mode(WIFI_STA);

  WiFi.begin(ssid, password);      // Mulai koneksi WiFi

  // Tunggu hingga terkoneksi
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");             // Indikator proses koneksi
  }

  // Tampilkan hasil koneksi
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());   // Alamat IP dari DHCP router
  Serial.println();
  delay(3000);

  ambilData();       // Panggil fungsi ambilData()
}
```

```
// ------------------------------------------------
// Fungsi sambungIn() juga berfungsi untuk menyambungkan ESP ke jaringan WiFi.
// Perbedaannya dengan sambungInternet() adalah:
// - Tidak memanggil fungsi ambilData()
// ------------------------------------------------
void sambungIn(){
  // Set your Static IP address
  //  IPAddress local_IP(192, 168, 0, 150);
  //  IPAddress gateway(192, 168, 0, 1);
  //  IPAddress subnet(255, 255, 255, 0);
  //  IPAddress primaryDNS(8, 8, 8, 8);
  //  IPAddress secondaryDNS(8, 8, 4, 4);
  Serial.print("Connecting to : ");
  Serial.println(ssid);
  // Configures static IP address
  //  if (WiFi.config(local_IP, gateway, subnet, primaryDNS, secondaryDNS)) {
  //  if (WiFi.config(local_IP, gateway, subnet)) {
  //    Serial.println("Static IP Configured");
  //  }else {
  //    Serial.println("Static IP Configuration Failed");
  //  }

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  Serial.println();
  delay(3000);
}

// Fungsi ambilData()
// ------------------------------------------------
// Fungsi ini bertanggung jawab untuk mengambil data awal pelanggan
// dari server menggunakan metode HTTP GET.
//
// Data yang diambil berupa file JSON yang berisi informasi pelanggan,
// seperti ID pelanggan, nomor pelanggan, nama, dan nilai pulsa.
//
// Setelah data diterima:
// - Data diparsing menggunakan ArduinoJson
// - Disusun ke format khusus dan dikirim ke Arduino melalui SoftwareSerial
//
// Jika gagal terhubung ke server (httpCode == -1), akan mencoba menyambung
// ulang ke internet melalui `sambungInternet()`.
// ------------------------------------------------
void ambilData(){
//  WiFi.mode(WIFI_STA);  // Ambil Data
```

```
HTTPClient http;          // HTTP client untuk request data

Serial.print("Request Link:");
Serial.println(host);     // Menampilkan URL tujuan

http.begin(host);         // Mulai request HTTP GET ke server

int httpCode = http.GET();              // Kirim request
String payload = http.getString();      // Ambil respon dari server (format JSON)

Serial.print("Response Code:");         // Contoh: 200 = OK
Serial.println(httpCode);               // Tampilkan kode http

Serial.print("Returned data from Server:");
Serial.println(payload);    // Tampilkan isi JSON

// Jika gagal koneksi ke server, coba sambung ulang internet
if(httpCode == -1){
  delay(5000);
  sambungInternet();
}else{
  // Jika sukses
  if(httpCode == 200){
    // Inisialisasi buffer JSON
    DynamicJsonBuffer jsonBuffer;
    JsonObject& root = jsonBuffer.parseObject(payload);
    JsonArray& requests = root[urlJSON];
    // Cek keberhasilan parsing
    if (!requests.success()) {
      Serial.println(F("Parsing failed!"));
      return;
    }
    // Decode JSON/Extract values
   // Ambil data pelanggan
    Serial.println(F("Response:"));
    String a = root[urlJSON][indexJSON]["id"].as<char*>();
    String b = root[urlJSON][indexJSON]["nopelanggan"].as<char*>();
    String c = root[urlJSON][indexJSON]["namapelanggan"].as<char*>();
    String d = root[urlJSON][indexJSON]["nilai"].as<char*>();
    // Format data untuk dikirim ke Arduino
    String allABCD = "#*" + a + "#@" + b + "#&" + c + "#!" + d + "##";
     // Serial.println(allABCD);

     // Serial.println(root[urlJSON][indexJSON]["id"].as<char*>());
     // Serial.println(root[urlJSON][indexJSON]["nopelanggan"].as<char*>());
     // Serial.println(root[urlJSON][indexJSON]["namapelanggan"].as<char*>());
     // Serial.println(root[urlJSON][indexJSON]["nilai"].as<char*>());
     // #*2#@PAM0002#&Pelanggan1#!90000##

    // Kirim ke Arduino (fungsi sendArduino)
    sendArduino(allABCD);
  }else{
    Serial.println("Error in response");
```

```
    }
  }

  http.end();  // Tutup koneksi
}

// Fungsi kirimData(String dataPulsa, String dataID)
// -------------------------------------------------
// Fungsi ini digunakan untuk mengirimkan data pulsa (nilai air yang
// ditambahkan atau dikonsumsi) ke server melalui HTTP GET request.
//
// Format URL:
//   http://lingindustri.com/pam/json/add.php?data=<pulsa>&num=<id>
//
// Proses:
// 1. Menyambungkan ulang ke internet (`sambungIn()`)
// 2. Membersihkan string (hapus newline, trim whitespace)
// 3. Menyusun URL request lengkap
// 4. Melakukan HTTP GET dan menampilkan hasil response
// 5. Memanggil kembali `ambilData()` untuk menyinkronkan ulang data lokal
// -------------------------------------------------
void kirimData(String dataPulsa, String dataID){
  sambungIn();              // Pastikan ESP sudah terkoneksi dengan internet

  // Bersihkan string dari newline dan whitespace
  dataPulsa.replace("\n", "");
  String pulsa = dataPulsa;
  urlData.trim();
  pulsa.trim();
  dataID.trim();
  urlPls.trim();
  String address;

  // URL Server
  // Susun URL untuk dikirim ke server
  Serial.println("--");
  address  = urlData;
  address += pulsa;
  address += urlPls;
  address += dataID;

  // Contoh URL:
  // http://lingindustri.com/pam/json/add.php?data=600&num=2

  address.replace("\n","");        // Hapus newline
  address.trim();                  // Hapus spasi
  Serial.println(address);         // Cetak URL yang akan dikirim
  delay(1000);                     // Delay agar request stabil


  HTTPClient http;
  http.begin(String(address));     // Siapkan HTTP GET
  int httpCode = http.GET();       // Kirim request
  String payload;
```

```
  if (httpCode > 0) {              // Check the returning code
      payload = http.getString();  // Ambil response dari server
      payload.trim();
      if( payload.length() > 0 ){
         Serial.println(payload + "\n"); // Tampilkan response
      }
  }

  ambilData();        // Ambil data terbaru setelah update ke server
  http.end();         // Tutup koneksi
}

// TambahanPintu
// Fungsi kirimPintu(String dataPintu, String dataID)
// ------------------------------------------------------------
// Fungsi ini digunakan untuk mengirim status kondisi pintu (terbuka/tertutup)
// dari Arduino ke server melalui ESP, menggunakan HTTP GET request.
//
// Status pintu dikirim melalui URL sebagai parameter data, contoh:
//   http://lingindustri.com/pam/json/pintu.php?data=PintuTerbuka&num=2
//
// Proses:
// 1. Memastikan koneksi internet aktif dengan `sambungIn()`
// 2. Membersihkan data dari karakter newline atau spasi
// 3. Menyusun URL lengkap untuk request
// 4. Mengirim request ke server dan mencetak respon
// 5. Memanggil kembali `ambilData()` untuk menyinkronkan data lokal
//
// Komponen dan Library:
// - ESP8266HTTPClient: untuk komunikasi HTTP GET
// - SoftwareSerial: digunakan untuk komunikasi ke Arduino
// - Server PHP: menerima data kondisi pintu
// ------------------------------------------------------------
void kirimPintu(String dataPintu, String dataID){
  sambungIn();                     // Pastikan terhubung ke WiFi

  // Bersihkan dan siapkan data
  dataPintu.replace("\n", "");
  String datapintu = dataPintu;
  urlData.trim();
  datapintu.trim();
  dataID.trim();
  urlPls.trim();
  String address;

  // Susun URL request lengkap
  Serial.println("--");
  address  = urlPintu;      // Berikan nilai address menjadi urlPintu
  address += datapintu;     // Tambahkan nilai address dengan datapintu
  address += urlPls;        // Tambahkan nilai address dengan urlPls
  address += dataID;        // Tambahkan nilai address dengan dataID
  // Contoh hasil akhir:
  // http://lingindustri.com/pam/json/pintu.php?data=PintuTerbuka&num=2
```

```
  address.replace("\n",""); // Hapus newline
  address.trim();           // Hapus spasi
  Serial.println(address);  // Tampilkan URL akhir untuk debug
  delay(1000);              // Jeda sebelum kirim request

  // Kirim HTTP GET ke server
  HTTPClient http;
  http.begin(String(address));     // Specify request destination
  int httpCode = http.GET();       // Kirim request GET
  String payload;
  if (httpCode > 0) {              // Jika request berhasil
      payload = http.getString();  // Ambil isi respon
      payload.trim();
      if( payload.length() > 0 ){
         Serial.println(payload + "\n"); // Tampilkan respon
      }
  }
  ambilData();  // Sinkronisasi ulang data
  http.end();   // Tutup koneksi
}

// Fungsi sendArduino(String nilaiPulsa)
// ------------------------------------------------------------
// Fungsi ini digunakan untuk mengirim data ke Arduino melalui
// komunikasi serial (mySerial) setelah data diterima dari server.
// Format data sudah disusun dalam bentuk string custom seperti:
//   #*2#@PAM0002#&Pelanggan1#!90000##
//
// Proses:
// 1. Menampilkan informasi ke Serial Monitor (debugging)
// 2. Mengirim string data ke Arduino melalui mySerial (SoftwareSerial)
// ------------------------------------------------------------
void sendArduino(String nilaiPulsa){
  Serial.println("GetData");              // Tampilkan indikator
  mySerial.println("GetData");            // Kirim sinyal ke Arduino
  //   #*2#@PAM0002#&Pelanggan1#!90000##
  Serial.println(nilaiPulsa);             // Tampilkan nilaiPulsa
  mySerial.println(nilaiPulsa);
}
```