LAMPIRAN

Lampiran 1. Program Utama AT Mega

```
//***Sistem Auto Reffil***//

      #include <ArduinoJson.h>

      #include "HCSR04.h"     // sub program running ultrasonic

      #include "flowMeter.h"  // sub program running Sensor flow meter

      #include "relay4Ch.h"   // sub program running relay


      byte counterHSD = 0;

      //////////////////////////////////////////////////////////////

      //Fungsi algoritma auto refill tanki atas

      void logicAutoTankiAtas (){

       //jika pambacaan tanki atas melebihi batas atas ketinggian tanki

       if (ultrasonicAtas > tinggiTankiAtas){

         //delay (50);

       }


       //jika pembacaan Sensor mencapai batas maksimum level hsd

       else if (ultrasonicAtas < maksimum){

         //perintah logic disini

         //resetValve ();
```

```
    //resetRelay ();
    relayHvOff();
    pompaOff();

    //menset status refill 'false' atau siap untuk pengisian selanjutnya
    refillStatus = false ;
    //delay(50);
  }
  //logic batas refill stage 2 antara nilai maksimal dan forecast atas tangki atas
  else if (ultrasonicAtas < tinggiTankiAtas){
    //if refill status true dan pembacaan Sensor < nilai forecast tanki atas
    if (refillStatus == true && ultrasonicAtas < forecastAtasTankiAtas ){
      //for pembacaan Sensor > nilai maksimum Sensor (seting clearance Sensor)
maka counter down
      for (ultrasonicAtas; ultrasonicAtas > maksimum; ultrasonicAtas--){
        //if pembacaan Sensor > nilai maksimum Sensor dan lebih kecil dari
forecast atas tanki atas
        if (ultrasonicAtas > maksimum && ultrasonicAtas <
forecastAtasTankiAtas){
          //perintah disini
          enableRelayValve ();
          relayHvOn ();
          valveOutAtasOn();
        valveIntakeBawahOn();

        pompaOn();


        continue ;  //iki opo aku lali

        delay(50);

      }

    }

  }

  //batas refill stage 1 antara nilai forecast dan minimal,

  //triger untuk memulai pengisian dari point ini

  if (ultrasonicAtas > forecastAtasTankiAtas ){

   //for pembacaan Sensor > nilai maksimum makan Sensor counter down

   for (ultrasonicAtas; ultrasonicAtas > maksimum; ultrasonicAtas--){

    //if pembacan Sensor > forecast atas tanki dan kurang dari tinggi tanki

    if (ultrasonicAtas > forecastAtasTankiAtas && ultrasonicAtas <
tinggiTankiAtas){

        //perintah disini

        enableRelayValve ();

        relayHvOn ();

        valveOutAtasOn();

        valveIntakeBawahOn();

        pompaOn();
        continue ;
        delay(50);
      }
    }
   refillStatus = true ;
  }
 }
}
//////////////////////////////////////////////////////////////
//Fungsi algoritma autorefill tanki bawah
void logicAutoTankiBawah (){
 //jika pambacaan tanki bawah melebihi batas atas ketinggian tanki
 if (ultrasonicBawah > tinggiTankiBawah){
   //perintah disini
   //delay(50);
 }
 //jika pembacaan Sensor mencapai batas maksimum level hsd
 else if (ultrasonicBawah < maksimum){
   //perintah disini
   resetValve ();
   resetRelay ();

   refillStatus = false ;
  //delay(50);
 }
 //batas refill stage 2 antara nilai maksimal dan forecast
 else if (ultrasonicBawah < tinggiTankiBawah){
  //if refill status true dan pembacaan Sensor < nilai forecast tanki bawah
  if (refillStatus == true && ultrasonicBawah < forecastAtasTankiBawah ){
   //for pembacaan Sensor > nilai maksimum Sensor (seting clearance Sensor)
maka counter down
   for (ultrasonicBawah; ultrasonicBawah > maksimum; ultrasonicBawah--){
    //if pembacaan Sensor > nilai maksimum Sensor dan lebih kecil dari
forecast atas tanki bawah
    if (ultrasonicBawah > maksimum && ultrasonicBawah <
forecastAtasTankiBawah){
     //perintah disini
     relayHvOn ();
     valveOutBawahOn();
     //valveIntakeBawahOn ();
     valveExternalOn();
     pompaOn();

     continue ;
     delay(50);
    }
  }
 }
 //batas refill stage 1 antara nilai forecast dan minimal,
 //triger untuk memulai pengisian dari point ini
 if (ultrasonicBawah > forecastAtasTankiBawah ){
  //for pembacaan Sensor > nilai maksimum makan Sensor counter down
  for (ultrasonicBawah; ultrasonicBawah > maksimum; ultrasonicBawah--){
   //if pembacaan Sensor > forecast atas tanki dan kurang dari tinggi tanki
   if (ultrasonicBawah > forecastAtasTankiBawah && ultrasonicBawah <
tinggiTankiBawah){
    //perintah disini
    relayHvOn ();
    valveOutBawahOn();
    //valveIntakeBawahOn ();
    valveExternalOn();
    pompaOn();

    continue ;
    delay(50);
   }
  }
  refillStatus = true ;
 }
```

```
  }
}
////////////////////////////////////////////////////////////
void loop (){
  while (Serial.available () == 0){
    logicAutoTankiAtas();  //fungsi auto refill tangki atas
    //logicAutoTankiBawah(); //fungsi auto refill tangki bawah
    //jajalPompa();

    ultrasonicRead(); //running ultrasonic Sensor
    flowRead();       //running flow Sensor

    if (Meter.getCurrentFlowrate()>1 && refillStatus == false){
      counterHSD = 1;
      //delay(50);
    }
    else {
      counterHSD = 0;
    }
    //format pengiriman paket data *refill,Us1,Us2,counter volume,flow,#
    //Serial.println
("*"+String(counterHSD)+","+konversi1(ultrasonicAtas)+","+konversi2(ultraso
        nicBawah)+","+konversi3(Meter.getTotalVolume())+","+konversi4(Meter.getCu
        rrentFlowrate())+","+"]");
          //delay (50);
          serialJson ();  //fungsi serialised data Sensor
         }

         readFromESP ();  //fungsi membaca command dari telegram via ESP8266
         delay (50);
}

//***Program Sensor Ultrasonik **//
        Ultrasonic uS1 (TriguS1,EchouS1);
        Ultrasonic uS2 (TriguS2,EchouS2);

        const int tinggiTankiAtas = 14 ;   //batas minimal tanki atas
        const int tinggiTankiBawah = 20 ;  //batas minimal tanki utama
        float forecast = 0.15;             //forecast 15% dari ketinggian tanki
        float forecastAtasTankiAtas = 0;   //batas bawah tanki atas
        float forecastAtasTankiBawah = 0;  //batas bawah tanki bawah
        int maksimum = 3 ;                 //clearance Sensor dengan HSD saat maksimum
        int ultrasonicAtas = 0 ;
        int ultrasonicBawah = 0 ;
        bool refillStatus = false;
```

```
//Fungsi persentase nilai forecast terhadap tinggi tanki
void batasBawah (){
  forecastAtasTankiAtas  = tinggiTankiAtas  - ((tinggiTankiAtas*forecast)+2);
  forecastAtasTankiBawah = tinggiTankiBawah -
(tinggiTankiBawah*forecast)+3;
}

//Fungsi baca data ultrasonic
void ultrasonicRead (){
  ultrasonicAtas  = uS1.Ranging(CM);
  delay (50);
  ultrasonicBawah = uS2.Ranging(CM);
  delay (50);

  batasBawah ();
}
```

```
//** Program flow Meter **//
    // connect a flow meter to an interrupt pin (see notes on your Arduino model for
    pin numbers)
    FlowMeter Meter = FlowMeter(2);
```

```
// set the measurement update period to 1s (1000 ms)
const unsigned long period = 100;
// define an 'interrupt service handler' (ISR) for every interrupt pin you use
void MeterISR() {
   // let our flow meter count the pulses
   Meter.count();
}

void setupFlow() {
   // enable a call to the 'interrupt service handler' (ISR) on every rising edge at
the interrupt pin
   // do this setup step for every ISR you have defined, depending on how many
interrupts you use
   attachInterrupt(INT0, MeterISR, RISING);

   // sometimes initializing the gear generates some pulses that we should ignore
   Meter.reset();
}

void flowRead() {
   // wait between output updates
   delay(period);
```

```
        // process the (possibly) counted ticks
        Meter.tick(period);


        // output some measurement result
        //Serial.println("Currently " + String(Meter.getCurrentFlowrate()) + " l/min, "
      + String(Meter.getTotalVolume())+ " l total.");
        // any other code can go here
        //
}


//** Program Modul  Relay 4 Ch **//
      //Fungsi Untuk Set Relay
      void setupRelay (){
       pinMode (pin1, OUTPUT); //relay high voltage
       pinMode (pin2, OUTPUT); //relay pompa
       pinMode (pin3, OUTPUT); //relay power 12VDC bawah
       pinMode (pin4, OUTPUT); //relay valve intake external
       pinMode (pin5, OUTPUT); //relay valve intake tanki bawah
       pinMode (pin6, OUTPUT); //relay valve output tanki atas
       pinMode (pin7, OUTPUT); //relay valve output tanki bawah
       pinMode (pin8, OUTPUT); //tambahan


       resetValve ();
```

```
  resetRelay ();
}
//Fungsi Test All Relay
void relayTest (){
  //for untuk menghidupkan relay satu persatu range 0. detik (Aktif Low)
  for (byte i=0 ; i<1 ; i++){ //setting berapa kali looping
    delay (200);
    digitalWrite (pin8,LOW);
    delay (200);
    digitalWrite (pin2,LOW);
    delay (500);
    digitalWrite (pin2, HIGH);
    digitalWrite (pin3,LOW);
    delay (200);
    digitalWrite (pin1,LOW);
    delay (200);
    digitalWrite (pin4,LOW);
    delay (200);
    digitalWrite (pin5,LOW);
    delay (200);
    digitalWrite (pin6,LOW);
    delay (200);
    digitalWrite (pin7,LOW);
    delay (200);

    resetValve ();
    resetRelay ();
  }
}

//Fungsi tegangan 220VAC ON
void relayHvOn (){
  digitalWrite (pin1, LOW);
}
//Fungsi tegangan 220VAC OFF
void relayHvOff (){
  digitalWrite (pin1, HIGH);
}
//Fungsi Pompa Utama ON
void pompaOn (){
  digitalWrite (pin2, LOW);
}
//Fungsi Pompa Utama OFF
void pompaOff (){
  digitalWrite (pin2, HIGH);
}
```

```
//Fungsi Enable Kontrol Relay 12 VDC
void enableRelayValve (){
  digitalWrite (pin3, LOW);
}
//Fungsi Disable Kontrol Relay 12 VDC
void disableRelayValve (){
  digitalWrite (pin3, HIGH);
}
//Fungsi Mengaktifkan Valve Pompa External
void valveExternalOn (){
  digitalWrite (pin4, LOW);
}
//Fungsi Mematikan Valve Pompa External
void valveExternalOff (){
  digitalWrite (pin4, HIGH);
}
//Fungsi Mengaktifkan Valve Tangki Utama
void valveIntakeBawahOn (){
  digitalWrite (pin5, LOW);
}
//Fungsi Memematikan Valve Tangki Utama
void valveIntakeBawahOff (){
  digitalWrite (pin5, HIGH);
```

```
}
//Fungsi Mengaktifkan Valve Tangki Atas
void valveOutAtasOn (){
  digitalWrite (pin6, LOW);
}
//Fungsi Mematikan Valve Tangki Atas
void valveOutAtasOff (){
  digitalWrite (pin6, HIGH);
}
//Fungsi Mengaktifkan/Bypass Valve Tangki Atas
void valveOutBawahOn (){
  digitalWrite (pin7, LOW);
}
//Fungsi Mematikan/Bypass Valve Tangki Atas
void valveOutBawahOff (){
  digitalWrite (pin7, HIGH );
}
void playRelay (){
  for (byte i=0; i<2; i++) {
    digitalWrite (pin8, LOW);
    delay (500);
    digitalWrite (pin8 , HIGH);
    delay (500);
```

```
        }
}
```

Lampiran 2. Program ESP 8266

```
//** universal bot _serial parser**//
        #define ARDUINOJSON_ENABLE_STD_STREAM 1
        // Wifi network station credentials
        #define WIFI_SSID "Waiwai" //Waiwai,Xperia SXZ Series
        #define WIFI_PASSWORD "tanyayangpunya"
        // Telegram BOT Token (Get from Botfather)
        #define BOT_TOKEN
        "1543164988:AAHxpFNdZZaBlQqfqhj_3Gd9RtPNiz1eG2c"

        //Json parser variable
        String answer ;
        int Us1, Us2, Pmp ;
        double vol, flo ;

        //inisiasi parsing data
        String dataIn;
        String data[10];
        int i;
        bool parsing = false;
```

```
void SerialBufferPacketData (){
 static byte idx = 0;
 char endMarker ='\n';
 char nSWRead ;

 if (Serial.available () > 0){
  newData = false ;

  while (Serial.available () > 0 && newData == false ){
   nSWRead = Serial.read ();

   if (nSWRead != endMarker){
    displayChars[idx] = nSWRead ;
    idx ++ ;
    if(idx >= numChars) {
     idx = numChars - 1 ;
    }
   }
   else {
    displayChars[idx] = '\0';
    idx = 0 ;
    newData = true ;
   }
```

```
    }
  }
}
/////////////////////////////////////////////
void handleNewMessages(int numNewMessages){
  //Serial.print("handleNewMessages ");
  //Serial.println(numNewMessages);

  String answer;
  for (int i = 0; i < numNewMessages; i++)
  {
    telegramMessage &msg = bot.messages[i];
    //Serial.println("Received " + msg.text);
    if (msg.text == "/help"){
      answer = "Jika anda membutuhkan bantuan tekan tanda '/' pada baris
Messsage Telegram";
      Serial.println ("1");
      delay (100);
    }
    else if (msg.text == "/start"){
      answer = "Selamat datang *" + msg.from_name + "* , apakah kita pernah
bertemu?";
      Serial.println ("@");
  }
  else if (msg.text == "/status"){
   Serial.println ("$");
   SerialBufferPacketData ();
   answer = displayChars ;

   displayChars [numChars] = '\0';

   //char megaReply []= {Serial.read ()};
   //deserializeJson (doc, megaReply);
   //String Us1 = doc ["data"][0];
   //String Us2 = doc ["data"][1];
   //String vol = doc ["data"][2];
   //answer = "Tanki Atas "+Us1+"%"+"Tanki Bawah "+Us2+" %"+"Counter
HSD "+vol+" Liter";
  }
  else if (msg.text == "/outputcheck"){
   Serial.println ("&");
  }
  else if (msg.text == "/extenalpump"){
   Serial.println ("?");
  }
  else{
      answer = "Tunggu sebentar";
    }

    bot.sendMessage(msg.chat_id, answer, "Markdown");
  }
}
////////////////////////////////////////////////
void bot_setup()
{
  const String commands = F("["
                            "{\"command\":\"help\",  \"description\":\"Bantuan\"},"
                            "{\"command\":\"start\", \"description\":\"Mulai interaksi
dengan MCU\"},"
                            "{\"command\":\"status\",\"description\":\"Membaca status
system\"}," // no comma on last command
                            "{\"command\":\"outputcheck\",\"description\":\"Mode output
check otomatis\"},"
                            "{\"command\":\"extenalpump\",\"description\":\"Trial mode
pengisian dari external\"}"
                            "]");
  bot.setMyCommands(commands);
  //bot.sendMessage("25235518", "Hola amigo!", "Markdown");
}
```

```
/////////////////////////////////////////////////
  // attempt to connect to Wifi network:
  configTime(0, 0, "pool.ntp.org");      // get UTC time via NTP
  secured_client.setTrustAnchors(&cert); // Add root certificate for
api.telegram.org
  Serial.print("Connecting to Wifi SSID ");
  Serial.print(WIFI_SSID);
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
    delay(100);
  }
  Serial.print("\nWiFi connected. IP address: ");
  Serial.println(WiFi.localIP());
  wifiConnected ();   //simbol wifi terkonek jaringan
  bot_setup();
  logoINKA();      //tampilkan logo UNMUH
  logoUnmuh();     //tampilkan logo INKA

  setupRTC();
  doa();
}
```

```
      ////////////////////////////////////////////
      void loop()
      {
        if (millis() - bot_lasttime > BOT_MTBS){
          int numNewMessages = bot.getUpdates(bot.last_message_received + 1);

          while (numNewMessages)
          {
            handleNewMessages(numNewMessages);
            numNewMessages = bot.getUpdates(bot.last_message_received + 1);
          }
            bot_lasttime = millis();
        }
         callRTC();
        parsingLoop ();

}
//** Program RTC DS3231**//
      #include "RTClib.h" //
      RTC_DS3231 rtc;
      DateTime now;
```

```
char daysOfTheWeek[7][12] = {"Sunday", "Monday", "Tuesday", "Wednesday",
"Thursday", "Friday", "Saturday"};
float rtcTemp;
String waktu;
/////////////////////////////////////////////
void setupRTC (){
 #ifndef ESP8266
  //while (!Serial);
 #endif

 delay(1000);
 if (! rtc.begin()){
  Serial.println ("RTC mu ilang bro");
  while (1);
 }

 if (rtc.lostPower ()){
  Serial.println ("Tegangan e ilang bro, jajal set waktune");
  rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
   }
}
/////////////////////////////////////////////
void callRTC(){
   now = rtc.now ();

   byte jam = now.hour();

   byte menit = now.minute();

   rtcTemp = rtc.getTemperature ();


   waktu = addNull(jam)+":"+addNull(menit);

}


//** Program OLED LCD Monitor **//

  #define SCREEN_WIDTH 128 // OLED display width, in pixels

  #define SCREEN_HEIGHT 64 // OLED display height, in pixels

  #define OLED_ADDR 0x3C

  ///////////////////////////////////////////////

  void wifiSearch (void){

   display.clearDisplay();

   display.drawBitmap (0,0, wifi_Search, 128, 64, WHITE);

   display.display();

  }

  ///////////////////////////////////////////////

  void wifiConnected (void){

   display.clearDisplay();

   display.drawBitmap (0,0, wifi_Connected, 128, 64, WHITE);

   display.display();
  //delay (1000);
}
```