LAMPIRAN

Lampiran 1.1 Seluruh Code Program

```
#define BLYNK_TEMPLATE_ID "TMPLl-X2ue2W"
#define BLYNK_TEMPLATE_NAME "IoT"
#define BLYNK_AUTH_TOKEN "UD4ItpqjK5QU3X_IRHW6S_Q4wTytH-Vf"
#define BLYNK_PRINT Serial

#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <TimeLib.h>
#include <NTPClient.h>
#include <WiFiUdp.h>

char auth[] = BLYNK_AUTH_TOKEN;
char ssid[] = "hospot";
char pass[] = "p2x87a8i";

#define ldrPin  36
#define relayPin  26
int ldrValue;
bool Mode = true;
#define  ledKMR1  33
#define  ledKMR2  17
#define  dapur  18
#define relayPin_RM 13
#define SENSOR 23
#define ACTION 16

String jamSekarang;
int jam;
int menit;
int detik;
String hari;
int tanggal;
int bulan;
int tahun;
String dmY;
int waktuuu;

WiFiUDP ntpUDP;
NTPClient timeClient(ntpUDP, "pool.ntp.org", 25200);
```

```
String weekDays[7]={"Minggu", "Senin", "Selasa", "Rabu", "Kamis", "Jum'at",
"Sabtu"};

String months[12]={"Januari", "Februari", "Maret", "April", "Mei", "Juni", "Juli",
"Agustus", "September", "Oktober", "November", "Desember"};

bool led_set[2];
long timer_start_set[2] = {0xFFFF, 0xFFFF};
long timer_stop_set[2] = {0xFFFF, 0xFFFF};
unsigned char weekday_set[2];

long NTP_server;
unsigned char day_of_week;

bool led_status[2];
bool update_blynk_status[2];
bool led_timer_on_set[2];

BlynkTimer timer;

BLYNK_WRITE (V6)
 {
   int pinValue = param.asInt();
      if (pinValue == 1){
    Mode = false;
  }
     if (pinValue == 0) {
   Mode = true;
  }
}

BLYNK_WRITE (V1)
 {
   int val = param.asInt();
      if ( led_timer_on_set[0] == 0 )
          led_set[0] = val;
     else
         update_blynk_status[0] = 1;
 }

BLYNK_WRITE (V2)
 {
   int val = param.asInt();
      if ( led_timer_on_set[1] == 0 )
```

```
            led_set[1] = val;
        else
            update_blynk_status[1] = 1;
  }

BLYNK_WRITE (V3)
 {
   unsigned char week_day;
   TimeInputParam t(param);
      if (t.hasStartTime() && t.hasStopTime() )
 {
timer_start_set[0] = (t.getStartHour() * 60 * 60) + (t.getStartMinute() * 60) +
t.getStartSecond();
timer_stop_set[0] = (t.getStopHour() * 60 * 60) + (t.getStopMinute() * 60) +
t.getStopSecond();

Serial.println(timer_start_set[0]);
Serial.println(String("Start Time: ") +
t.getStartHour() + ":" +
t.getStartMinute() + ":" +
t.getStartSecond());

Serial.println(String("Stop Time: ") +
t.getStopHour() + ":" +
t.getStopMinute() + ":" +
t.getStopSecond());

    for (int i = 1; i <= 7; i++)
 {
    if (t.isWeekdaySelected(i))
 {
    week_day |= (0x01 << (i-1));
       Serial.println(String("Day ") + i + " is selected");
     }
    else
    {
        week_day &= (~(0x01 << (i-1)));
    }
}
    weekday_set[0] = week_day;
   }
   else
  {
timer_start_set[0] = 0xFFFF;
timer_stop_set[0] = 0xFFFF;
```

```
  }
}

BLYNK_WRITE (V4)
 {
   unsigned char week_day;

TimeInputParam t(param);

   if (t.hasStartTime() && t.hasStopTime() )
   {
timer_start_set[1] = (t.getStartHour() * 60 * 60) + (t.getStartMinute() * 60) +
t.getStartSecond();
timer_stop_set[1] = (t.getStopHour() * 60 * 60) + (t.getStopMinute() * 60) +
t.getStopSecond();

Serial.println(String("Start Time: ") +
t.getStartHour() + ":" +
t.getStartMinute() + ":" +
t.getStartSecond());

Serial.println(String("Stop Time: ") +
t.getStopHour() + ":" +
t.getStopMinute() + ":" +
t.getStopSecond());

    for (int i = 1; i <= 7; i++)
   {
    if (t.isWeekdaySelected(i))
   {
     week_day |= (0x01 << (i-1));
       Serial.println(String("Day ") + i + " is selected");
   }
   else
  {
   week_day &= (~(0x01 << (i-1)));
  }
}

  weekday_set[1] = week_day;
  }
  else
  {
timer_start_set[1] = 0xFFFF;
timer_stop_set[1] = 0xFFFF;
```

```
  }
}


BLYNK_WRITE (InternalPinRTC)
{
const unsigned long DEFAULT_TIME = 1357041600;
unsigned long blynkTime = param.asLong();

if (blynkTime >= DEFAULT_TIME)
{
setTime(blynkTime);
timeClient.update();

day_of_week = weekday();

if ( day_of_week == 1 )
day_of_week = 7;
else
day_of_week -= 1;

NTP_server = (timeClient.getHours()* 60 * 60) + (timeClient.getMinutes()* 60) +
timeClient.getSeconds();

time_t epochTime = timeClient.getEpochTime();
jam = timeClient.getHours();
menit = timeClient.getMinutes();
detik = timeClient.getSeconds();
hari = weekDays[timeClient.getDay()];
struct tm *ptm = gmtime ((time_t *)&epochTime);
tanggal = ptm->tm_mday;
bulan = ptm->tm_mon+1;
tahun = ptm->tm_year+1900;

jamSekarang = timeClient.getFormattedTime();
Serial.print("JAM : ");
Serial.println(jamSekarang);

dmY = hari + ", "+ tanggal + " - " + bulan + " - " + tahun;
Serial.print("HARI : ");
Serial.println(dmY);
Serial.println("");
  }
}
```

```
BLYNK_CONNECTED ()
 {
   Blynk.sendInternal("rtc", "sync");
  }



void checkTime()
{
  Blynk.sendInternal("rtc", "sync");
}

void led_mng()
{
bool time_set_overflow;
bool led_status_buf[2];

for (int i=0; i<2; i++)
{
led_status_buf[i] = led_status[i];
time_set_overflow = 0;


if ( timer_start_set[i] != 0xFFFF && timer_stop_set[i] != 0xFFFF)
{

if ( timer_stop_set[i] < timer_start_set[i] ) time_set_overflow = 1;

if (((((time_set_overflow == 0 && (NTP_server >= timer_start_set[i]) &&
(NTP_server < timer_stop_set[i])) ||
(time_set_overflow && ((NTP_server >= timer_start_set[i]) || (NTP_server <
timer_stop_set[i])))) &&
(weekday_set[i] == 0x00 || (weekday_set[i] & (0x01 << (day_of_week - 1) )))) )
{
led_timer_on_set[i] = 1;
}
else
led_timer_on_set[i] = 0;
}
else
led_timer_on_set[i] = 0;

if ( led_timer_on_set[i] )
{
led_status[i] = 1;
```

```
led_set[i] = 0;
}
else
{
led_status[i] = led_set[i];
}

if ( led_status_buf[i] != led_status[i] )
update_blynk_status[i] = 1;
}
digitalWrite(14, !led_status[0]);
digitalWrite(27, !led_status[1]);
}


void blynk_update ()
{
    if ( update_blynk_status[0] )
  {
     update_blynk_status[0] = 0;
    Blynk.virtualWrite(V1, led_status[0]);
  }

   if ( update_blynk_status[1] )
  {
    update_blynk_status[1] = 0;
   Blynk.virtualWrite(V2, led_status[1]);
  }
}


BLYNK_WRITE (V0)
  {
   int value = param.asInt();
     if (value == 1)
    {
       digitalWrite(relayPin_RM, LOW);
    }
     if (value == 0)
   {
       digitalWrite(relayPin_RM, HIGH);
    }
}


BLYNK_WRITE (V5)
```

```
{
  if (Mode == false){
     if (param.asInt() == 0){
        digitalWrite(relayPin, HIGH);
  } else {
        digitalWrite(relayPin, LOW);
  }
 }
}
BLYNK_WRITE (V7)
 {
   int value = param.asInt();
    if (value == 1)
  {
    digitalWrite(ledKMR1, LOW);
   }
    if (value == 0)
  {
    digitalWrite(ledKMR1, HIGH);
   }
}


BLYNK_WRITE (V8)
 {
   int value = param.asInt();
     if (value == 1)
 {
    digitalWrite(ledKMR2, LOW);
  }
     if (value == 0)
 {
   digitalWrite(ledKMR2, HIGH);
  }
}


BLYNK_WRITE (V9)
{
  int value = param.asInt();
    if (value == 1)
{
  digitalWrite(dapur, LOW);
}
     if (value == 0)
{
```

```
    digitalWrite(dapur, HIGH);
  }
}


void sensorLDR()
 {
   if(Mode == true){
      ldrValue = analogRead(ldrPin);
        Serial.print("Nilai sensor LDR : ");
Serial.println(ldrValue);
Blynk.virtualWrite(V1, ldrValue);

     if (ldrValue > 2000){
        digitalWrite(relayPin, LOW);
     }else{
       digitalWrite(relayPin, HIGH);
     }
   }
}


void IR_KM ()
 {
    int L =digitalRead(SENSOR);
       Serial.print("Nilai sensor kamar mandi = ");
       Serial.println(L);

    if(L == 0){
       digitalWrite(ACTION,LOW);
       Serial.print("Lampu ON ,");
    }else{
       digitalWrite(ACTION,HIGH);
       Serial.print("Lampu OFF ,");
   }
   delay(500);
}

void setup()
{
Serial.begin(115200);

pinMode(14, OUTPUT);
pinMode(27, OUTPUT);

pinMode(ldrPin, INPUT);
```

```
pinMode(relayPin, OUTPUT);

pinMode(ledKMR1, OUTPUT);
pinMode(ledKMR2, OUTPUT);
digitalWrite(ledKMR1, HIGH);
digitalWrite(ledKMR2, HIGH);

pinMode(dapur, OUTPUT);
digitalWrite(dapur, HIGH);

pinMode(relayPin_RM, OUTPUT);
digitalWrite(relayPin_RM, HIGH);

pinMode(SENSOR, INPUT_PULLUP);
pinMode(ACTION, OUTPUT);

Blynk.begin(auth, ssid, pass);
timer.setInterval(1000L, checkTime);

}

void loop()
{
 Blynk.run();
 timer.run();
 led_mng();
 blynk_update();
 IR_KM();
 sensorLDR();
}
```