```
#define BLYNK_PRINT Serial
#include "DHT.h"
#include <SPI.h>
#include <Ethernet.h>
#include <BlynkSimpleEthernet.h>
#include <SimpleTimer.h>
#include <VirtualWire.h>
int RF_TX_PIN = 29;   //tx pin voor poort
#define DHTPIN 30 // The pin you connect to
#define DHTTYPE DHT22   // DHT 11 Change this if you have a DHT22
//#define BLYNK_MAX_SENDBYTES 128
DHT dht(DHTPIN, DHTTYPE); // Change this to

// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "d9f7f9893dff4b80bcdd17958e398931";

// drukknoppen
const int btnPin22 = 22;     // drukknop keuken licht
const int btnPin2 = 2;       // drukknop wasplaats licht
const int btnPin3 = 3;       // drukknop wasplaats afzuiging
const int btnPin5 = 5;       // drukknop achtertuin spot 1000w
const int btnPin6 = 6;       // drukknop eetplaats licht
const int btnPin7 = 7;       // drukknop zitplaats zetel licht
const int btnPin9 = 9;       // drukknop zitplaats midden licht
const int btnPin11 = 11;     // drukknop zitplaats tv licht
const int btnPin12 = 12;     // drukknop trap licht
const int btnPin13 = 13;     // drukknop badkamer spiegel licht
const int btnPin14 = 14;     // drukknop badkamer bad licht
const int btnPin15 = 15;     // drukknop badkamer afzuiging
const int btnPin16 = 16;     // drukknop alles aan
const int btnPin17 = 17;     // drukknop alles uit

// relais
const int ledPin31 = 31;      // relais keuken licht
const int ledPin32 = 32;      // relais wasplaats licht
const int ledPin33 = 33;      // relais wasplaats afzuiging
const int ledPin34 = 34;      // relais achtertuin spot 1000w
const int ledPin35 = 35;      // relais eetplaats licht
const int ledPin36 = 36;      // relais zitplaats zetel licht
const int ledPin37 = 37;      // relais zitplaats midden licht
const int ledPin38 = 38;      // relais zitplaats tv licht
const int ledPin39 = 39;      // relais trap licht
const int ledPin40 = 40;      // relais badkamer spiegel licht
const int ledPin41 = 41;      // relais badkamer bad licht
const int ledPin42 = 42;      // relais badkamer afzuiging


SimpleTimer timer;
void checkPhysicalButton22();
void checkPhysicalButton2();
void checkPhysicalButton3();
```

```
void checkPhysicalButton5();
void checkPhysicalButton6();
void checkPhysicalButton7();
void checkPhysicalButton9();
void checkPhysicalButton11();
void checkPhysicalButton12();
void checkPhysicalButton13();
SimpleTimer timer2;
void checkPhysicalButton14();
void checkPhysicalButton15();
void checkPhysicalButton16();
void checkPhysicalButton17();

int ledState31 = LOW;
int ledState32 = LOW;
int ledState33 = LOW;
int ledState34 = LOW;
int ledState35 = LOW;
int ledState36 = LOW;
int ledState37 = LOW;
int ledState38 = LOW;
int ledState39 = LOW;
int ledState40 = LOW;
int ledState41 = LOW;
int ledState42 = LOW;
int ledState44 = HIGH;

int btnState22 = HIGH;
int btnState2 = HIGH;
int btnState3 = HIGH;
int btnState5 = HIGH;
int btnState6 = HIGH;
int btnState7 = HIGH;
int btnState9 = HIGH;
int btnState11 = HIGH;
int btnState12 = HIGH;
int btnState13 = HIGH;
int btnState14 = HIGH;
int btnState15 = HIGH;
int btnState16 = HIGH;
int btnState17 = HIGH;

// Every time we connect to the cloud...
BLYNK_CONNECTED() {
  // Request the latest state from the server
 Blynk.syncVirtual(V22);
 Blynk.syncVirtual(V2);
 Blynk.syncVirtual(V3);
 Blynk.syncVirtual(V5);
 Blynk.syncVirtual(V6);
 Blynk.syncVirtual(V7);
 Blynk.syncVirtual(V9);
```

```
 Blynk.syncVirtual(V11);
 Blynk.syncVirtual(V12);
 Blynk.syncVirtual(V13);
 Blynk.syncVirtual(V14);
 Blynk.syncVirtual(V15);
 Blynk.syncVirtual(V16);
 Blynk.syncVirtual(V17);
 Blynk.syncVirtual(V44);
}

// begin bediening keuken
BLYNK_WRITE(V22) {
  ledState31 = param.asInt();
  digitalWrite(ledPin31, ledState31);
}

void checkPhysicalButton22()
{
  if (digitalRead(btnPin22) == HIGH) {
    // btnState is used to avoid sequential toggles
    if (btnState22 != LOW) {

      // Toggle LED state
      ledState31 = !ledState31;
      digitalWrite(ledPin31, ledState31);

      // Update Button Widget
      Blynk.virtualWrite(V22, ledState31);
    }
    btnState22 = LOW;
  } else {
    btnState22 = HIGH;
  }
}
// einde bediening keuken

// begin bediening wasplaats licht
BLYNK_WRITE(V2) {
  ledState32 = param.asInt();
  digitalWrite(ledPin32, ledState32);
}

void checkPhysicalButton2()
{
  if (digitalRead(btnPin2) == HIGH) {
    // btnState is used to avoid sequential toggles
    if (btnState2 != LOW) {

      // Toggle LED state
      ledState32 = !ledState32;
      digitalWrite(ledPin32, ledState32);
```

```
      // Update Button Widget
      Blynk.virtualWrite(V2, ledState32);
    }
    btnState2 = LOW;
  } else {
    btnState2 = HIGH;
  }
}
// einde bediening wasplaats licht

// begin bediening wasplaats afzuiging
BLYNK_WRITE(V3) {
  ledState33 = param.asInt();
  digitalWrite(ledPin33, ledState33);
}

void checkPhysicalButton3()
{
  if (digitalRead(btnPin3) == HIGH) {
    // btnState is used to avoid sequential toggles
    if (btnState3 != LOW) {

      // Toggle LED state
      ledState33 = !ledState33;
      digitalWrite(ledPin33, ledState33);

      // Update Button Widget
      Blynk.virtualWrite(V3, ledState33);
    }
    btnState3 = LOW;
  } else {
    btnState3 = HIGH;
  }
}
// einde bediening wasplaats afzuiging

// begin bediening achtertuin spot 1000w
BLYNK_WRITE(V5) {
  ledState34 = param.asInt();
  digitalWrite(ledPin34, ledState34);
}

void checkPhysicalButton5()
{
  if (digitalRead(btnPin5) == HIGH) {
    // btnState is used to avoid sequential toggles
    if (btnState5 != LOW) {

      // Toggle LED state
      ledState34 = !ledState34;
      digitalWrite(ledPin34, ledState34);
```

```
      // Update Button Widget
      Blynk.virtualWrite(V5, ledState34);
    }
    btnState5 = LOW;
  } else {
    btnState5 = HIGH;
  }
}
// einde bediening achtertuin spot 1000w

// begin bediening eetplaats licht
BLYNK_WRITE(V6) {
  ledState35 = param.asInt();
  digitalWrite(ledPin35, ledState35);
}

void checkPhysicalButton6()
{
  if (digitalRead(btnPin6) == HIGH) {
    // btnState is used to avoid sequential toggles
    if (btnState6 != LOW) {

      // Toggle LED state
      ledState35 = !ledState35;
      digitalWrite(ledPin35, ledState35);

      // Update Button Widget
      Blynk.virtualWrite(V6, ledState35);
    }
    btnState6 = LOW;
  } else {
    btnState6 = HIGH;
  }
}
// einde bediening eetplaats licht

// begin bediening zitplaats zetel licht
BLYNK_WRITE(V7) {
  ledState36 = param.asInt();
  digitalWrite(ledPin36, ledState36);
}

void checkPhysicalButton7()
{
  if (digitalRead(btnPin7) == HIGH) {
    // btnState is used to avoid sequential toggles
    if (btnState7 != LOW) {

      // Toggle LED state
      ledState36 = !ledState36;
      digitalWrite(ledPin36, ledState36);
```

```
      // Update Button Widget
      Blynk.virtualWrite(V7, ledState36);
    }
    btnState7 = LOW;
  } else {
    btnState7 = HIGH;
  }
}
// einde bediening zitplaats zetel licht

// begin bediening zitplaats midden licht
BLYNK_WRITE(V9) {
  ledState37 = param.asInt();
  digitalWrite(ledPin37, ledState37);
}

void checkPhysicalButton9()
{
  if (digitalRead(btnPin9) == HIGH) {
    // btnState is used to avoid sequential toggles
    if (btnState9 != LOW) {

      // Toggle LED state
      ledState37 = !ledState37;
      digitalWrite(ledPin37, ledState37);

      // Update Button Widget
      Blynk.virtualWrite(V9, ledState37);
    }
    btnState9 = LOW;
  } else {
    btnState9 = HIGH;
  }
}
// einde bediening zitplaats midden licht

// begin bediening zitplaats tv licht
BLYNK_WRITE(V11) {
  ledState38 = param.asInt();
  digitalWrite(ledPin38, ledState38);
}

void checkPhysicalButton11()
{
  if (digitalRead(btnPin11) == HIGH) {
    // btnState is used to avoid sequential toggles
    if (btnState11 != LOW) {

      // Toggle LED state
      ledState38 = !ledState38;
      digitalWrite(ledPin38, ledState38);
```

```
      // Update Button Widget
      Blynk.virtualWrite(V11, ledState38);
    }
    btnState11 = LOW;
  } else {
    btnState11 = HIGH;
  }
}
// einde bediening zitplaats tv licht

// begin bediening trap licht
BLYNK_WRITE(V12) {
  ledState39 = param.asInt();
  digitalWrite(ledPin39, ledState39);
}

void checkPhysicalButton12()
{
  if (digitalRead(btnPin12) == HIGH) {
    // btnState is used to avoid sequential toggles
    if (btnState12 != LOW) {

      // Toggle LED state
      ledState39 = !ledState39;
      digitalWrite(ledPin39, ledState39);

      // Update Button Widget
      Blynk.virtualWrite(V12, ledState39);
    }
    btnState12 = LOW;
  } else {
    btnState12 = HIGH;
  }
}
// einde bediening trap licht

// begin bediening badkamer spiegel licht
BLYNK_WRITE(V13) {
  ledState40 = param.asInt();
  digitalWrite(ledPin40, ledState40);
}

void checkPhysicalButton13()
{
  if (digitalRead(btnPin13) == HIGH) {
    // btnState is used to avoid sequential toggles
    if (btnState13 != LOW) {

      // Toggle LED state
      ledState40 = !ledState40;
      digitalWrite(ledPin40, ledState40);
```

```
      // Update Button Widget
      Blynk.virtualWrite(V13, ledState40);
    }
    btnState13 = LOW;
  } else {
    btnState13 = HIGH;
  }
}
// einde bediening badkamer spiegel licht

// begin bediening badkamer bad licht
BLYNK_WRITE(V14) {
  ledState41 = param.asInt();
  digitalWrite(ledPin41, ledState41);
}

void checkPhysicalButton14()
{
  if (digitalRead(btnPin14) == HIGH) {
    // btnState is used to avoid sequential toggles
    if (btnState14 != LOW) {

      // Toggle LED state
      ledState41 = !ledState41;
      digitalWrite(ledPin41, ledState41);

      // Update Button Widget
      Blynk.virtualWrite(V14, ledState41);
    }
    btnState14 = LOW;
  } else {
    btnState14 = HIGH;
  }
}
// einde bediening badkamer bad licht

// begin bediening badkamer bad afzuiging
BLYNK_WRITE(V15) {
  ledState42 = param.asInt();
  digitalWrite(ledPin42, ledState42);
}

void checkPhysicalButton15()
{
  if (digitalRead(btnPin15) == HIGH) {
    // btnState is used to avoid sequential toggles
    if (btnState15 != LOW) {

      // Toggle LED state
      ledState42 = !ledState42;
      digitalWrite(ledPin42, ledState42);
```

```
      // Update Button Widget
      Blynk.virtualWrite(V15, ledState42);
    }
    btnState15 = LOW;
  } else {
    btnState15 = HIGH;
  }
}
// einde bediening badkamer bad afzuiging


// begin bediening alles uit
BLYNK_WRITE(V16) {
  ledState31 = param.asInt();
  digitalWrite(ledPin31, HIGH);
  Blynk.virtualWrite(V22, HIGH);
  ledState32 = param.asInt();
  digitalWrite(ledPin32, HIGH);
  Blynk.virtualWrite(V2, HIGH);
  ledState33 = param.asInt();
  digitalWrite(ledPin33, HIGH);
  Blynk.virtualWrite(V3, HIGH);
  ledState34 = param.asInt();
  digitalWrite(ledPin34, HIGH);
  Blynk.virtualWrite(V5, HIGH);
  ledState35 = param.asInt();
  digitalWrite(ledPin35, HIGH);
  Blynk.virtualWrite(V6, HIGH);
  ledState36 = param.asInt();
  digitalWrite(ledPin36, HIGH);
  Blynk.virtualWrite(V7, HIGH);
  ledState37 = param.asInt();
  digitalWrite(ledPin37, HIGH);
  Blynk.virtualWrite(V9, HIGH);
  ledState38 = param.asInt();
  digitalWrite(ledPin38, HIGH);
  Blynk.virtualWrite(V11, HIGH);
  ledState39 = param.asInt();
  digitalWrite(ledPin39, HIGH);
  Blynk.virtualWrite(V12, HIGH);
  ledState40 = param.asInt();
  digitalWrite(ledPin40, HIGH);
  Blynk.virtualWrite(V13, HIGH);
  ledState41 = param.asInt();
  digitalWrite(ledPin41, HIGH);
  Blynk.virtualWrite(V14, HIGH);
  ledState42 = param.asInt();
  digitalWrite(ledPin42, HIGH);
  Blynk.virtualWrite(V15, HIGH);
 }
 void checkPhysicalButton16()
{
```

```
  if (digitalRead(btnPin16) == HIGH) {
    // btnState is used to avoid sequential toggles
    if (btnState16 != LOW) {

      ledState31 = !ledState31;
      digitalWrite(ledPin31, HIGH);
      Blynk.virtualWrite(V22, HIGH);
      ledState32 = !ledState32;
      digitalWrite(ledPin32, HIGH);
      Blynk.virtualWrite(V2, HIGH);
      ledState33 = !ledState33;
      digitalWrite(ledPin33, HIGH);
      Blynk.virtualWrite(V3, HIGH);
      ledState34 = !ledState34;
      digitalWrite(ledPin34, HIGH);
      Blynk.virtualWrite(V5, HIGH);
      ledState35 = !ledState35;
      digitalWrite(ledPin35, HIGH);
      Blynk.virtualWrite(V6, HIGH);
      ledState36 = !ledState36;
      digitalWrite(ledPin36, HIGH);
      Blynk.virtualWrite(V7, HIGH);
      ledState37 = !ledState37;
      digitalWrite(ledPin37, HIGH);
      Blynk.virtualWrite(V9, HIGH);
      ledState38 = !ledState38;
      digitalWrite(ledPin38, HIGH);
      Blynk.virtualWrite(V11, HIGH);
      ledState39 = !ledState39;
      digitalWrite(ledPin39, HIGH);
      Blynk.virtualWrite(V12, HIGH);
      ledState40 = !ledState40;
      digitalWrite(ledPin40, HIGH);
      Blynk.virtualWrite(V13, HIGH);
      ledState41 = !ledState41;
      digitalWrite(ledPin41, HIGH);
      Blynk.virtualWrite(V14, HIGH);
      ledState42 = !ledState42;
      digitalWrite(ledPin42, HIGH);
      Blynk.virtualWrite(V15, HIGH);
     }
    btnState16 = LOW;
  } else {
    btnState16 = HIGH;
  }
}
// einde bediening alles uit

// begin bediening alles aan
BLYNK_WRITE(V17) {
  ledState31 = param.asInt();
  digitalWrite(ledPin31, LOW);
```

```
  Blynk.virtualWrite(V22, LOW);
  ledState32 = param.asInt();
  digitalWrite(ledPin32, LOW);
  Blynk.virtualWrite(V2, LOW);
  ledState33 = param.asInt();
  digitalWrite(ledPin33, LOW);
  Blynk.virtualWrite(V3, LOW);
  ledState34 = param.asInt();
  digitalWrite(ledPin34, LOW);
  Blynk.virtualWrite(V5, LOW);
  ledState35 = param.asInt();
  digitalWrite(ledPin35, LOW);
  Blynk.virtualWrite(V6, LOW);
  ledState36 = param.asInt();
  digitalWrite(ledPin36, LOW);
  Blynk.virtualWrite(V7, LOW);
  ledState37 = param.asInt();
  digitalWrite(ledPin37, LOW);
  Blynk.virtualWrite(V9, LOW);
  ledState38 = param.asInt();
  digitalWrite(ledPin38, LOW);
  Blynk.virtualWrite(V11, LOW);
  ledState39 = param.asInt();
  digitalWrite(ledPin39, LOW);
  Blynk.virtualWrite(V12, LOW);
  ledState40 = param.asInt();
  digitalWrite(ledPin40, LOW);
  Blynk.virtualWrite(V13, LOW);
  ledState41 = param.asInt();
  digitalWrite(ledPin41, LOW);
  Blynk.virtualWrite(V14, LOW);
  ledState42 = param.asInt();
  digitalWrite(ledPin42, LOW);
  Blynk.virtualWrite(V15, LOW);
   }
   void checkPhysicalButton17()
{
  if (digitalRead(btnPin17) == HIGH) {
    // btnState is used to avoid sequential toggles
    if (btnState17 != LOW) {

      // Toggle LED state
  digitalWrite(ledPin31, LOW);
  Blynk.virtualWrite(V22, LOW);
  digitalWrite(ledPin32, LOW);
  Blynk.virtualWrite(V2, LOW);
  digitalWrite(ledPin33, LOW);
  Blynk.virtualWrite(V3, LOW);
  digitalWrite(ledPin34, LOW);
  Blynk.virtualWrite(V5, LOW);
  digitalWrite(ledPin35, LOW);
  Blynk.virtualWrite(V6, LOW);
```

```
  digitalWrite(ledPin36, LOW);
  Blynk.virtualWrite(V7, LOW);
  digitalWrite(ledPin37, LOW);
  Blynk.virtualWrite(V9, LOW);
  digitalWrite(ledPin38, LOW);
  Blynk.virtualWrite(V11, LOW);
  digitalWrite(ledPin39, LOW);
  Blynk.virtualWrite(V12, LOW);
  digitalWrite(ledPin40, LOW);
  Blynk.virtualWrite(V13, LOW);
  digitalWrite(ledPin41, LOW);
  Blynk.virtualWrite(V14, LOW);
  digitalWrite(ledPin42, LOW);
  Blynk.virtualWrite(V15, LOW);
     }
  }
// einde bediening alles aan

}


void setup()
{
 vw_set_tx_pin(RF_TX_PIN); // Setup transmit pin
 vw_setup(2000); // Transmission speed in bits per second.
 pinMode(26, INPUT);
 pinMode(13, OUTPUT);


  // Debug console
  Serial.begin(9600);

  Blynk.begin(auth);

  pinMode(ledPin31, OUTPUT);
  pinMode(ledPin32, OUTPUT);
  pinMode(ledPin33, OUTPUT);
  pinMode(ledPin34, OUTPUT);
  pinMode(ledPin35, OUTPUT);
  pinMode(ledPin36, OUTPUT);
  pinMode(ledPin37, OUTPUT);
  pinMode(ledPin38, OUTPUT);
  pinMode(ledPin39, OUTPUT);
  pinMode(ledPin40, OUTPUT);
  pinMode(ledPin41, OUTPUT);
  pinMode(ledPin42, OUTPUT);

  pinMode(btnPin22, INPUT);
  pinMode(btnPin2, INPUT);
  pinMode(btnPin3, INPUT);
  pinMode(btnPin5, INPUT);
  pinMode(btnPin6, INPUT);
```

```
  pinMode(btnPin7, INPUT);
  pinMode(btnPin9, INPUT);
  pinMode(btnPin11, INPUT);
  pinMode(btnPin12, INPUT);
  pinMode(btnPin13, INPUT);
  pinMode(btnPin14, INPUT);
  pinMode(btnPin15, INPUT);
  pinMode(btnPin16, INPUT);
  pinMode(btnPin17, INPUT);

  digitalWrite(ledPin31, ledState31);
  digitalWrite(ledPin32, ledState32);
  digitalWrite(ledPin33, ledState33);
  digitalWrite(ledPin34, ledState34);
  digitalWrite(ledPin35, ledState35);
  digitalWrite(ledPin36, ledState36);
  digitalWrite(ledPin37, ledState37);
  digitalWrite(ledPin38, ledState38);
  digitalWrite(ledPin39, ledState39);
  digitalWrite(ledPin40, ledState40);
  digitalWrite(ledPin41, ledState41);
  digitalWrite(ledPin42, ledState42);

  // Setup a function to be called every 100 ms
  timer.setInterval(100L, checkPhysicalButton22);
  timer.setInterval(100L, checkPhysicalButton2);
  timer.setInterval(100L, checkPhysicalButton3);
  timer.setInterval(100L, checkPhysicalButton5);
  timer.setInterval(100L, checkPhysicalButton6);
  timer.setInterval(100L, checkPhysicalButton7);
  timer.setInterval(100L, checkPhysicalButton9);
  timer.setInterval(100L, checkPhysicalButton11);
  timer.setInterval(100L, checkPhysicalButton12);
  timer.setInterval(100L, checkPhysicalButton13);
  timer2.setInterval(100L, checkPhysicalButton14);
  timer2.setInterval(100L, checkPhysicalButton15);
  timer2.setInterval(100L, checkPhysicalButton16);
  timer2.setInterval(100L, checkPhysicalButton17);
  timer2.setInterval(2000, sendDHT);
}
void sendDHT()
{
//Read the Temp and Humidity from DHT
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  int hum = (int) h;
  int tem = (int) t;
//Write values to V04 and V05
  Blynk.virtualWrite(30, hum);
  Blynk.virtualWrite(31, tem);
}
void loop()
```

prog def rf433

```
{
  Blynk.run();
  timer.run();
  timer2.run();
{
 int sensorVal = digitalRead(26);
 if (sensorVal == HIGH) {
   digitalWrite(13, HIGH);
   const char *msg = "0";
   vw_send((uint8_t *)msg, strlen(msg));
   delay(20);
 }
 else {
   digitalWrite(13, LOW);
   const char *msg = "1";
   vw_send((uint8_t *)msg, strlen(msg));
   delay(20);
 }
}
}
```