

IOT
**Energieverbruik
helper**

hightech

Versie 1.0, April 2020

Deze handleiding werd ontwikkeld door **voornaam + naam**.
voor Maakbib (STEM-partnerschap VLAIO)



en valt onder de Creative Commons licentie



www.maakbib.be

www.decreativestem.be

www.vlaio.be/nl

www.stem-academie.be

Wat?

Tijd:

+3u

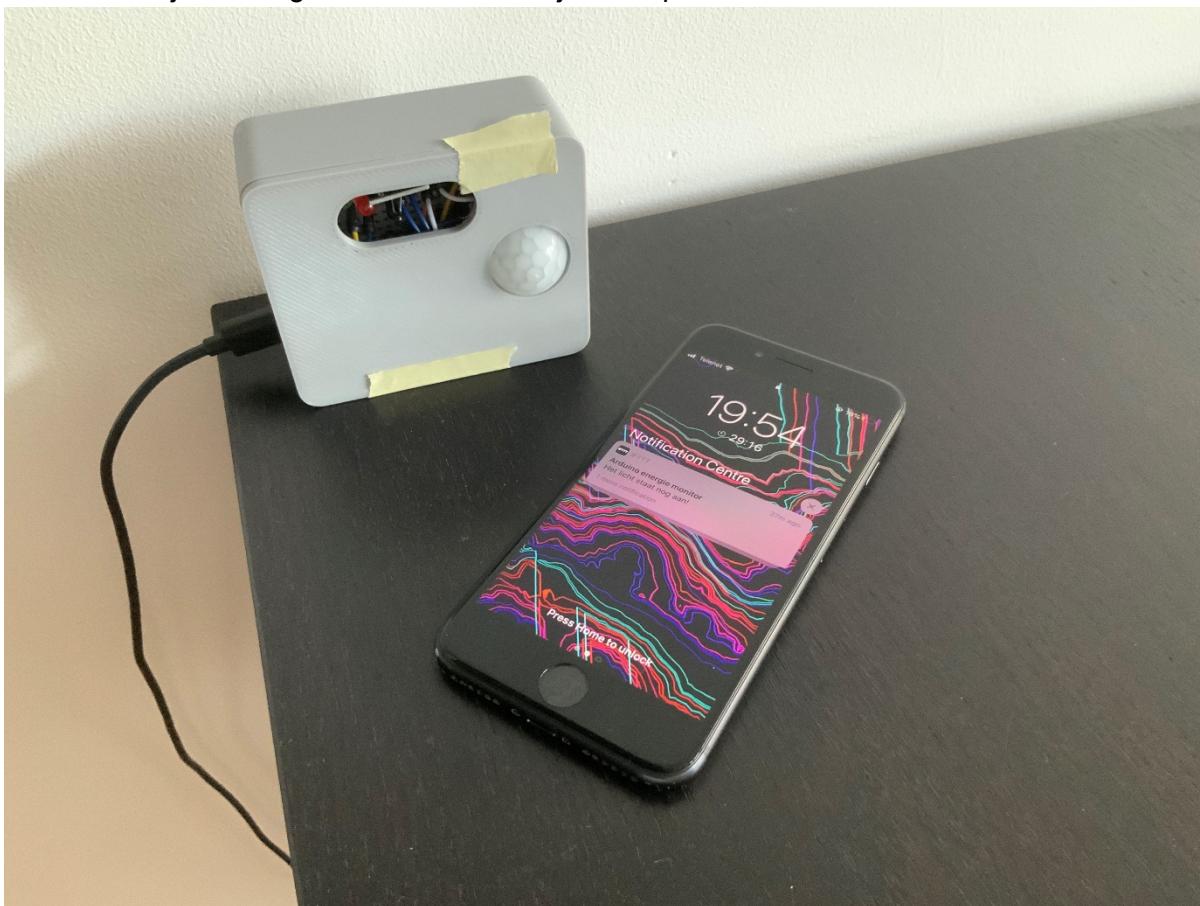
Soort activiteit:

Individueel

Maak je eigen IOT energieverbruik hulp, en krijg een melding op je smartphone als het licht of de verwarming nog aan staat wanneer er tussen een bepaald tijdsinterval niemand meer in de kamer is.

Leer hoe de ESP32 programmeert met de Arduino IDE

Ontdekt hoe je meldingen kan sturen naar je smartphone met Arduino.



(Afgewerkte IOT energieverbruik helper in behuizing naast smartphone met melding)

Inhoud

Stap 1: Arduino software configureren

Stap 2 : Componenten assembleren

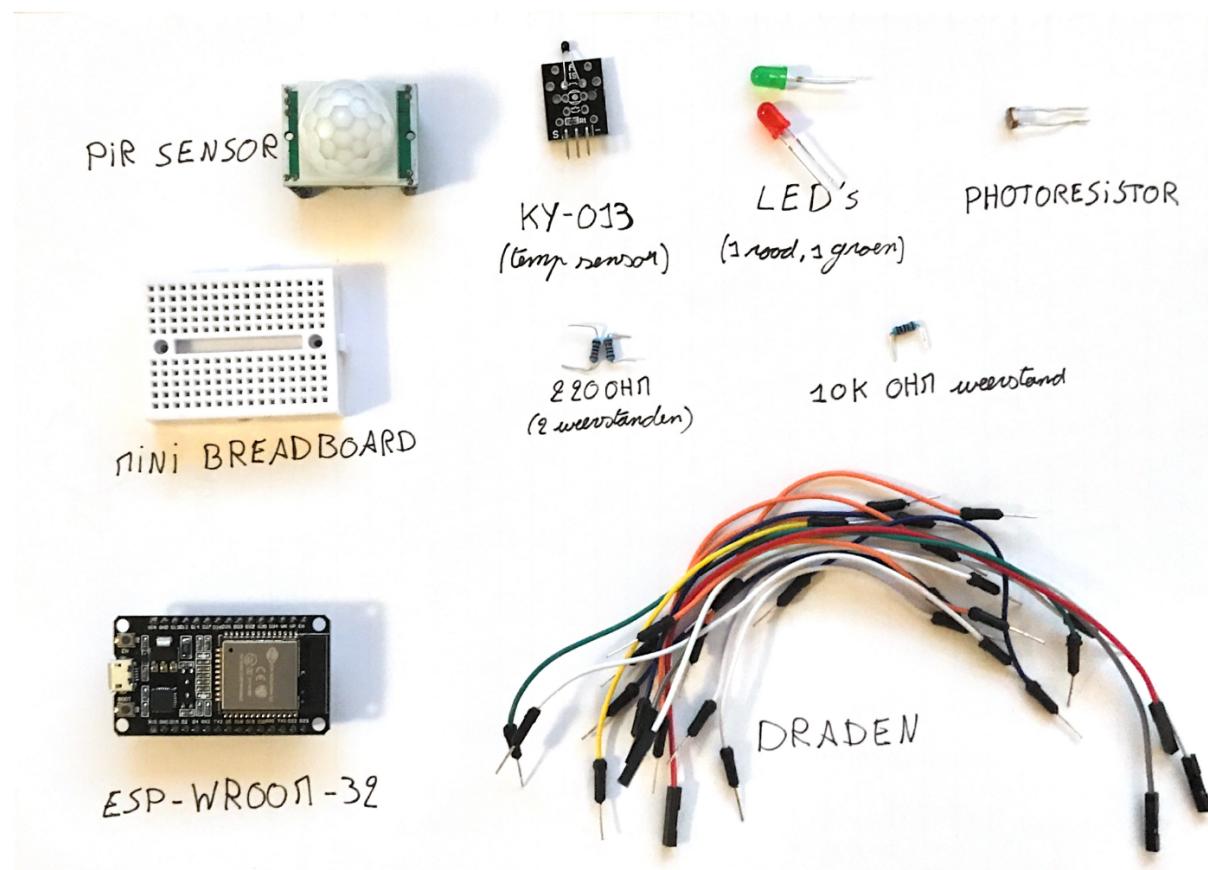
Stap 3: GSM melding sturen

Stap 4: Finale Code

Stap 5: De behuizing

Stap 6: Plaatsen voor gebruik

Materiaal



*PIR sensor

<https://www.kiwi-electronics.nl/pir-bewegingssensor>

*KY-013 temperatuur sensor

<https://www.cheaptch.nl/myxl-ky-13-analog-temperature-sensor-module-voor-a.html>

Photoresistor

https://nl.banggood.com/20Pcs-5MM-Light-Dependent-Resistor-Photoresistor-GL5528-LDR-p-943459.html?gpla=1&gmcCountry=BE¤cy=EUR&createTmp=1&utm_source=googleshopping&utm_medium=cpc_bgcs&utm_content=xibei&utm_campaign=xibei-pla-beg-pc-nl-all-0314&gclid=CjwKCAjw_LL2BRAkEiwAv2Y3SdOIPPEAAAnN2u16GF3x10iapwzTBTNUGWogxzv3UXWETyHte3GWtUWRoCeOwQAvD_BwE&cur_warehouse=CN

Mini Breadboard

Draden

Weerstanden (10k OHM en 220 OHM)

LED's

https://nl.banggood.com/Basic-Starter-Kit-UNO-R3-Mini-Breadboard-LED-Jumper-Wire-Button-With-Box-For-Geekcreit-for-Arduino-products-that-work-with-official-Arduino-boards-p-1161006.html?gpla=1&gmcCountry=BE¤cy=EUR&createTmp=1&utm_source=googleshopping&utm_medium=cpc_bgcs&utm_content=xibei&utm_campaign=xibei-ssc-beg-nl-ele-

[1221&gclid=CjwKCAjw_LL2BRAkEiwAv2Y3SWfGPpbN1Vc-jm9PcLict3B1BX1qEFMdl5EISaTiZtk60txfWzG6jhoC7UsQAvD_BwE&cur_warehouse=CN](https://www.bol.com/nl/p/esp-wroom-32-esp32-esp-32s-ontwikkelbord-development-board-2-4-ghz-dual-mode-wifi-bluetooth-dual-cores-microcontroller-processor-geintegreerd-met-antenne-rf-amp-filter-ap-sta-voor-arduino-ide/9200000114634593/?country=BE)

ESP-WROOM-32

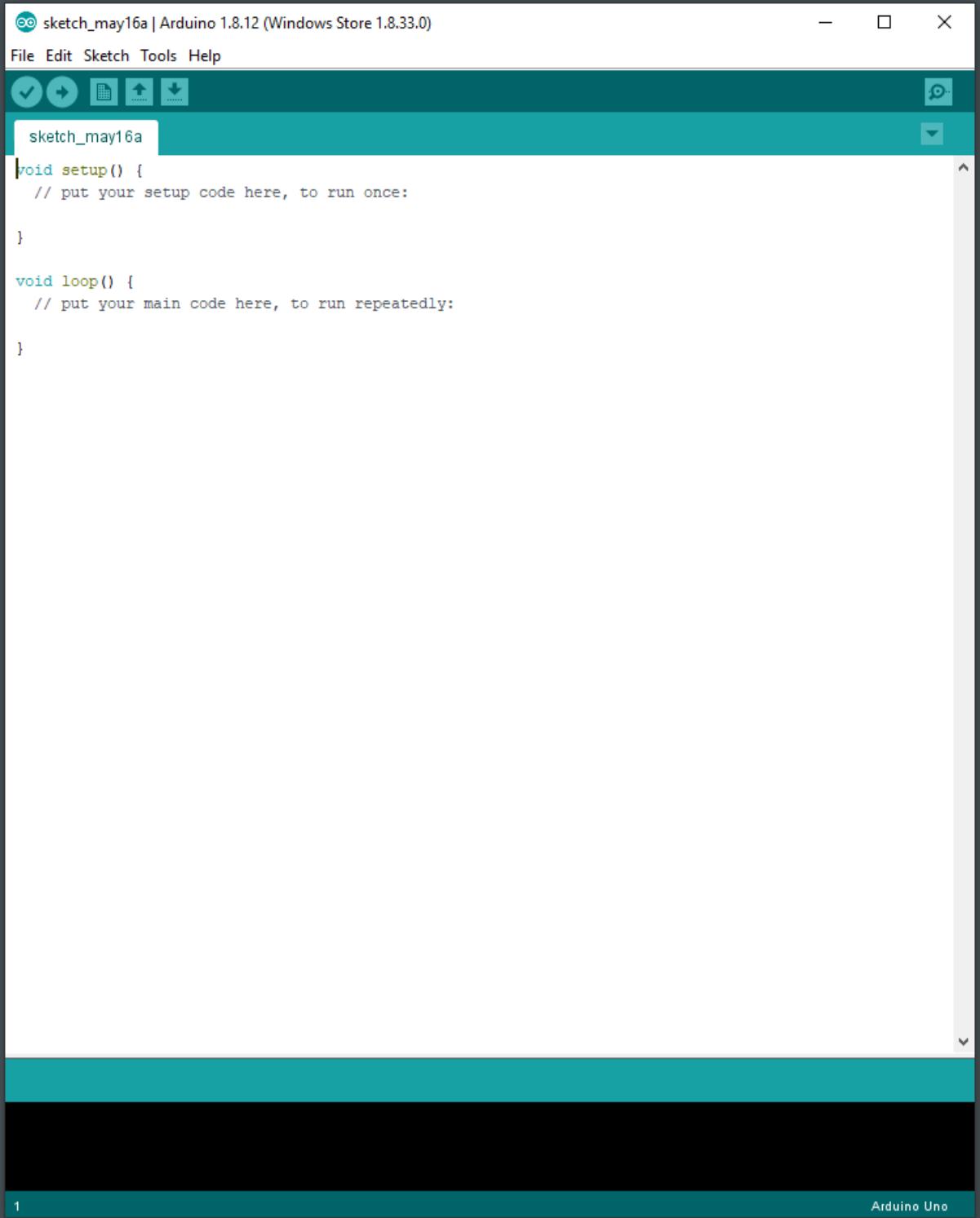
<https://www.bol.com/nl/p/esp-wroom-32-esp32-esp-32s-ontwikkelbord-development-board-2-4-ghz-dual-mode-wifi-bluetooth-dual-cores-microcontroller-processor-geintegreerd-met-antenne-rf-amp-filter-ap-sta-voor-arduino-ide/9200000114634593/?country=BE>

Tools

- *Computer
- *Micro usb kabel
- *Wifi
- *smartphone
- *3Dprinter
- *Lijmpistool

Stap 1: Arduino software configureren

1.1 Start **Arduino IDE** op, Dit is het programma om de Arduino Code in te schrijven. En ziet er zo uit:



The screenshot shows the Arduino IDE interface. The title bar reads "sketch_may16a | Arduino 1.8.12 (Windows Store 1.8.33.0)". The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for save, undo, redo, and upload. The main workspace shows a code editor with the following content:

```
void setup() {
  // put your setup code here, to run once:

}

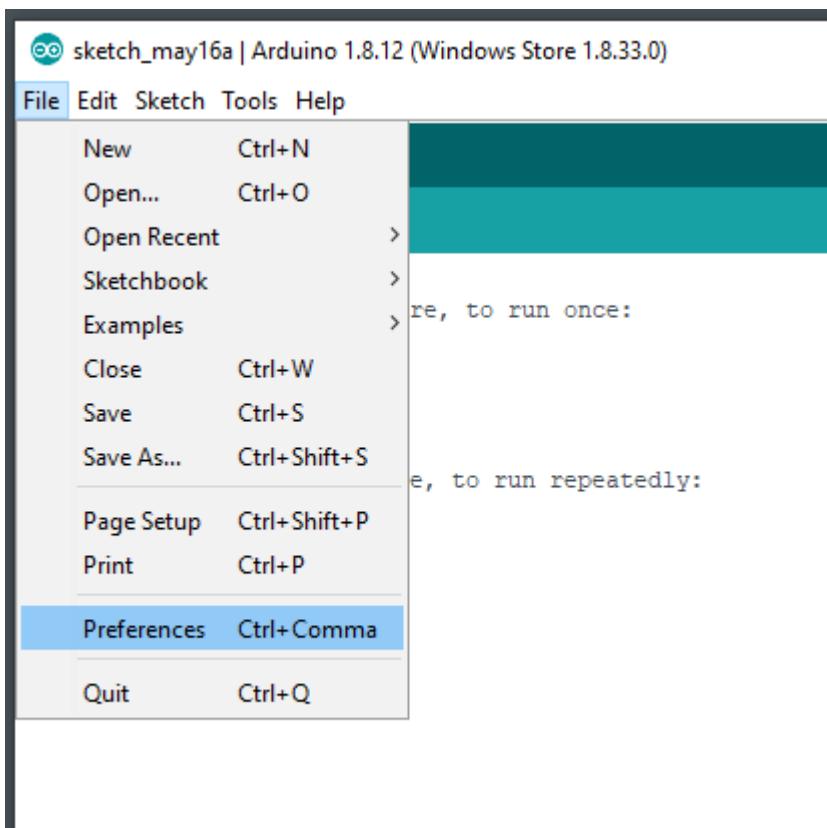
void loop() {
  // put your main code here, to run repeatedly:

}
```

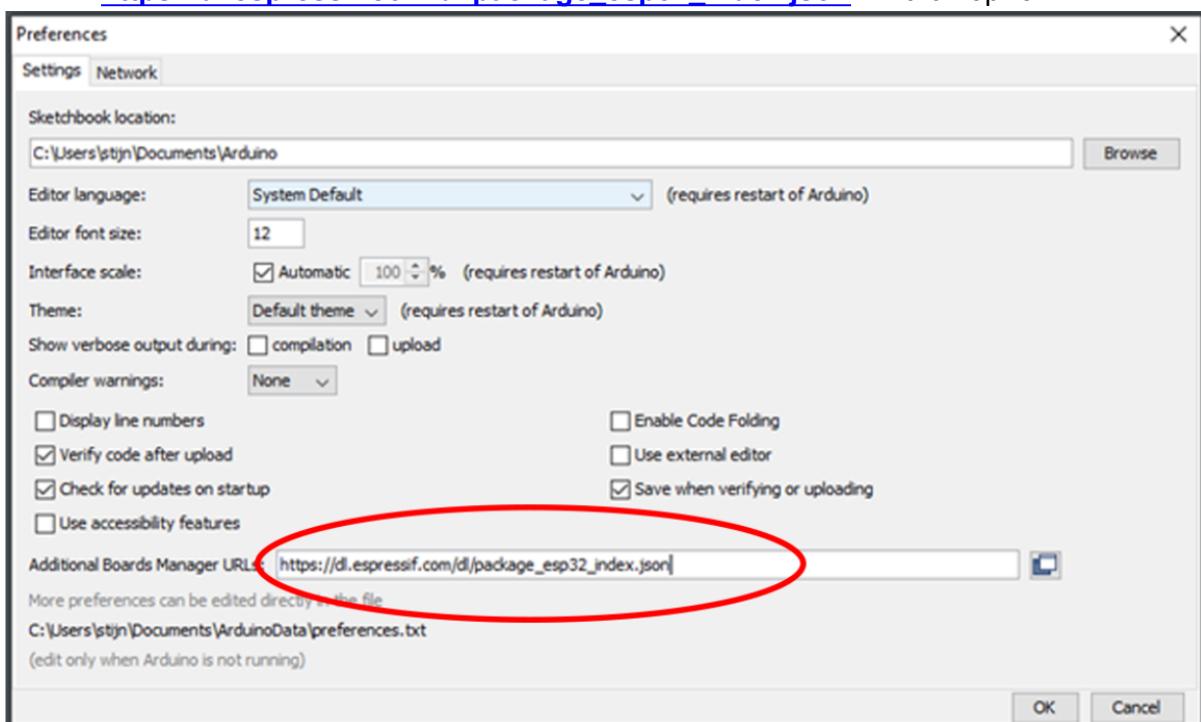
The status bar at the bottom indicates "1" and "Arduino Uno".

Als de Arduino IDE nog niet op je computer geïnstalleerd is, kan je het hier gratis downloaden: <https://www.arduino.cc/en/Main/Software>

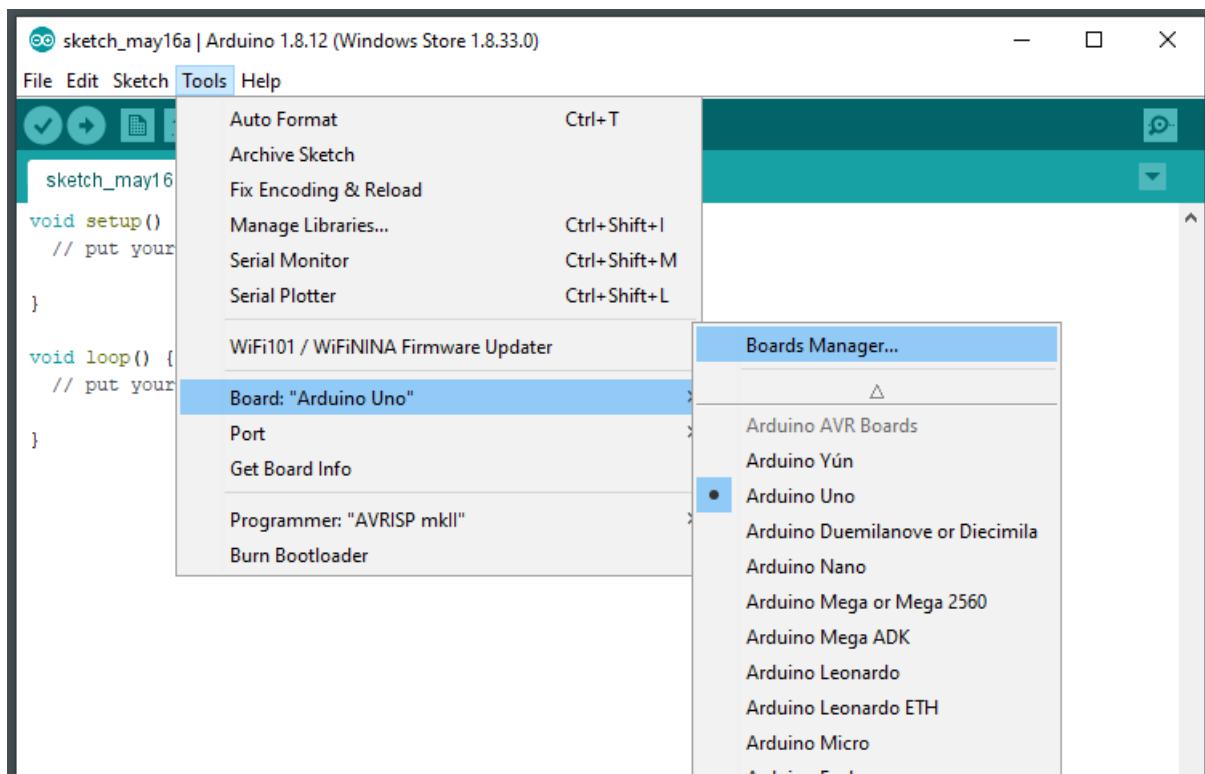
1.2 Ga naar de **Preferences**, onder FILE→PREFERENCES



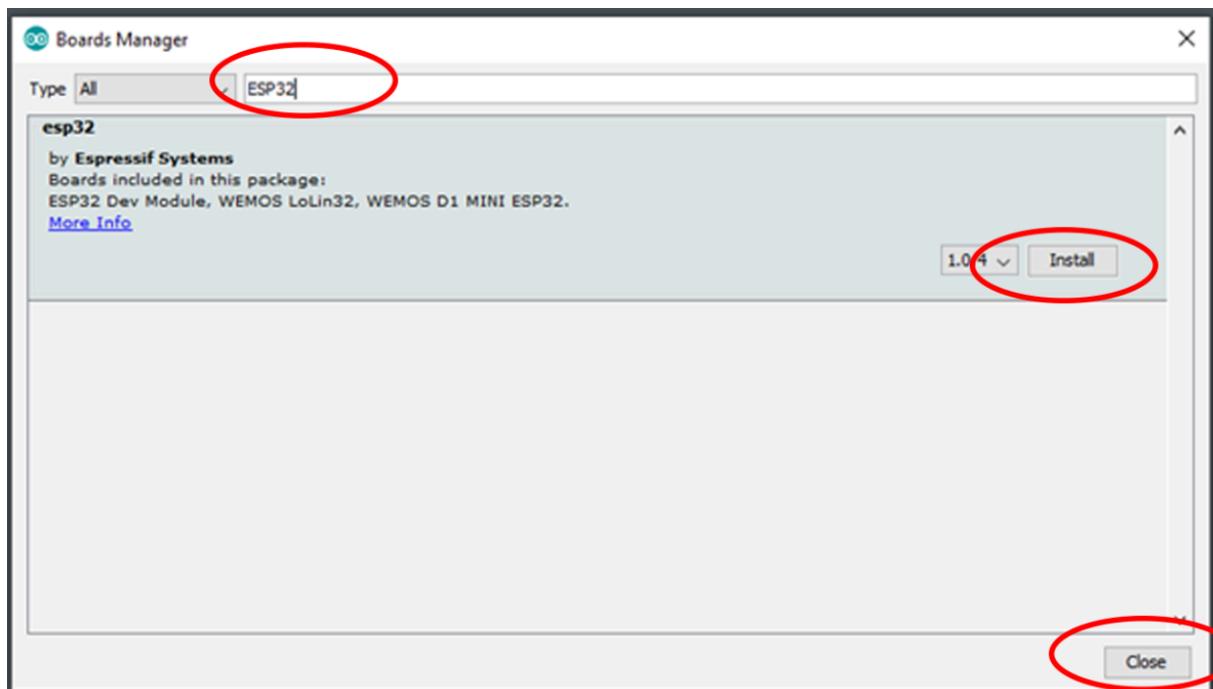
**1.3 Copy Paste het volgende in het “Additional Boards Manager URLs” vak.
https://dl.espressif.com/dl/package_esp32_index.json** En druk op “ok”.



1.4 Open de Board Manager, onder TOOLS→BOARD: “....”→ Boards Manager



1.5 Typ “ESP32” in de zoekbalk en installeer het pakket.



En druk op “close”

1.6 Test de Arduino

Verbind de Arduino met de Computer via de usb kabel.

1.7 Open het voorbeeld “Wifi Scan” , onder File → Examples→WiFi→WiFiScan

The screenshot shows the Arduino IDE interface. The title bar reads "WiFiScan | Arduino 1.8.12 (Windows Store 1.8.33.0)". The menu bar includes File, Edit, Sketch, Tools, and Help. The "File" menu is open, showing options like New, Open..., Open Recent, Sketchbook, Examples, Close, Save, Save As..., Page Setup, Print, Preferences, and Quit. The "Examples" option is highlighted. A code editor window displays the "WiFiScan" sketch, which uses the WiFi library to scan for available networks. The code includes comments explaining the setup and loop functions. In the bottom-left corner of the code editor, there is a message: "Done uploading." followed by several lines of serial output showing the progress of writing data to memory. To the right of the code editor, a large context menu is open under the "Examples" heading. This menu is organized into sections: "Built-in Examples", "Examples for any board", "Examples for ESP32 Dev Module", and "Examples from Custom Libraries". The "WiFiScan" option is visible in the "Examples for WiFi" section, which is part of the "WiFi" category. The "WiFiScan" option is highlighted with a blue selection bar.

```
WiFiScan | Arduino 1.8.12 (Windows Store 1.8.33.0)
File Edit Sketch Tools Help
New Ctrl+N
Open... Ctrl+O
Open Recent >
Sketchbook >
Examples >
Close Ctrl+W
Save Ctrl+S
Save As... Ctrl+Shift+S
Page Setup Ctrl+Shift+P
Print Ctrl+P
Preferences Ctrl+Comma
Quit Ctrl+Q
void setup() {
    WiFi.disconnect();
    delay(100);

    Serial.println("Setup");
}

void loop()
{
    Serial.println("scan start");

    // WiFi.scanNetworks will return the number of networks found
    int n = WiFi.scanNetworks();
    Serial.println("scan done");
    if (n == 0) {
        Serial.println("no networks found");
    } else {
        Serial.print(n);
        Serial.println(" networks found");
        for (int i = 0; i < n; i++) {
            // Print SSID
            Serial.print(i);
            Serial.print(" ");
            Serial.print(WiFi.SSID(i));
            Serial.print(" (");
            Serial.print(WiFi.RSSI(i));
            Serial.print(")");
            Serial.println("");
            delay(10);
        }
    }
    Serial.println("");
}

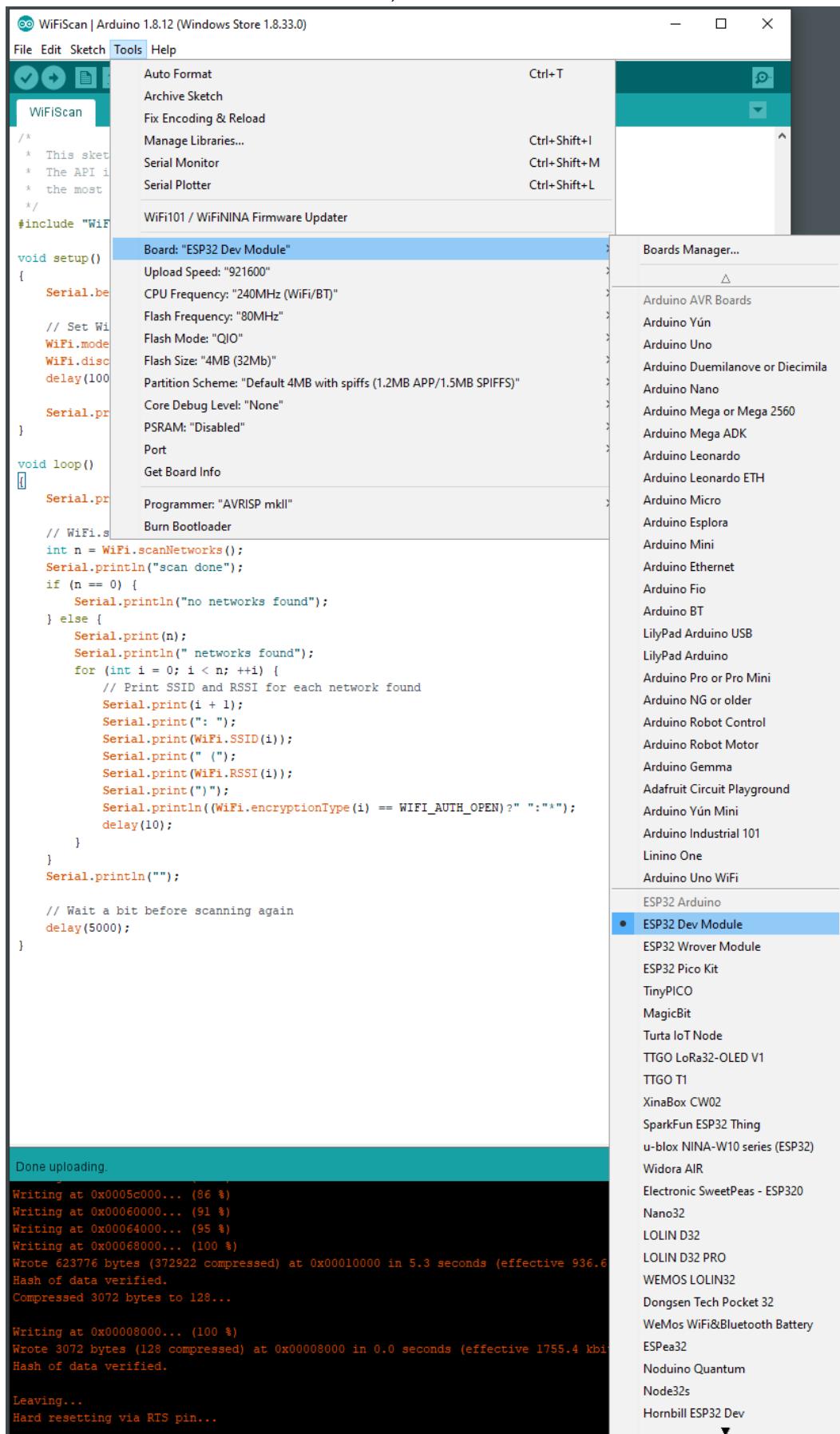
// Wait a bit before scanning again
delay(5000);
}

Done uploading.
Writing at 0x0005c000... (8)
Writing at 0x00060000... (9)
Writing at 0x00064000... (9)
Writing at 0x00068000... (1)
Wrote 62376 bytes (372922
Hash of data verified.
Compressed 3072 bytes to 12
Writing at 0x00008000... (100 %)

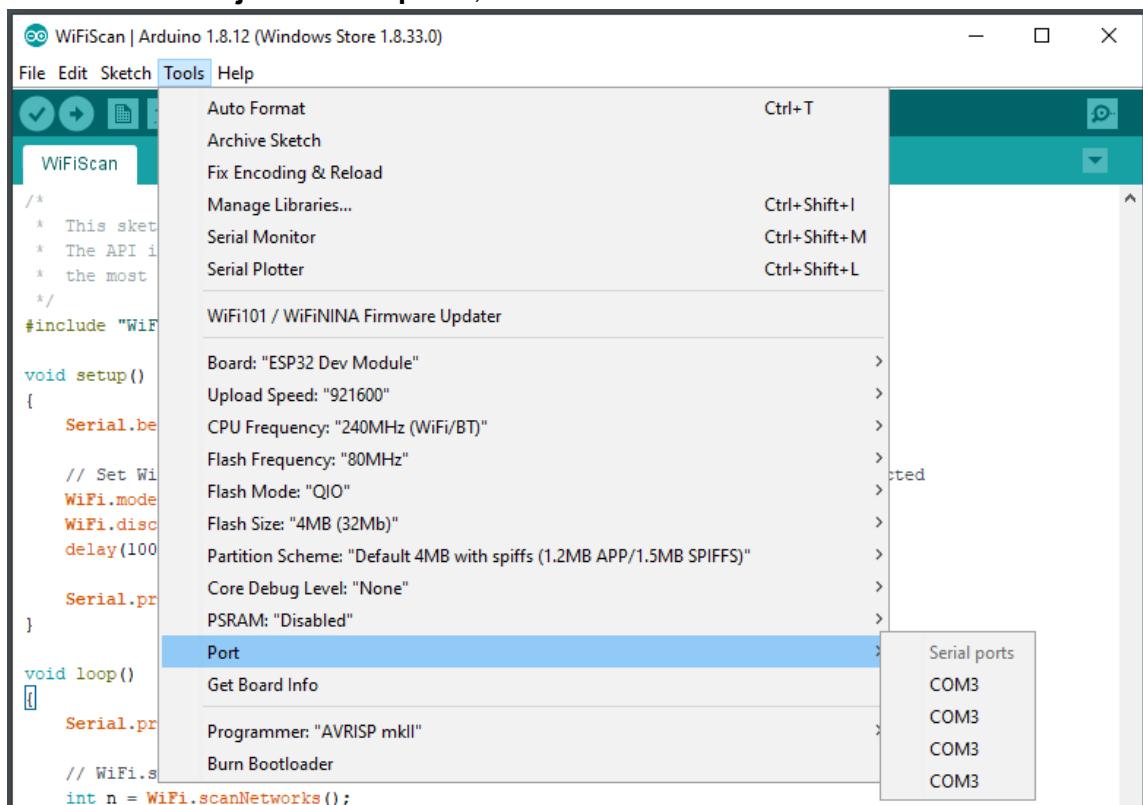
File Edit Sketch Tools Help
Examples >
Built-in Examples
01.Basics >
02.Digital >
03.Analog >
04.Communication >
05.Control >
06.Sensors >
07.Display >
08.Strings >
09.USB >
10.StarterKit_BasicKit >
11.ArduinoISP >
Examples for any board
Adafruit Circuit Playground >
Bridge >
Ethernet >
Firmata >
LiquidCrystal >
SD >
Stepper >
Temboo >
RETIRED >
Examples for ESP32 Dev Module
ArduinoOTA >
BluetoothSerial >
DNSServer >
EEPROM >
ESP32 >
ESP32 Async UDP >
ESP32 Azure IoT Arduino >
ESP32 BLE Arduino >
ESPmDNS >
FFat >
HTTPClient >
HTTPUpdate >
NetBIOS >
Preferences >
SD(esp32) >
SD_MMC >
SimpleBLE >
SPI >
SPIFFS >
Ticker >
Update >
WebServer >
WiFi >
WiFiClientSecure >
Examples from Custom Libraries
Adafruit NeoPixel >
Time >

- □ ×
WiFiScan
WiFiSmartConfig
WiFiTelnetToSerial
WiFiUDPClient
WPS
```

1.8 Selecteer “ESP32 Dev Module”, onder Tools→Board”...”→ESP32 Dev Module

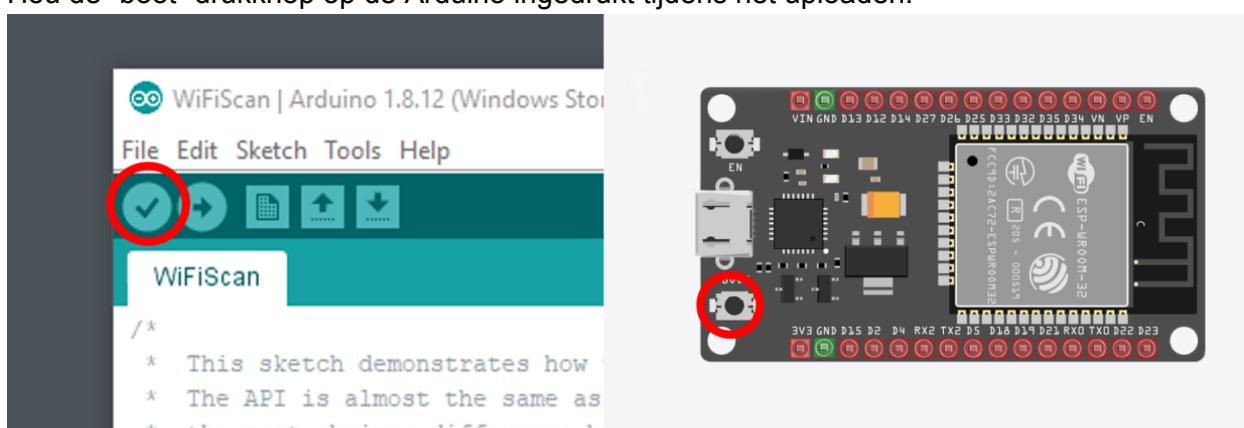


1.9 Selecteer de juiste COM poort, onder Tools → Port

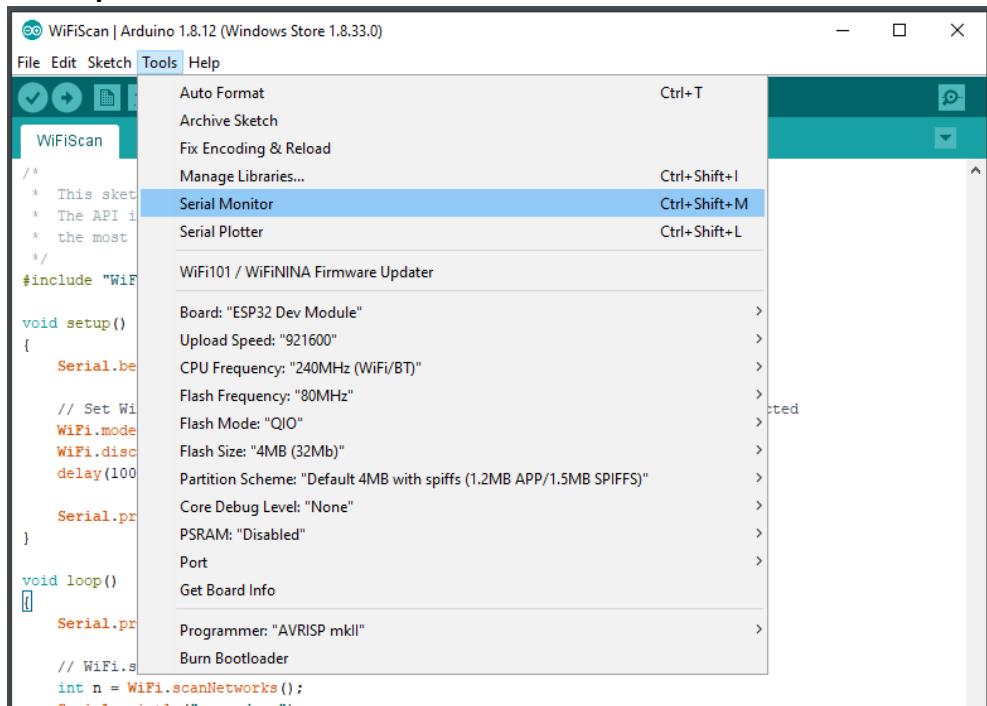


1.10 Upload de code naar de Arduino, door op de pijl links boven de drukken.

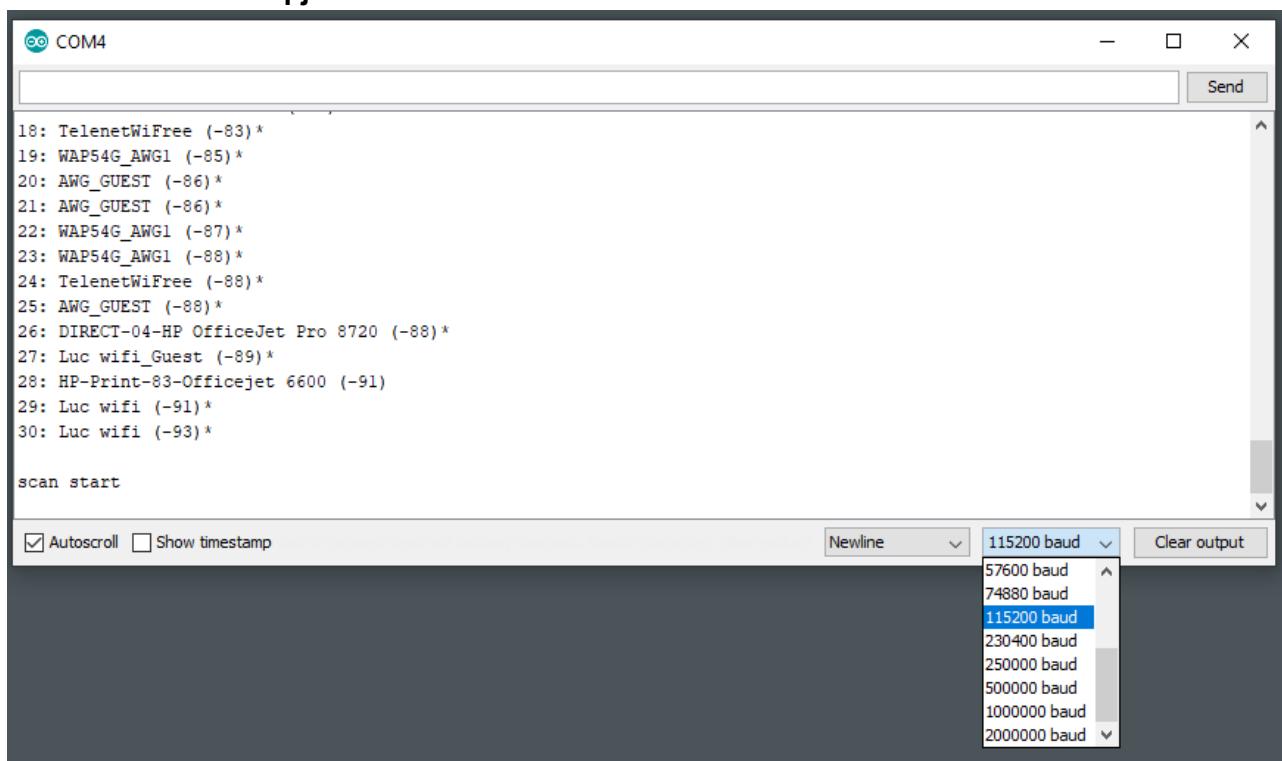
Hou de "boot" drukknop op de Arduino ingedrukt tijdens het uploaden.



1.11 Open de serial monitor, onder Tools→ Serial Monitor

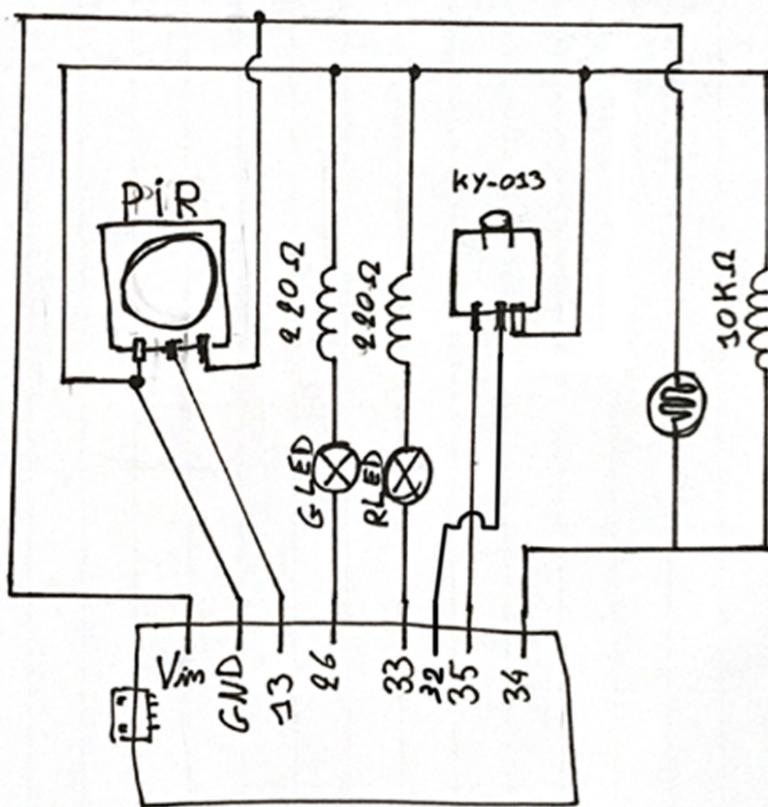
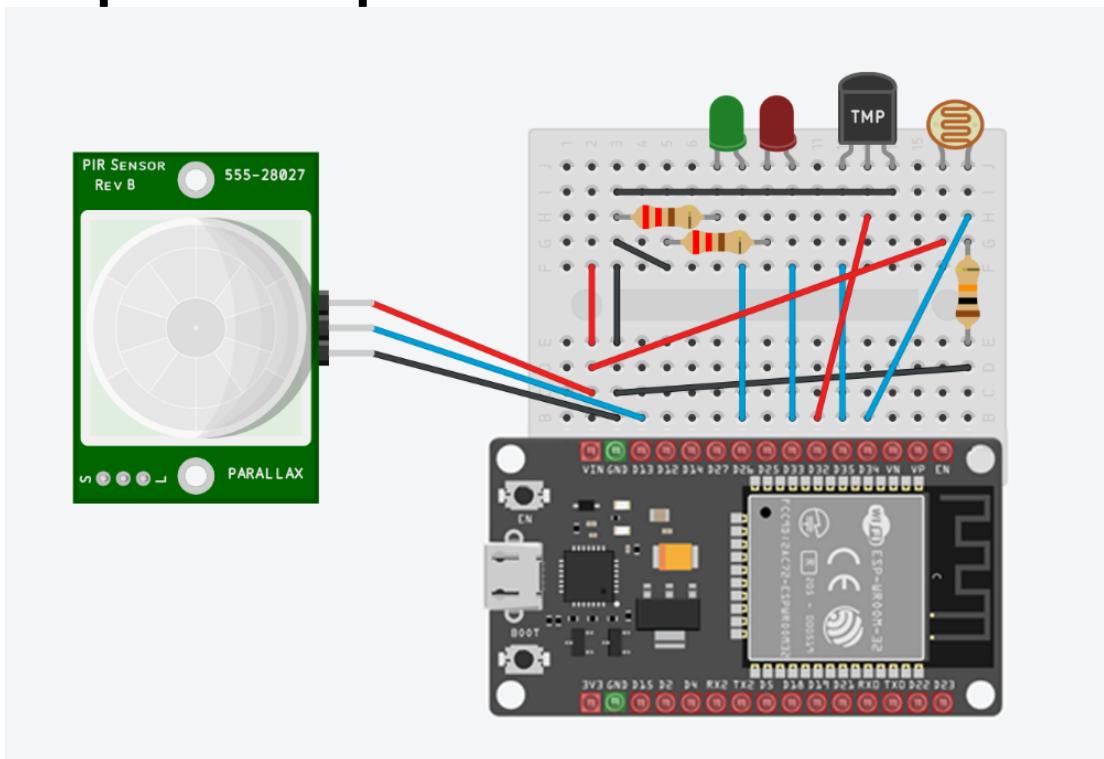


1.12 Zet het Baud tapje naar 115200

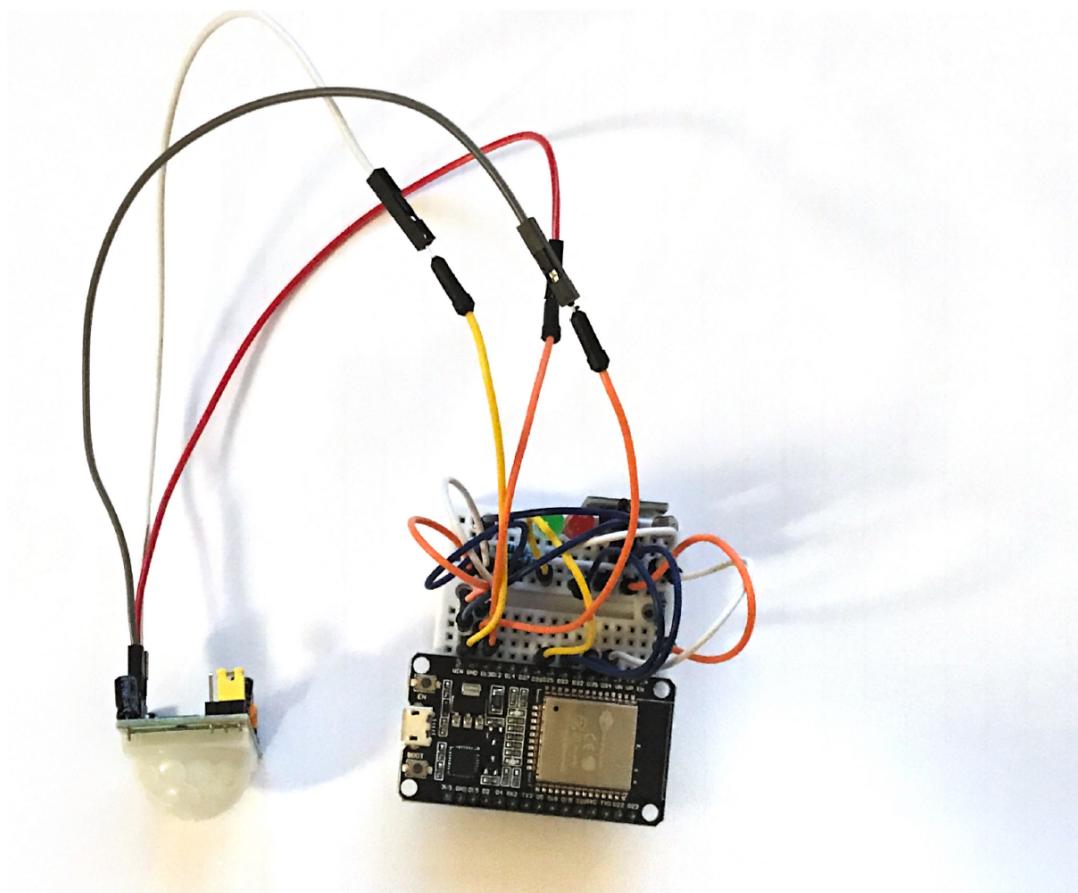


Als je nu een lijst krijgt van alle wifi netwerken in de buurt zoals hier boven,
werk alles zoals het moet!

Stap 2 : Componenten assembleren



Assembleer alle componenten zoals hierboven weergegeven.



Als alles goed in elkaar gestoken is zou het geheel er ongeveer zo moeten uit zien.

Om nu te testen of we alles juist hebben verbonden en alles werkt, gaan we een test Code uploaden.

Deze gaat de waarden van de inputs (temperatuur sensor, lichtsensor en bewegingssensor) in de Serial monitor weer geven. En de groene en rode led zullen aan gaan wanneer er beweging gedetecteerd wordt.

De testcode (op de foto hieronder) is terug te vinden in de bijlage onder de naam “Aansluitingen_test”, zo moet je die niet overtypen!

2.1 open de testcode “Aansluitingen_test”

The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** Aansluitingen_test | Arduino 1.8.12 (Windows Store 1.8.33.0)
- Menu Bar:** File Edit Sketch Tools Help
- Toolbar:** Includes icons for Open, Save, Print, and Upload.
- Sketch Editor:** Displays the code for 'Aansluitingen_test'. The code initializes pins for GreenLED (26), RedLED (33), PIRsens (13), OnBoardLED (2), TempSens (35), and LichtSens (34). It sets up Serial communication at 115200 bps. The setup() function initializes pins and starts serial output. The loop() function reads analog values from LichtSens and TempSens, prints them to the serial monitor, and checks the PIR sensor status. If motion is detected, it prints "beweging gedetecteerd".

```
const int GreenLED = 26;
const int RedLED = 33;
const int PIRsens = 13;
const int OnBoardLED= 2;
const int TempSens = 35;
const int LichtSens = 34;
int TempSensValue = 0;
int LichtSensValue = 0;

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

    pinMode(GreenLED, OUTPUT);
    pinMode(RedLED, OUTPUT);
    //pinMode(PIRsens, INPUT);
    |
    pinMode(32, OUTPUT);
    digitalWrite(32, HIGH);

    digitalWrite(GreenLED, LOW);
    digitalWrite(RedLED, LOW);

    //int beweging_status ;
}

void loop()
{

    LichtSensValue = analogRead(LichtSens);
    TempSensValue = analogRead (TempSens);

    Serial.print( "Tempvalue:");
    Serial.print ( TempSensValue);
    Serial.print( "    ");

    Serial.print( "Lichtvalue:");
    Serial.print ( LichtSensValue);
    Serial.print( "    ");

    int beweging_status = digitalRead(PIRsens);
    digitalWrite(GreenLED, beweging_status);
    digitalWrite(RedLED, beweging_status);
    if (beweging_status == HIGH)
    {
        Serial.println("beweging gedetecteerd");
    }
    else {
        Serial.println ("geen beweging");
    }

    delay(1000);
}

Done uploading.
Wrote 214992 bytes (109795 compressed) at 0x000010000 in 1.6 seconds (effective 1067.0 KBPS)...
Hash of data verified.
Compressed 3072 bytes to 128...
Writing at 0x00008000... (100 %)
```
- Serial Monitor:** Shows the upload progress and a message indicating the hash of the data was verified.
- Status Bar:** Shows the text "ESP32 Dev Module on COM4" and the number 19.

2.2 Upload de test code, vergeet niet op de “boot” knop op de Arduino te drukken tijdens het uploaden.

2.3 Open de serial monitor, en zet de baud op 115200.

```
Tempvalue:4095 Lichtvalue:1207 geen beweging
Tempvalue:4095 Lichtvalue:1676 beweging gedetecteerd
Tempvalue:4095 Lichtvalue:2327 beweging gedetecteerd
Tempvalue:4095 Lichtvalue:2310 geen beweging
Tempvalue:4095 Lichtvalue:1712 geen beweging
Tempvalue:4095 Lichtvalue:1685 beweging gedetecteerd
Tempvalue:4095 Lichtvalue:2319 beweging gedetecteerd
Tempvalue:4095 Lichtvalue:2316 geen beweging
Tempvalue:4095 Lichtvalue:1717 geen beweging
Tempvalue:4095 Lichtvalue:1699 geen beweging
Tempvalue:4095 Lichtvalue:1712 geen beweging
Tempvalue:4095 Lichtvalue:1712 geen beweging
Tempvalue:4095 Lichtvalue:1717 geen beweging
Tempvalue:4095 Lichtvalue:1717 geen beweging
Tempvalue:4095 Lichtvalue:1714 geen beweging
Tempvalue:4095 Lichtvalue:1715 geen beweging
Tempvalue:4095 Lichtvalue:1717 geen beweging
Tempvalue:4095 Lichtvalue:1712 geen beweging
Tempvalue:4095 Lichtvalue:1708 geen beweging
```

Autoscroll Show timestamp Newline 115200 baud Clear output

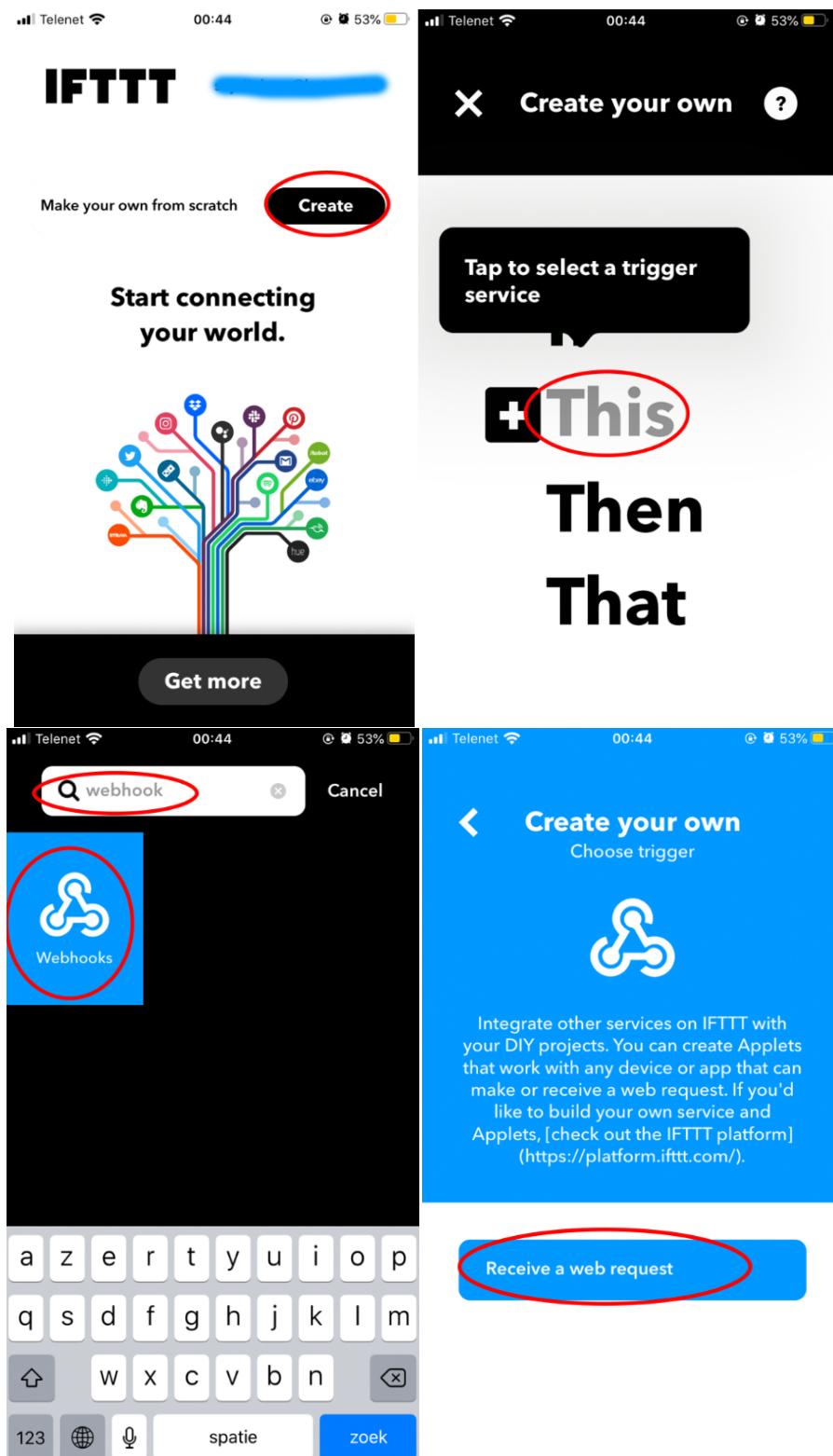
Als de serial monitor er zo uit ziet en de LED's aan gaan wanneer ja voor de bewegingssensor zwaait is alles in orde!

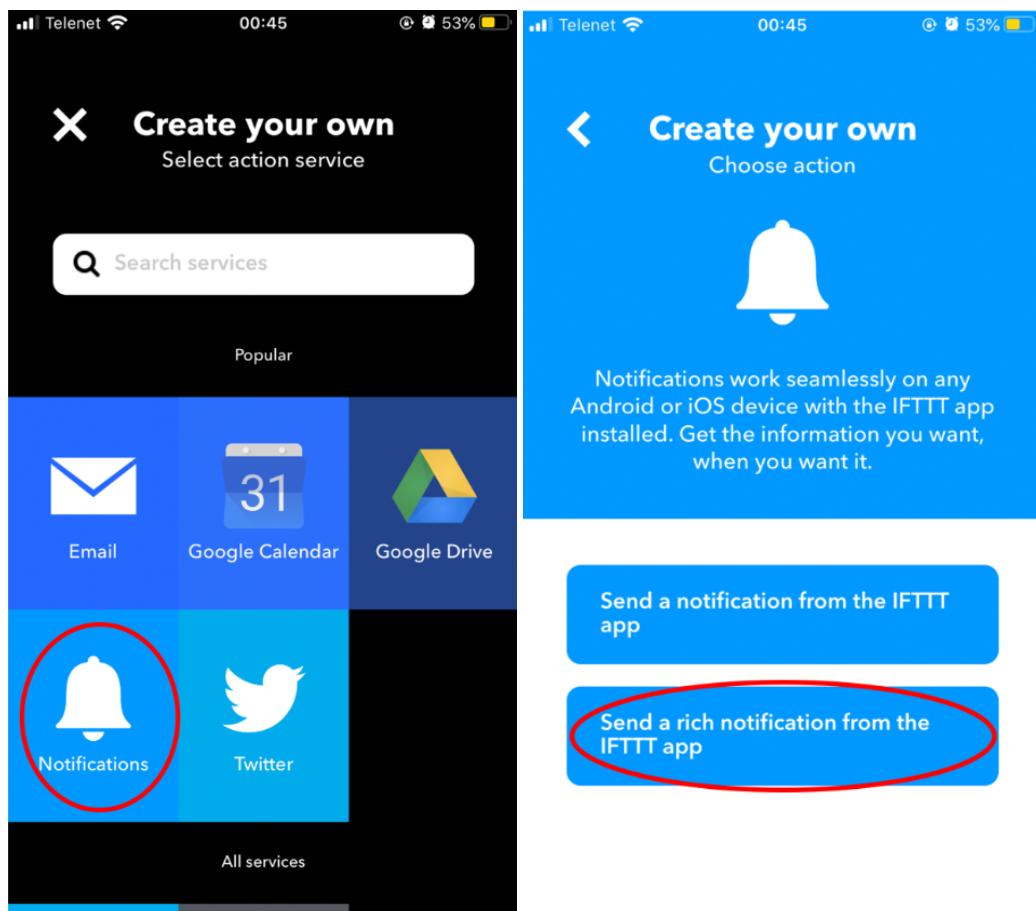
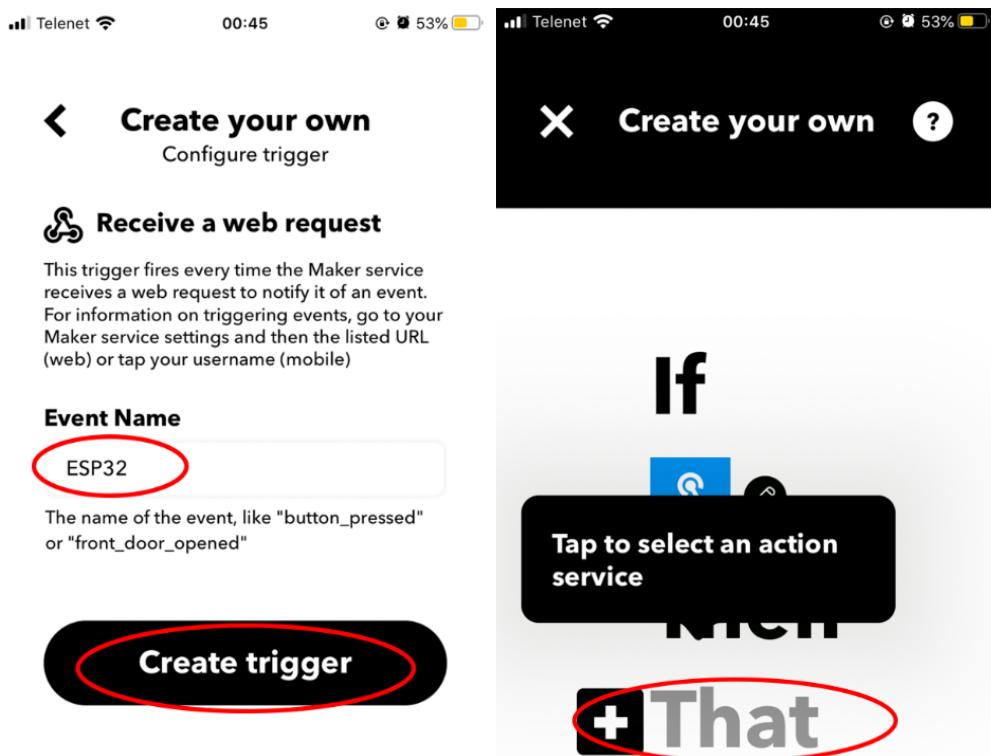
En kunnen we nu verder naar het volgende hoofdstuk: **GSM meldingen sturen.**

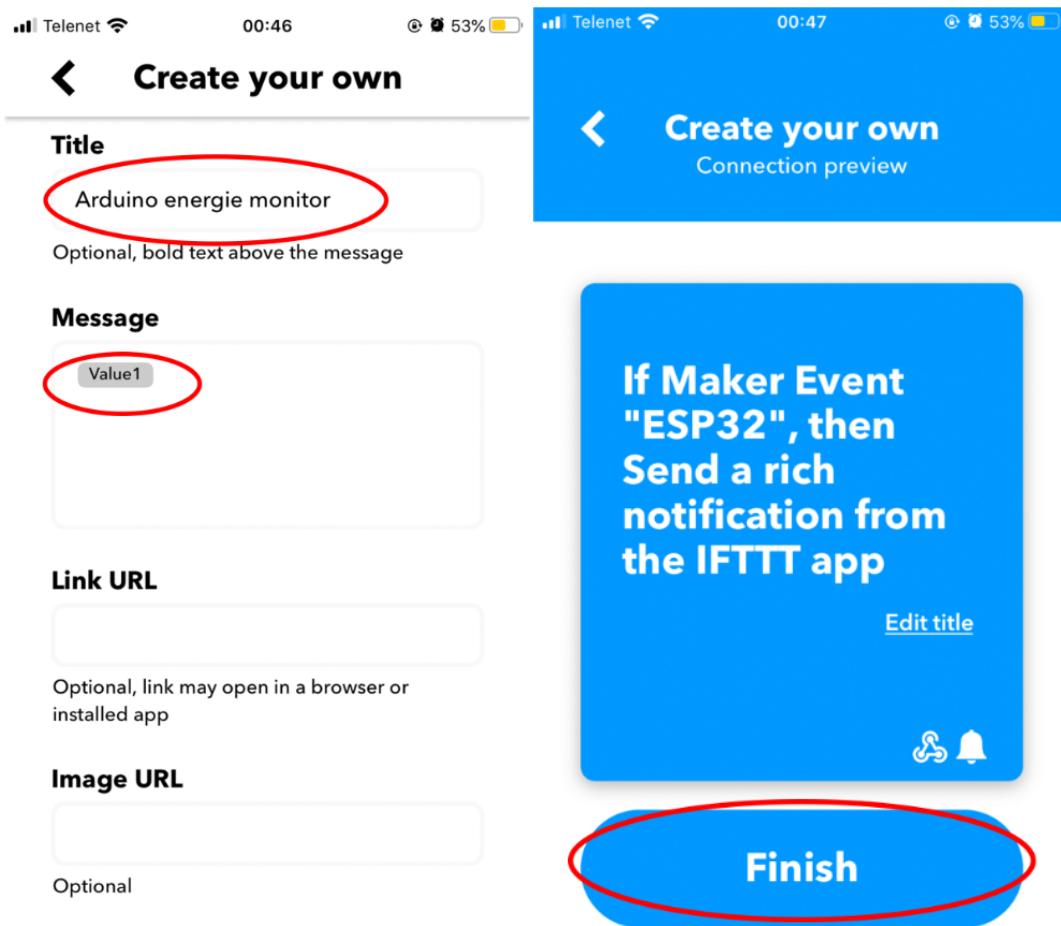
Stap 3: GSM melding sturen

3.1 Download de IFTTT APP op je smartphone en maak een account.

3.2 Maak de applet



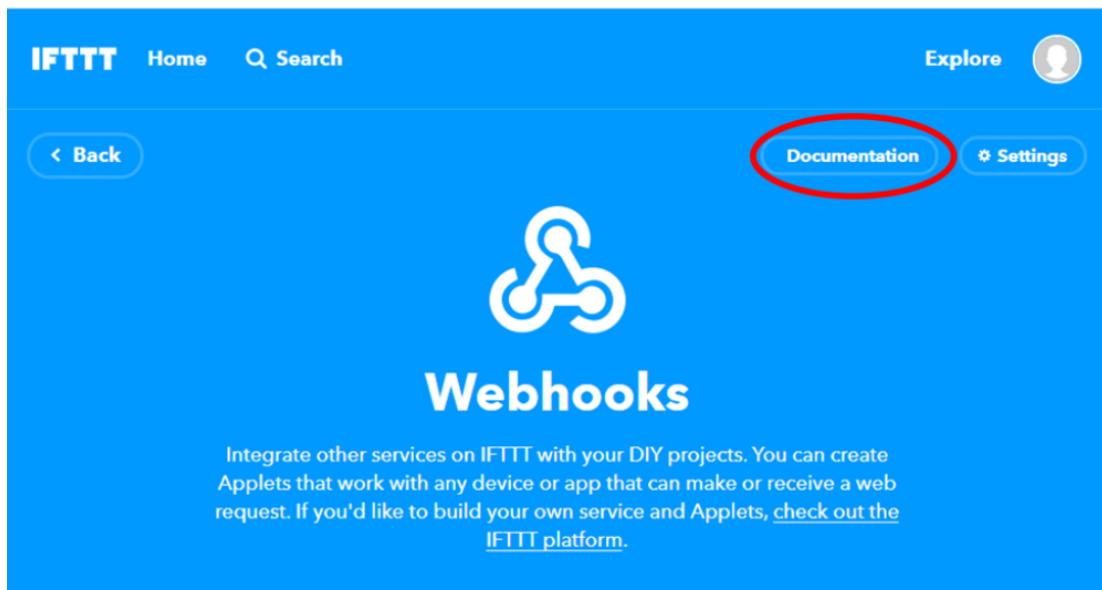




Laat de “link URL” en “image URL” leeg

3.3 Je Persoonlijke IFTTT Code opvragen

Surf naar: https://ifttt.com/maker_webhooks
check dat je ingelogd bent!
En druk op “Documentation”, dan opent er een webpagina met jouw Persoonlijke Code.

A screenshot of the IFTTT Webhooks trigger key page. It shows a blue header with the IFTTT logo and the word "Webhooks". Below it, a message says "Your key is: dX9xoRWDZvllhJ7M" (the key is highlighted with a red oval). There are back and forward navigation buttons. The main content area is titled "To trigger an Event" and provides instructions for making a POST or GET web request to the URL `https://maker.ifttt.com/trigger/{event}/with/key/dX9xoRWDZvllhJ7M`. It also shows an example JSON body: `{ "value1": "████████", "value2": "████████", "value3": "████████" }`. A note states that optional parameters can be passed as query parameters or form variables. A curl command is provided for testing: `curl -X POST https://maker.ifttt.com/trigger/{event}/with/key/dX9xoRWDZvllhJ7M`. A "Test It" button is at the bottom.

Arduino test Code

Om te testen of alles juist is ingesteld, gaan we een test Code uploaden naar de Arudino.

3.4 Open de code “ GSM_notificatie_test” (deze is te vinden in de bijlage)

The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** GMS_notificatie_test | Arduino 1.8.12 (Windows Store 1.8.33.0)
- Menu Bar:** File Edit Sketch Tools Help
- Toolbar:** Includes icons for Open, Save, Upload, and Download.
- Code Editor:** Displays the C++ code for the sketch. The code initializes WiFi, connects to a network, and sends a POST request to an IFTTT trigger URL. It includes comments explaining the code's purpose and parameters.
- Serial Monitor:** Shows the upload progress and the output of the uploaded code. The output indicates the code was uploaded successfully and is now running on the ESP32 Dev Module connected via COM4.

```
#include <WiFi.h>
#include <HTTPClient.h>

const char* ssid = "WIFI_NAAM";
const char* password = "wifi_wachtwoord";
String KEY = "XXXXXXXXXXXXXX";
String Event_marker = "ESP32";

const char* serverName = "http://maker.ifttt.com/trigger/ESP32/with/key/dX9xoRWDZvlIhJ7MHpGkHF";

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

    WiFi.begin(ssid, password);
    Serial.println("Connecting");
    while(WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.println("");
    Serial.print("Connected to WiFi network:");
    Serial.println(ssid);

    if(WiFi.status()== WL_CONNECTED) //Check WiFi connection status
    {
        HTTPClient http;
        http.begin(serverName); // Your Domain name with URL path or IP address with port

        http.addHeader("Content-Type", "application/x-www-form-urlencoded"); // Specify content-type header
        String httpRequestData = "value1=" + String("Test notificaties Arduino"); // Data to send with HTTP POST
        int httpResponseCode = http.POST(httpRequestData); // Send HTTP POST request

        Serial.print("HTTP Response code: ");
        Serial.println(httpResponseCode);

        http.end(); // Free resources
    }
    else {
        Serial.println("WiFi Disconnected");
    }
}

void loop() {
```

Done uploading.
Compressed 3072 bytes to 128...
Writing at 0x00008000... (100 %)
Wrote 3072 bytes (128 compressed) at 0x00008000 in 0.0 seconds (effective 2048.0 kbit/s)...
Hash of data verified.
Leaving...
Hard resetting via RTS pin...
8
ESP32 Dev Module on COM4

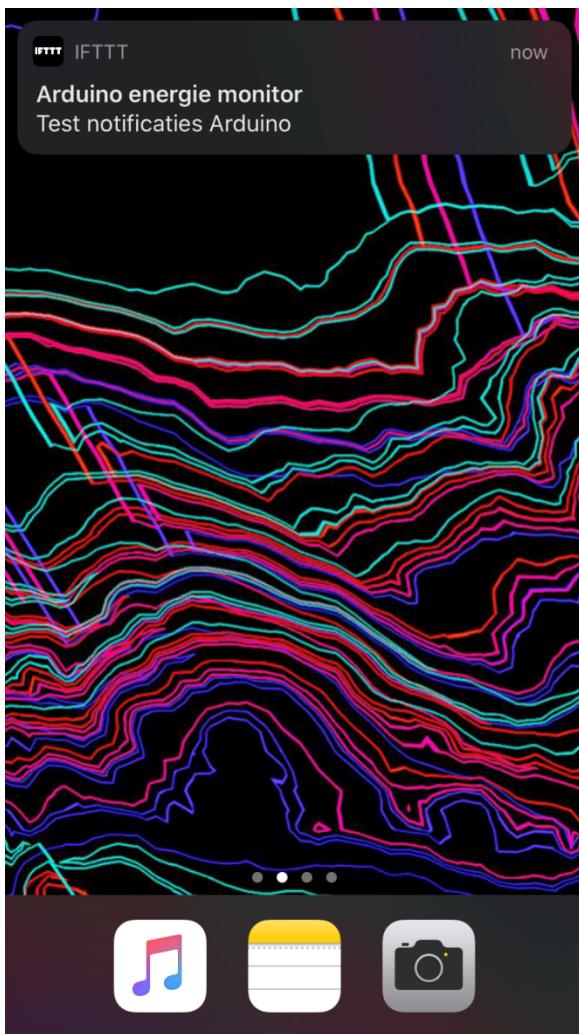
3.5 Vervang (WIFI_NAAM) door je eigen wifi netwerk, bv.: telenet-80415.

Vervang (wifi_wachtwoord) door je eigen wifi wachtwoord.

Vervang (XXXXXXXXXXXX) door de persoonlijke Code die je in stap 3 hebt opgevraagd.

3.6 Upload de code naar de Arduino, vergeet niet de “boot” drukknop ingedrukt te houden tijdens het uploaden.

3.7 Ontvang een melding!



Als alles goed is verlopen zou je na enkele seconden een melding moeten krijgen op de Smartphone.

Stap 4: Finale Code

4.1 Open de finale code, deze is te vinden in de bijlage als “volledige_code”.

The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** Volledige_code | Arduino 1.8.12 (Windows Store 1.8.33.0)
- Menu Bar:** File Edit Sketch Tools Help
- Code Area:** The main code area contains C++ code for an ESP32. The code includes declarations for WiFi, HTTPClient, and TimeLib libraries. It defines constants for SSID, password, and a key. It also defines pins for GreenLED (26), RedLED (33), and a motion sensor (13). Variables include temperature and light sensors, their respective values, and ranges. It uses unsigned integers for time calculations in minutes and seconds. The code includes setup and loop functions.
- Serial Monitor:** Below the code area, the serial monitor shows the following output:

```
Done Saving.  
Hash of data verified.  
Compressed 3072 bytes to 128...  
Writing at 0x00008000... (100 %)  
Wrote 3072 bytes (128 compressed) at 0x00008000 in 0.0 seconds (effective 1890.5 kbit/s)...  
Hash of data verified.  
Leaving...  
Hard resetting via RTS pin...
```
- Status Bar:** At the bottom right, it says "ESP32 Dev Module on COM4".

4.2 Vervang (WIFI_NAAM) door je eigen wifi netwerk, bv.: telenet-80415.

Vervang (wifi_wachtwoord) door je eigen wifi wachtwoord.

Vervang (XXXXXXXXXXXX) door de persoonlijke Code die je in stap 3 hebt opgevraagd.

4.3 Parameters van de code

Het eerste deel van de code tot aan de sterretjes lijn vanonder zijn alle instellingen voor de code.

Hier kan je instellen vanaf wanneer de IOT energieverbruik helper inschakelt door de waarde van "uur_start" en "minuten_start" aan te passen.

Nu is uur_start = 20 en minuten_start= 0, dus schakelt de helper in om 20u 's avonds.

Hetzelfde geld voor "uur_stop" en "minuten_stop".

Deze staan nu op 8 en 15, wat wilt zeggen dat de helper om 20u 's avonds ingeschakeld wordt en 's morgens om 8:15 uitgeschakeld wordt.

Ook kan je hier de "tijd_geen_beweging_tot_melding_sturen" aanpassen, dit is zoals de naam doet vermoeden: hoe lang er niemand aanwezig mag zijn (in minuten) als de helper actief is voordat hij een melding stuurt als het licht of de verwarming nog aan zou staan. Momenteel staat die op 1, wat wilt zeggen dat er al een melding kan gestuurd worden na 1 minuut.

```
#include <WiFi.h>
#include <HTTPClient.h>
#include <TimeLib.h>           //      tijd                bibliothek (voor interne klok)

const char* ssid = "WIFI_NAAM";
const char* password = "wifi_wachtwoord";
String KEY = "XXXXXXXXXXXXXX";

String Event_marker = "ESP32" ;

String serverName = "http://maker.ifttt.com/trigger/" + Event_marker + "/with/key/" + KEY ;

const int GreenLED = 26;
const int RedLED = 33;
const int bewegingssensor = 13;

const int temperatuursensor = 35;
const int lichtsensor = 34;
int temperatuursensorvalue = 0;
int lichtsensorvalue = 0;
int bewegingssensorvalue = 0;

int max_lichtsensorvalue = 0;
int min_lichtsensorvalue = 4095;

int max_temperatuursensorvalue = 0;
int min_temperatuursensorvalue = 4095;

unsigned int uur_start = 21 ;          //uur vanaf wanneer er gescand mag worden
unsigned int minuten_start=0;          //minuten na dit uur
unsigned long startT = (uur_start - 1) * 60 + minuten_start ; //start tijdstip in minuten

unsigned int uur_stop = 8;             //uur tot wanneer er gescand mag worden
unsigned int minuten_stop = 15;        //minuten na dit uur
unsigned long stopT = (uur_stop - 1 ) * 60 + minuten_stop ; //stop tijdstip in minuten

unsigned long minuten_sinds_middernacht ;
unsigned int tijd_tot_volgende_scan= 1; //hoe lang te wachten om opnieuw te scannen wanneer er beweging is (in minuten)
unsigned long scan_interval= (tijd_tot_volgende_scan*60*1000);
unsigned int tijd_geen_beweging_tot_melding_sturen= 1; //hoe lang er geen beweging mag zijn voor we een melding sturen (in minuten)
unsigned long buffer_tijd= (tijd_geen_beweging_tot_melding_sturen*60*1000);
unsigned int berichten_verstuurd = 0;

unsigned int toegestane_hoeveelheid_meldingen = 1;

//*****
```

4.4 Void Setup

Dit deel van de code zal slechts een keer uitgevoerd worden wanneer de Arduino opstart.
Dus zetten we hier alle commando's die maar een keer moeten gebeuren.

Zoals de pinMode commando's die vertellen de Arduino welke pins outputs moeten zijn,
want op de ESP32 kunnen de meeste pins zowel outputs als inputs zijn.
Ook zeggen we hier of die pinnen hoog of laag moeten zijn bij het opstarten.

Daarna maken we verbinding met de wifi, synchroniseren we de interne klok en kalibreren
we de analoge sensoren.

Deze drie commando's zijn geschreven als modules zodat we die eenvoudig in onze code
kunnen gebruiken, de volledige modules van deze drie onderdelen komen in de volgende
puntjes aan bod.

Als laatste definiëren we de tijd als "minuten_sinds_middernacht" omdat dit makkelijker
rekent verder in de code.

```
void setup() {  
  
    Serial.begin(115200);  
  
    pinMode(GreenLED, OUTPUT);  
    pinMode(RedLED, OUTPUT);  
  
    pinMode(32, OUTPUT);           //extra 3.3V pin voor temp sensor  
    digitalWrite(32, HIGH);        //altijd aan  
  
    digitalWrite(GreenLED, LOW);  
    digitalWrite(RedLED, LOW);  
  
  
    maak_verbinding();          //start verbinding met wifi  
    delay(400);  
    sync_tijd();  
  
    kalibreren();  
    delay(3000);  
  
  
    unsigned int uur_sinds_middernacht = hour();  
    unsigned int min_sinds_middernacht = minute();  
    //Serial.println(uur_sinds_middernacht);  
    //Serial.println(min_sinds_middernacht);  
    Serial.println( startT );  
    Serial.println( stopT );  
    minuten_sinds_middernacht = uur_sinds_middernacht * 60 + min_sinds_middernacht ;  
    //Serial.println( minuten_sinds_middernacht );  
}
```

4.5 Void Loop

Dit is het deel van de code die voortdurend zal draaien.
En is dus de meest basis versie van onze hele code.

Dit deel van de code zorgt ervoor dat wanneer de helper actief is en er voor de bepaalde tijd (buffer_tijd) geen beweging meer is geweest in de ruimte, er een melding zal gestuurd worden als het licht of de kachel nog aan staat.

Als er iemand in de kamer is tijdens deze buffer_tijd zal de timer resetten en zal de helper weer wachten voor een melding te verzenden moest het licht of de kachel nog aan staan.

```
void loop() {  
  
    bewegingssensorvalue= digitalRead(bewegingssensor);  
  
    if ( ( minuten_sinds_middernacht > startT ) || (minuten_sinds_middernacht < stopT ) ){           // als het tussen de start en stop tijdstippe  
        if (bewegingssensorvalue == HIGH){  
            berichten_verstuurd = 0;  
            return;  
        }  
        if(berichten_verstuurd<toegestane_hoeveelheid_meldingen){                                //als er nog meldingen mogen gestuurd  
            if (bewegingssensorvalue == LOW){  
                //als er geen beweging is  
                unsigned long currentTime = millis();  
                //start timer  
                while ((millis()-currentTime) < buffer_tijd ) {  
                    if (bewegingssensorvalue == HIGH){  
                        berichten_verstuurd = 0;  
                        return;  
                    }  
                }  
                Serial.println("al een tijd geen beweging");  
                Serial.print("check sensoren ");  
                check_sensoren_en_stuur_meldingen();  
                // als er na de timer nog geen beweging is,  
            }  
        }  
        else{  
            return;  
        }  
    }  
    else {  
        berichten_verstuurd = 0;  
    }  
}
```

4.6 Commando's

Dit zijn de stukken code die geschreven zijn als op te roepen modules en die we al eerder hebben gebruikt in 'Void Setup" en "Void Loop".

4.6.1 Maak_verbinding comando

Deze module verbind de Arduino met het wifi netwerk.

En laat de rode LED branden zolang de module bezig is met de verbinding tot stand te brengen.

En Switcht naar de groene LED als de verbinding geslaagd is

```
void maak_verbinding() {  
  
    digitalWrite(RedLED, HIGH);           //Rode led aan  
    digitalWrite(GreenLED, LOW);          //Groene led uit  
  
    WiFi.begin(ssid, password);  
    Serial.print("Verbinding maken met Wifi netwerk: "); //melden via serial monitor dat wifi verbinding opgestart is  
    Serial.println(ssid);  
  
    while(WiFi.status() != WL_CONNECTED) {           //zolang de wifi nog niet verbonden is, stuur ja puntjes via de serial monitor (zo: . . . .  
        delay(500);  
        Serial.print(".");  
    }  
  
    Serial.println("");  
    Serial.print("Verbonden met WiFi netwerk:"); //melden via serial monitor dat wifi verbinding geslaagd is  
    Serial.println(ssid);  
  
    digitalWrite(GreenLED, HIGH); //Groene led aan  
    digitalWrite(RedLED, LOW);   //Rode led uit  
  
}  
//***** einde "maak_verbinding" comando
```

4.6.2 sync_tijd commando

Deze module synchroniseert te interne klok van de Arduino met het internet door een Google tijd server te contacteren.

Ook hier weer gaat de rode LED branden zolang de module bezig is en switcht deze naar de groene LED als alles succesvol is verlopen.

```
void sync_tijd() {  
  
    digitalWrite(RedLED, HIGH); //Rode led aan  
    digitalWrite(GreenLED, LOW); //Groene led uit  
  
    WiFiClient client;  
    while (!client.connect("google.com", 80)) {  
        Serial.println("connection failed, retrying...");  
    }  
  
    client.print("HEAD / HTTP/1.1\r\n\r\n");  
  
    while(!client.available()) {  
        yield();  
    }  
  
    while(client.available()){  
        if (client.read() == '\n') {  
            if (client.read() == 'D') {  
                if (client.read() == 'a') {  
                    if (client.read() == 't') {  
                        if (client.read() == 'e') {  
                            if (client.read() == ':') {  
                                client.read();  
                                String theDate = client.readStringUntil('\r');  
                                String het_uur = theDate.substring(17, 19) ;  
                                String de_minuut = theDate.substring(20, 22);  
                                String de_seconde = theDate.substring(23, 25);  
                                Serial.println(theDate);  
                                Serial.println(het_uur);  
                                Serial.println(de_minuut);  
                                Serial.println(de_seconde);  
                                unsigned long set_time = (((het_uur.toInt()+1)*3600)+((de_minuut.toInt())*60)+(de_seconde.toInt()));  
                                adjustTime(set_time);  
                                client.flush();  
                                client.stop();  
                                digitalWrite(GreenLED, HIGH); //Groene led aan  
                                digitalWrite(RedLED, LOW); //Rode led uit  
                                return; // theDate  
                            }  
                        }  
                    }  
                }  
            }  
        }  
    }  
}  
//***** einde "sync_tijd" commando
```

4.6.3 kalibreren commando

Dit commando kalibreert de temperatuur en de licht sensor naar de kamer en positie waarin de helper wordt geplaatst.

Ook hier weer gaat de rode LED branden zolang de module bezig is en switcht deze naar de groene LED als alles succesvol is verlopen.

```
void kalibreren(){

    digitalWrite(RedLED, HIGH); //Rode led aan
    digitalWrite(GreenLED, LOW); //Groene led uit
    delay(1000);
    digitalWrite(RedLED, LOW); //Rode led uit

    delay(1000);

    unsigned long currentTime = millis(); //unsigned want dan kan een dubbel zo groot getal opgeslagen worden (Negatieve waarden hebben we niet nodig)

    while ((millis()-currentTime) < 5000) { // 5 seconden de tijd om te kalibreren
        lichtsensorvalue = analogRead (lichtsensor);
        if (lichtsensorvalue > max_lichtsensorvalue) {
            max_lichtsensorvalue = lichtsensorvalue;
        }
        if (lichtsensorvalue < min_lichtsensorvalue) {
            min_lichtsensorvalue = lichtsensorvalue;
        }
        temperatuursensorvalue = analogRead (temperatuursensor);
        if (temperatuursensorvalue > max_temperatuursensorvalue) {
            max_temperatuursensorvalue = temperatuursensorvalue;
        }
        if (temperatuursensorvalue < min_temperatuursensorvalue) {
            min_temperatuursensorvalue = temperatuursensorvalue;
        }
    }
    Serial.println(max_lichtsensorvalue);
    max_lichtsensorvalue= (max_lichtsensorvalue*1.2);
    Serial.println(max_lichtsensorvalue);

    Serial.println(min_temperatuursensorvalue);
    min_temperatuursensorvalue= (min_temperatuursensorvalue*0.99);
    Serial.println(min_temperatuursensorvalue);
    digitalWrite(GreenLED, HIGH);
    digitalWrite(RedLED, LOW);
}

***** einde "kalibreren" commando
```

4.6.4 Stuur_melding commando

Deze module stuurt een bepaald stukje tekst naar de Webhook server zodat dit naar de app op je smartphone kan verzonden worden.

```
void stuur_melding(String message) {  
  
    if(WiFi.status() == WL_CONNECTED) {  
        //Check WiFi connection status  
  
        HTTPClient http;  
  
        http.begin(serverName);  
        // Your Domain name with URL path or IP address with path  
  
        http.addHeader("Content-Type", "application/x-www-form-urlencoded");  
        // Specify content-type header  
  
        String httpRequestData = "value1=" + message; // Data to send with HTTP POST  
  
        int httpResponseCode = http.POST(httpRequestData);  
        // Send HTTP POST request  
  
        Serial.print("HTTP Response code: ");  
        Serial.println(httpResponseCode);  
  
        http.end();  
    }  
    else {  
        Serial.println("WiFi Disconnected");  
    }  
} //***** einde "stuur_melding" commando
```

4.6.5 check_sensoren_en_stuur_meldingen comando

Deze module lijst de temperatuur en licht sensor en bepaald of het licht of de kachel nog aan staat.

En als een van de twee, of beiden nog aan staan stuurt deze module de juiste melding naar je smartphone.

Door eerst het juiste stukje tekst naar de stuur_melding module te sturen.

```
void check_sensoren_en_stuur_meldingen(){

    lichtsensorvalue= analogRead(lichtsensor);
    temperatuursensorvalue= analogRead(temperatuursensor);

    if ((lichtsensorvalue > max_lichtsensorvalue) && (temperatuursensorvalue > min_temperatuursensorvalue)){
        stuur_melding("Het licht staat nog aan!");
        Serial.println ("het licht staat nog aan");

        berichten_verstuurd=berichten_verstuurd+1;
    }

    if ((lichtsensorvalue < max_lichtsensorvalue) && (temperatuursensorvalue < min_temperatuursensorvalue)){
        stuur_melding("De kachel staat nog aan!");
        Serial.println ("De kachel staat nog aan");

        berichten_verstuurd=berichten_verstuurd+1;
    }

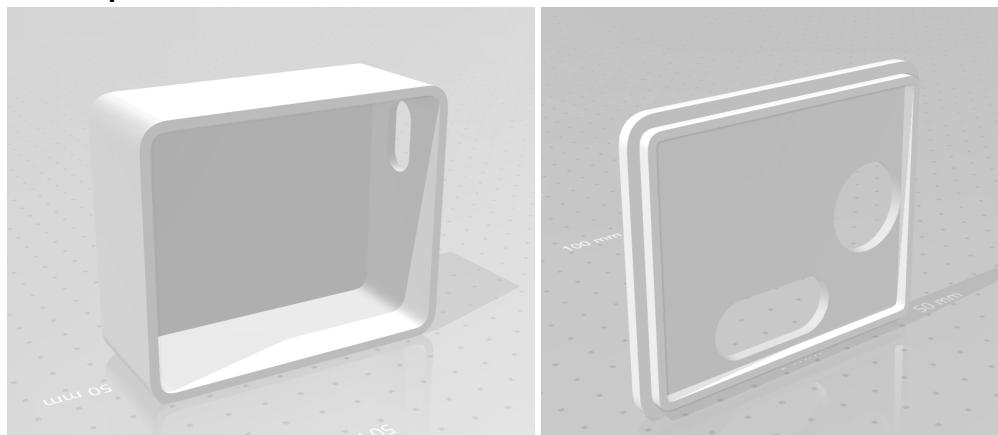
    if ((lichtsensorvalue > max_lichtsensorvalue) && (temperatuursensorvalue < min_temperatuursensorvalue)){
        stuur_melding("Het licht én de kachel staan nog aan!");
        Serial.println ("het licht én de kachel staan nog aan");

        berichten_verstuurd=berichten_verstuurd+1;
    }
}
//***** einde "check_sensoren_en_stuur_meldingen" comando
```

Stap 5: De behuizing

5.1 3Dprint de behuizing

De STL files hiervoor zijn te vinden in de bijlage als “**3D print case bottom**” en “**3D print case top**”

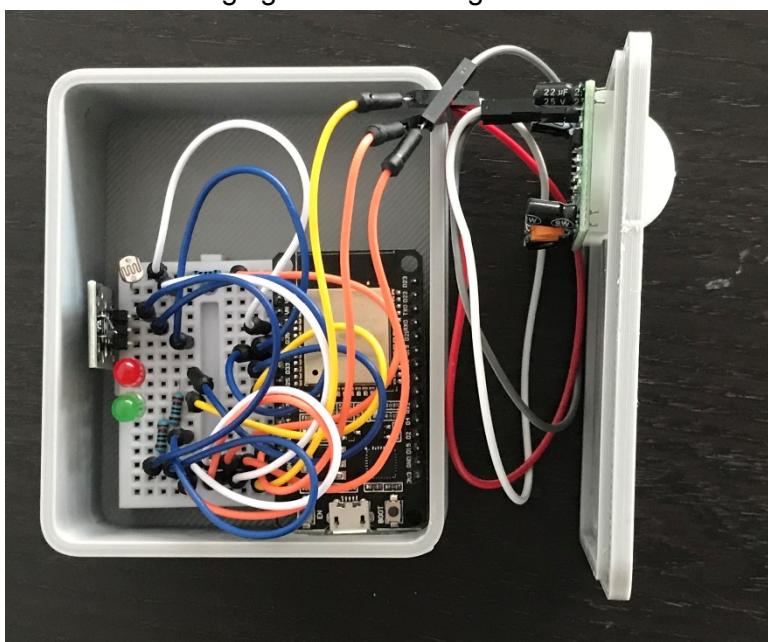




5.2 Plaats de componenten

gebruik het lijmpistool of dubbelzijdige tape op het mini breadboard in de behuizing te lijmen zodat de usb poort overeen komt met het gat.

En duw de bewegingssensor in het gat in het deksel zoals op de foto hieronder.



5.3 Sluit de behuizing

Plaats het dekseltje zo op de behuizing zodat het kot over de LED's en sensoren komt te liggen, zoals op de foto hieronder.



Afhankelijk van de gebruikte 3d printer kan het zijn dat je een beetje plakband nodig hebt om het dekseltje deftig op de behuizing te houden.

Stap 6: Plaatsen voor gebruik

Om de helper nuttig te kunnen gebruiken is het best dat je hem dicht bij een verwarming plaatst en op een plaats waar niet veel zonlicht valt. Maar een plaats die de binnenverlichting wel goed oplicht.

6.1 Zet de helper op de gekozen plaats

6.2 Zorg ervoor dat het buiten donker is, de binnenverlichting uit is en dat de kachel uit staat.

6.3 Steek de stekker van de helper in het stopcontact.

Voor het optimale resultaat doe je dit best 's avonds.

Wanneer je de stekker in steekt zal de helper na enkele seconden beginnen te zijn sensoren te kalibreren, dus het is best om wat afstand te nemen voor een accurate kalibratie.

6.4 Klaar!

Nu krijg je een melding iedere keer het licht of de kachel nog aan staat en er al een bepaalde tijd niemand in de kamer is geweest wanneer de helper actief is!

Weetje(s)

Met de Arduino en IFTTT app kan je nog veel meer doen dan enkel meldingen sturen!

Je kan bijvoorbeeld Alexa iets laten zeggen als er nog een licht aan staat of een email sturen naar iemand om het licht te gaan uit zetten, om er maar twee te noemen.