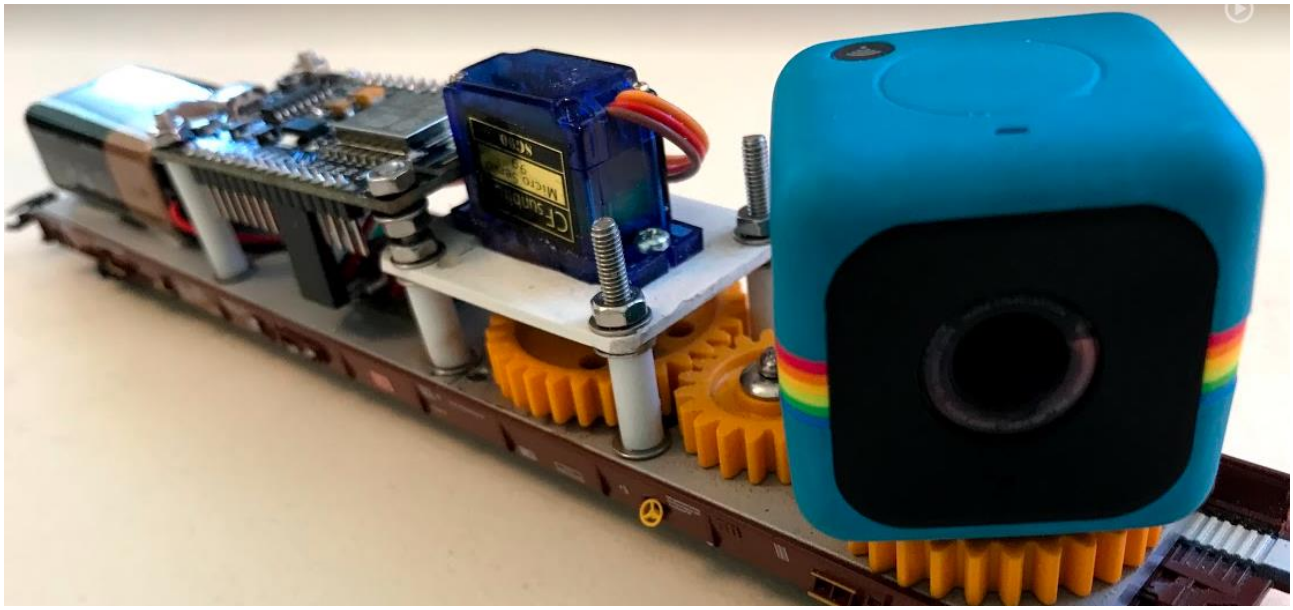
How to build a Loc camera with panning.

Document created 3-6-2020

Rev A: 9-6-2020 – hardcoded WIFI credentials are replaced with log-on box.

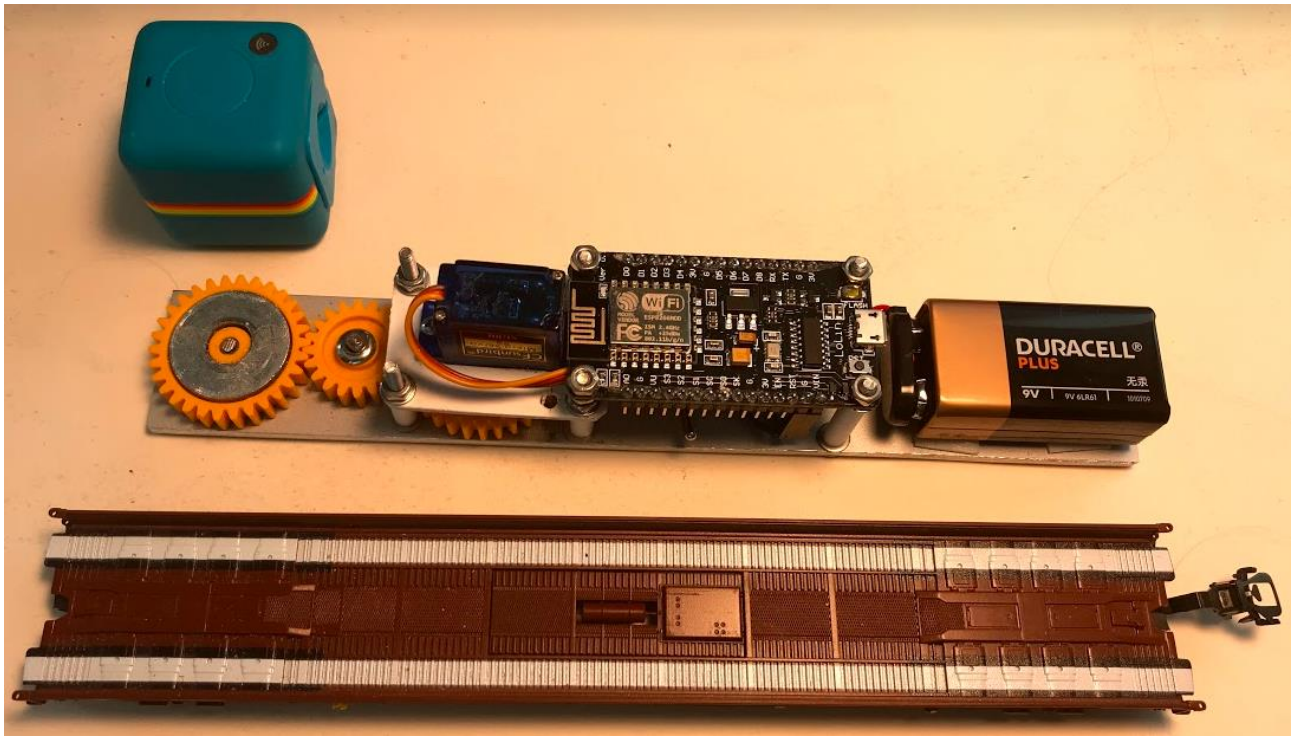
By Jens Krogsgaard, jenskrogsgaard@gmail.com, Syrenvaenget 10, 3520 Farum, Denmark - +4542669987



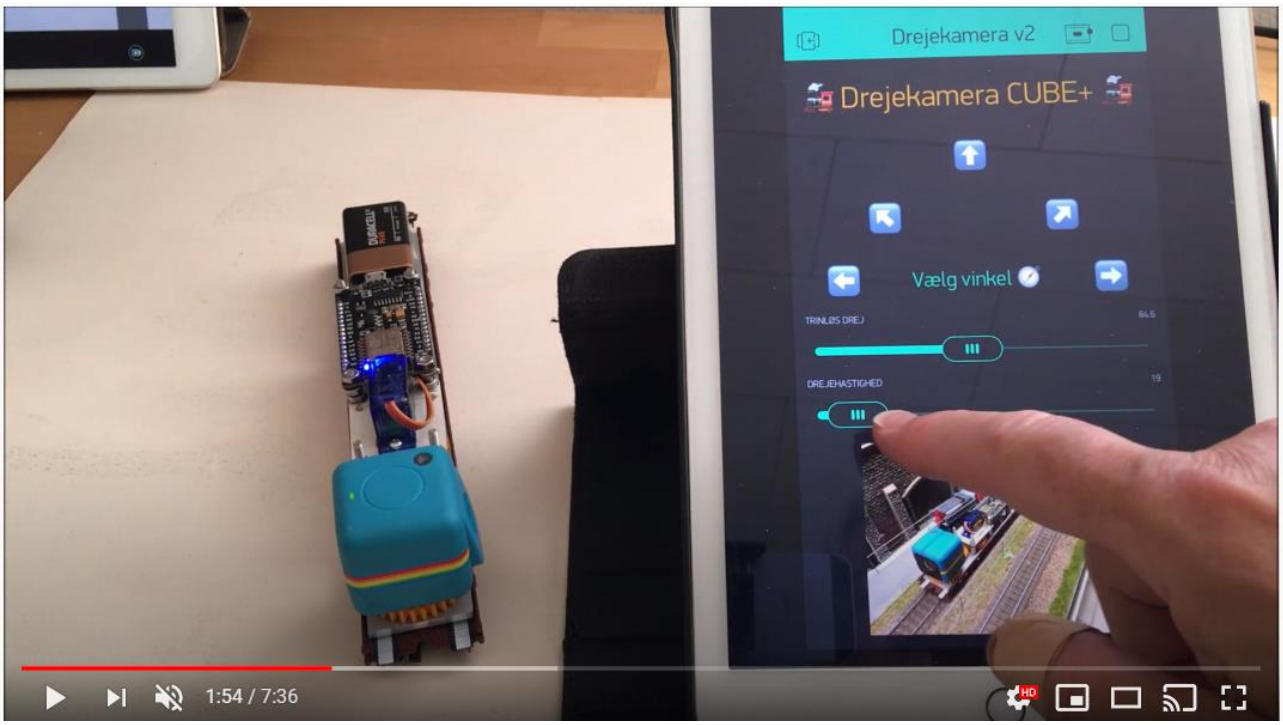
Content

1 Summary

This document contains a description of how I designed and programmed a camera wagon. The camera can pan and it is controlled by an ESP9266 Node MCU. To operate the camera, I have made an a Blynk app.



Check the YouTube video regarding this project: - click on the picture to start the video



1.1 Construction of the wagon

1.1.1 Bottom plate

Metal plate – 31 x 200 x 3 mm. The weight of the plate stabilizes the wagon when driving

Bought in Bauhaus and cut out.

The bottom plate is designed to match the wagon from Rollende Landstraße / Rolling Road

1.1.2 Gear

I have used 3 gear-wheels:

- wheel 31,5 mm
- 1 wheel 21,5 mm is used.

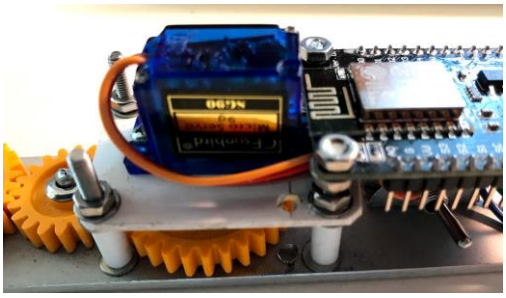
Bought at Conrad.de: https://www.conrad.de/de/search.html?search=237663

Shafts 4 mm – bought at bauhaus

Glue a metal disc onto the gear-wheel to carry the camera - check that it is magnetic

1.1.3 Servomotor

https://www.elextra.dk/details/H34768/servomotor-mikro-3-72vdc-120ms-60-9g



The servo is mounted on a 31 x 44 x 2 mm plastic plate

The screws are 3 mm - from the Bauhaus. The bushings are plastic tubes.

1.1.4 ESP8266 – Node MCU

https://www.conrad.de/de/p/joy-it-entwickler-platine-node-mcu-esp8266-wifi-1613301.html

1.1.5 Power supply

I have used a 9V battery - Here you might consider a different solution so you don't have to change the battery.

A toggle switch to disconnect battery power is also necessary.

1.1.6 Camera

Polaroid Cube+ - wifi.

Unfortunately, it does not appear to be available anymore

Polaroid Cube+ 1440p Mini Lifestyle Action Camera with Wi-Fi & Image Stabilization (Black)

by Polaroid

420 ratings | 143 answered questions

Available from these sellers.

Color: **Black**

1 option from $251.88 | 4 options from $70.00 | 2 options from $119.02

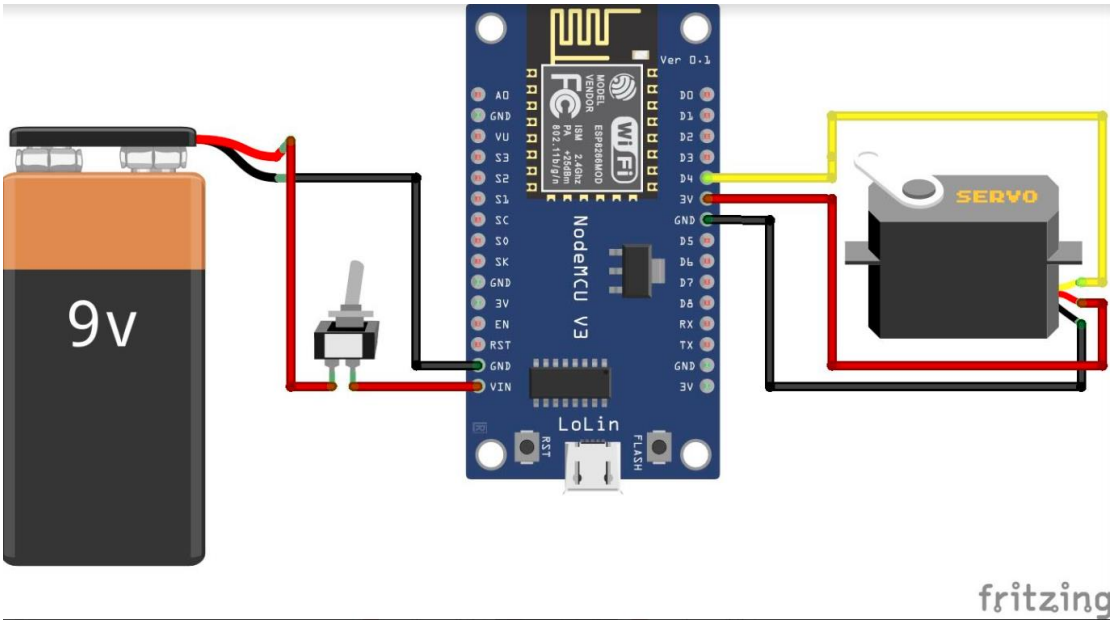
- World's Funnest, Cutest Lifestyle Action Camera in Light & Tiny Cubic Package
- NEW! Wi-Fi + FREE App; Shoot, View, Save, Print & Share with Your Mobile Device
- 8MP still images; Selectable 1440p / 1080p / 720p Video Rate; Full Image & Video Stabilization
- Built-In Rechargeable Battery for up to 107 Minutes of Continuous Recording Per Charge
- 124° Wide-Angle Lens; Magnetic/Clip Mounting Options; Includes MicroSD Card & Polaroid Bumper Case

There is a newer model of this item:

Polaroid Cube+ Live Streaming 1440p Mini Lifestyle Action Camera with Wi-Fi & Image Stabilization (Black)

Currently unavailable.

1.1.7 Connect Node CMU with servo and battery



The servo with its three wires is connected in this way:

- Yellow – signal – D4
- Red – 3v
- Black – Ground

The battery is connected to GND and VIN

1.2 BLYNK – app

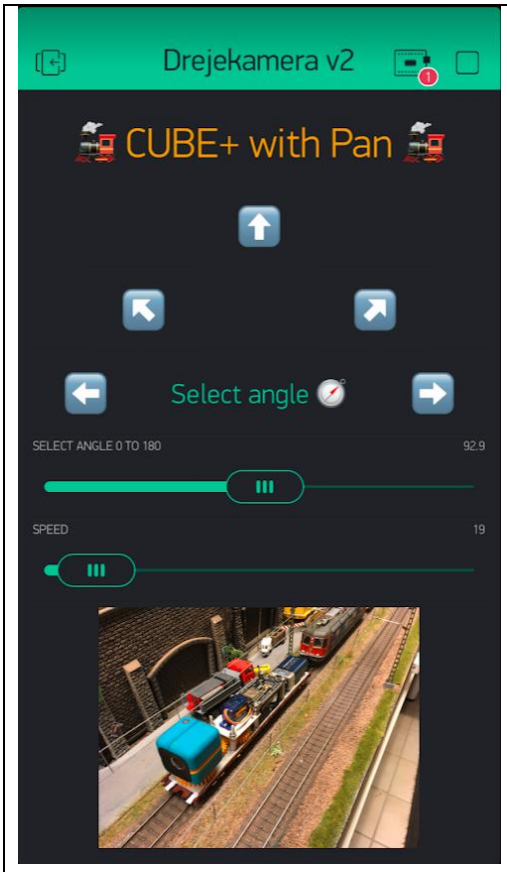
There are many videos on YouTube describing how to work with Blynk.

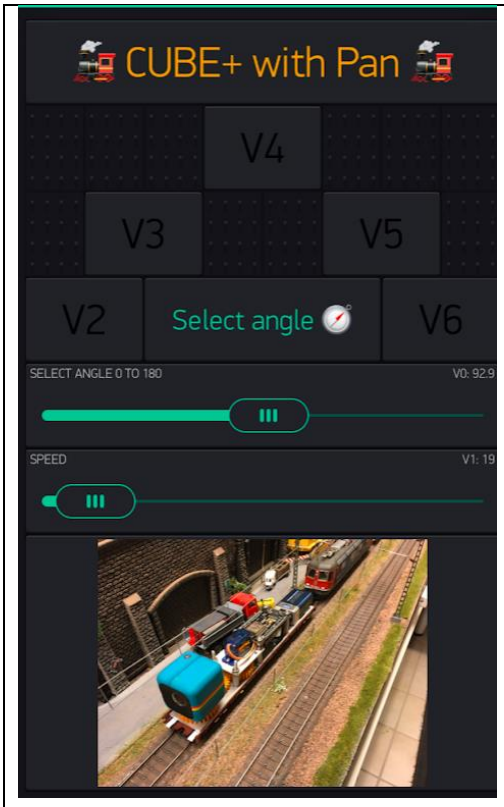
Take a look at this video: https://www.youtube.com/watch?v=EYrEjC3QEew&t=8s

Install the Blynk app on your Mobile or iPad and follow the instructions in the video above.

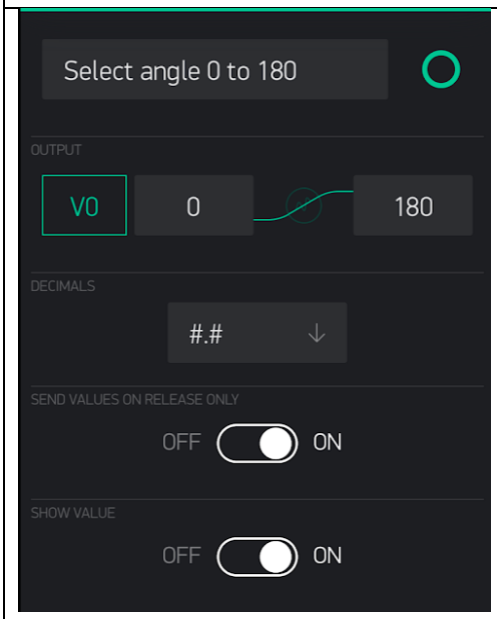
Make sure to get the authorization code – you shall use it later.

Below a description of the Blynk app to control the Servo:

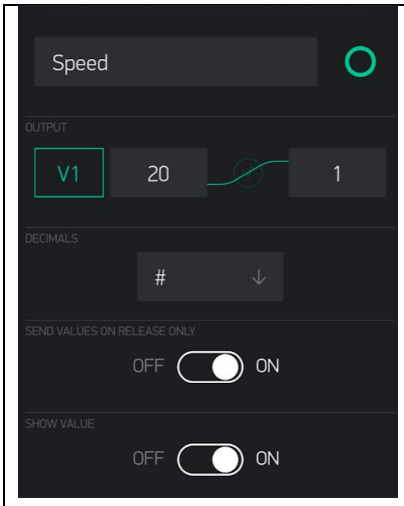
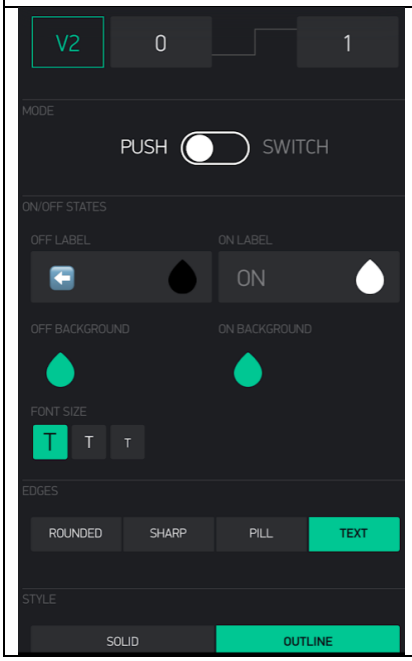
	This is the finished application. There are 5 buttons with arrows and 2 horizontal sliders. The other elements are just texts and pictures - you can compose them as you like.

<table>
<tr>
<td> 
</td>
<td>This is Design view.

Each of the sliders and the angle-buttons have a virtual pin.

• V0 – select angle slider
• V1 – speed slider
• V2 – 0 degree
• V3 – 45 degree
• V4 – 90 degree
• V5 – 135 degree
• V6 – 180 degree</td>
</tr>
<tr>
<td> 
</td>
<td>Detail for: The Select angle slider.

The values are 0 to 180</td>
</tr>
</table>

	Detail for: The select speed slider NB: the values goes from 20 to 0 The speed is implemented as an delay in milliseconds between each change of degree. Example – go from 45 to 90 degree. We loop from 45 to 90 – that is 45 steps. In each step we have a delay – if the value of the is small – for example 5 – then the speed is fast. If the delay is high – for example 18 – then the speed is slow
	Detail for: This is button 0 degree – V2. The other 4 buttons are identical – of course another pin (v3 – v4 – v5 – v6) and another label

1.3 Coding the Node MCU – ESP8266

Coding of the Node MCU is done in the Arduino environment.

If you are new in Arduino coding you might want to have a look on this video:

https://www.youtube.com/watch?v=p06NNRq5NTU&t=331s

In then code below there is no hardcoded wifi credentials. Also the BLYNK token can be replaced with your own token.

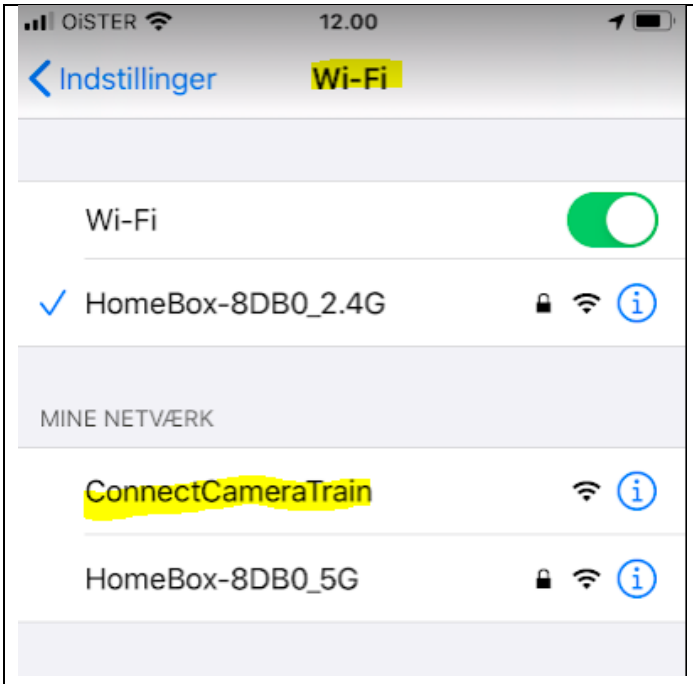
Paste in the code to your Arduino environment - compile it and send it to the ESP8266.

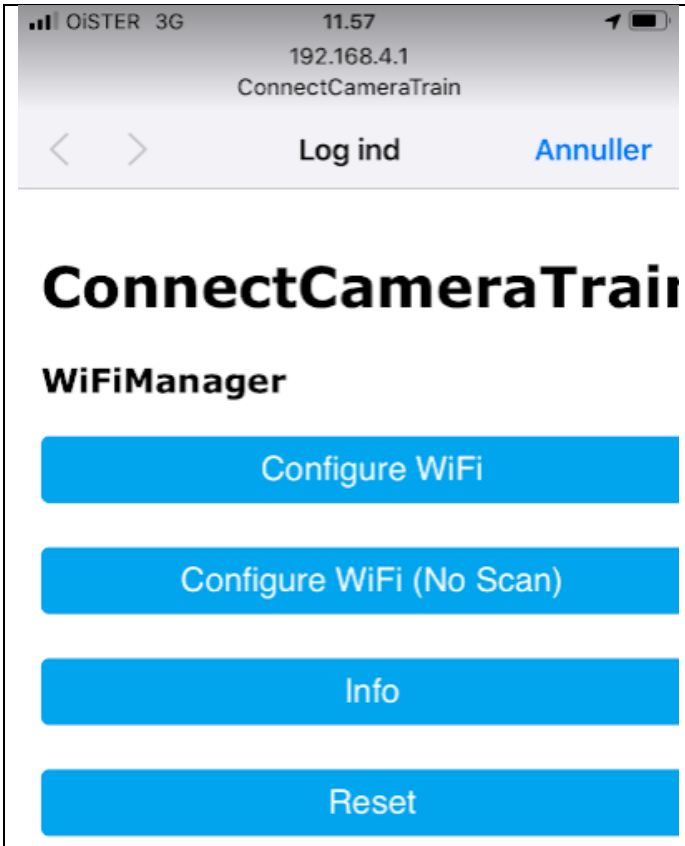
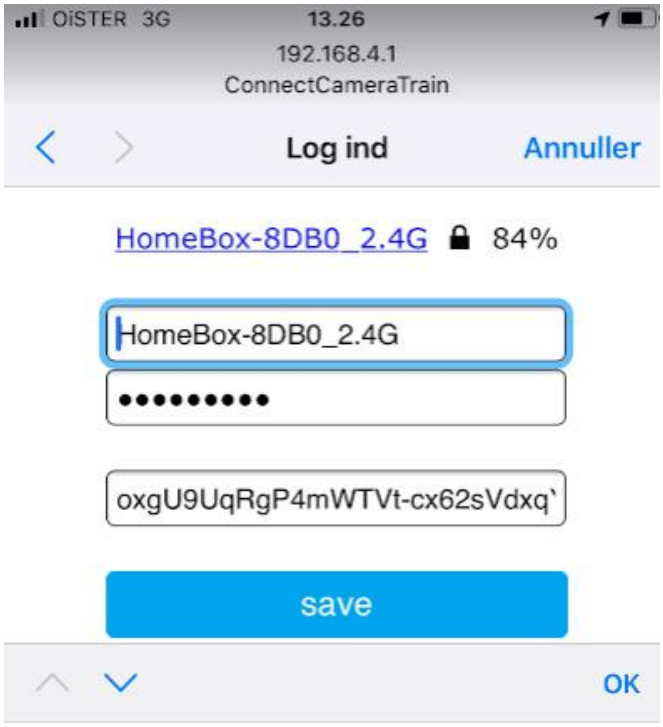
Follow this procedure to connect the Camera Train to your wifi-network:

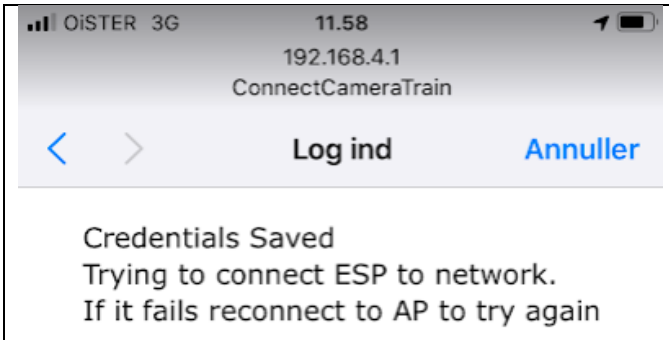
1 – Find your Blynk token – from mail or in the Blynk app. Copy this token to the clipboard

2 – Turn on the camera-train with the switch button.

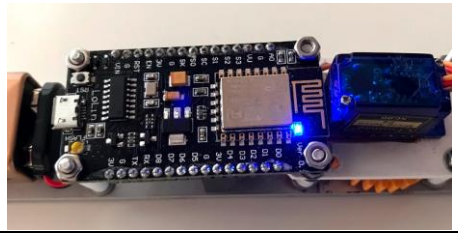
3 – On your Mobile/Ipad or PC – find the hotspot 'ConnectCameraTrain – and choose this hotspot. See Examples below

	Choose the hotspot ConnectCamaraTrain

	Choose the first button – 'Configure WiFi
	Choose your WiFI network and type in the password. The next field is for the Blynk Token (see previous chapter). Paste in this token here. Click Save.



If everything is Ok – you should now see this massage (picture to the left) And there must be a blue-light on the ESP8266 – as shown below.



Code

```
/*  Cube Camera Pan – v3
   Servo controle - speed and movement 0 to 180 degree
   31-5-2020 - Jens Krogsgaard
   rev A - 9-6-2020: Hardcoded wifi credentials and blynk token is replaces with logon-box.
*/
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <Servo.h>
#include <DNSServer.h>
#include <WiFiManager.h>        //https://github.com/tzapu/WiFiManager
#include <EEPROM.h>
#include <DoubleResetDetector.h>

#define DRD_TIMEOUT 10
#define DRD_ADDRESS 0

DoubleResetDetector drd(DRD_TIMEOUT, DRD_ADDRESS);

// auth kode fra BLYNK app - Drejekamera V3 - defualt kan bee replaced
char auth[] = "oxgU9UqRgP4mWTVt-cx62sVdxqY-IUEx";

int oldPos; // servo - old position
int newPos; // servo - new position
int stepPos; // step position for servo - used when you step from old to new position
int servoSpeed; // speed of servo - delay - big value - slow, small value fast

Servo servo;

void setup()
{
   // initial values
  EEPROM.begin(512);
  Serial.begin(115200);
```

```
  // WIFI - connect
  WiFiManager wifiManager;
  // wifiManager.resetSettings();    //Uncomment this to wipe WiFi settings from EEPROM on
boot.  Comment out and recompile/upload after 1 boot cycle.
  // Input blynk token
  WiFiManagerParameter BlynkToken("auth", "Blynk Token", auth, 34);
  wifiManager.addParameter(&BlynkToken);

  if (drd.detectDoubleReset()) {
    // when reset button is activated by a double click a reconnect to wifi i initiated.
    Serial.println("Double Reset Detected");
    digitalWrite(LED_BUILTIN, LOW);
    wifiManager.startConfigPortal("ConnectCameraTrain");

  } else {
    Serial.println("No Double Reset Detected");
    digitalWrite(LED_BUILTIN, HIGH);
    // temporary hotspot ConnectCameraTrain is credted
    wifiManager.autoConnect("ConnectCameraTrain");
  }

  strcpy(auth, BlynkToken.getValue());

  //if you get here you have connected to the WiFi
  Serial.println("connected...yeey :)");
  pinMode(BUILTIN_LED, OUTPUT);     // Initialize the BUILTIN_LED pin as an output, I like
blinkies.
  Blynk.begin(auth, WiFi.SSID().c_str(), WiFi.psk().c_str());

  servoSpeed = 10;
  servo.attach(2); // 2 means D4 pin of ESP8266
}

// Slider angle - 0 to 180 degree
BLYNK_WRITE(V0) {
  turnServo(param.asInt());
}

// Slider speed - from 20 to 0
BLYNK_WRITE(V1) {
  servoSpeed = param.asInt();
}


// Button - 0 degree
BLYNK_WRITE(V2) {
  turnServo(0);
}
// Button - 45 degree
```

```
BLYNK_WRITE(V3) {
  turnServo(45);
}
// Button - 90 degree
BLYNK_WRITE(V4) {
  turnServo(90);
}
// Button - 135 degree
BLYNK_WRITE(V5) {
  turnServo(135);
}
// Button - 180 degree
BLYNK_WRITE(V6) {
  turnServo(180);
}

// Turn servo an angle
// Speed is implemented as delay between each angle
//   long delay - slow speed
//   short delay - fast speed

void turnServo(int turnTo)
{
  oldPos = servo.read();
  newPos = turnTo;

  if (oldPos <= newPos)
  {
    for (stepPos = oldPos ; stepPos <= newPos; stepPos += 1)
    {
      servo.write(stepPos);
      delay(servoSpeed);
    }
  }
  else
  {
    for (stepPos = oldPos ; stepPos >= newPos; stepPos -= 1)
    {
      servo.write(stepPos);
      delay(servoSpeed);
    }
  }
}

void loop()
{
  Blynk.run();
}
```