

# Titan Router

## V6 Firmware

---

Sending data to  
MyDevices platform

## Scenario Details

TITAN routers have all the typical functionalities of 4G/3G/2G routers, as well as a series of added features that make them one of the most feature-packed routers on the market.

One of the added features is the datalogger, where the TITAN router can store a number of types of records in its non-volatile memory in JSON format. These records can come from MODBUS readings, SERIAL data captures via the RS232 / RS485 ports, or GPS positions, etc. These JSON-type records are stored in the TITAN router's internal non-volatile memory and can subsequently be sent to remote platforms via protocols such as HTTP, HTTPS, MQTT, MQTTS, FTP and FTPS.

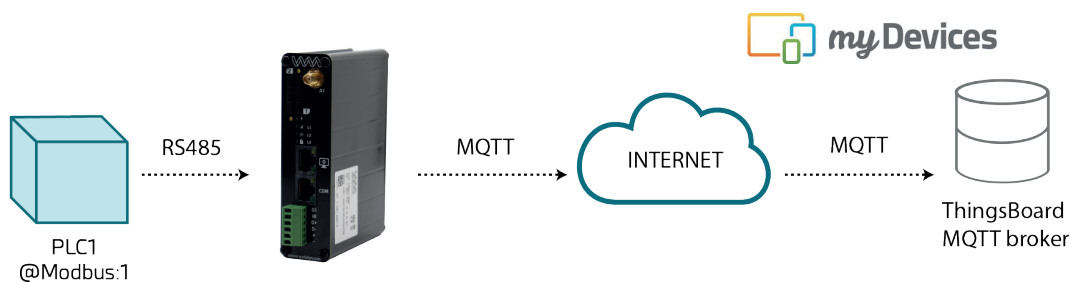
As mentioned, the TITAN router stores the JSON registers in its internal memory in a proprietary format by default. This can sometimes be a problem when communicating with platforms that expect to receive information in a certain format (i.e. a format other than JSON, the one used by the TITAN router).

In this application note, we will guide you through an entire example of how send to data to the well-known MyDevices' Cayenne platform (<https://cayenne.mydevices.com>), which requires the sent JSONs to be in a special format.

In this particular application note, we will assume that 2 registers are to be read from 2 PLCs with Modbus communications connected to a Webdyn - EasyTunnel via their RS485 port.

More specifically, the aim is for the WebDyn-Easytunnel device to read the Modbus registers with addresses 30000 and 30001 from the PLC every minute. Register 30000 corresponds to the measured temperature, and register 30001 to the humidity level. The temperature and humidity readings must be sent in real time to a dashboard on the MyDevices' Cayenne platform. You must also be able to change the Modbus registers 340002 and 30003 of the PLC from the Cayenne platform, allowing you to write an analogue value between 0 and 255 to the registers 30002 and 30003.

## 1. WAN mobile configuration



The Webdyn - EasyTunnel must communicate with the My.Devices.com's Cayenne platform via 4G/3G/2G communications, so the "Mobile> Basic Settings" section must be configured correctly according to the SIM card used.

**Mobile > Basic Settings**

**Mobile WAN**

Mobile WAN: Enabled (IP active) Enable Wireless WAN interface

Sim Mode: SIM1 Sim selection

SIM1 APN: movistar.es SIM Card 1 APN

SIM1 Username: MOVISTAR SIM Card 1 username

SIM1 Password: ..... SIM Card 1 password

SIM1 Pin: SIM Card 1 PIN

SIM1 Auth: None SIM card 1 authentication

**Network selection**

Network selection: Auto (4G/3G/2G) Network selection

DNS selection: Get DNS from Operator Preferred DNS1

DNS1: 8.8.8.8 Preferred DNS2

DNS2: 8.8.4.4

## 2. Configuring the RS485 serial port

The two PLCs will connect to 9600,8,N,1 via the RS485 serial port, so the "Serial Settings> Serial Port2-RS485" section must be configured by setting the parameters as shown below.

**Serial Gateway > Com2 Settings**

**Serial Gateway**

Baudrate: 9600 Baudrate of serial port

Data bits: 8 Number of data bit

Parity: none Parity

Stop bits: 1 Number of stop bits

Timeout ms: 50 msec without serial data before sending (default: 50)

☐ Allow local embedded AT commands Ex.: <MTXTUNNEL>AT</MTXTUNNEL>

☐ Allow remote embedded AT commands Ex.: <MTXTUNNELR>AT</MTXTUNNELR>

☐ Allow incoming GSM call (CSD Data Call) Only TCP Server and TCP Client functions or Nothing. 2G (CSD) network required.

☒ Function: Nothing or used by External Device or Script

☐ Function: Serial - IP Gateway (TCP Server)

TCP Local Port: 20011 Listening TCP Port (1 ... 65535)

Temporal client RS232: ☐ Check if you need a temporal TCP Client when data is present at serial port.

Temporal client Wakeup: DDHHMM. Example: XX2200 starts a temporal client every day at 22:00

Temporal client time: 60 Seconds for temporal client

Temporal client Random: 0 Seconds. Random time for temporal client

SSL/TLS enabled: ☐ SSL/TLS Enabled (SSL Certs needed)

## 3. Configuring the Modbus section

In this configuration section, "External Devices > Modbus Devices", only the port that will be used to read the PLC must be configured. As you need to use an RS485 port, you must also use the COM2 port of the Webdyn-Easy-Tunnel.

Mobile

Status

Basic Settings

Keep Online

Ethernet

Basic Settings

Firewall

Authorized IPs

MAC Filter

Serial Settings

Serial Port1-RS232

Serial Port2-RS485

SSL Certificates

External Devices

Logger configuration

ModBus Devices

Generic Serial Device

Temperature Sensor

IEC102 Meter

GPS Receiver

Other

AT Command

DynDns

Private DynDns

Sms control

Periodic Autoreset

External Devices

ModBus RTU / TCP

Enabled:

☒

Enable Modbus Devices

Serial Port:

Serial Port 2

Select the connected serial port if needed

Logger:

☐

Check if logger must be used

Please, configure logger before using this option

SAVE CONFIG

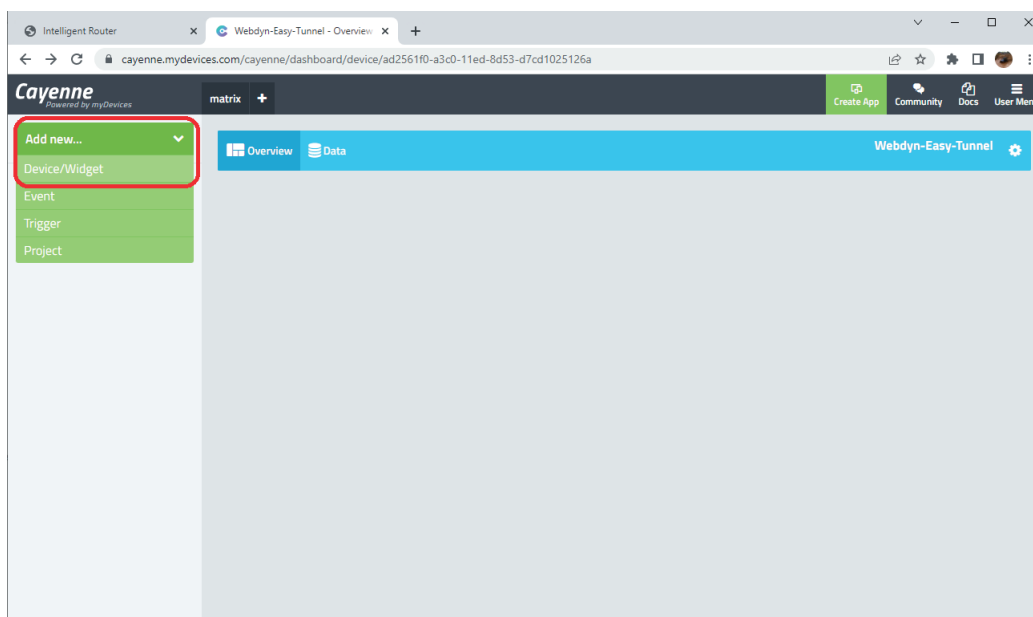
VIEW LOG

Dev. name / ID	Addr.	Command	Start	Num word/bit	Reg Type	Period
Device name / ID:						
Address:						
Command:		0x01				
Start:						
Number Words / Bits:						
Reg Type:		WORD				
Period:		1				

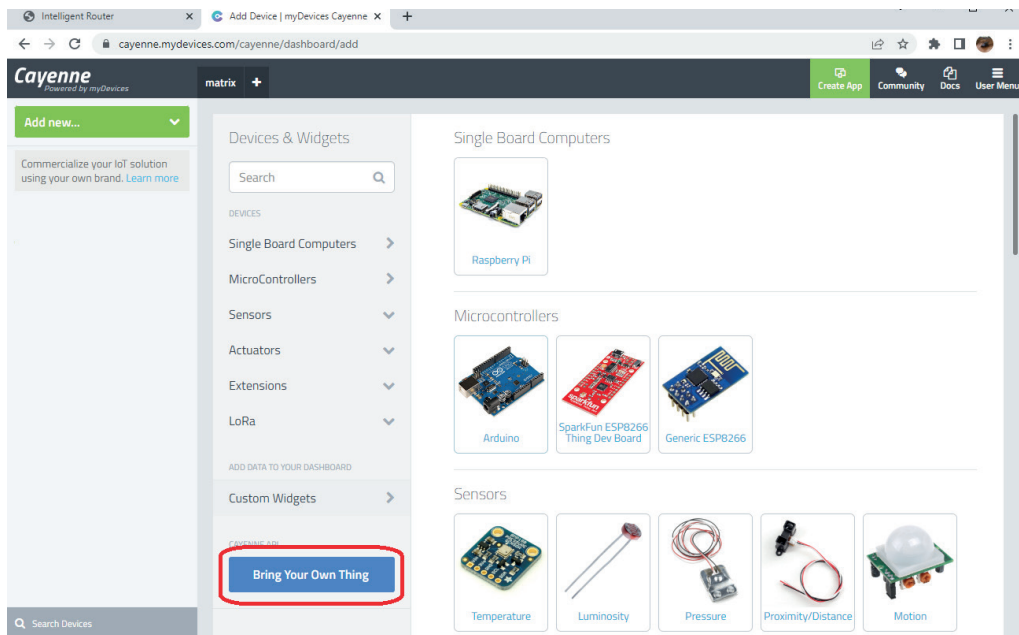
## 4. Configuring the MQTT section and the MyDevices' Cayenne platform

You must also correctly configure the "Other - Mqtt" section of the Titan-based device so that it can connect to the Cayenne platform. But first, you must go to the web platform to add the Webdyn-Easy-tunnel device and obtain the mqtt authentication attributes.

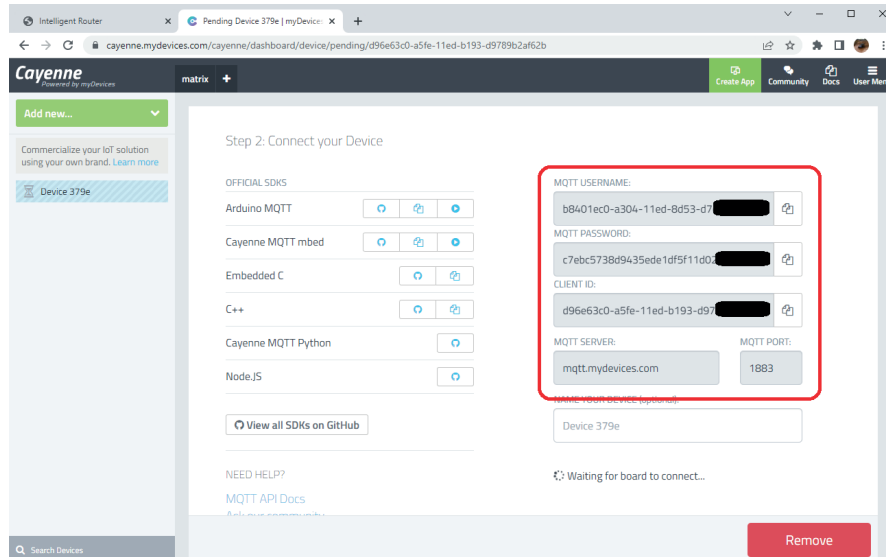
Once you have logged in to your <https://cayenne.mydevices.com> account, click on Add New >Device Widget,



Then, select the “Bring Your Own Thing” option.



Once this is done, a display will appear like the one shown in the screenshot below, where you will need to copy the data related to the MQTT connection (marked in red below) in order to integrate them into the MQTT section of the Webdyn-Easy-Tunnel device. This display will continue to appear until the connection is established.

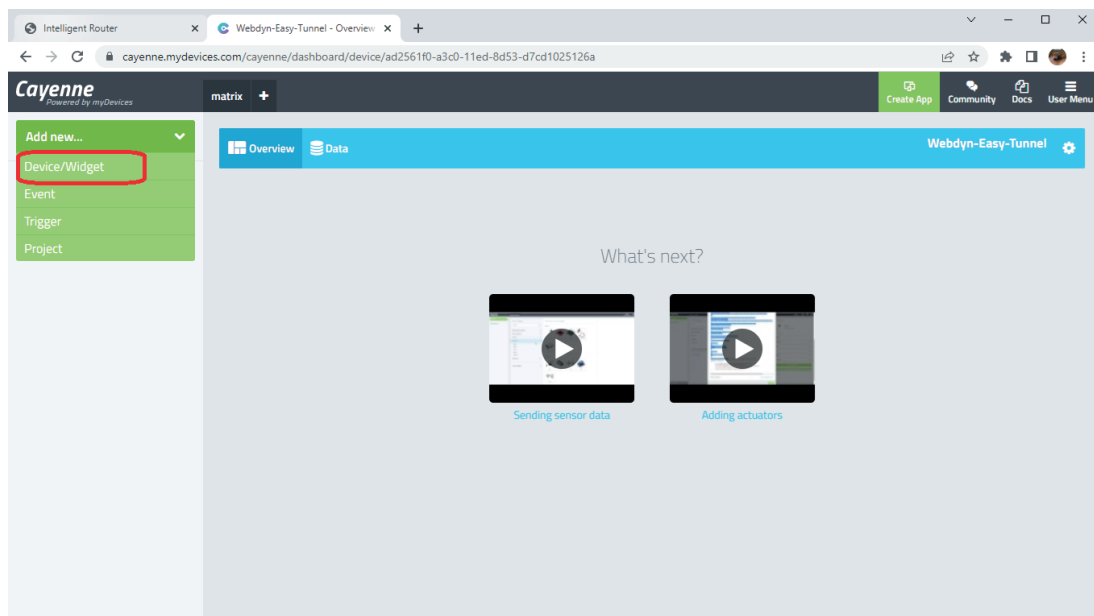


In the "Other > Mqtt" configuration section of the Webdyn-EasyTunnel, you will use the configuration data obtained in the previous display.

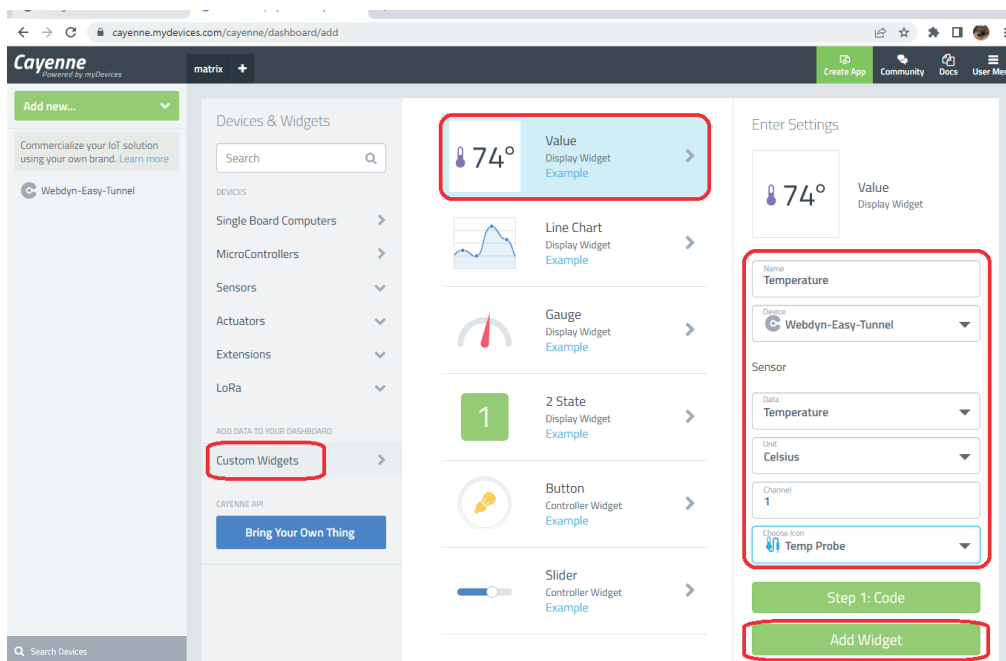
<b>External Devices</b> <ul style="list-style-type: none"> <li>Logger configuration</li> <li>ModBus Devices</li> <li>Generic Serial Device</li> <li>Temperature Sensor</li> <li>IEC102 Meter</li> <li>GPS Receiver</li> </ul> <b>Other</b> <ul style="list-style-type: none"> <li>AT Command</li> <li>DynDns</li> <li>Private DynDns</li> <li>Sms control</li> <li>Periodic Autoreset</li> <li>Time Servers</li> <li>Remote Console</li> <li>Snmp</li> <li>Tacacs+</li> <li><b>Mqtt</b></li> <li>Http / Https</li> <li>User Permissions</li> <li>Passwords Web UI</li> <li>CA Certificates</li> </ul>	<b>Enabled:</b> <input checked="" type="checkbox"/>	Enable MQTT client
	<b>MQTT Broker</b> <input type="text" value="tcp://mqtt.mydevices.com:18"/>	Destination MQTT Broker. Examples: tcp://test.mosquitto.org:1883 ssl://test.mosquitto.org:8883 (certificate needed) ssl://test.mosquitto.org:8884 (certificates needed)
	<b>MQTT Username</b> <input type="text" value="b8401ec0-a304-11ed-8d53-c"/>	MQTT Username (blank if not used)
	<b>MQTT Password</b> <input type="password" value=""/>	MQTT Password (blank if not used)
	<b>MQTT ID</b> <input type="text" value="ad2561f0-a3c0-11ed-8d53-d"/>	Device identification
	<b>MQTT Qos</b> <input type="text" value="1"/>	MQTT Quality Of Service (0 ... 2)
	<b>MQTT Keepalive</b> <input type="text" value="60"/>	Seconds for keepalive (30 ... 3600)
	<b>MQTT Persistence</b> <input type="checkbox"/>	Data persistence
	<b>MQTT AT Topic</b> <input type="text"/>	This topic will be subscribed for receiving AT Commands (usefull for individual device)
	<b>MQTT AT Resp Topic</b> <input type="text"/>	This topic will be used for publishing the AT Command Responses of AT Topic

Once the section has been configured, restart the device from the "Other > Reboot" menu. Following the reboot, the device will connect to the MyDevices' Cayenne platform after a few seconds and will appear on the left-hand side of the screen. This will be called the "Webdyn-Easy-Tunnel".

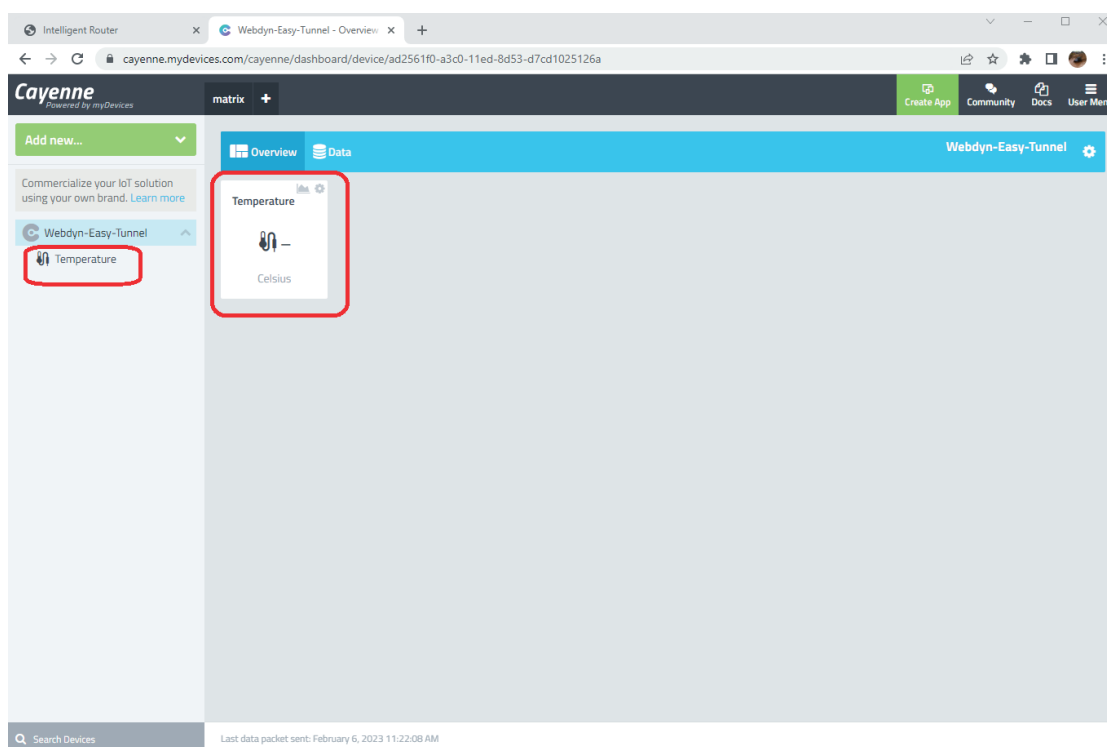
You must now add 3 Widgets: one for the temperature sensor, one for the humidity sensor and two others to remotely change the value of the PLC registers 30002 and 30003. Click again on "Add new > Device/Widget".



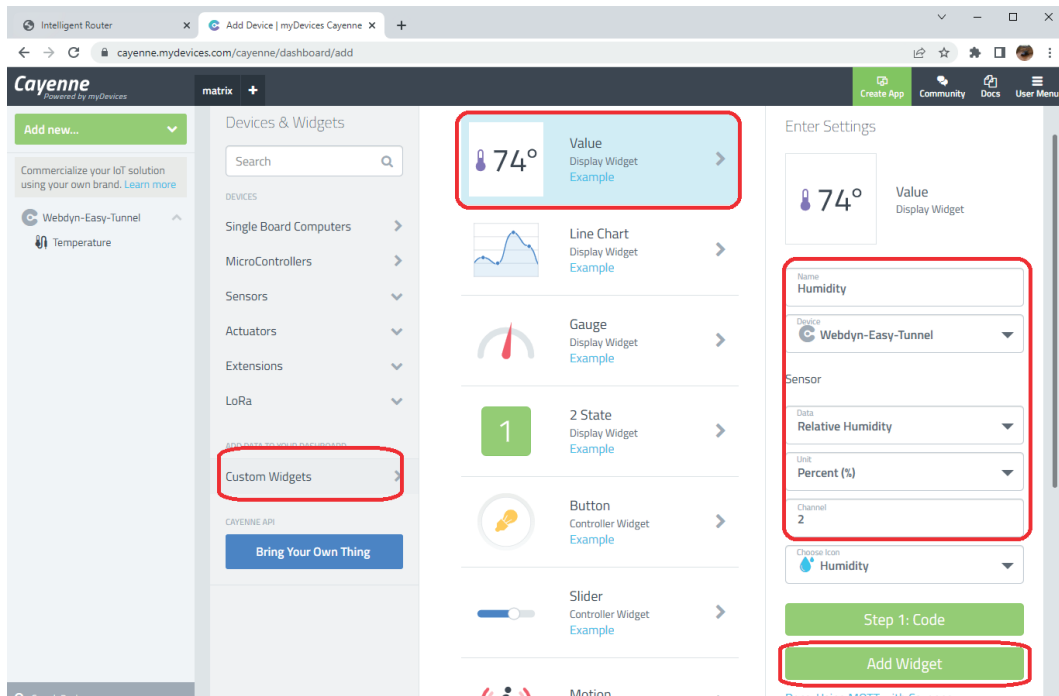
Then, select "Custom Widgets" > "Display Widget", as shown in the screenshot below, and select / fill in the indicated data. Please note that the Cayenne platform will use channel "1" for the temperature sensor.



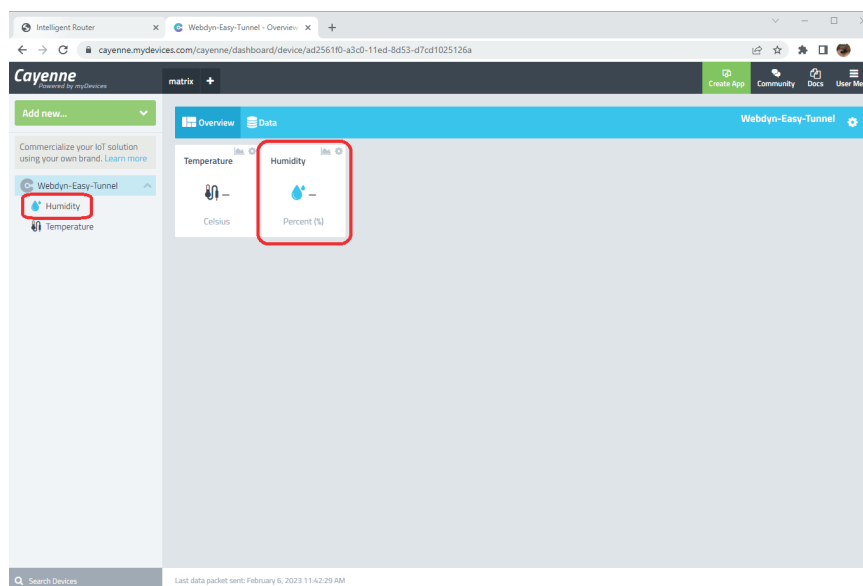
Finally, click on the "Add Widget" button and the newly created widget will appear on screen.



Now, repeat the same process for the humidity widget, assigning channel 2 in this instance.

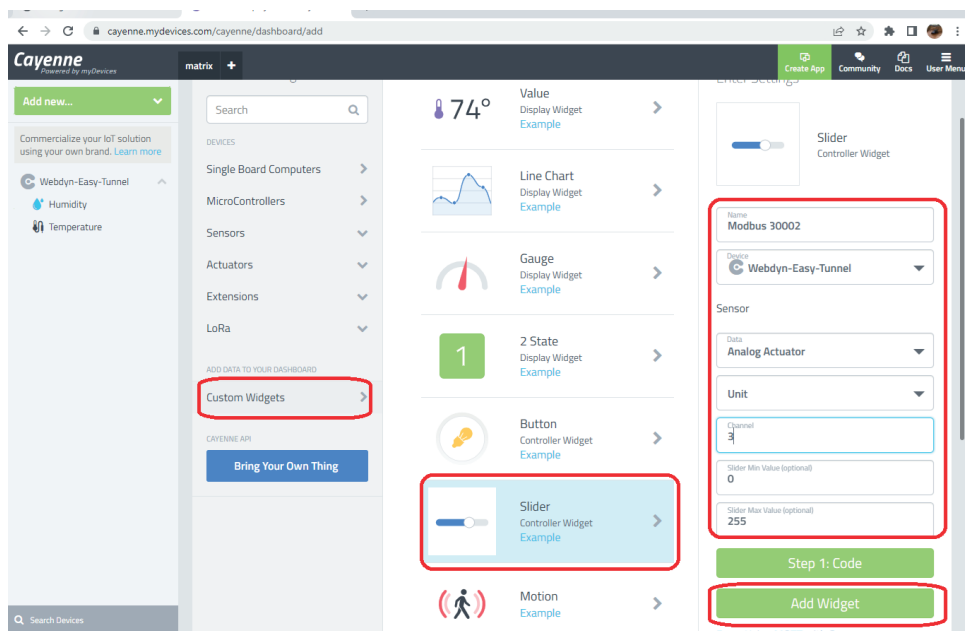


Click on the "Add Widget" button and the newly created widget will appear on screen.

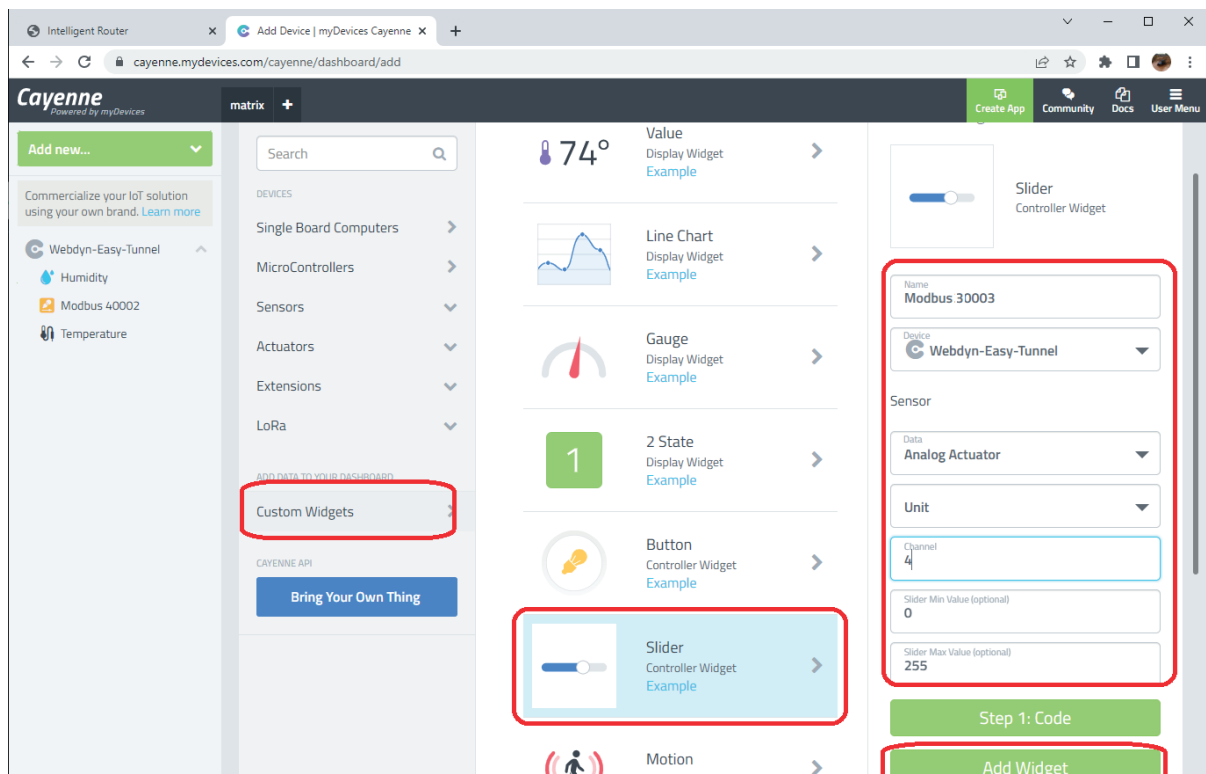


Then, add the widget to write an analogue value between 0 and 255 in register 30002 of the PLC. This widget will be assigned channel 3.

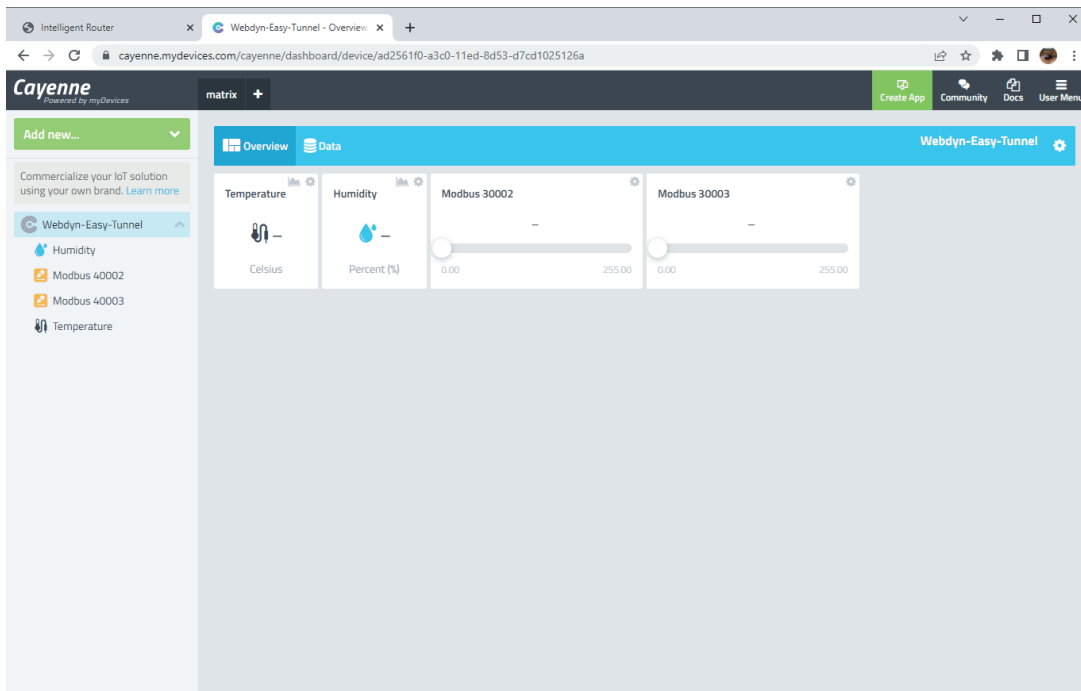




Then, add the widget to write an analogue value between 0 and 255 in register 30003 of the PLC. This widget will be assigned channel 4.



At this point, the dashboard display will look like this:



You must now write a script within the Webdyn-Easy-Tunnel to manage the sending and receiving of data from the platform.

## 5. Configuring the SCRIPT to read the Modbus registers of the PLC and manage Topics

In this example, a script will be used to read the Modbus registers from the PLC and send them to the platform. You will also integrate the required code into this script to accept the command sent from the MyDevices' Cayenne platform, in order to change the value of the Modbus registers 30002 and 30003 of the PLC. Add the following script in the "Other > Titan scripts" section:



The full code of which is detailed here:

```
mtx.println("EXAMPLE MyDevices.com - Running...");

var USERNAME="b8401ec0-a304-11ed-8d53-aaaaaaaaaaaa";
var CLIENTID="ad2561f0-a3c0-11ed-8d53-aaaaaaaaaaaa";
var subscribed=false;
var cont=9;

while (true)
{
    //***** CHECK IF MQTT CONNECTION IS OK
    if (mtx.mqttIsConnected())
    {
        //***** SUBSCRIBE TO THE MQTT TOPICS (for receiving data from platform)
        if (subscribed==false)
        {
            subscribedTopic0=mtx.mqttSubscribe("v1/" + USERNAME + "/things/" +
CLIENTID + "/cmd/3",0);
            subscribedTopic1=mtx.mqttSubscribe("v1/" + USERNAME + "/things/" +
CLIENTID + "/cmd/4",1);
            if ((subscribedTopic0)&&(subscribedTopic1))
                subscribed=true;
        }
        else
        {
            //***** EVERY 10 SECONDS READ AND SEND MODBUS DATA ...
            cont++;
        }
        if (cont==10)
        {
            cont=0;
            var res=mtx.modbusRTUGetWords(1,3,30000,2);
```

```

//***** IF THE MODBUS READING WAS CORRECT ...
if (res!==null)
{
    mtx.println("Temperature:" + res[0]/10);
    mtx.println("Humidity:" + res[1]);

    //*** BUILDING TOPIC AND JSON DATA
    var topic="v1/" + USERNAME+ "/things/" + CLIENTID + "/data/
json";
    var data="[{"channel\:1,\"value\":" + res[0]/10 + ",\"type\":"temp\", \"unit\":"c\"},";
    data=data + "{"channel\:2,\"value\":" + res[1] + ",\"type\":"rel_hum\", \"unit\":"p\"}]";

    //*** SENDING DATA TO CAYENNE (MyDevices.com)
    var r=mtx.mqttSend(data,topic,0);
}

}

//***** CHECK TOPICS ...
for (var topicID=0;topicID<2;topicID++)
{
    //***** IF THERE IS DATA AVAILABLE IN TOPIC
    var res=mtx.mqttGetArray(topicID)
    if(res!==null)
    {
        //***** CONVERT RECEIVED ARRAY BYTE INTO STRING
        var stringData=mtx.byteArrayToString(res,0,res.length);
        //***** READ seq AND value SENT FROM MYDEVICES

        var seq=stringData.substring(0, stringData.indexOf(","));
        var value=stringData.substring(stringData.indexOf(",")+1);
        //***** IF RECEIVED DATA COMES FROM TOPIC 0
        if (topicID==0)
        {

```

PLATFORM

```

        var array = [0];
        array[0]=value;
        var res=mtx.modbusRTUSetWords(1,16,30002,array);
    }
    //***** IF RECEIVED DATA COMES FROM TOPIC 1
    else if (topicID==1)
    {
        var array = [0];
        array[0]=value;
        var res=mtx.modbusRTUSetWords(1,16,30003,array);
    }

    //***** RESPONSE OK TO CAYENNE WITH SAME SEQ
    mtx.mqttSend("ok," + seq,"v1/" + USERNAME + "/things/" +
CLIENTID + "/response",0);
    }
    }
    }
    }
else
    subscribed=false;

    //***** 1 SECOND PAUSE
    mtx.pause(1000);
}

```

Essentially, commands sent from the MyDevices' Cayenne platform are sent to the mqtt topics:  
v1/<USERNAME>/things/<CLIENTID>/cmd/<CHANNEL>

The script therefore subscribes to 2 topics, namely channels 2 and 3, i.e., the channels that will receive the commands to change the Modbus registers 30002 and 30003.

Depending on whether the script receives data in one or the other topic, it will write to register 30002 or 30003.

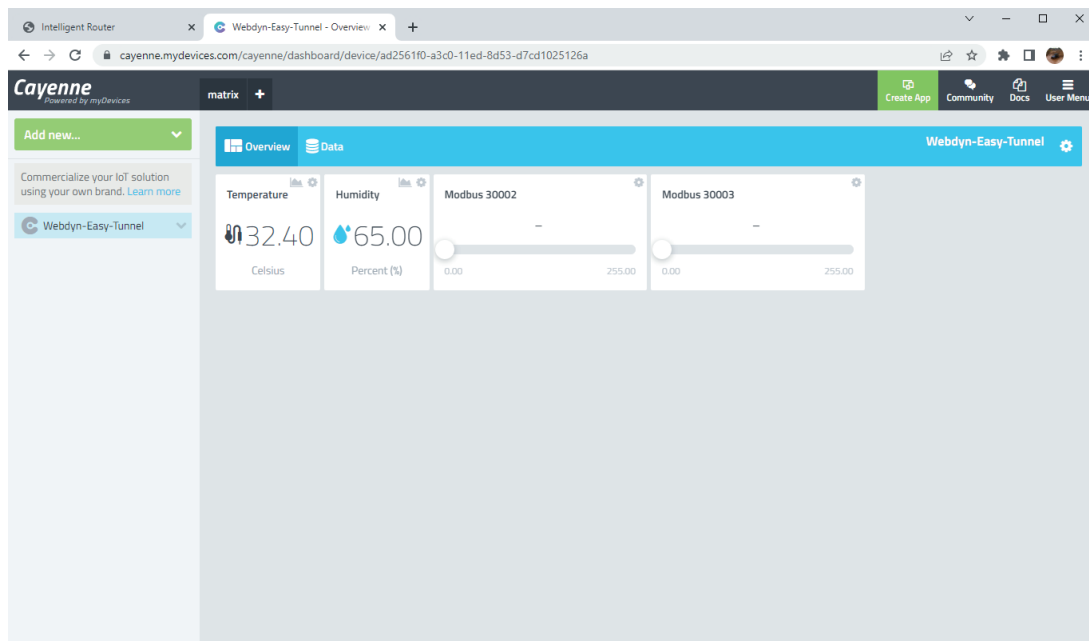
More detailed information on the JSON format for sending data to the platform can be found here:  
<https://docs.mydevices.com/docs/device/mqtt>

## 6. Testing the example

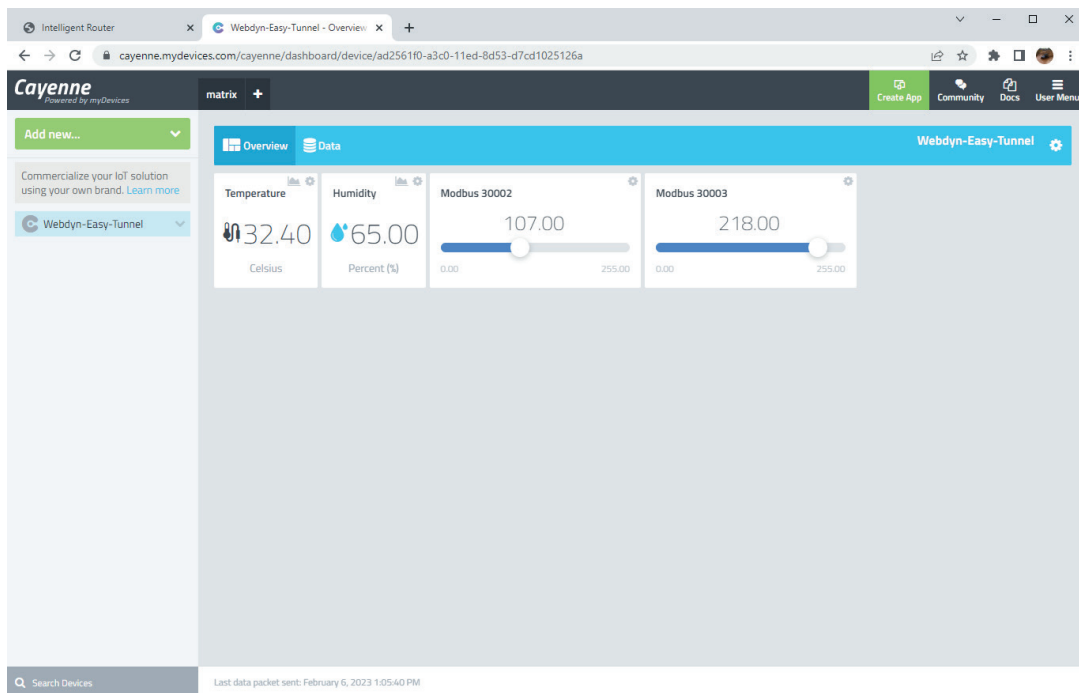
The only thing left for you to do now is to execute your String. Once executed, you should be able to see how the temperature and humidity are being read in the Logs display:



These values should also be reflected in the dashboard of the MyDevices' Cayenne platform:



Similarly, if you change the values of the sliders from the Cayenne platform (to alter the values of the Modbus registers 30002 and 30003), you can check that everything works correctly.



Mbslave1

ID = 1: F = 03

	Alias	30000
0		324
1		65
2		107
3		218
4		
5		
6		
7		
8		
9		

Any questions?

Please direct your enquiries to [iotsupport@mtxm2m.com](mailto:iotsupport@mtxm2m.com)