Lesson Two – MQTT Virtual Button

Internet and Virtual Button with MQTT

Check NodeMCU connection

See last module - INSTALL USB DRIVERS
set port plugged into

Download and run NodeMCUButton code to board.

Blink slow but when button pressed, blink quick.

Set up Losant Account

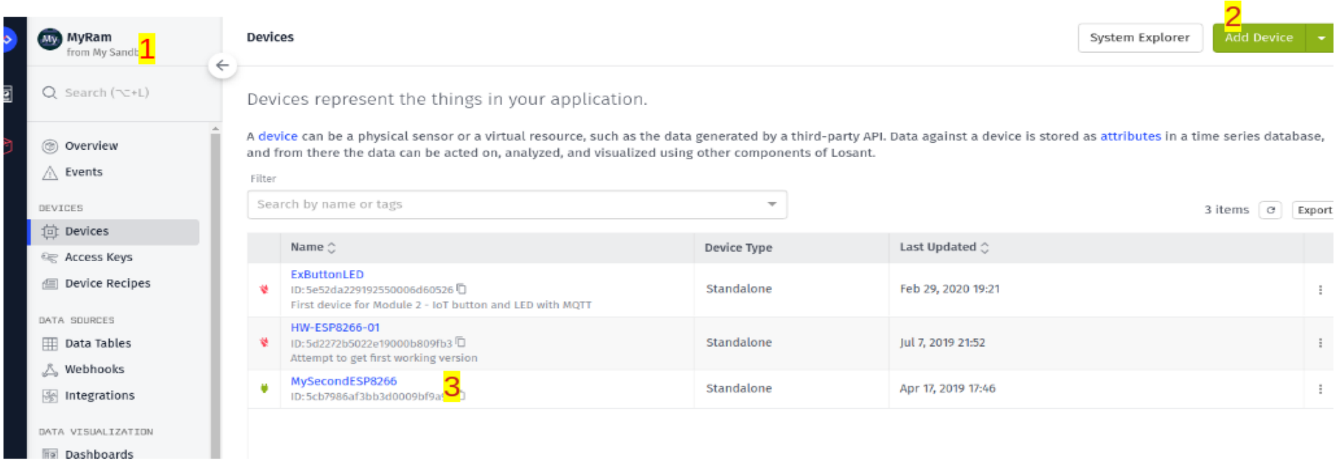
These are simplified instructions similar to the following at Losant:
https://docs.losant.com/getting-started/walkthrough/

Section 3. LOSANT SETUP

Create Account

If you don't already have an account, navigate to https://accounts.losant.com/create-account to register.

Add Device



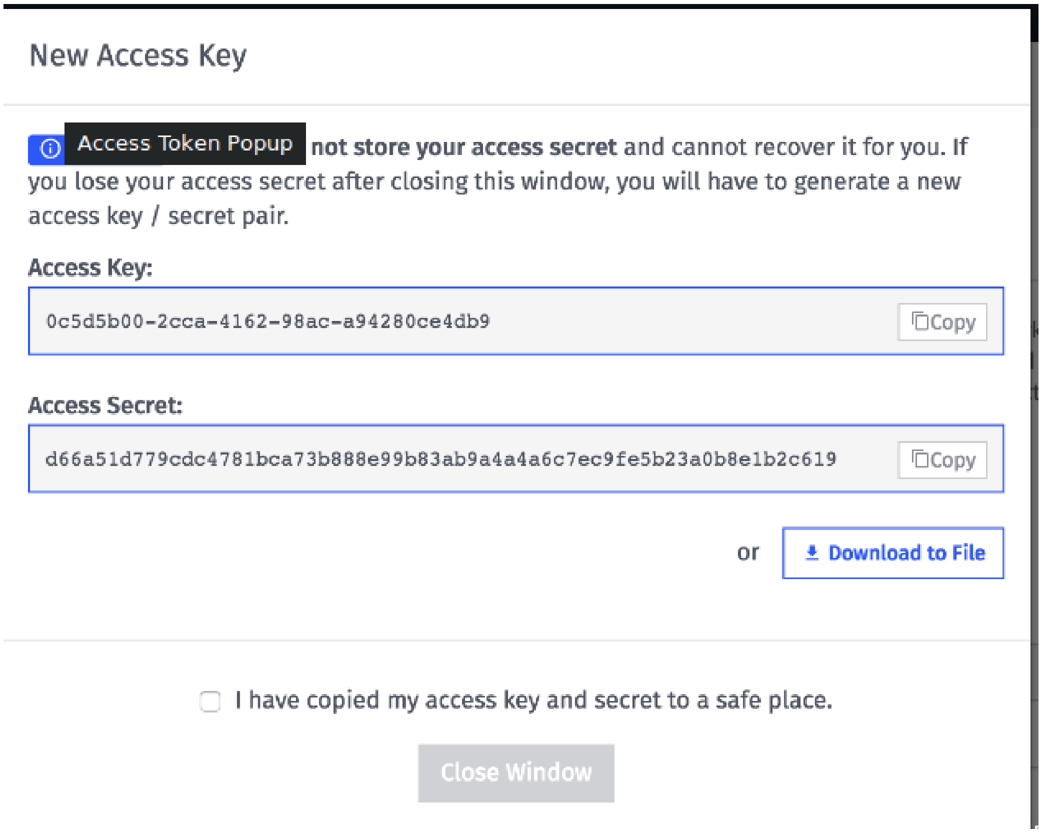
Devices represent the things in your application.

A device can be a physical sensor or a virtual resource, such as the data generated by a third-party API. Data against a device is stored as attributes in a time series database, and from there the data can be acted on, analyzed, and visualized using other components of Losant.

Name	Device Type	Last Updated
ExButtonLED ID:5e52da229192550006d60526 First device for Module 2 - IoT button and LED with MQTT	Standalone	Feb 29, 2020 19:21
HW-ESP8266-01 ID:5d2272b5022e19000b809fb3 Attempt to get first working version	Standalone	Jul 7, 2019 21:52
MySecondESP8266 ID:5cb7986af3bb3d0009bf9a	Standalone	Apr 17, 2019 17:46

Generate An Access Key

NOTE: Generating Access Keys for a device will override older device keys for same device. Old device keys for same device will not work.



not store your access secret and cannot recover it for you. If you lose your access secret after closing this window, you will have to generate a new access key / secret pair.

Access Key: 0c5d5b00-2cca-4162-98ac-a94280ce4db9

Access Secret: d66a51d779cdc4781bca73b888e99b83ab9a4a4a6c7ec9fe5b23a0b8e1b2c619

□ I have copied my access key and secret to a safe place.

Download to File for this module

Devices:
https://docs.losant.com/devices/overview/

“Add Device”

Choosing a Device Type
Standalone

Device Configuration

DEVICE TAGS
Tags are also a great place to store things like device configuration or threshold values, as the tag keys and values for a device are available to use within workflows

Device Attributes
Attributes are properties of a device that define the data that can be reported as state against that device. State data cannot be reported against the device unless the values in the state report have first been defined as attributes.

A device’s attributes can be viewed and modified under the “Attributes” tab of a device’s configuration page.

Lesson Two – MQTT Virtual Button

Device State
Device state is one of the core communication and data components of the Losant Platform.
A device's state represents a snapshot of the device at some point in time.
For example, if the device has a temperature sensor, it might report the temperature every minute of so.
Losant stores every state request and makes them available in visualizations and workflow triggers.

USING STATE
Once your device is reporting state, the information is most commonly used for dashboard visualizations and workflow triggers.

There are many dashboard blocks that show state information.
The blocks can either show the real-time value as it's reported or show aggregates over time.
For a thermostat, for example, you might want to graph the average indoor temperature per day over the last 30 days.

Device state can also trigger workflows. Workflows are the primary way your devices will connect with each other and the outside world. A typical workflow for a thermostat might be to send an SMS if the indoor temperature drops below 40 degrees, which would indicate the user's furnace is malfunctioning.

5. WORKSHOP 1 – INTERNET BUTTON

Code URL:

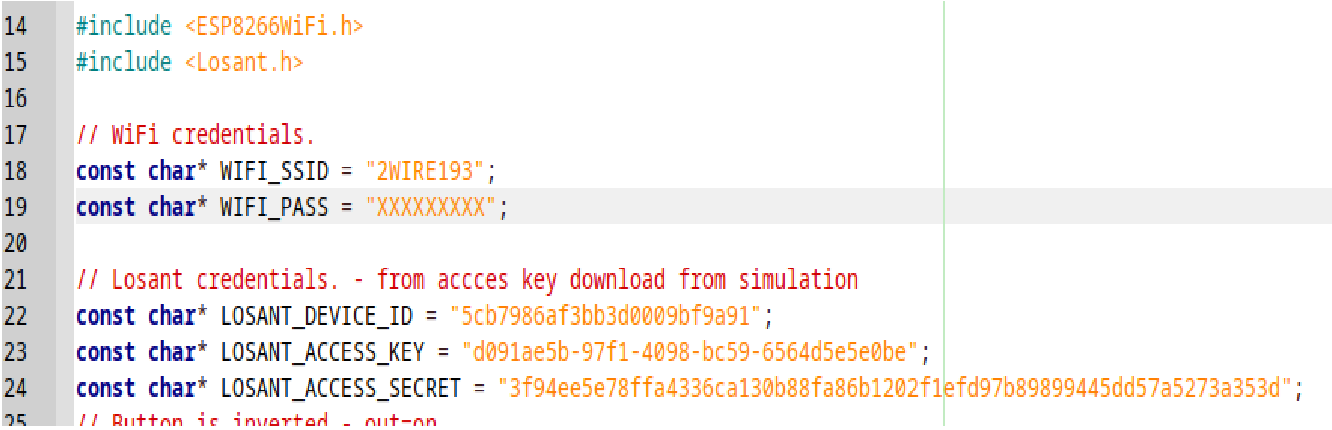
http://web.eng.fiu.edu/watsonh/EEL4730/MQTT/sketch_esp8266LosantSimple.ino

Save as a Sketch file within the Arduino IDE.

Edit the following variables at the top of the ESP8266 source file.

1. WIFI_SSID: The name of your WiFi network.
2. WIFI_PASS: Your WiFi password.
3. LOSANTDEVICEID: After creating your device, the device ID is printed on the page in a gray box. You can also find it next to the name of your device on your application's "View All Devices" page.
4. LOSANTACCESSKEY: Set this to the access key you generated after creating the Losant application.

5.LOSANTACCESSSECRET: Set this to the access secret you generated after creating the Losant application.



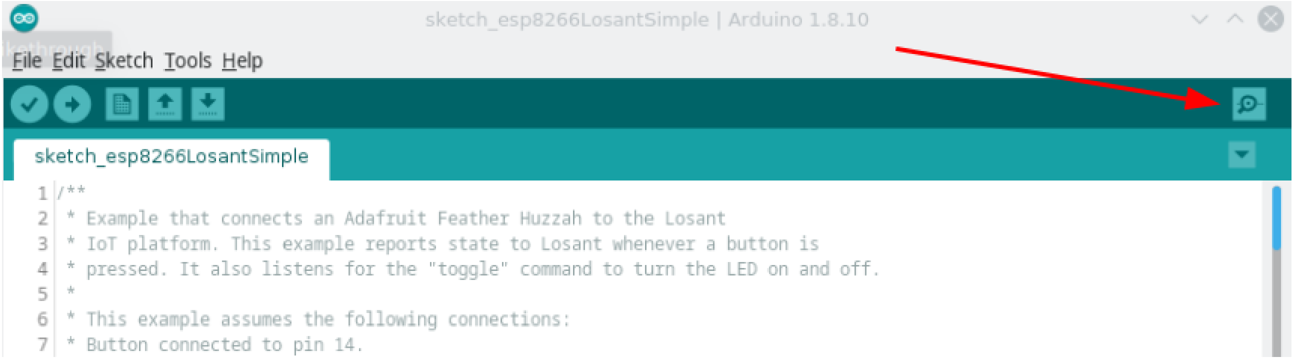
```
#include <ESP8266WiFi.h>
#include <Losant.h>

// WiFi credentials.
const char* WIFI_SSID = "2WIRE193";
const char* WIFI_PASS = "XXXXXXXXX";

// Losant credentials. - from accces key download from simulation
const char* LOSANT_DEVICE_ID = "5cb7986af3bb3d0009bf9a91";
const char* LOSANT_ACCESS_KEY = "d091ae5b-97f1-4098-bc59-6564d5e5e0be";
const char* LOSANT_ACCESS_SECRET = "3f94ee5e78ffa4336ca130b88fa86b1202f1efd97b89899445dd57a5273a353d";
// Button is inverted - out-on
```

Save the sketch. Then compile and upload to the NodeMCU board.

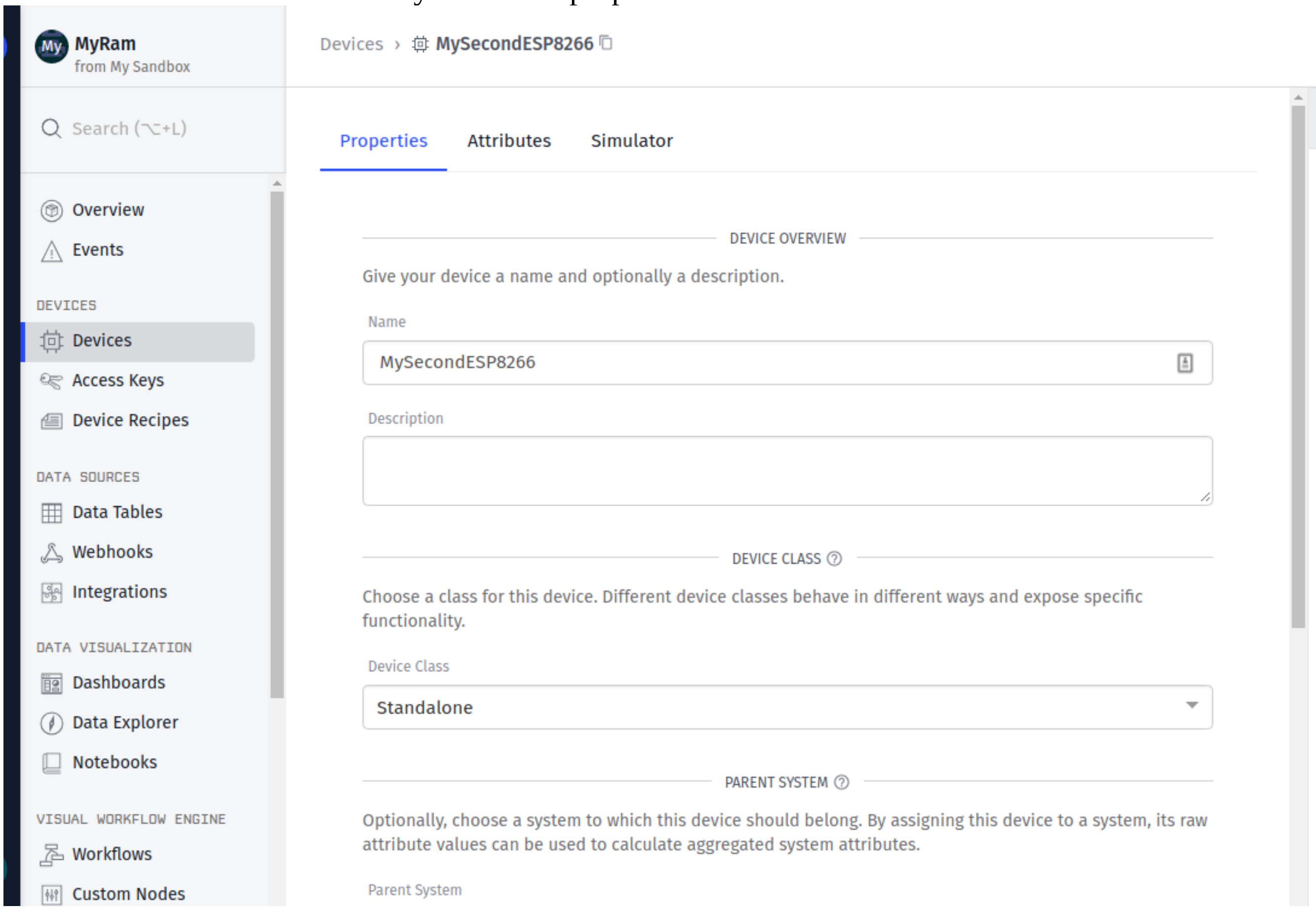
Once the code is compiled and uploaded, open the Serial Monitor Screen:


sketch_esp8266LosantSimple | Arduino 1.8.10
File Edit Sketch Tools Help
sketch_esp8266LosantSimple

```
/**
 * Example that connects an Adafruit Feather Huzzah to the Losant
 * IoT platform. This example reports state to Losant whenever a button is
 * pressed. It also listens for the "toggle" command to turn the LED on and off.
 *
 * This example assumes the following connections:
 * Button connected to pin 14.
```

Lesson Two – MQTT Virtual Button

Go to Losant ‘Devices’ and edit your device properties – choose Standalone



DEVICE OVERVIEW

Give your device a name and optionally a description.

Name: MySecondESP8266

Description:

DEVICE CLASS

Choose a class for this device. Different device classes behave in different ways and expose specific functionality.

Device Class: Standalone

PARENT SYSTEM

Optionally, choose a system to which this device should belong. By assigning this device to a system, its raw attribute values can be used to calculate aggregated system attributes.

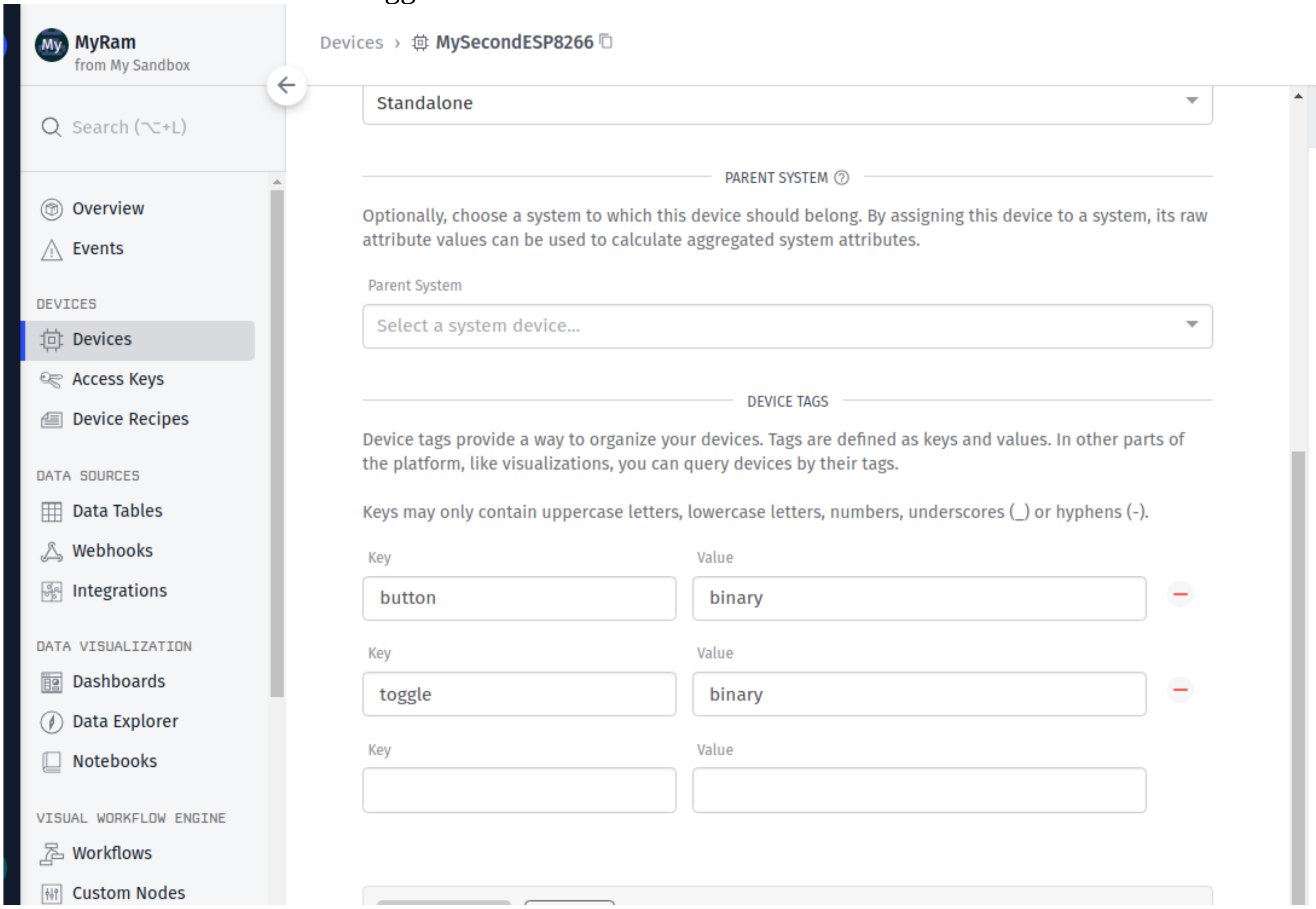
Parent System

Lesson Two – MQTT Virtual Button

At the bottom of the same screen, set the attributes for communication

The first key is the receiving key which is ‘button’ and binary data. This is what is received from the Node MCU when the ‘Flash’ button is pressed.

The second key is the command key which is ‘toggle’ and also binary. This is the command that will be sent to the NodeMCU to toggle the on board LED.



Devices › MySecondESP8266

Standalone

PARENT SYSTEM

Optionally, choose a system to which this device should belong. By assigning this device to a system, its raw attribute values can be used to calculate aggregated system attributes.

Parent System: Select a system device...

DEVICE TAGS

Device tags provide a way to organize your devices. Tags are defined as keys and values. In other parts of the platform, like visualizations, you can query devices by their tags.

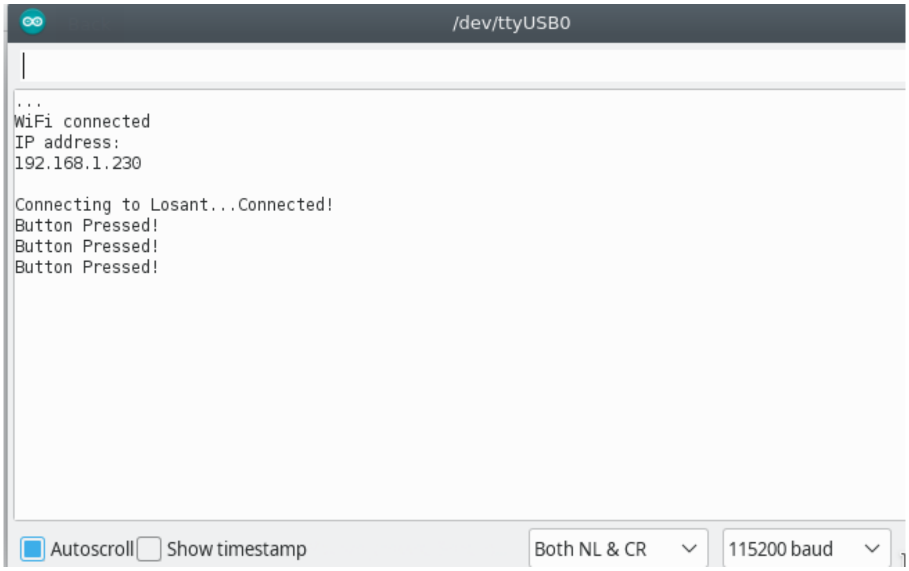
Keys may only contain uppercase letters, lowercase letters, numbers, underscores (_) or hyphens (-).

Key	Value
button	binary
toggle	binary

Save the device and attributes on Losant.

Lesson Two – MQTT Virtual Button

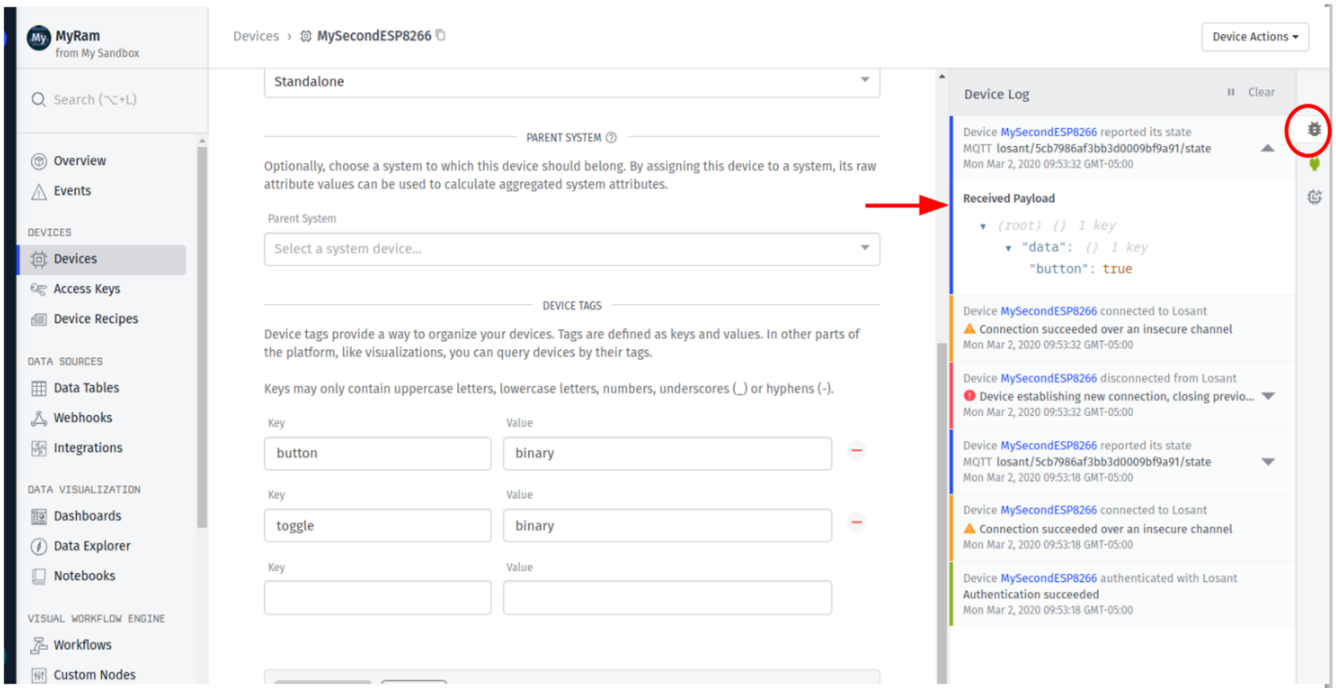
On the NodeMCU, press the button labeled “Flash” on the board and watch for the response in the Serial Monitor



```
...
WiFi connected
IP address:
192.168.1.230

Connecting to Losant...Connected!
Button Pressed!
Button Pressed!
Button Pressed!
```

Go to the Device Log Screen on Losant and press the button – see the response:



Standalone

Optionally, choose a system to which this device should belong. By assigning this device to a system, its raw attribute values can be used to calculate aggregated system attributes.

Parent System: Select a system device...

Device tags provide a way to organize your devices. Tags are defined as keys and values. In other parts of the platform, like visualizations, you can query devices by their tags.

Keys may only contain uppercase letters, lowercase letters, numbers, underscores (_) or hyphens (-).

Key	Value
button	binary
toggle	binary

```
Device MySecondESP8266 reported its state
MQTT losant/5cb7986af3bb3d0009bf9a91/state
Mon Mar 2, 2020 09:53:32 GMT-05:00

Received Payload
  ▾ (root) {} 1 key
    ▾ "data": {} 1 key
       "button": true

Device MySecondESP8266 connected to Losant
⚠ Connection succeeded over an insecure channel
Mon Mar 2, 2020 09:53:32 GMT-05:00

Device MySecondESP8266 disconnected from Losant
❗ Device establishing new connection, closing previo...
Mon Mar 2, 2020 09:53:32 GMT-05:00

Device MySecondESP8266 reported its state
MQTT losant/5cb7986af3bb3d0009bf9a91/state
Mon Mar 2, 2020 09:53:18 GMT-05:00

Device MySecondESP8266 connected to Losant
⚠ Connection succeeded over an insecure channel
Mon Mar 2, 2020 09:53:18 GMT-05:00

Device MySecondESP8266 authenticated with Losant
Authentication succeeded
Mon Mar 2, 2020 09:53:18 GMT-05:00
```

Every time the button is pressed, the firmware is publishing the state { "button": true } to Losant. Currently, this firmware only reports when the button goes to true, it does not report the button going back to false. So if you hold the button down for a long time, it will only report when the button is initially pressed.

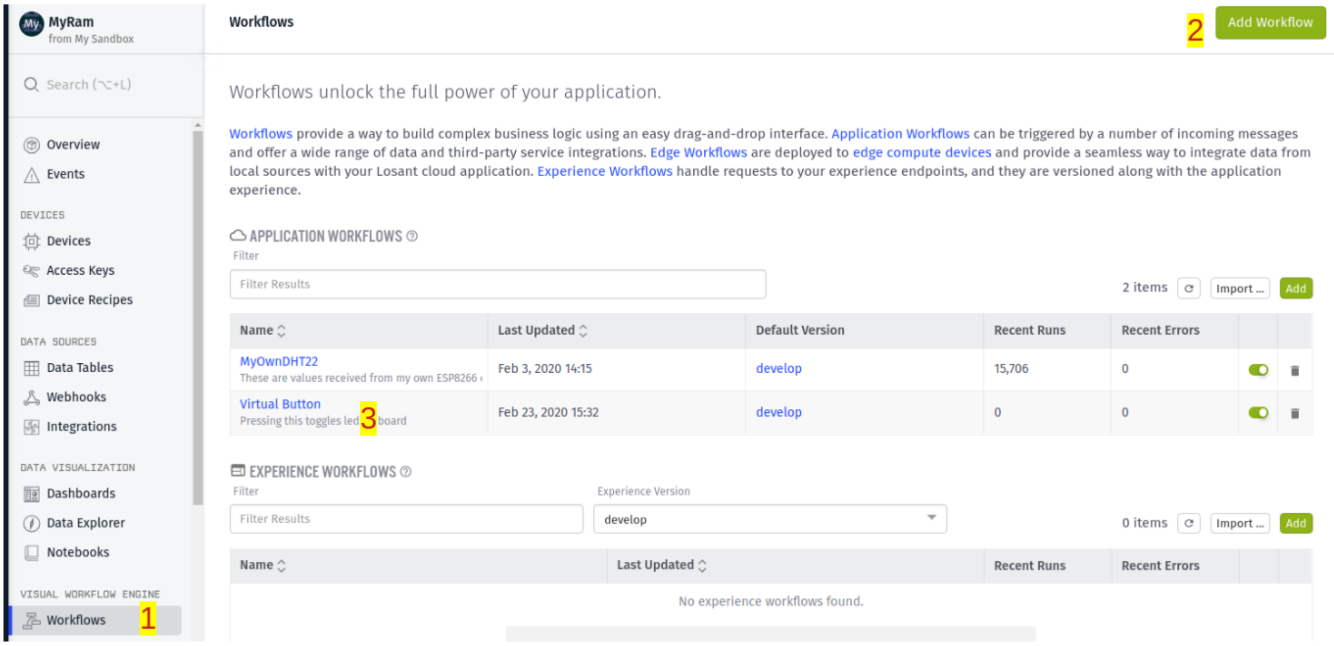
7. WORKSHOP 2 – REMOTE CONTROL LED

Send command from Losant to the NodeMCU to show how a Workflow virtual button can do something in the physical world.

Skip the wiring, we are using the LED on the NodeMCU Board

The firmware is already flashed into the NodeMCU from the InternetButton example above.

Create a Workflow for the device



Workflows

Workflows unlock the full power of your application.

Workflows provide a way to build complex business logic using an easy drag-and-drop interface. Application Workflows can be triggered by a number of incoming messages and offer a wide range of data and third-party service integrations. Edge Workflows are deployed to edge compute devices and provide a seamless way to integrate data from local sources with your Losant cloud application. Experience Workflows handle requests to your experience endpoints, and they are versioned along with the application experience.

APPLICATION WORKFLOWS

Filter: Filter Results

2 items Import ... Add

Name	Last Updated	Default Version	Recent Runs	Recent Errors		
MyOwnDHT22 These are values received from my own ESP8266 (	Feb 3, 2020 14:15	develop	15,706	0	●	🗑
Virtual Button Pressing this toggles led 3 board	Feb 23, 2020 15:32	develop	0	0	●	🗑

EXPERIENCE WORKFLOWS

Filter: Filter Results

Experience Version: develop

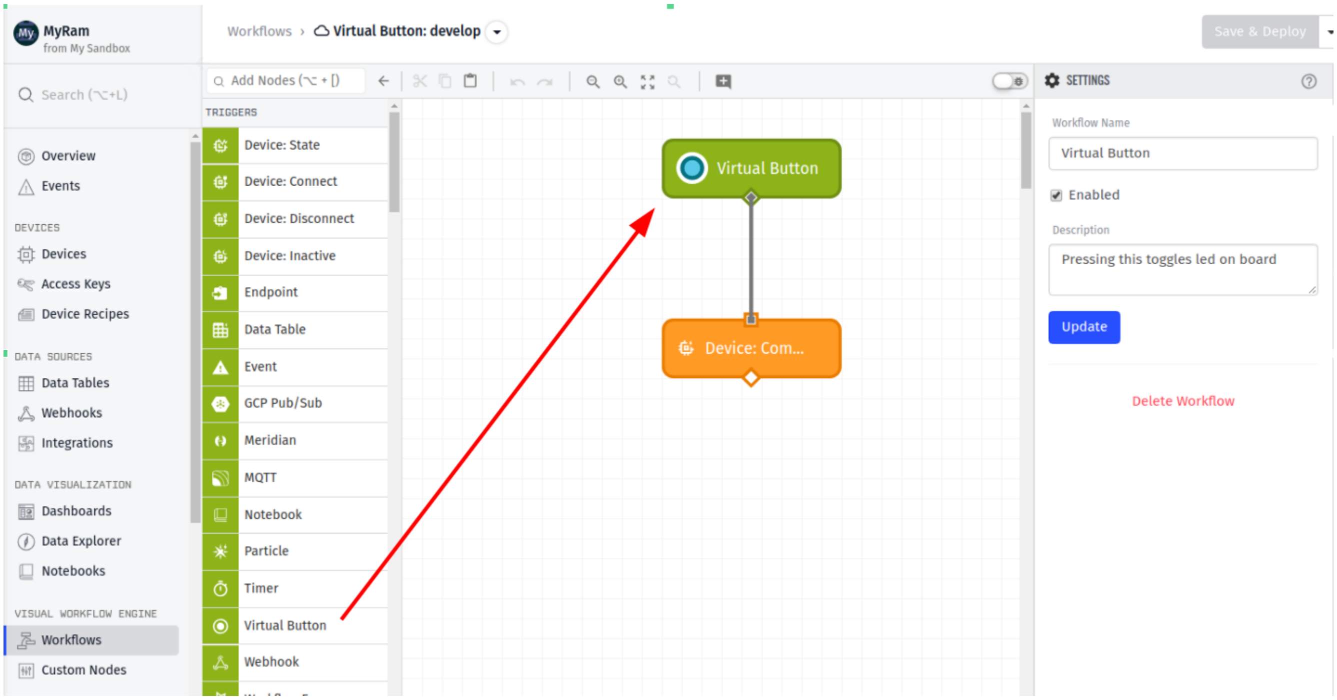
0 items Import ... Add

Name	Last Updated	Recent Runs	Recent Errors
No experience workflows found.			

Lesson Two – MQTT Virtual Button

Create the Virtual Button

Open the Workflow, select Virtual button under triggers and place onto the develop space.

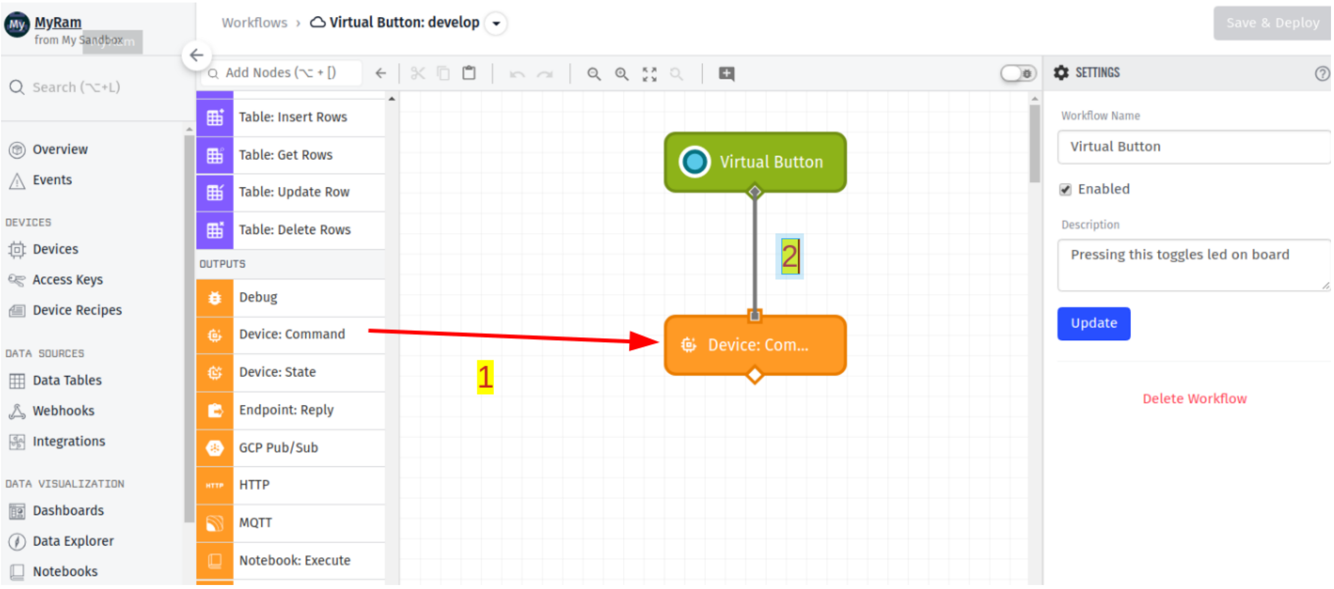


Workflow Name: Virtual Button

☑ Enabled

Description: Pressing this toggles led on board

Now add Device Output command and connect with the Virtual Button



Workflow Name: Virtual Button

☑ Enabled

Description: Pressing this toggles led on board

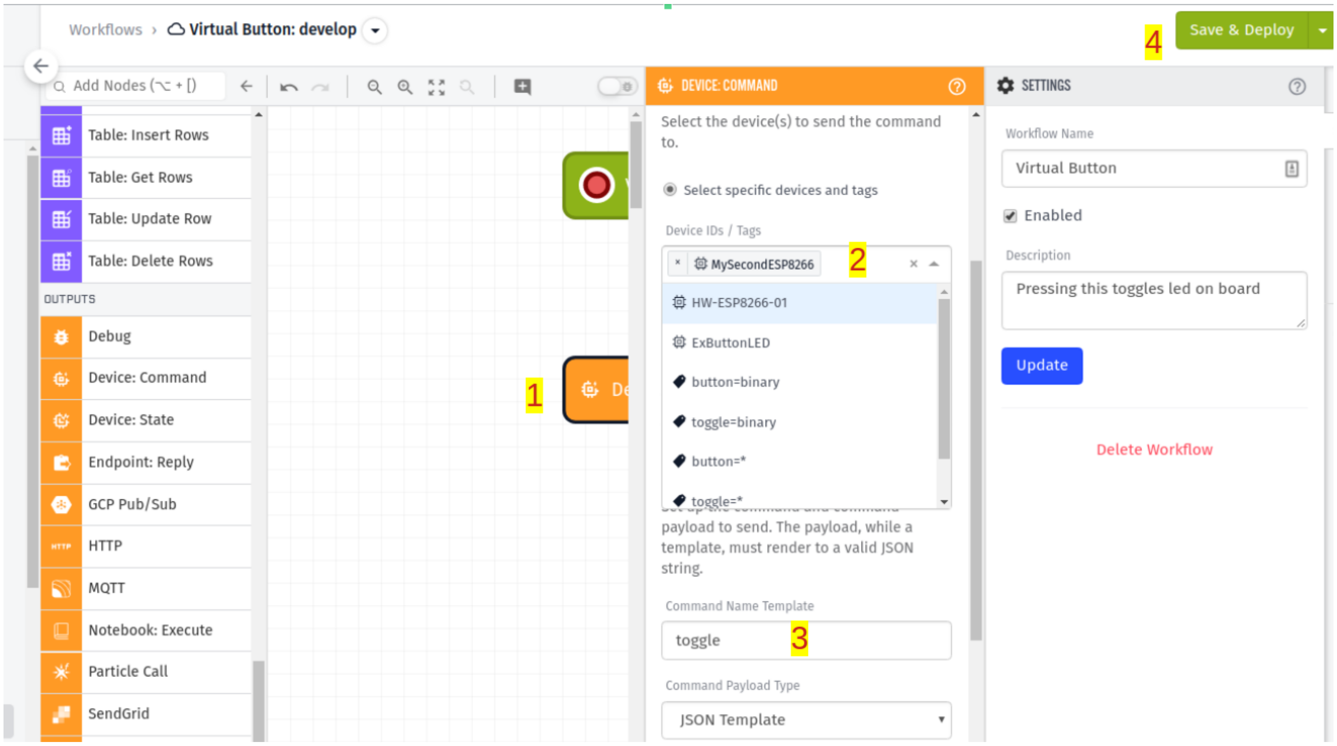
Lesson Two – MQTT Virtual Button

1. Select the Device Command. The Device Command values can then be set.

2. From the drop down list, select the Device to connect to the command

3. Enter the command key word ‘toggle’ which will be sent when the Virtual Button is selected

4. Save and deploy the Workflow



Select the device(s) to send the command to.

● Select specific devices and tags

Device IDs / Tags: MySecondESP8266

... payload to send. The payload, while a template, must render to a valid JSON string.

Command Name Template: toggle

Command Payload Type: JSON Template

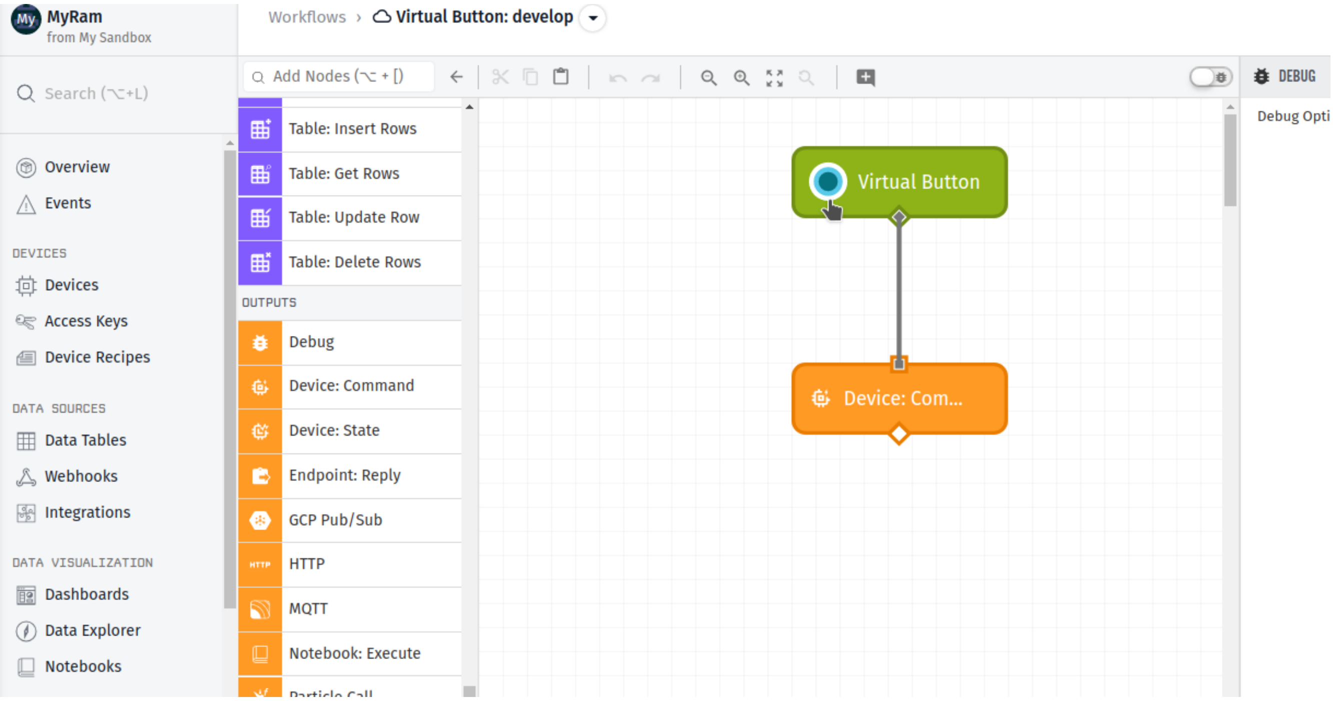
Workflow Name: Virtual Button

☑ Enabled

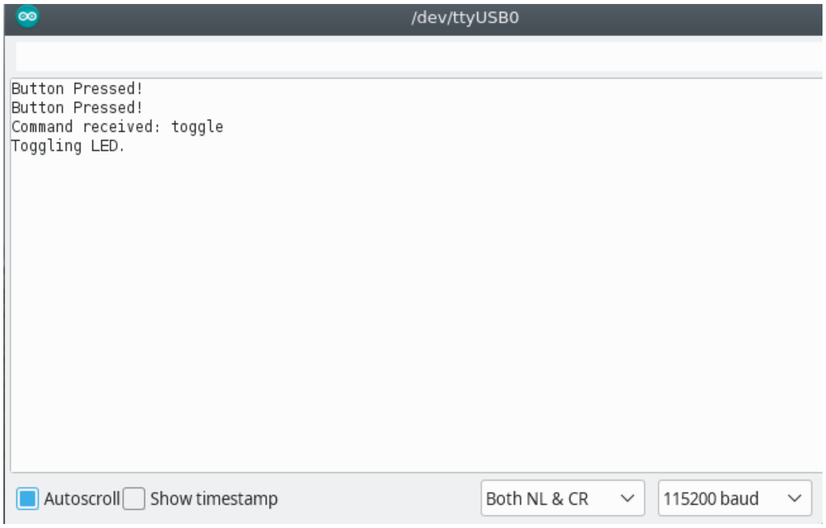
Description: Pressing this toggles led on board

Lesson Two – MQTT Virtual Button

Click on the Virtual button, it should toggle the on board LED



The Serial monitor should show the received command.

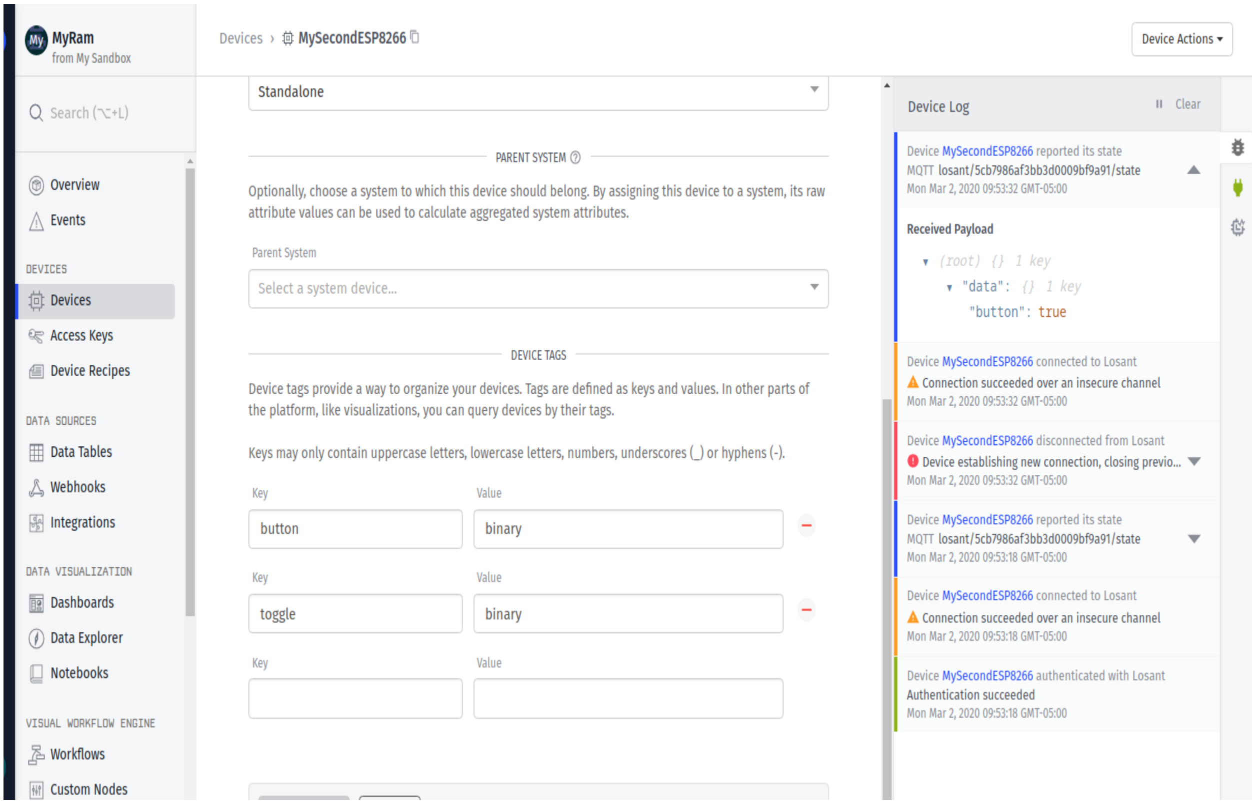


```
Button Pressed!
Button Pressed!
Command received: toggle
Toggling LED.
```

Take screen shot of Serial monitor and LED

Lesson Two – MQTT Virtual Button

Also take screen shot of debug screen device log and turn both in for the Lesson Submission



Standalone

PARENT SYSTEM

Optionally, choose a system to which this device should belong. By assigning this device to a system, its raw attribute values can be used to calculate aggregated system attributes.

Parent System: Select a system device...

DEVICE TAGS

Device tags provide a way to organize your devices. Tags are defined as keys and values. In other parts of the platform, like visualizations, you can query devices by their tags.

Keys may only contain uppercase letters, lowercase letters, numbers, underscores (_) or hyphens (-).

Key	Value
button	binary
toggle	binary

Device Log

```
Device MySecondESP8266 reported its state
MQTT losant/5cb7986af3bb3d0009bf9a91/state
Mon Mar 2, 2020 09:53:32 GMT-05:00

Received Payload
  ▾ (root) {} 1 key
     ▾ "data": {} 1 key
        "button": true

Device MySecondESP8266 connected to Losant
⚠ Connection succeeded over an insecure channel
Mon Mar 2, 2020 09:53:32 GMT-05:00

Device MySecondESP8266 disconnected from Losant
❗ Device establishing new connection, closing previo...
Mon Mar 2, 2020 09:53:32 GMT-05:00

Device MySecondESP8266 reported its state
MQTT losant/5cb7986af3bb3d0009bf9a91/state
Mon Mar 2, 2020 09:53:18 GMT-05:00

Device MySecondESP8266 connected to Losant
⚠ Connection succeeded over an insecure channel
Mon Mar 2, 2020 09:53:18 GMT-05:00

Device MySecondESP8266 authenticated with Losant
Authentication succeeded
Mon Mar 2, 2020 09:53:18 GMT-05:00
```