Microsoft Azure AnyCloud™ Exercise:

Add Multimeter Click to the WBZ451 OOB Example

Table of Contents

MICROCHIP
UNIVERSITY

Introduction

You will learn how to modify both the device firmware and the corresponding IoT Central application(s) to incorporate additional telemetry into the existing OOB WBZ451 Curiosity Board AnyCloud™ example.

Required Hardware Tools:

- WFI32-IoT Development Board (Part No. EV36W50A)
- WBZ451 Curiosity Board (Part No. EV96B94A)
- Multimeter click (Part No. MIKROE-3116)
- mikroBUS Xplained Pro (Part No. ATMBUSADAPTER-XPRO)
- XPRO-Adapter click (Part No. MIKROE-4123)
- USB-to-UART Serial Adapter
- Adjustable DC Power Supply

Required Software Tools:

- MPLAB Integrated Production Environment (IPE) v6.05
- MPLAB X Integrated Development Environment (IDE) v6.05
- MPLAB® XC32 Compiler version 4.20
- Terminal emulator software

Upon completion, you will:

- Connect the WBZ451 Curiosity Board to your own Azure IoT Central application using a WFI32-IoT Development Board as the UART-to-Cloud serial bridge
- Add new sensors (via Multimeter click) to the WBZ451 Curiosity Board
- Implement a new Digital Twins Model Interface (DTMI) to enable the Multimeter click telemetry to be communicated to the IoT Central application
- Create a dashboard in the IoT Central application to visualize the telemetry data reported by the Multimeter click
- Extend the IoT Central application to manage devices at scale by creating analytics queries, running jobs, and setting rules

Lab 1 – Set up the AnyCloud™ OOB Demo for the WBZ451 Curiosity Board as the Host MCU

Purpose:

Verify that the WBZ451 Curiosity Board can successfully connect to Microsoft Azure through the WFI32-IoT Development Board (which has been programmed with the latest release of AnyCloud™ firmware)

Overview:

In this exercise we will use an Out-Of-Box demo to provision the WFI32-IoT Development Board with the latest release of the AnyCloud™ firmware to serve as the "UART-to-Cloud" serial bridge. The WBZ451 Curiosity Board will be programmed with Host MCU firmware to emulate an IoT device connecting to IoT Central.

Procedure:

Complete the procedures outlined in Microchip's **AzureDemo_AnyCloud** repository (up until the section titled "Adding Extra Sensors to the Embedded Firmware Example" which is achieved with the remainder of the lab exercises ☺)

NOTE: If you had previously gone through the repository and successfully connected the WBZ451 Curiosity Board to IoT Central, proceed to Lab 2.

Results:

The WBZ451 Curiosity Board successfully connects to IoT Central and reports its own temperature sensor values to the Cloud application. All Cloud-writable properties can be changed from within the IoT Central application.

Summary:

By completing this lab exercise, you will have verified that the Azure AnyCloud™ firmware and corresponding example OOB demo application works with your hardware – and is now ready for some additional customizations.

Lab 2 – Add Sensors to Device

Purpose:

Learn how to add extra sensors to an existing IoT device that has already successfully connected to Azure IoT Central

Overview:

In this exercise we will add extra sensors to an existing Azure IoT device by installing the MikroElektronika Multimeter click board onto the WBZ451 Curiosity Board and then modifying the "IoT Plug and Play" Device Twin Model Interface (DTMI) to accommodate the additional telemetry data reported by the Multimeter click board.

MULTIMETER CLICK ELECTRICAL SPECIFICATIONS

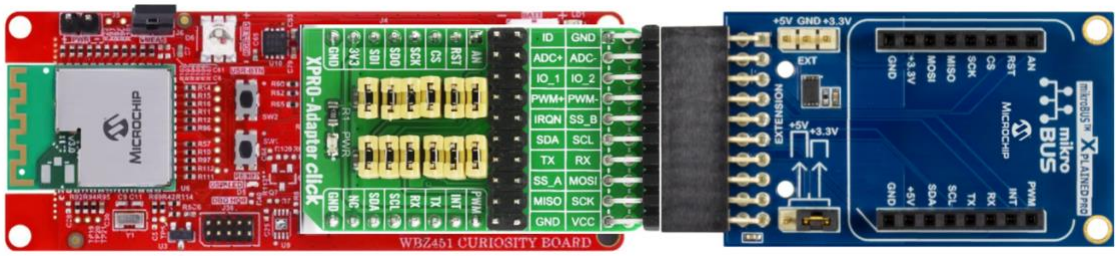
Description	Min	Type	Max	Unit
Measurement resolution	-	$5e^{-4}$	-	V/bit
Current measurement range	0.001	-	1.024	A
Voltage measurement range	0.008	-	17.068	V
Resistance measurement range	0.25	-	20M	Ω
Capacitance measurement range	0.001	-	2.2	µF

Procedure:

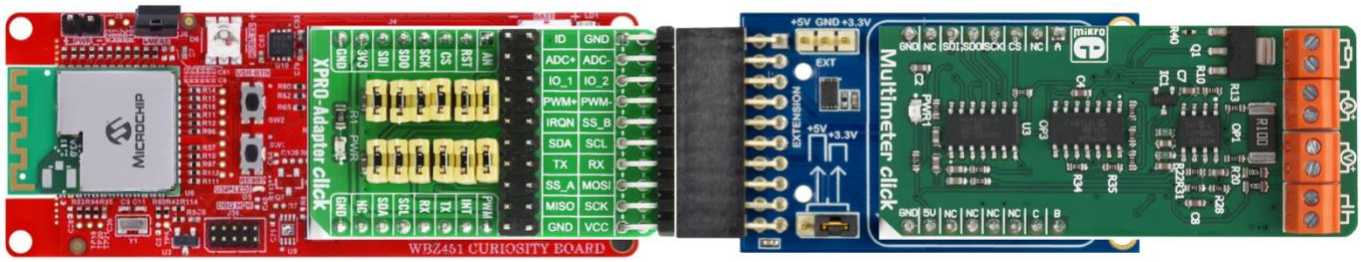
STEP 1: Assemble the multiple boards required to add the Multimeter click

Step 1a: Install the **XPRO-Adapter click** onto the WBZ451 Curiosity Board's mikroBUS socket

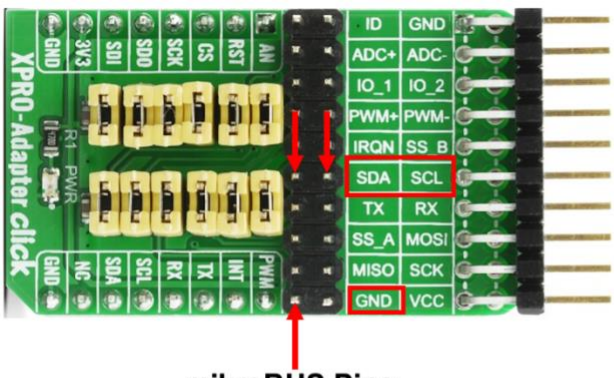
Step 1b: Connect the **mikroBUS Xplained Pro** to the XPRO-Adapter click

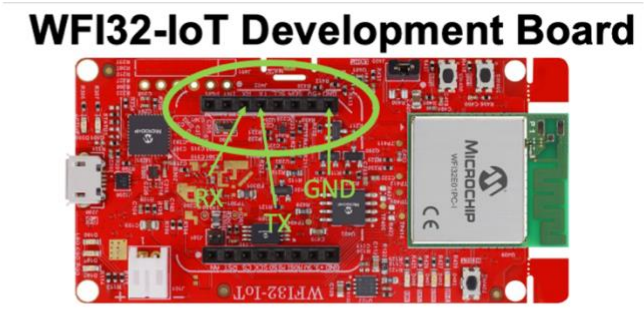


Step 1c: Install the **Multimeter click** onto the mikroBUS Xplained Pro



Step 1d: Connect 3 jumper wires between the XPRO-Adapter click & WFI32-IoT Development Board as shown in the following wiring diagram





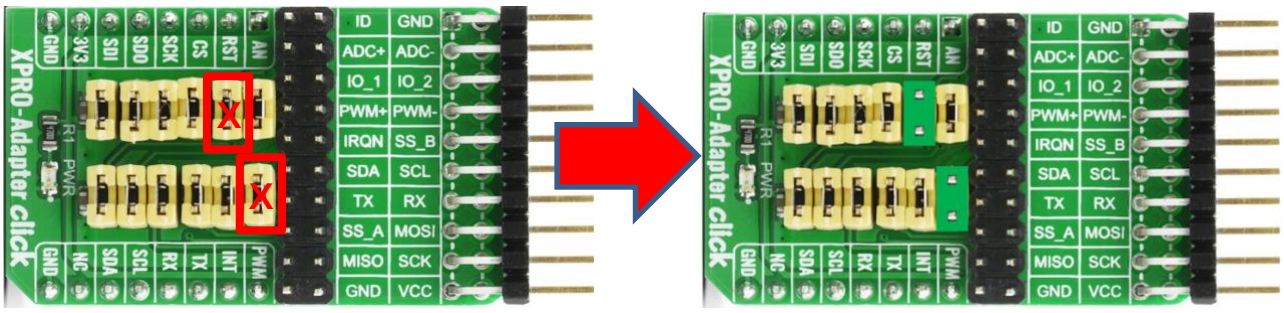
mikroBUS Pins

Pin Name	Pin Number
GND	9
SDA	11
SCL	12

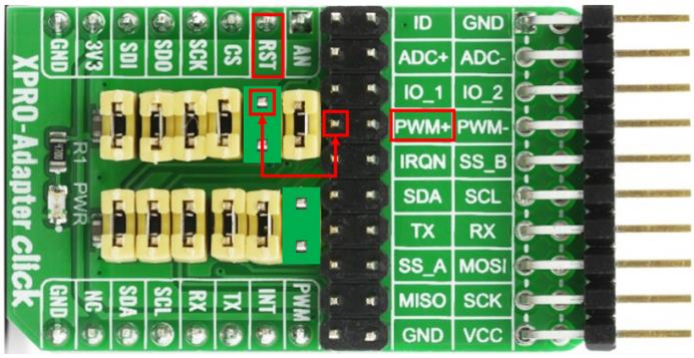
mikroBUS Socket (J402)

Pin Number	Pin Name
9	GND
13	TX
14	RX

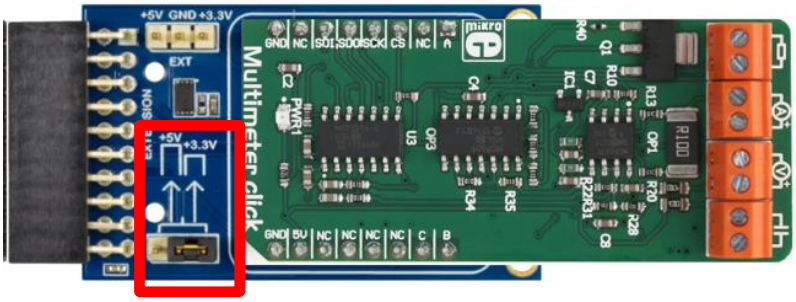
Step 1e: On the XPRO-Adapter click board, **remove** the mechanical jumpers next to **RST** & **PWM**



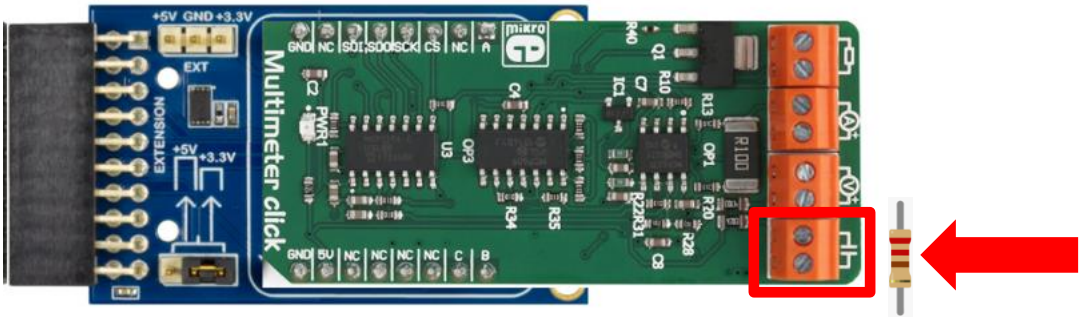
Step 1f: Use a jumper wire to connect the pin *closest* to **RST** directly to the nearby **PWM+** pin



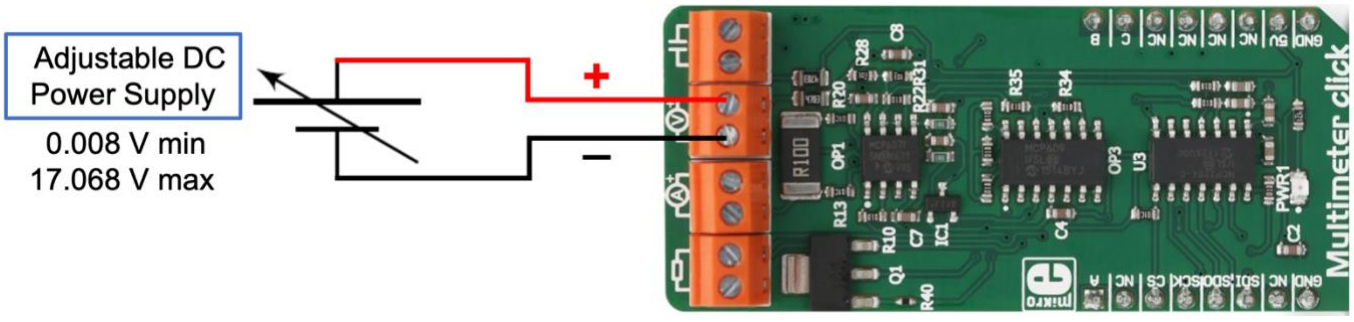
Step 1g: Enable the +5V jumper setting on the mikroBUS Xplained Pro board (diagram shows the default jumper setting for +3.3V)



Step 1h: Connect a resistor (any value in the range from 0.25 to 20M Ω) to the resistance sensing terminal block of the Multimeter click (if no resistor is installed, the sensor reading will simply be communicated as zero)

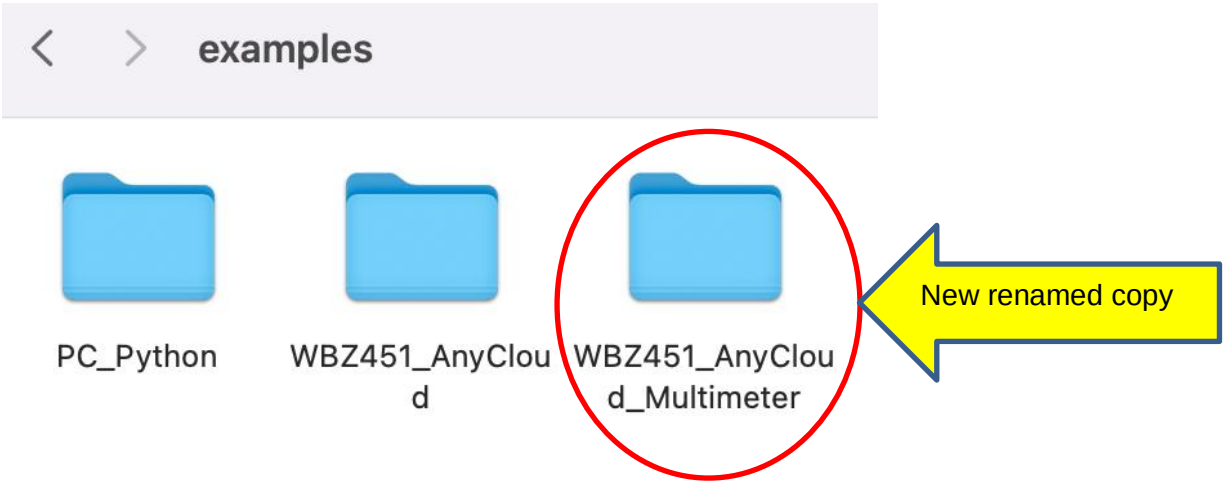


Step 1i: Connect the positive and negative wires of an adjustable DC power supply to the Multimeter click's voltage measurement terminal block but more importantly, do **not** enable the power supply just yet…



STEP 2: Create a Working Copy of the WBZ451_AnyCloud OOB Demo Project

Using a File Explorer (or equivalent) window, navigate to the **\examples** folder (in your clone\download of the **AzureDemo_AnyCloud** repository). Make a copy of the **WBZ451_AnyCloud** folder and rename it to **WBZ451_AnyCloud_Multimeter**



STEP 3: Launch the MPLAB X IDE

Step 3a: If not already open, launch the MPLAB X IDE

Step 3b: Close any projects that are currently opened by selecting [File → Close All Projects] using the MPLAB X main toolbar

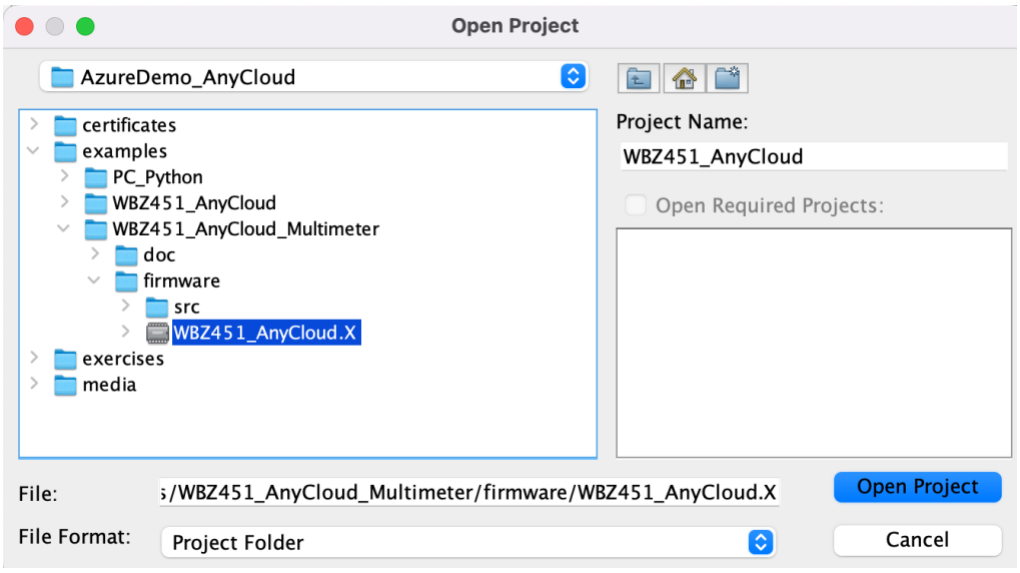
STEP 4: Open the Newly Created Project

Step 4a: From the MPLAB X main toolbar, select [File → Open Project]

Step 4b: Navigate to the **AzureDemo_AnyCloud\examples\WBZ451_AnyCloud_Multimeter\firmware** folder

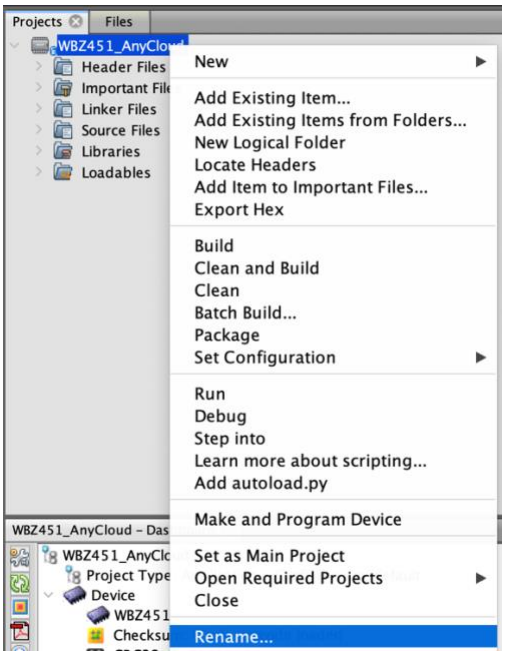
Step 4c: Select (click on) the **WBZ451_AnyCloud.X** project folder

Step 4d: Click on the **Open Project** button

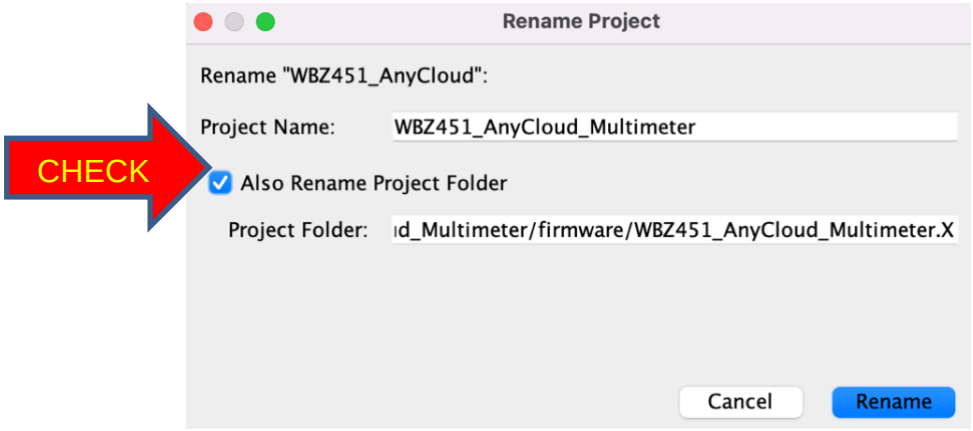


STEP 5: Rename the Project to *WBZ451_AnyCloud_Multimeter*

Step 5a: In the **Projects** view, right-click on the **WBZ451_AnyCloud** project and select **Rename**



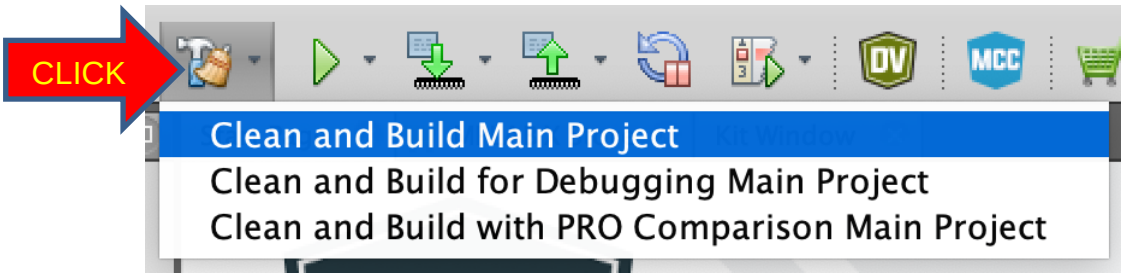
Step 5b: Check the box "**Also Rename Project Folder"** and then type **WBZ451_AnyCloud_Multimeter** for the **Project Name**



Step 5c: Click the **Rename** button

STEP 6: Clean and Build the Project

Step 6a: Click on the **Clean and Build Main Project** icon in the MPLAB X main toolbar



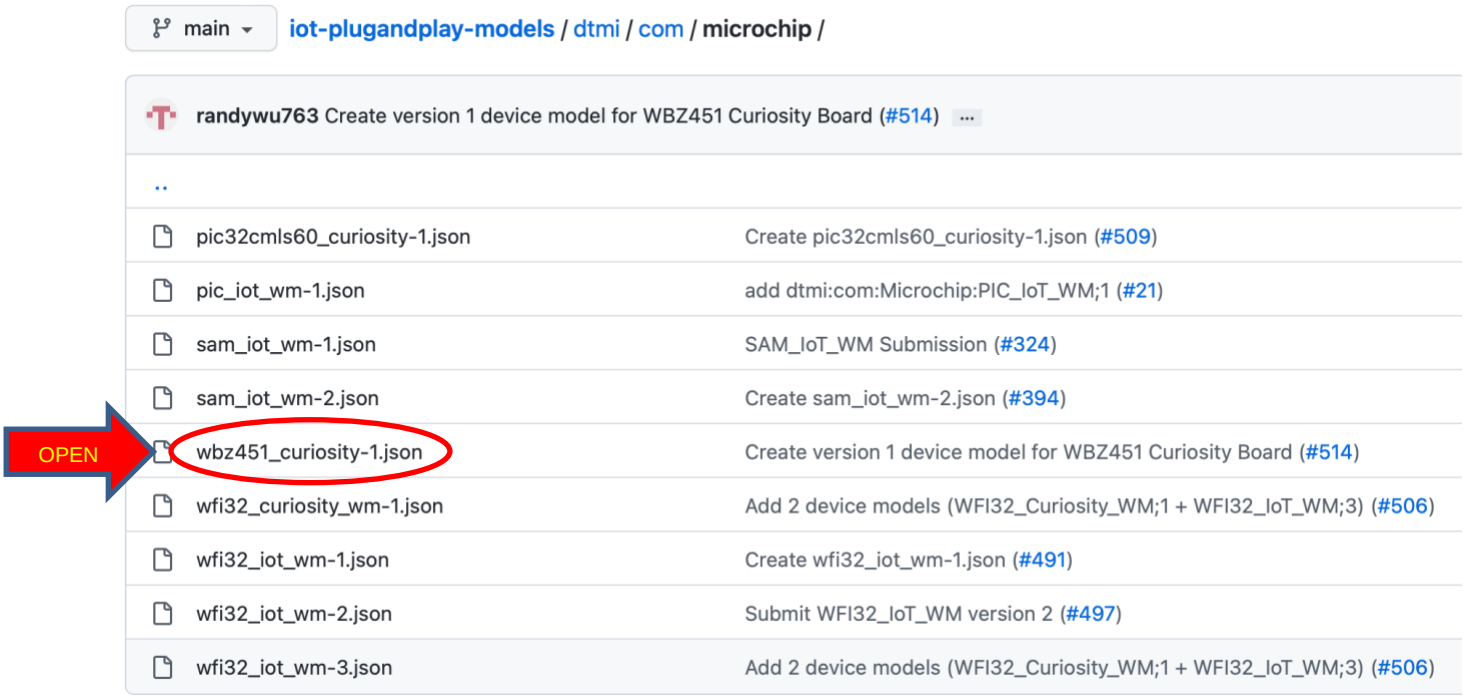
Step 6b: Verify that the build is successful before proceeding; otherwise review the previous steps to make sure each one has been implemented correctly. Do not proceed until the project builds successfully. If you need to refer to a working copy of the project, the solution can be found in the **\exercises\solutions\WBZ451_AnyCloud_Multimeter** folder

STEP 7: Update the name of the Digital Twins Model Interface (DTMI)

By adding new telemetry to the existing example, a new device model interface needs to be created to incorporate the extra sensor data that's being communicated from Device to Cloud.

Step 7a: The existing WBZ451 AnyCloud OOB example emulates the DTMI named **dtmi:com:Microchip:WBZ451_Curiosity;1** which is used as the device model for a stand-alone WBZ451 Curiosity Board. As is the case for all of Microchip's publicly-released DTMI's, this particular one is published in Microsoft's IoT Plug and Play Models Repository and can be accessed by following the absolute directory path **/dtmi/com/microchip**

Step 7b: Access this **microchip** sub-folder in Microsoft's repository and download the JSON file corresponding to the WBZ451 Curiosity Board ("**wbz451_curiosity-1.json"**). Save this file to your local drive as it will be used in the next lab. You may need to just open the file from within the web browser and then copy and paste the contents of the file into a newly-created file.



Step 7c: Determine the name of the new DTMI to be created. For the purposes of this exercise, let's just extend the model ID name to identify the addition of the Multimeter click:

dtmi:com:Microchip:WBZ451_Curiosity_Multimeter;1 (where '1' denotes version 1)

Step 7d: Open (double-click) the **app_rio2_config.h** header file located under

[WBZ451_AnyCloud_Multimeter → Source Files]

Step 7e: Confirm that the `USE_AZURE` definition is enabled as the MQTT Cloud service

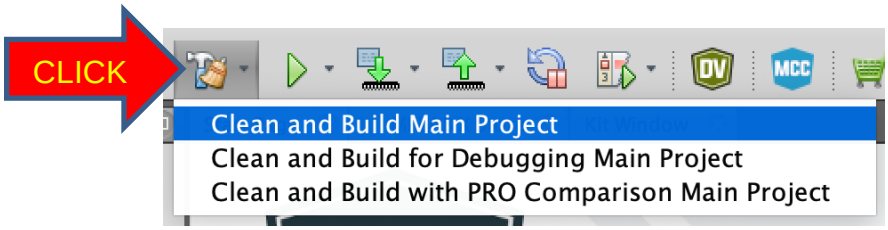
```
53  //// ONLY define ONE MQTT Cloud service
54
55  //#define USE_MOSQUITTO
56  //#define USE_AWS
57  #define USE_AZURE
```

Step 7f: Enable the definition `MULTIMETER_CLICK` for including Multimeter click support and operations. Notice that a different Digital Twins Model Interface (DTMI) is selected to represent the new Model ID corresponding to the addition of the Multimeter click

```
136 #define MULTIMETER_CLICK // Comment out if no Multimeter click is installed
137 #ifdef MULTIMETER_CLICK
138     #define MODEL_ID "dtmi:com:Microchip:WBZ451_Curiosity_Multimeter;1"
139 #else
140     #define MODEL_ID "dtmi:com:Microchip:WBZ451_Curiosity;1"
141 #endif /* MULTIMETER_CLICK */
```

STEP 8: Clean and Build the Project

Step 8a: Click on the **Clean and Build Main Project** icon in the MPLAB X main toolbar



Step 8b: Verify that the build is successful before proceeding; otherwise review the previous steps (after the last successful build) to make sure each one has been implemented correctly. You will not be able to proceed with the remainder of the lab exercises until the project builds successfully with your code modifications.

Results:

You have made the necessary changes to the embedded project required to send the Multimeter click data to IoT Central. The following 2 tasks were accomplished:

1. Updated the model ID in the embedded project. This allows us to create a new device template in Azure, which will define our updated interfaces (adding the new sensors)
2. Enabled the application code to send telemetry messages on behalf of the Multimeter click board's sensors

Summary:

At this point, the device project should be able to successfully connect to Azure IoT Central using the newly-updated device model interface. Now it's time to switch over to the Cloud side and modify the IoT Central application to display the additional telemetry being reported by the Multimeter click.

Lab 3 – Modify the IoT Central Application

Purpose:

Learn how to add a device template to an IoT Central application and create a dashboard to visualize telemetry.

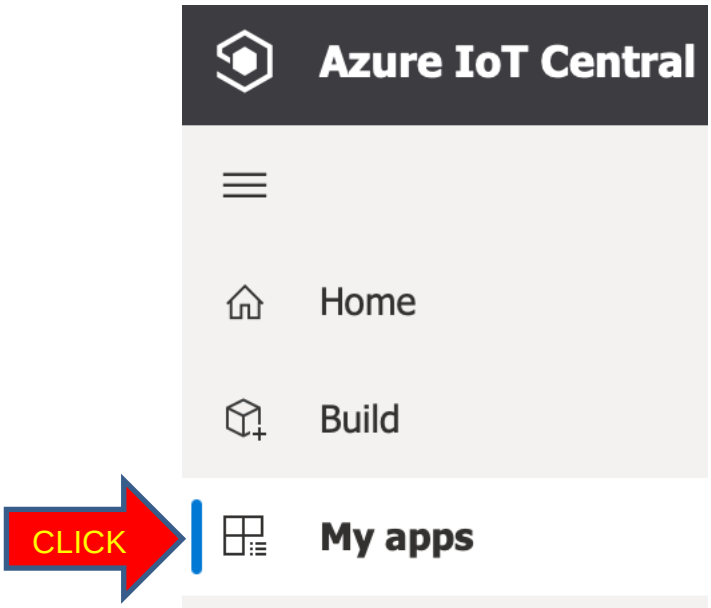
Overview:

In this exercise we will create a new device template in the existing IoT Central application so that all devices that have the updated "IoT Plug and Play" model interface will be able to visually present its data on a dashboard. Upon successful completion of the lab exercise, you should see the Multimeter click board's sensor values get updated periodically in a Last Known Value (LKV) tile that has been added to a new dashboard in the IoT Central application.

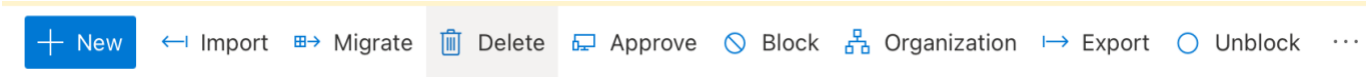
Procedure:

STEP 1: Access the existing IoT Central Demo Application

Step 1a: Sign into the IoT Central Portal and click on **My apps**



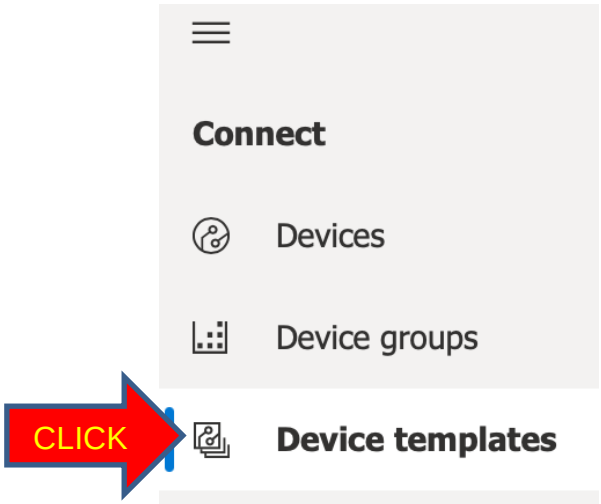
Step 1b: Click on the name of the demo application which was created/used back in Lab 1



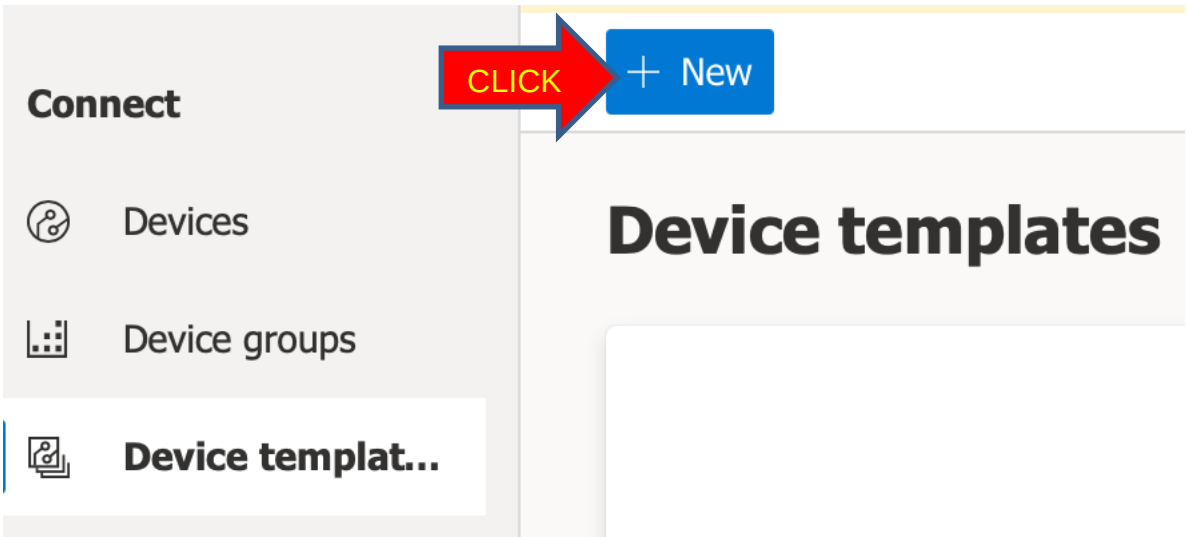
STEP 2: Create the Device Template for the New Multimeter Click DTMI

In this step, we will simply modify the existing "WBZ451_Curiosity;1" device template that was being used for the OOB AnyCloud example and then rename it for the new model ID

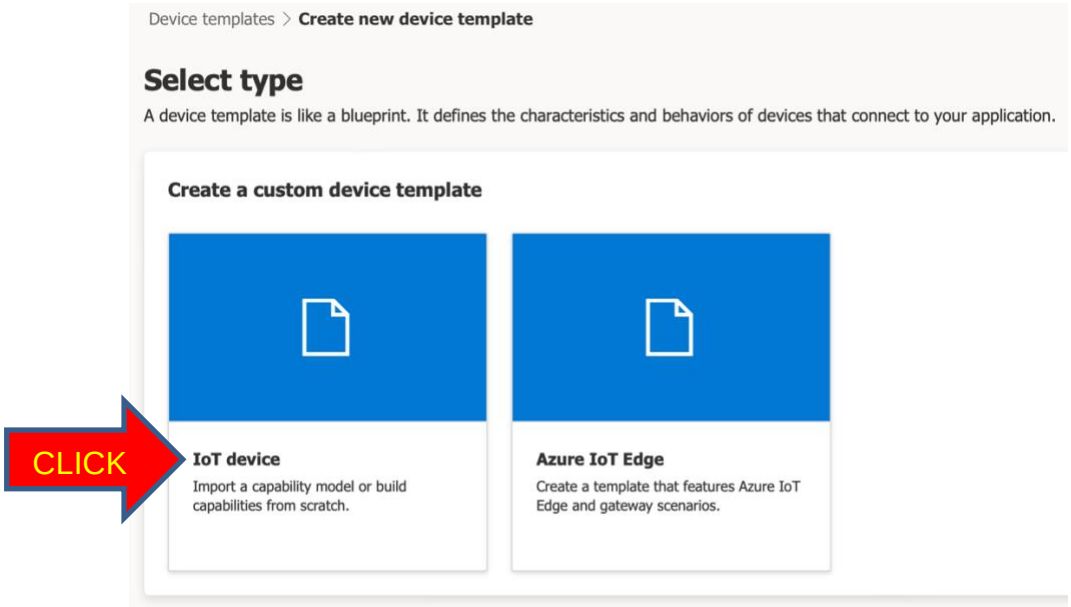
Step 2a: Using the navigation pane on the left-hand side, select [**Connect → Device templates**]



Step 2b: Click on the [+ New] button

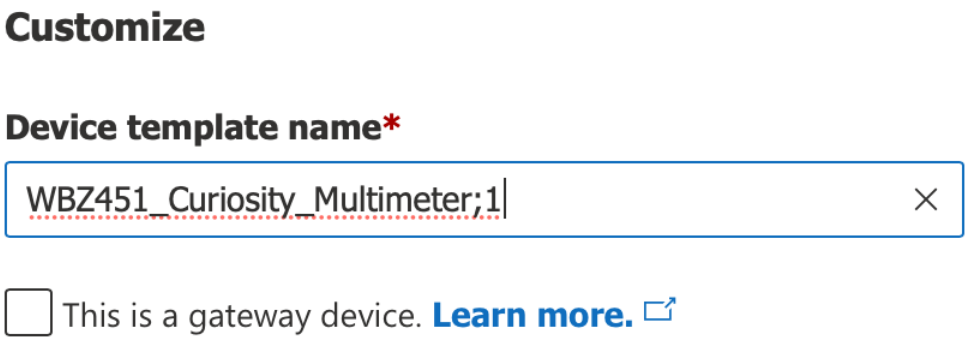


Step 2c: Click the **IoT device** tile



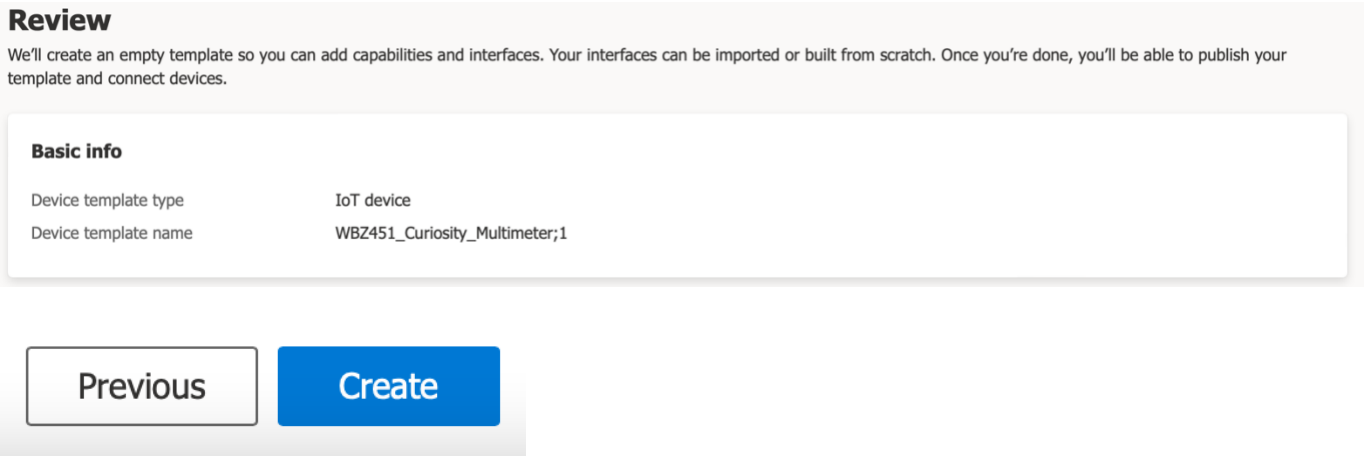
Step 2d: Click on the **Next: Customize** button

Step 2e: Set “Device template name” to **WBZ451_Curiosity_Multimeter;1**



Step 2f: Click the **Next: Review** button

Step 2g: Click the **Create** button



Step 2h: Create a copy of the **wbz451_curiosity-1.json** file (which should have been downloaded earlier from the IoT Plug and Play Models Repository) and rename it to **wbz451_curiosity_multimeter-1.json**. Open the file using the text editor of your choice and make the following edits:

- Change the model ID designation from dtmi:com:Microchip:WBZ451_Curiosity;1 to dtmi:com:Microchip:WBZ451_Curiosity_Multimeter;1

```
1   {
2       "@id": "dtmi:com:Microchip:WBZ451_Curiosity_Multimeter;1",
3       "@type": "Interface",
```

- Change the “description” & “displayName” to accurately reflect the hardware configuration

```
266     "description": {
267         "en": "WBZ451 Curiosity Board & Multimeter Click"
268     },
269     "displayName": {
270         "en": "WBZ451 Curiosity Board & Multimeter Click"
271     },
```

- Copy the lines shown below and paste the entire block of code just below the line beginning with “contents” (line #4 towards the top of the JSON file). This expands the existing device model to include 4 additional telemetry values reported from the Multimeter click board:
  1. voltage (mV)
  2. current (mA)
  3. capacitance (nF)
  4. resistance (Ω)

COPY BEGIN

```
{
    "@type": [
        "Telemetry",
        "Voltage",
        "NumberValue"
    ],
    "description": {
        "en": "Voltage reading in mV from the Multimeter click"
    },
    "displayName": {
        "en": "MULTIMETER_voltage"
    },
    "name": "MULTIMETER_voltage",
    "schema": "double",
    "unit": "millivolt"
},
{
    "@type": [
        "Telemetry",
        "Current",
        "NumberValue"
    ],
    "description": {
        "en": "Current reading in mA from the Multimeter click"
    },
    "displayName": {
        "en": "MULTIMETER_current"
    },
    "name": "MULTIMETER_current",
    "schema": "double",
    "unit": "milliampere"
},
{
    "@type": [
```

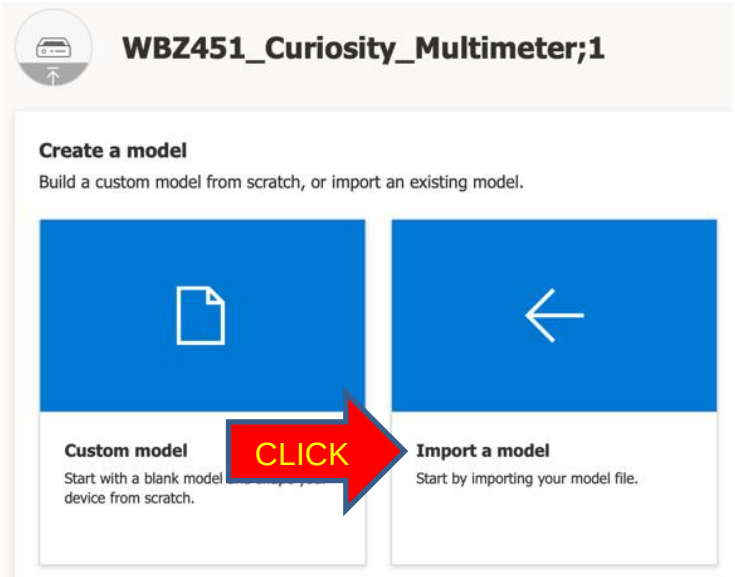
```
            "Telemetry",
            "Capacitance",
            "NumberValue"
        ],
        "description": {
            "en": "Capacitance reading in nF from the Multimeter click"
        },
        "displayName": {
            "en": "MULTIMETER_capacitance"
        },
        "name": "MULTIMETER_capacitance",
        "schema": "double",
        "unit": "nanofarad"
    },
    {
        "@type": [
            "Telemetry",
            "Resistance",
            "NumberValue"
        ],
        "description": {
            "en": "Resistance reading in ohms from the Multimeter click"
        },
        "displayName": {
            "en": "MULTIMETER_resistance"
        },
        "name": "MULTIMETER_resistance",
        "schema": "double",
        "unit": "ohm"
    },
```

COPY END

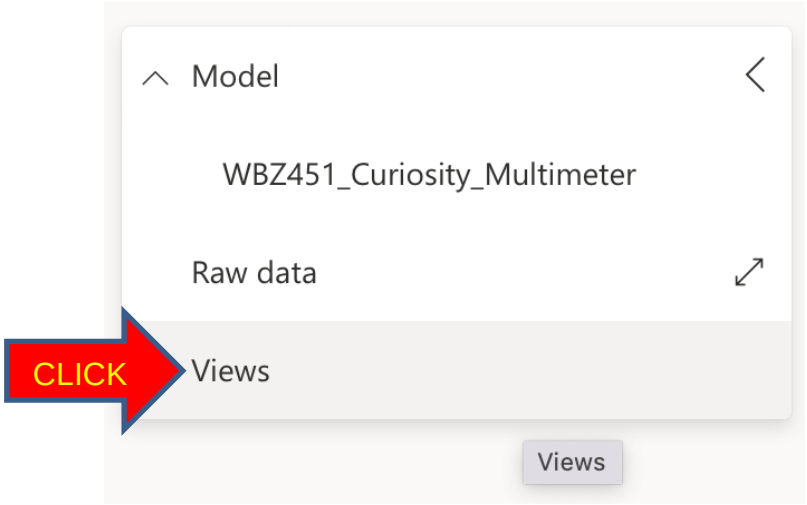
- Confirm that all the lines were copied over correctly into the file. The entire block contains all 4 telemetry definitions covering each of the Multimeter click's measurement sensors

- Save changes to the JSON file – this will be used to import the device model into the application's device template

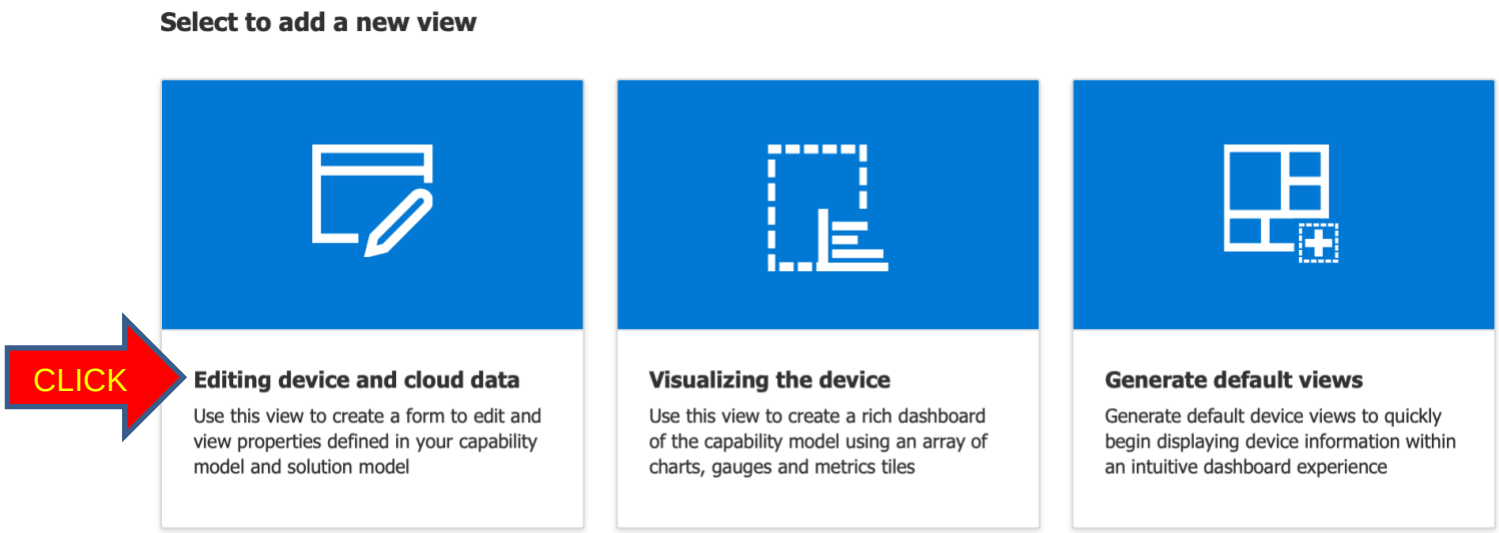
Step 2i: Click the **Import a model** tile



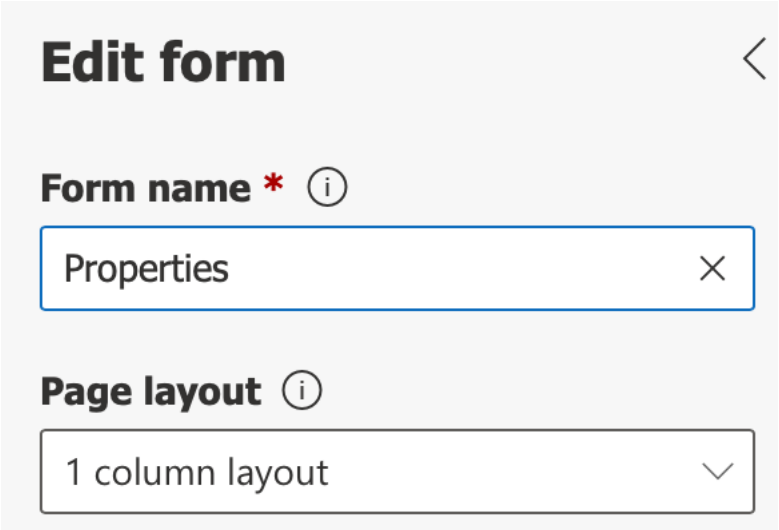
Step 2j: Click on the **Views** category underneath the model name



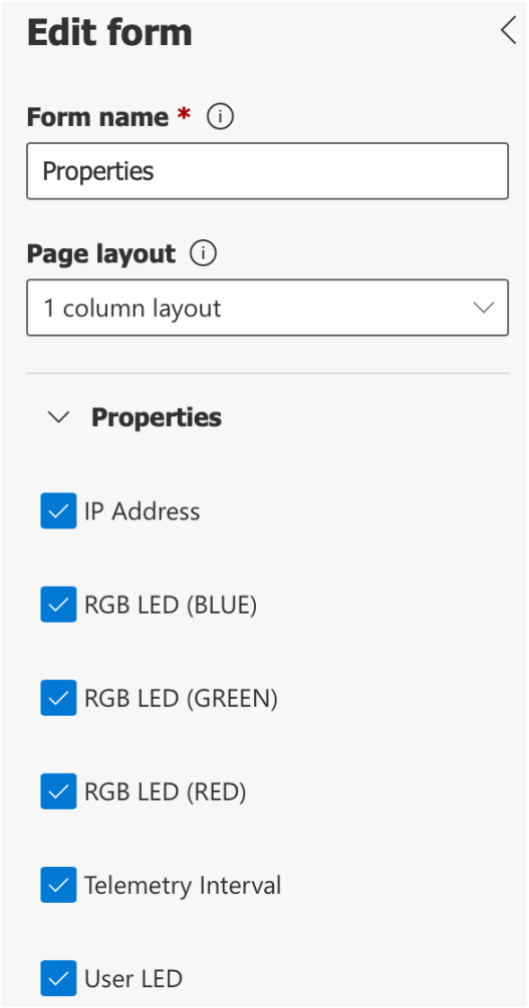
Step 2k: Click on the **Editing device and cloud data** tile



Step 2l: Type in **Properties** for the **Form name**



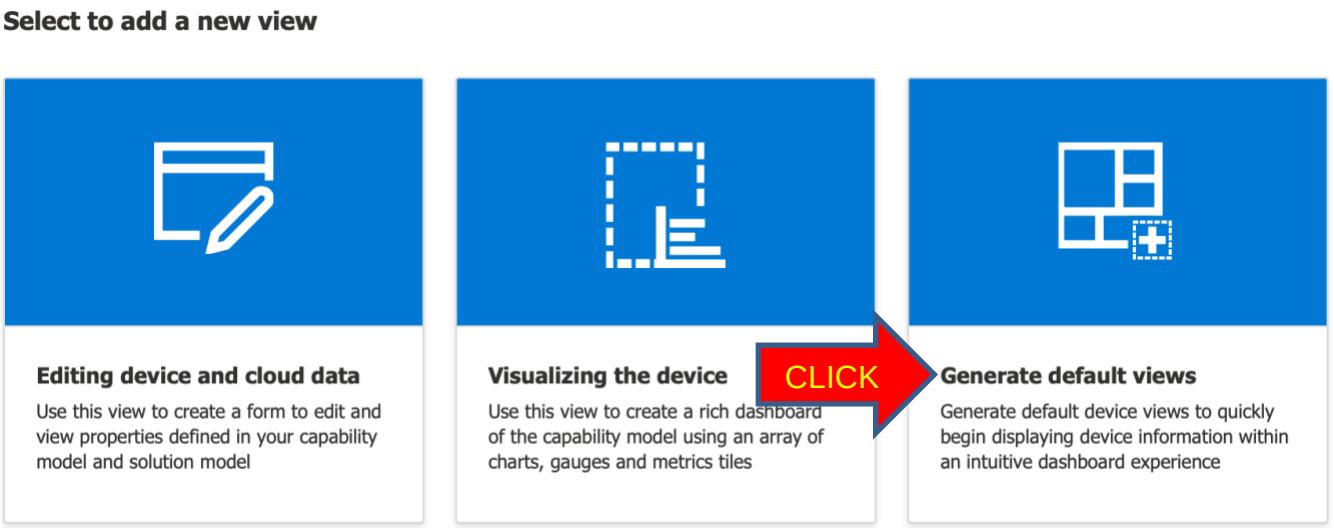
Step 2m: Click on the **Properties** category and check all the boxes



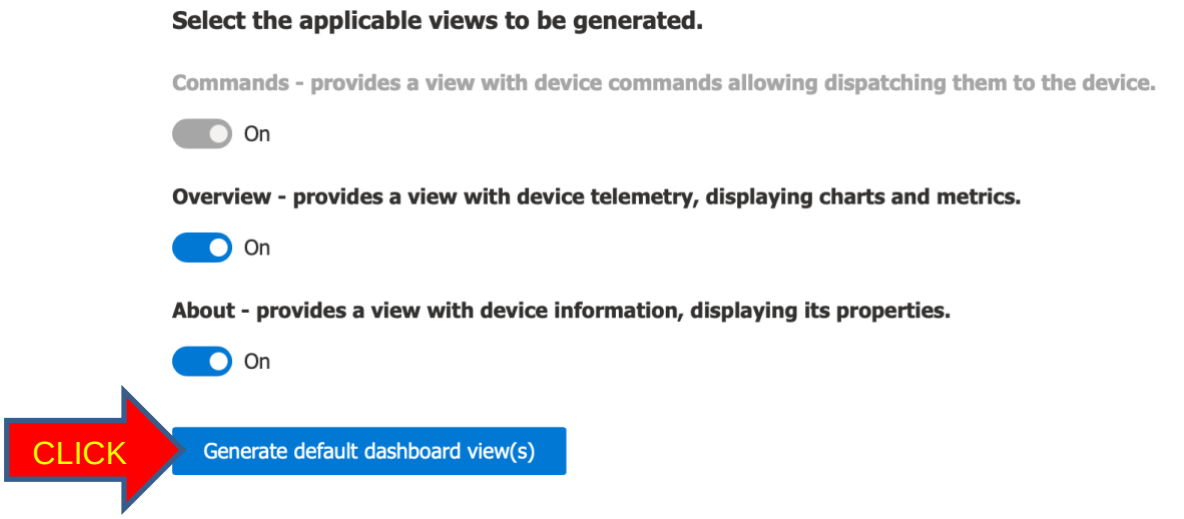
Step 2n: Click on the **Add section** button (bottom of the page), click on the **Save** icon (top of the page), and then click on the **Back** icon

Step 2o: Click on the **Views** category underneath the device model name

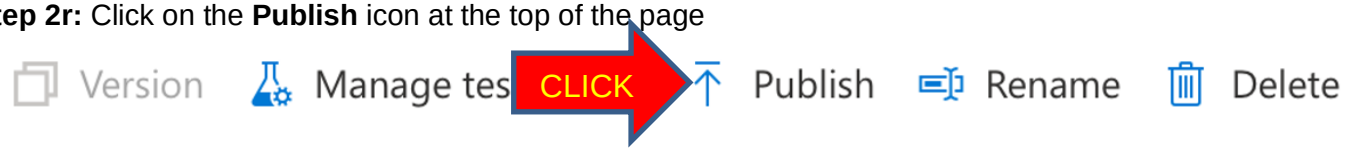
Step 2p: Click on the **Generate default views** tile



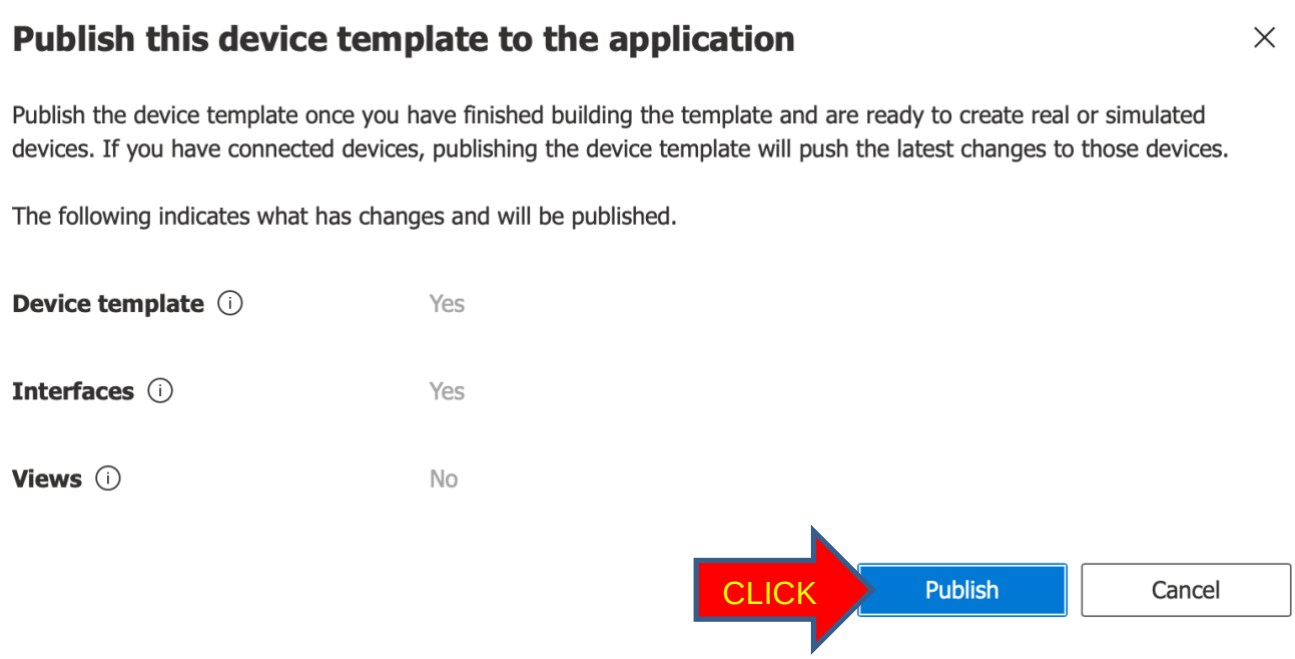
Step 2q: Click on the **Generate default dashboard view(s)** button



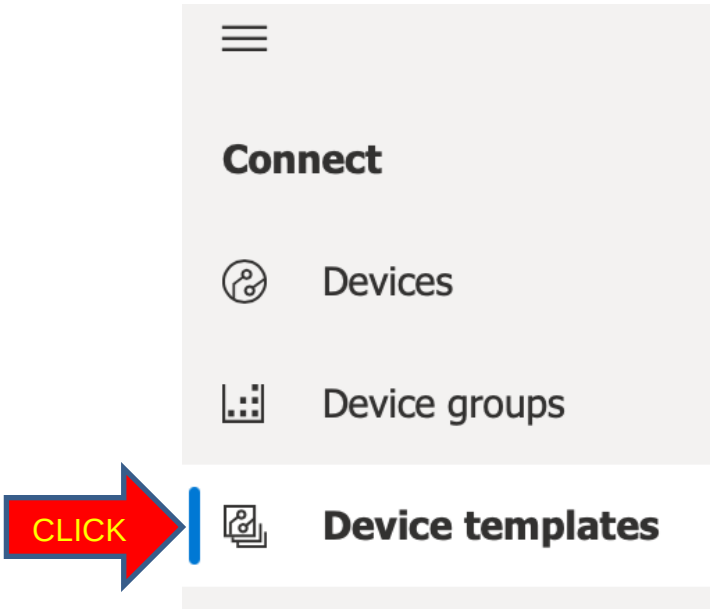
Step 2r: Click on the **Publish** icon at the top of the page



Step 2s: When a pop-up dialogue box appears, click on the **Publish** button



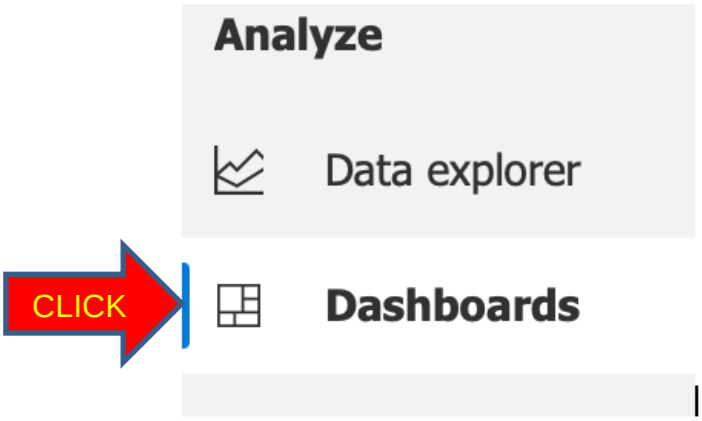
Step 2t: Using the navigation pane on the left-hand side, select [**Connect → Device templates**]



Step 2u: Confirm the new device template shows up in the **Device templates** list

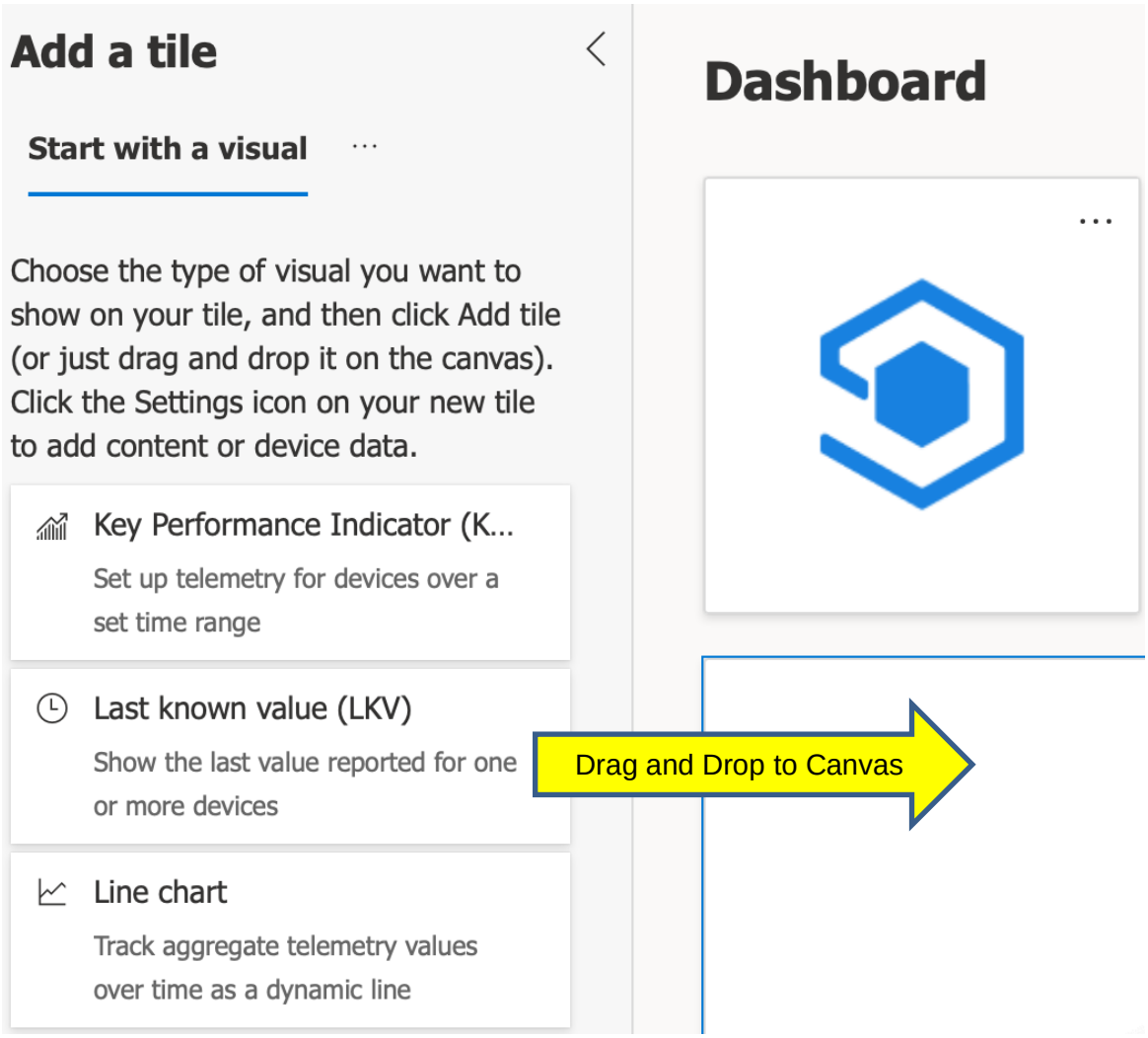
STEP 3: Create a New Dashboard to Display the Multimeter Click Sensor Values

Step 3a: Using the navigation pane on the far left-hand side of the page, click on [**Analyze → Dashboards**]

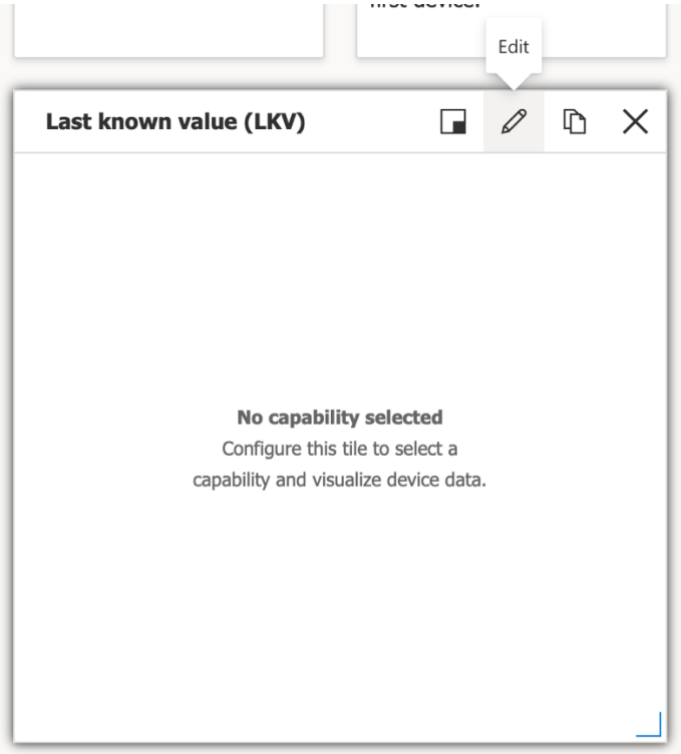


Step 3b: Click on the **Edit** icon

Step 3c: Drag the **Last known value (LKV)** tile onto the canvas



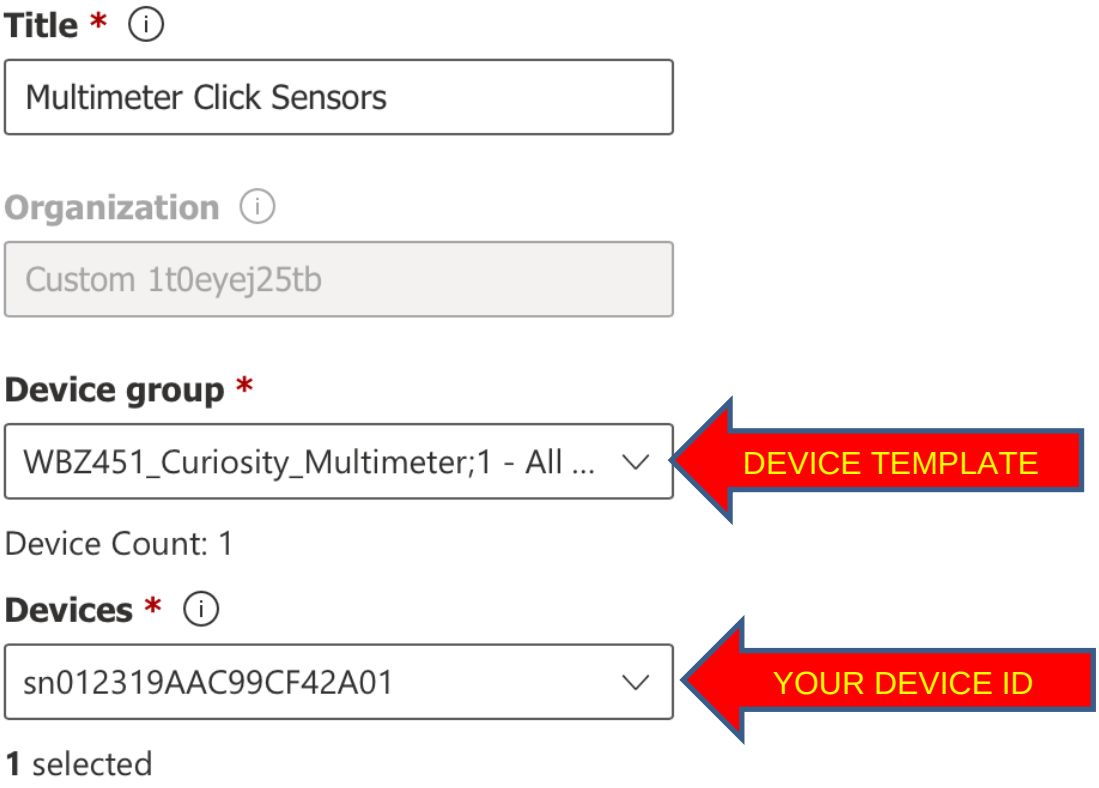
Step 3d: Click the **Edit** icon in the upper right-hand corner of the newly-created LKV tile



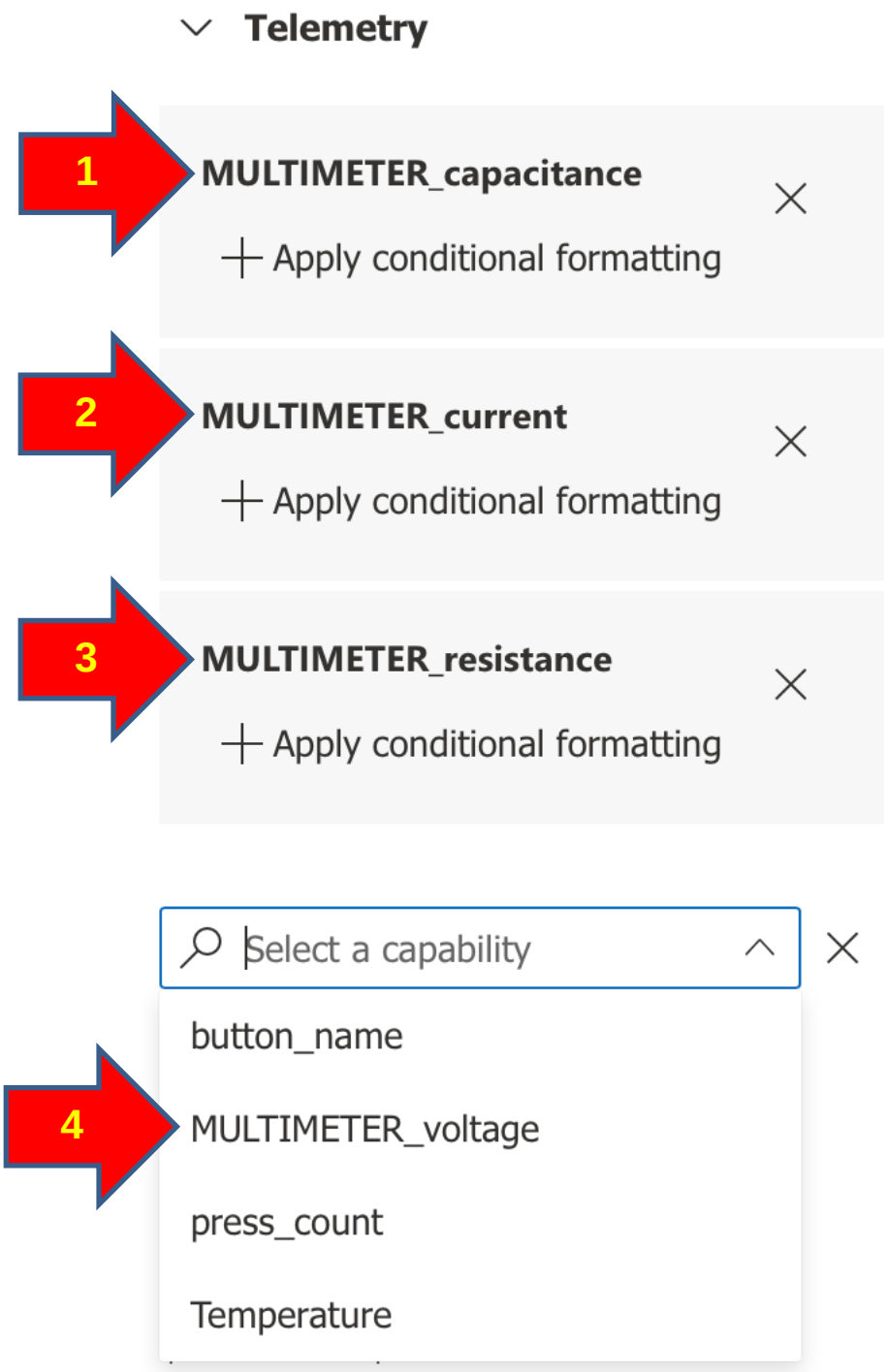
Step 3e: Change the Title of the tile to something meaningful (e.g. **Multimeter Click Sensors**)

Step 3f: Select **WBZ451_Curiosity_Multimeter;1** for **Device group**

Step 3g: Select your device name under **Devices**

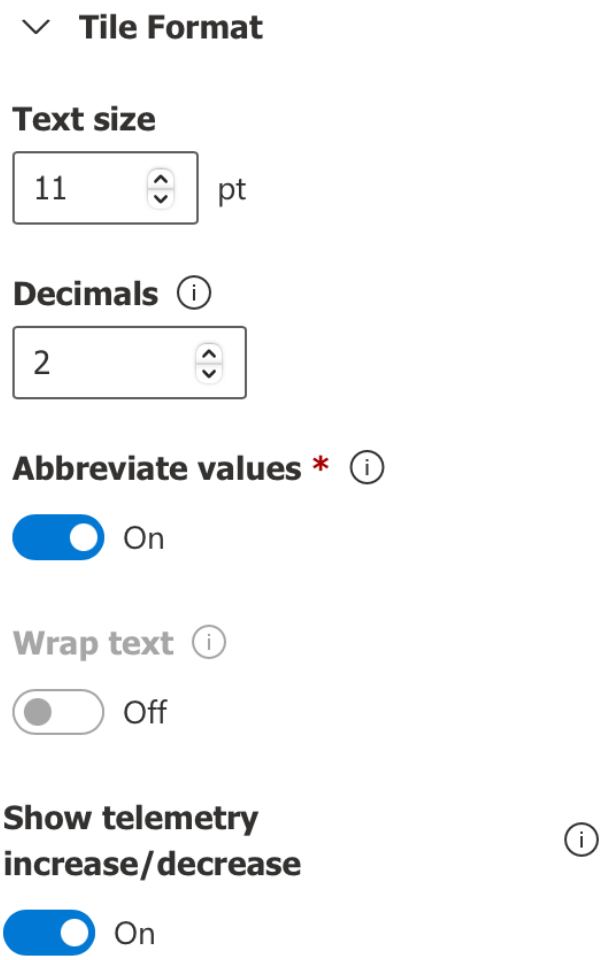


Step 3h: Under the **Telemetry** category, click on **+Capability** to add each of the 4 MULTIMETER telemetry values (do each one at a time)



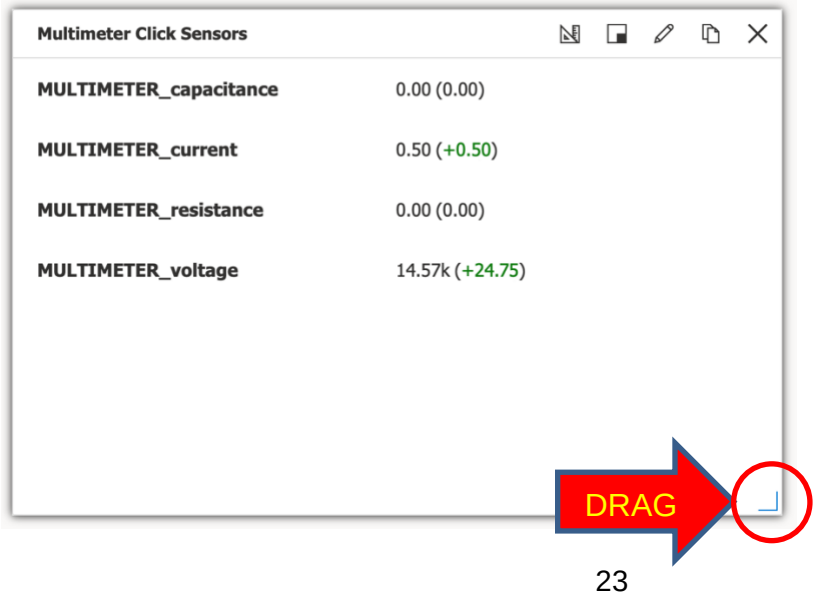
Step 3i: Under the **Tile Format** category, set the following options to these suggested values:

- **Text Size**: 11
- **Decimals**: 2
- **Abbreviate values**: On
- **Show telemetry increase/decrease**: On



Step 3j: Click the **Update** button to finalize the edits to this specific tile

Step 3k: Select and drag the bottom right corner of the tile to adjust the overall size for better readability of all names and values

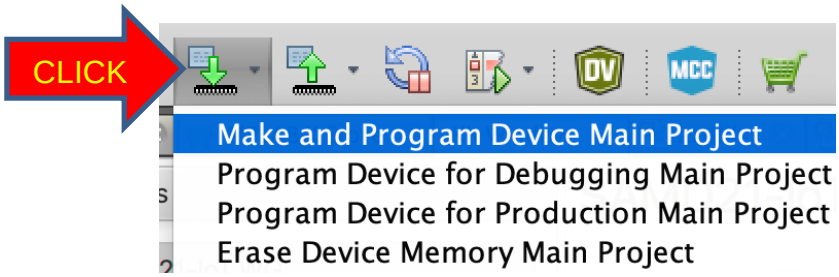


Step 3l: Click the **Save** icon (located towards the top of the page) to finalize edits to the dashboard

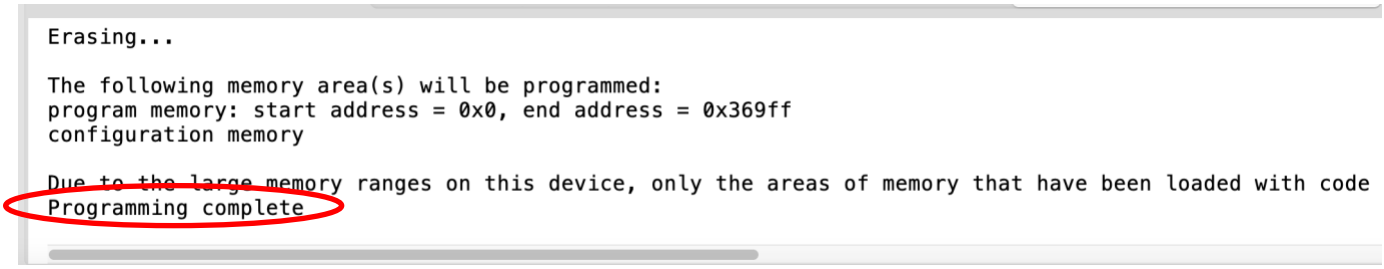
STEP 4: Press the RESET button on the WFI32-IoT AnyCloud™ bridge board

STEP 5: Program the WBZ451 Curiosity Board

Step 5a: Confirm that the WBZ451 Curiosity Board is connected to the PC and is powered ON. Click on the **Make and Program Device Main Project** icon in the MPLAB X main toolbar

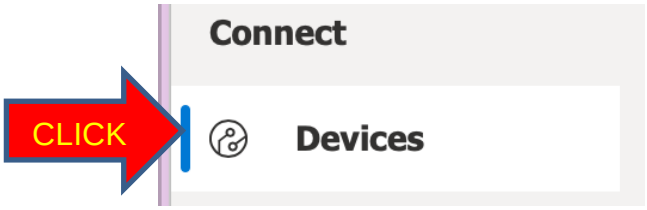


Step 5b: Verify that the programming phase was successful

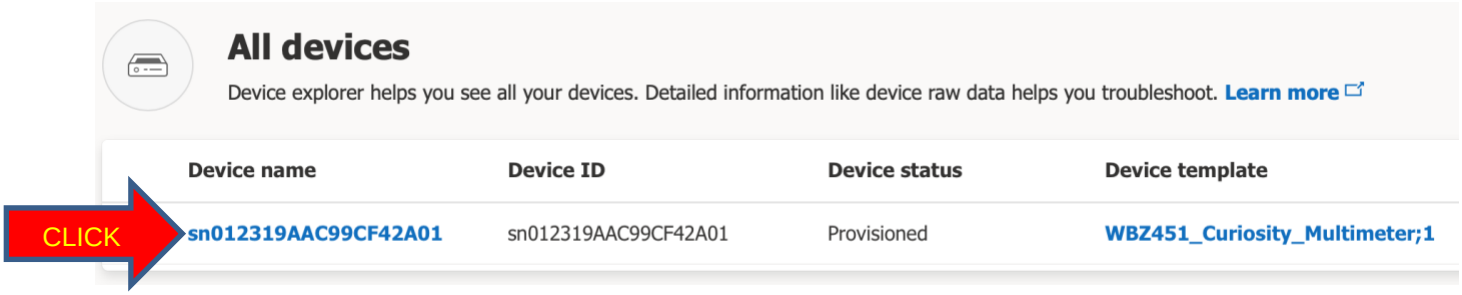


STEP 6: Confirm Device Telemetry Received by the Cloud

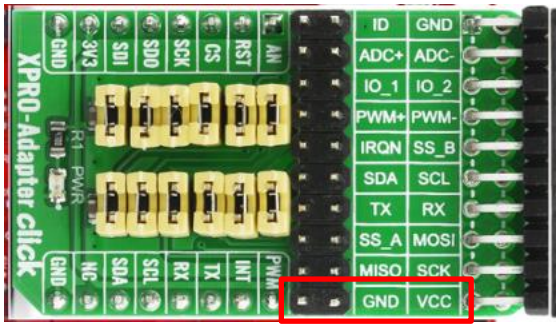
Step 6a: Using the navigation pane on the far left-hand side of the page, click on [**Connect** → **Devices**]



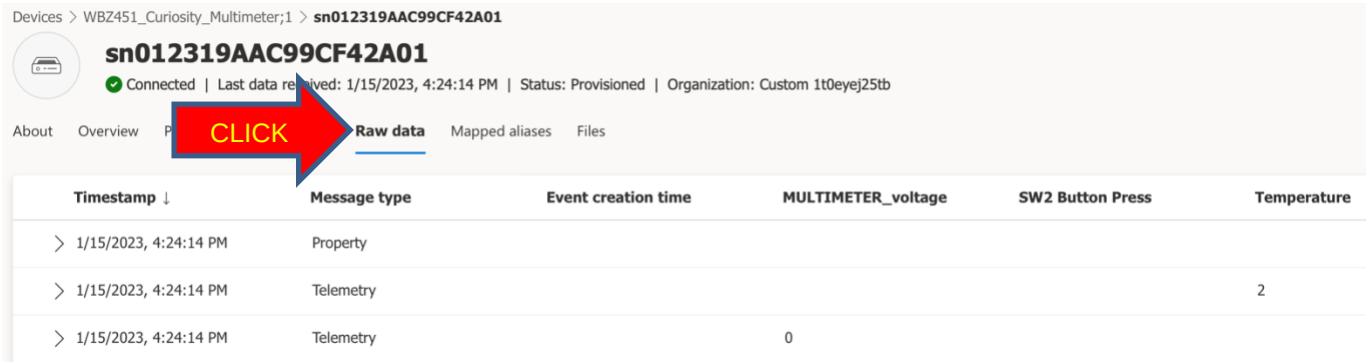
Step 6b: Click on your device name



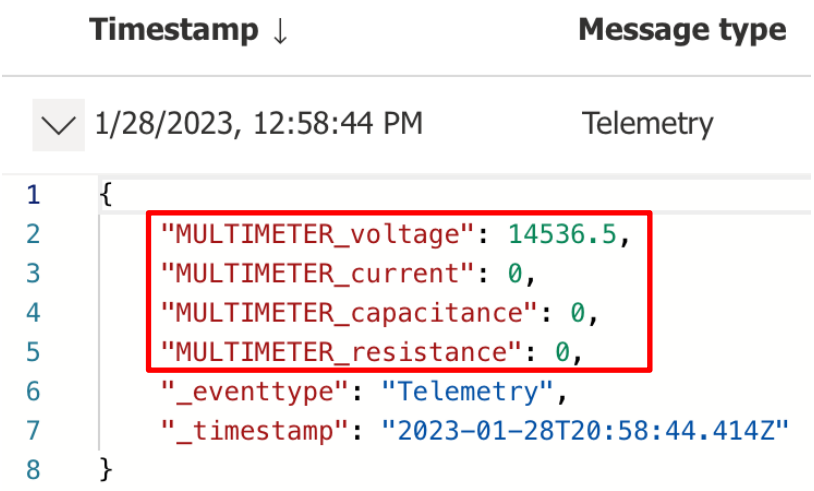
Step 6c: Enable the DC power supply and adjust the voltage to any value which falls within the input range of the Multimeter click voltage sensor (8 mV min to 17.068k mV max). If no DC power supply is available, you can use jumper wires to hook up the voltage sensing terminals to the **VCC** and **GND** pins on the XPRO-Adapter click board



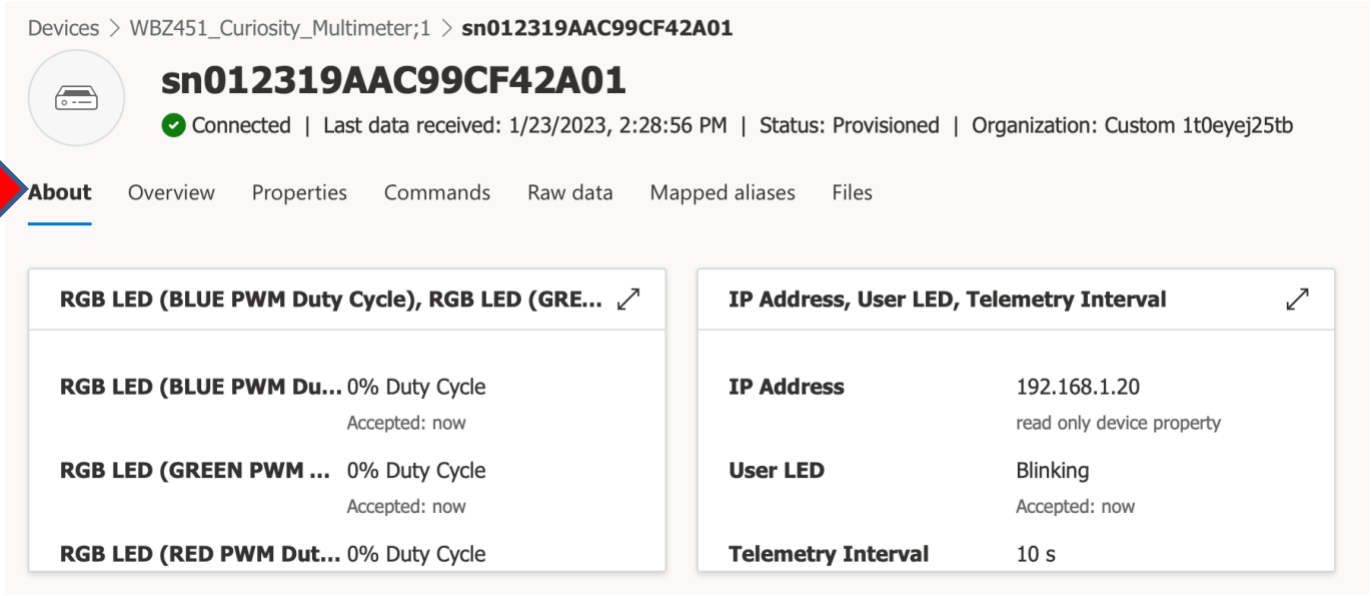
Step 6d: Click on the **Raw data** tab and confirm that property and telemetry messages are currently being received



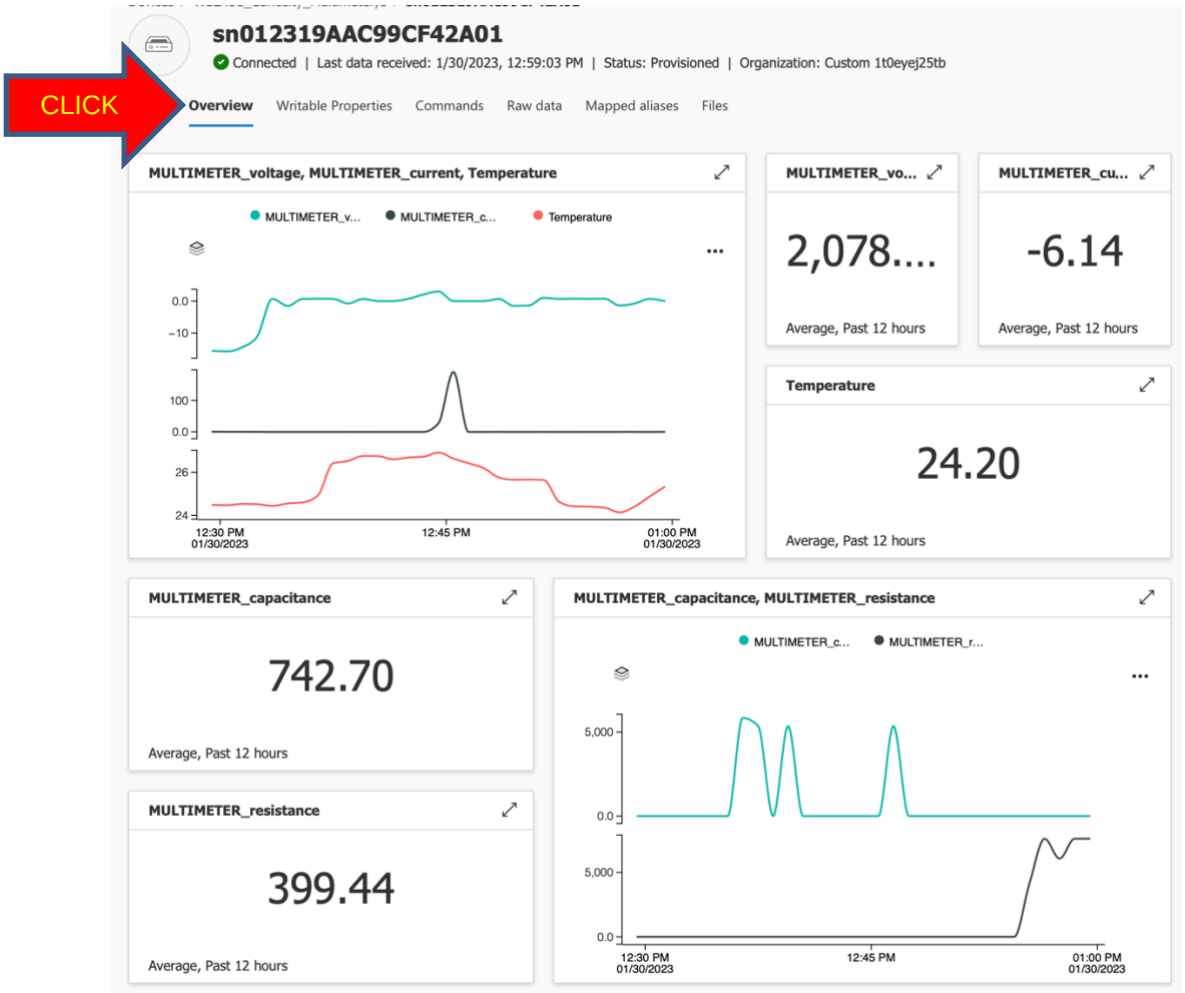
Step 6e: Expand one of the telemetry messages and observe the Multimeter click's voltage, current, capacitance, and resistance values being reported



Step 6f: Click on the **About** tab for a convenient way of viewing of all the device properties at once

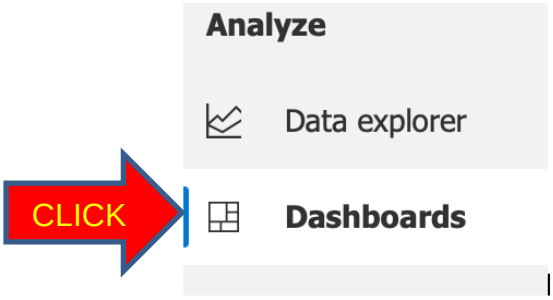


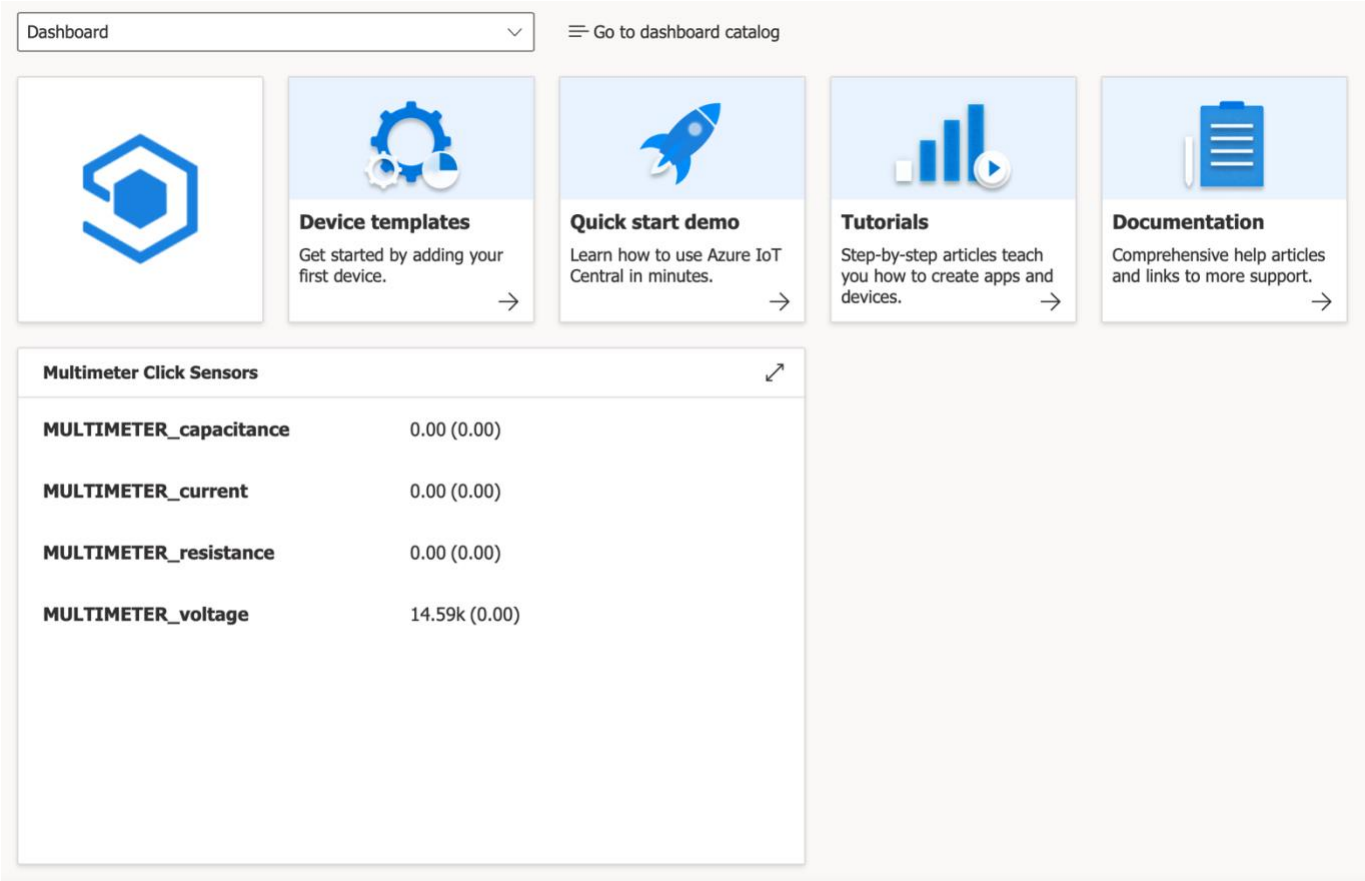
Step 6g: Click on the **Overview** tab for a convenient way of viewing the last known values & averages for all telemetry which has been reported. Use the cursor and hover over any part of each of the line graphs to see the average value for a specific date/time in the history of logged events



STEP 7: Access the Dashboard to Monitor the Multimeter Click Data

Using the navigation pane on the far left-hand side of the page, click on [**Analyze → Dashboards**]





Results:

The last reported values for the Multimeter click's sensors (voltage, current, capacitance, resistance) will be displayed on the Dashboard. The Dashboard can be easily edited to add more tiles to visualize telemetry in various formats (e.g. averages, bar charts, line graphs, heat maps, images, event history, state history, pie charts, key performance indicators, markdowns, external content, labels, maps).

Summary:

In this exercise, we added a new device template to the existing IoT Central application so that the device with the updated Plug and Play model interface was able to report the Multimeter click board telemetry.

Lab 4 – Extend the IoT Central Application

Purpose:

Learn how to leverage some of the extended features offered by IoT Central to help analyze and manage your devices at scale.

Overview:

In this exercise we will extend the IoT Central application beyond the visual dashboard to help us manage the devices that are currently connected to Azure. Some of these extended features include creating analytics queries, running jobs, and settings rules.

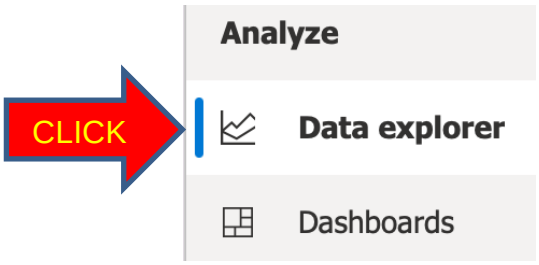
Procedure:

STEP 1: Create an Analytics Query

Azure IoT Central provides rich analytics capabilities to analyze historical trends and correlate telemetry from your devices. A query allows the IoT Central application to interrogate a device (or a group of devices) for various pieces of information. The following are examples of how you can use a query:

- Get the last sequence of telemetry values reported by a device
- Get the last 24 hours of data from devices that are in the same room
- Analyze telemetry trends from devices over a specific time frame (e.g. multiple days)

Step 1a: On the left-hand navigation pane under **Analyze**, select **Data explorer**

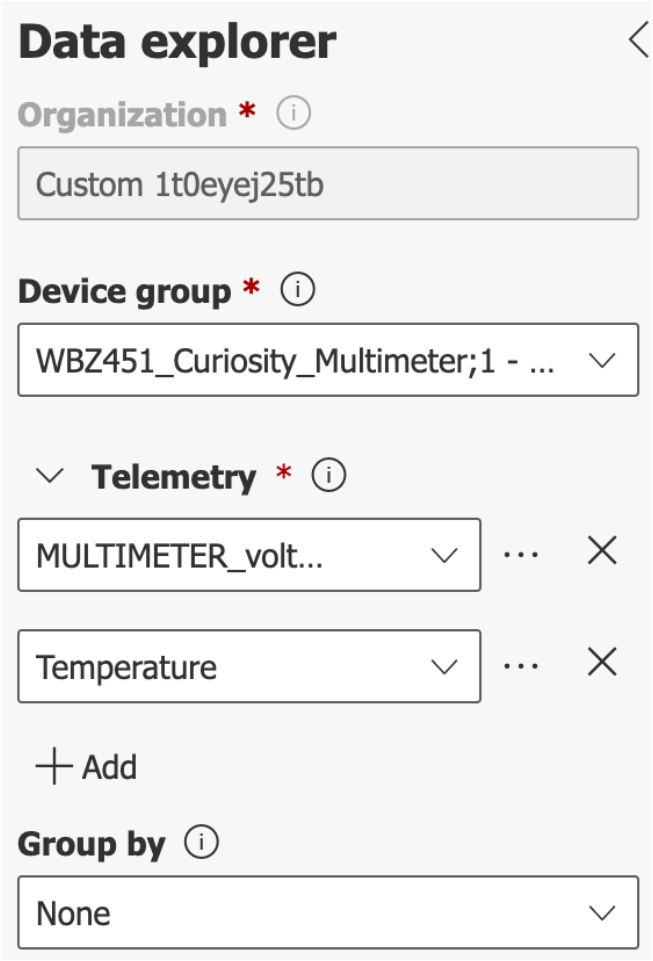


Step 1b: Click on the **+ New query** button

Step 1c: Select the **Device group** (e.g. WBZ451_Curiosity_Multimeter;1)

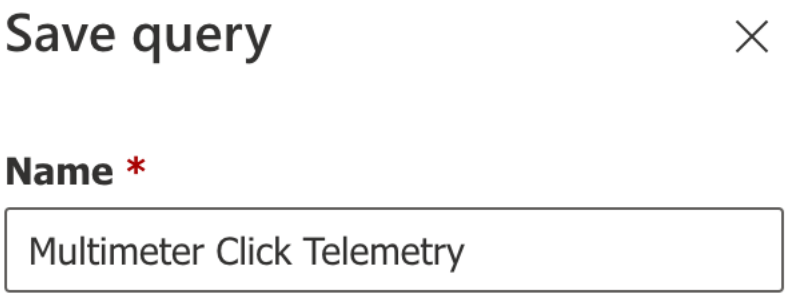
Step 1d: Under **Telemetry**, select **Temperature**

Step 1e: Click **+Add** and select **MULTIMETER_voltage** (repeat to add any of the other Multimeter click sensors)

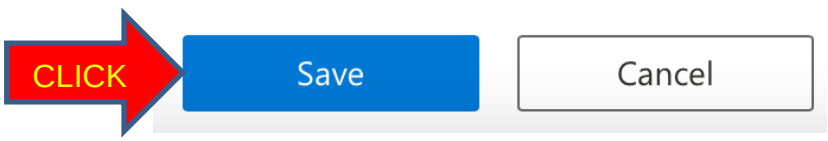


Step 1f: Click on the **Save** icon

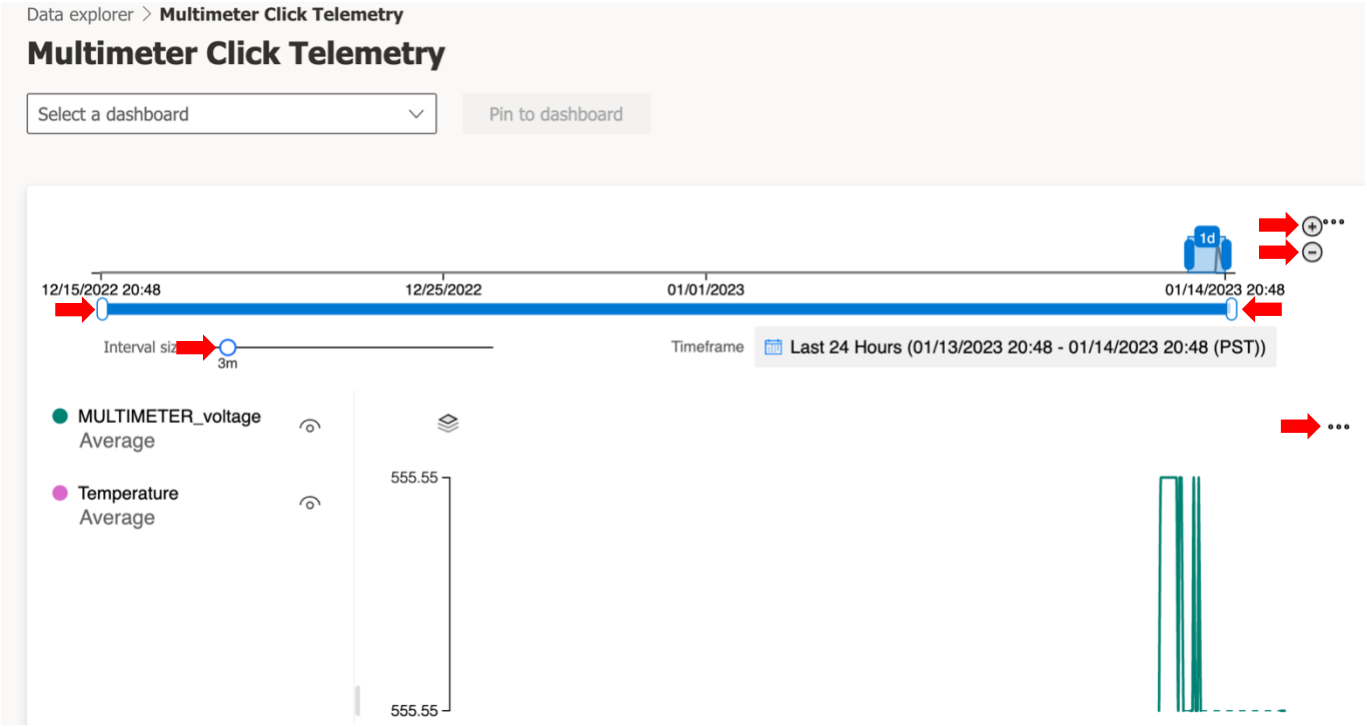
Step 1g: Type in a name for the query (e.g. Multimeter Click Telemetry)



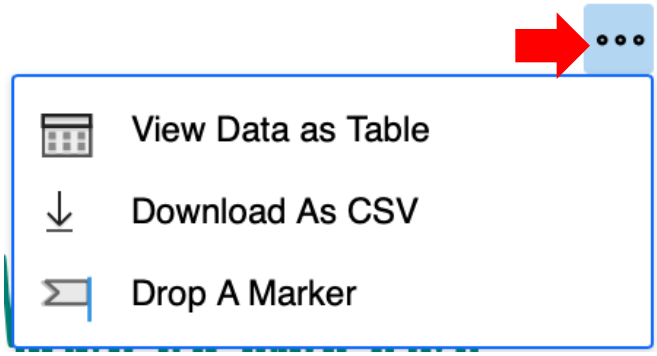
Step 1h: Click on the **Save** button



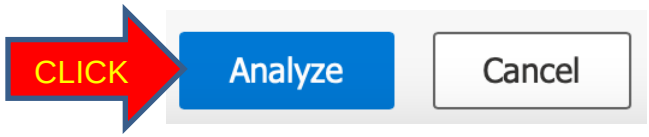
Step 1i: Click on the various sliders and buttons to interact with your data



Step 1j: Click on the ellipsis to bring up additional functions



Step 1k: Click on the **Analyze** button to refresh the data



Step 1l: For additional guidance on analytics queries, consult the Microsoft Azure IoT Central documentation**:** https://docs.microsoft.com/en-us/azure/iot-central/core/howto-create-analytics

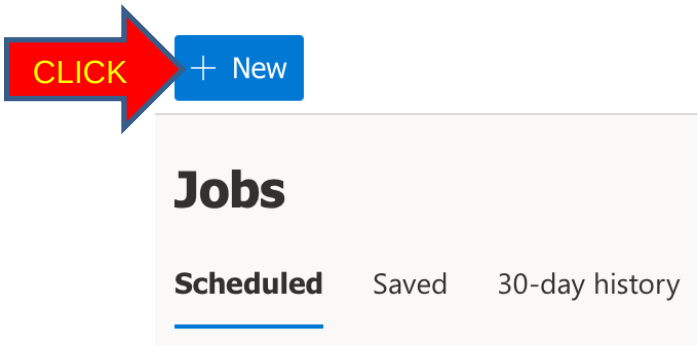
STEP 2: Run a Job

You can use Azure IoT Central to manage your connected devices at scale through jobs. Jobs let you do bulk updates to device and cloud properties as well as run commands supported by the device model. You can also use CSV files to import and export devices in bulk.

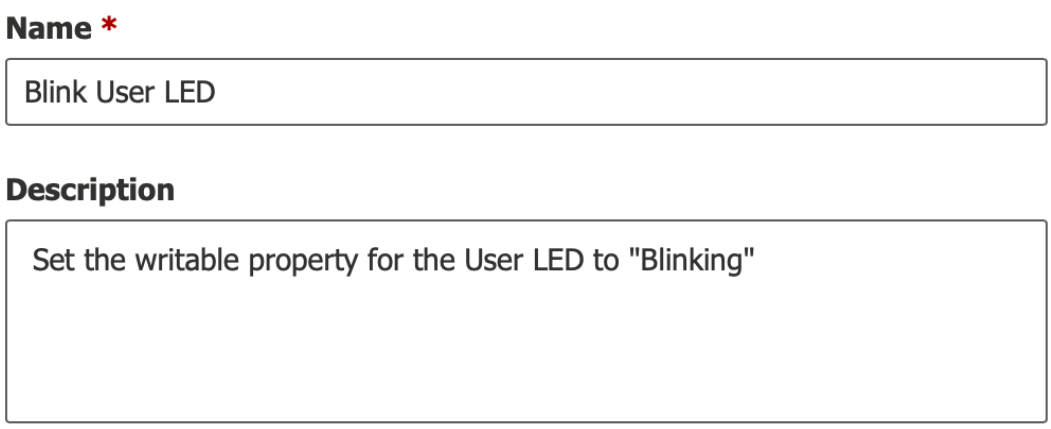
We will create a job and then run the job to set one of the writable properties (e.g. Yellow LED).

Step 2a: On the left-hand navigation pane under **Manage**, select **Jobs**

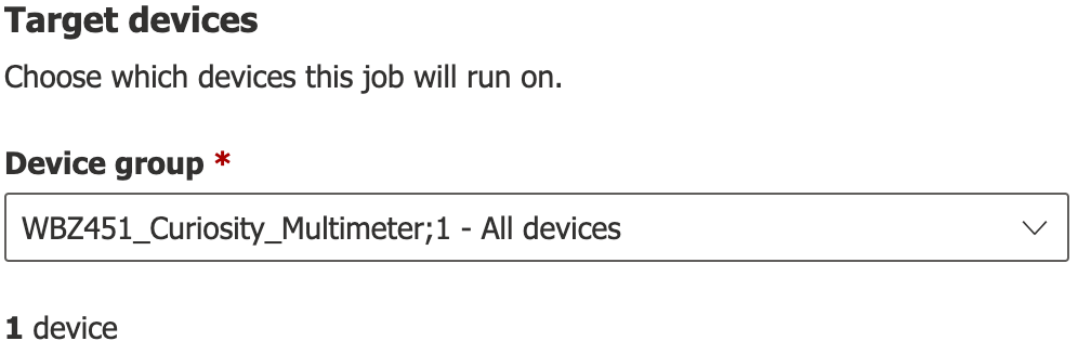
Step 2b: Click the **+ New** button



Step 2c: On the **Configure your job** page, enter a name and description to identify the job you're creating. For this example, let's create a job that puts the User LED into Blinking mode (e.g. Name = "Blink User LED")

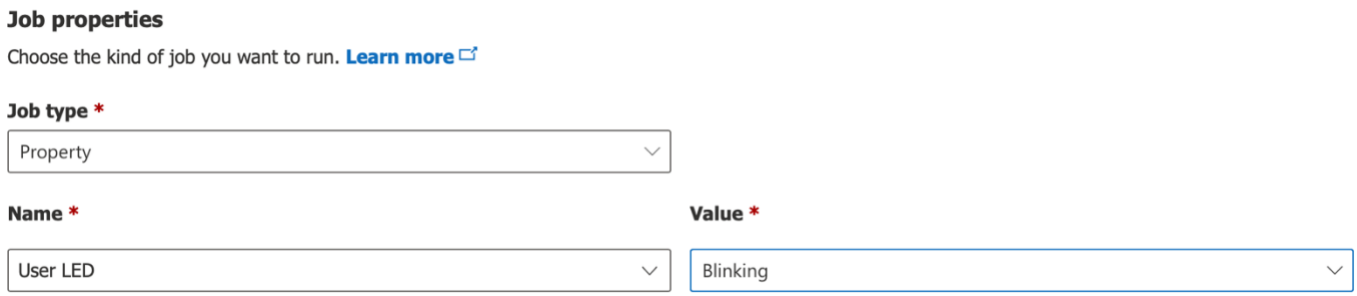


Step 2d: Select the target device group that you want your job to apply to (e.g. WBZ451_Curiosity_Multimeter;1). You can see how many devices your job configuration applies to just below the **Device group** selection



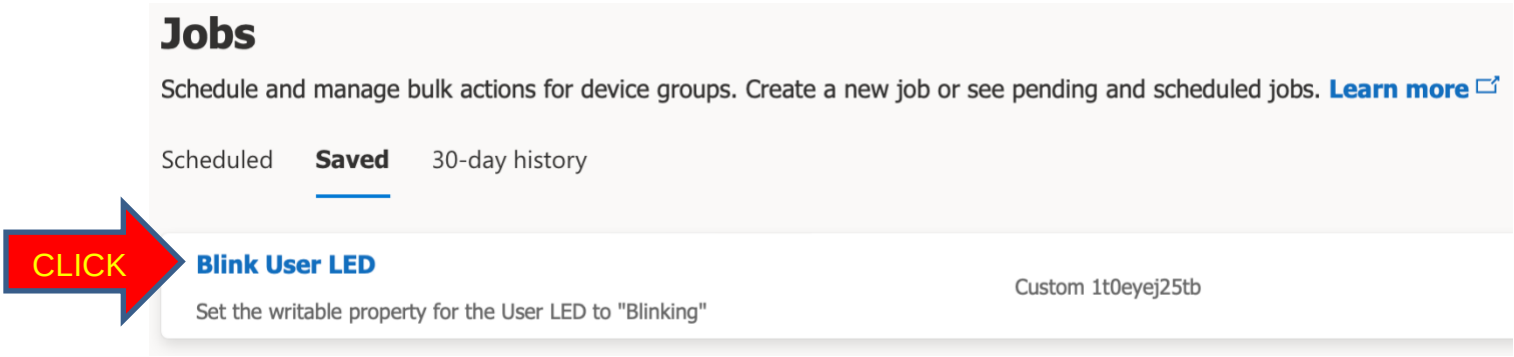
Step 2e: In the Job properties box, select the following:

- Job type = Property
- Name = User LED
- Value = Blinking



Step 2f: Click on **Save and exit** to add the job to the list of saved jobs on the **Jobs** page

Step 2g: Click on the newly-created job in the list of **Saved** jobs

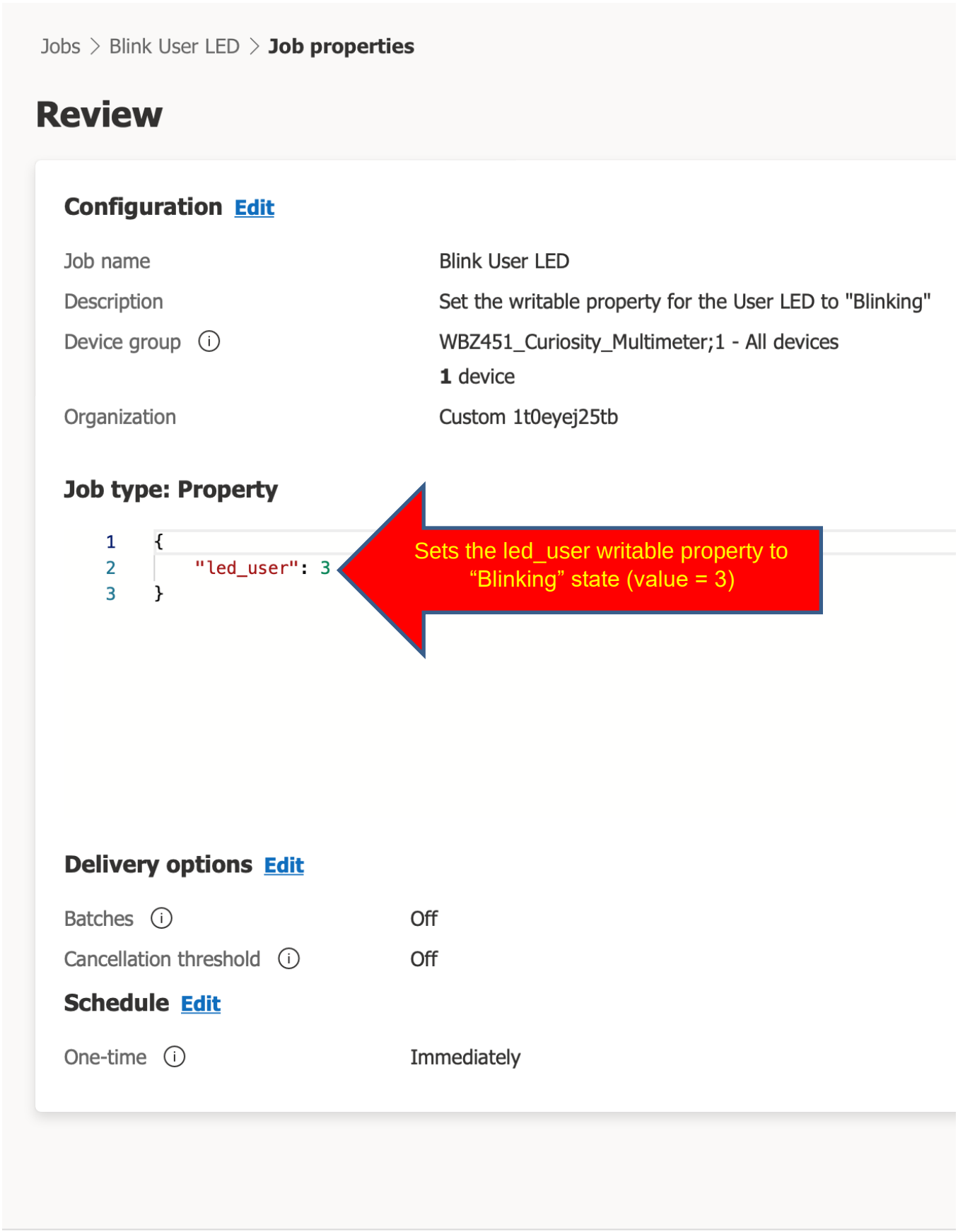


Step 2h: Select **Next** to move to the **Delivery Options** page. For now leave each option as **Enable = Off**. The Delivery Options page lets you set the delivery options for this job:

- **Batches** let you stagger jobs for large numbers of devices. The job is divided into multiple batches and each batch contains a subset of the devices. The batches are queued and run in sequence
- **Cancellation threshold** lets you automatically cancel a job if the number of errors exceeds your set limit. The threshold can apply to all the devices in the job, or to individual batches

Step 2i: Select **Next** to move to the **Schedule** page. This page lets you enable a schedule to run the job in the future. You can set up a job to run one time, daily, or weekly. For now leave the **Schedule Enable = No**

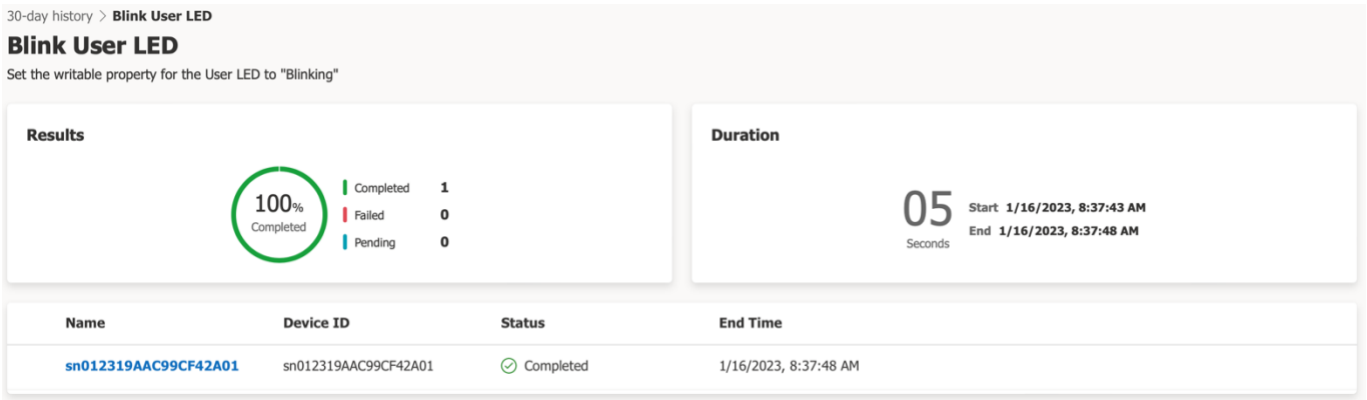
Step 2j: Select **Next** to move to the **Review** page. The **Review** page shows the job configuration details


Jobs > Blink User LED > Job properties
Review
Configuration Edit
Job name
Blink User LED
Description
Set the writable property for the User LED to "Blinking"
Device group
WBZ451_Curiosity_Multimeter;1 - All devices
1 device
Organization
Custom 1t0eyej25tb
Job type: Property
1 {
2 "led_user": 3
3 }
Sets the led_user writable property to "Blinking" state (value = 3)
Delivery options Edit
Batches
Off
Cancellation threshold
Off
Schedule Edit
One-time
Immediately
Previous
Run

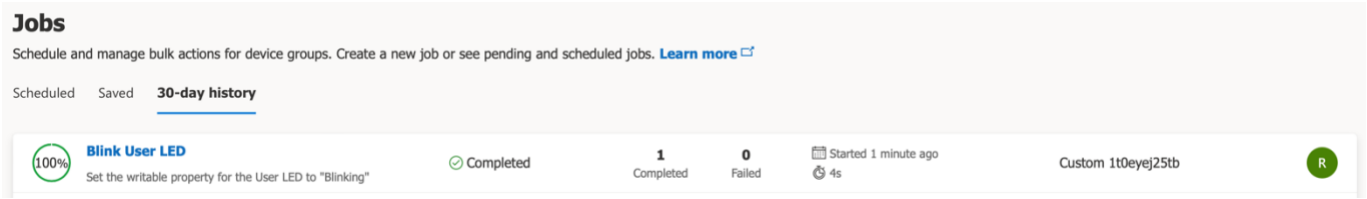
Step 2k: The job details page shows information about scheduled jobs. When the scheduled job executes, you see a list of the job instances. The scheduled job execution should also part of the **30-day history** job list. On this page, you can **Unschedule** the job or **Edit** the scheduled job. You can return to a scheduled job from the list of scheduled jobs.

Step 2l: In the job wizard, you can choose to not schedule a job, and just run it immediately. Click on the **Run** button to run the job now. When the job has finished execution, the User LED on the WBZ451 Curiosity Board should be toggling due to the property update.

Step 2m: A job goes through pending, running, and completed phases. The job execution details contain result metrics, duration details, and a device list grid. When the job is complete, you can click on the **Results log** icon to download a CSV file of your job details, including the devices and their status values. This information can be useful for troubleshooting.



Step 2n: The job now appears in the **30-day history** list on the **Jobs** page. This page shows currently running jobs and the history of any previously run or saved jobs.



Step 2o: For additional guidance on managing jobs, consult the Microsoft Azure IoT Central documentation**:** https://docs.microsoft.com/en-us/azure/iot-central/core/howto-manage-devices-in-bulk

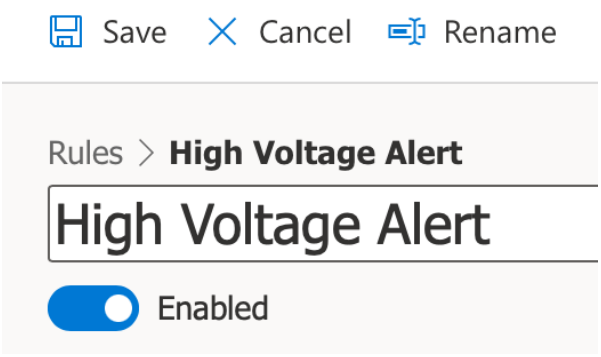
STEP 3: Set a Rule

Rules in IoT Central serve as a customizable response tool that triggers on actively monitored events from connected devices. You can define one or more actions that happen when a rule triggers. For example, when a specific condition has been detected, an e-mail message can be sent out as a response to the condition. In this step, we will set a rule that sends out an e-mail message whenever the voltage measurement of the Multimeter click surpasses a specified voltage threshold.

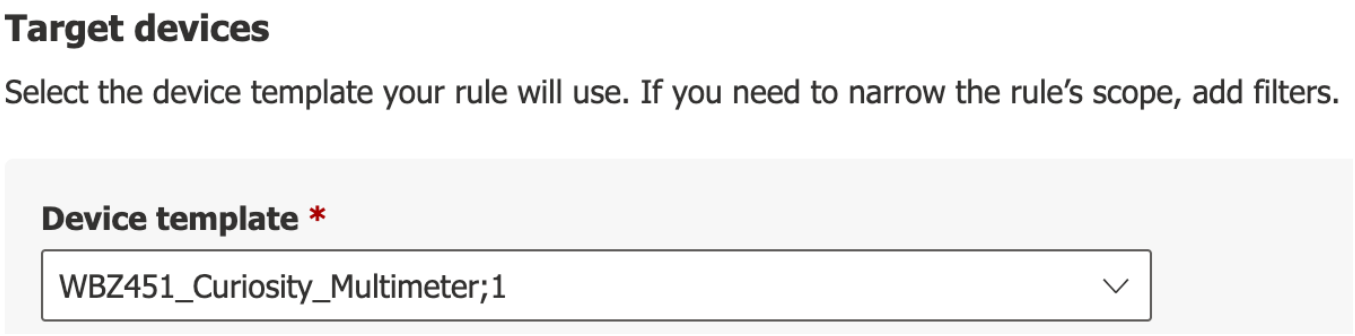
Step 3a: On the left-hand navigation pane under **Extend**, select **Rules**

Step 3b: Click the **+ New** button

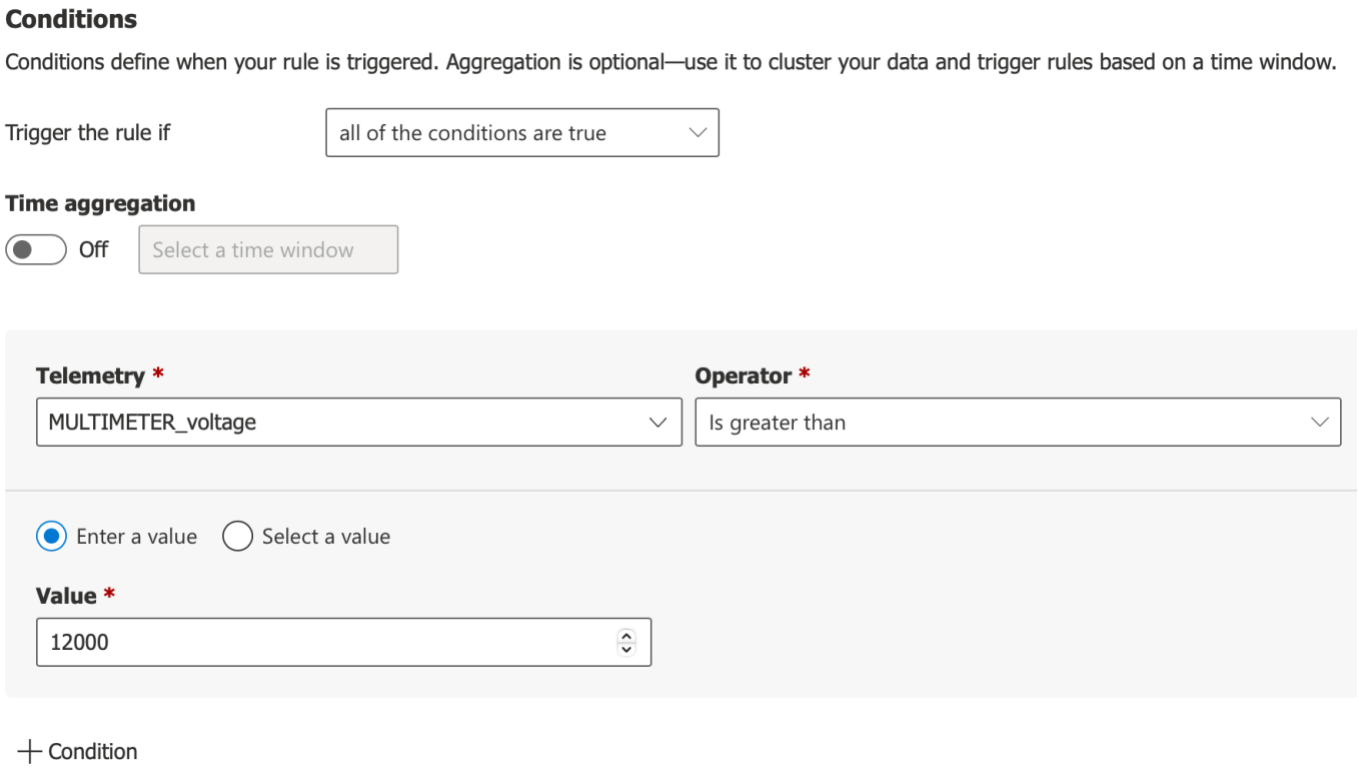
Step 3c: Enter a name for the new rule (e.g. “High Voltage Alert”)



Step 3d: Select the Device Template (e.g. WBZ451_Curiosity_Multimeter;1)

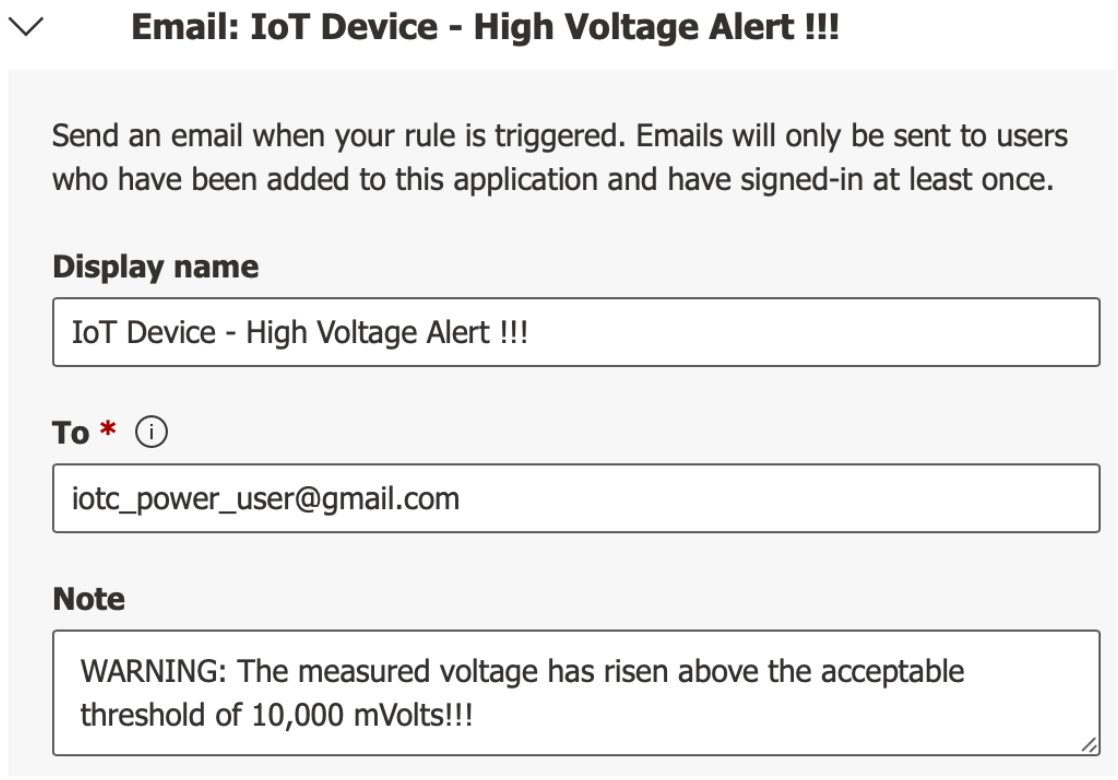


Step 3e: Set the condition(s) for generating the trigger for the rule (e.g. the Multimeter click's voltage measurement is greater than 12,000 mV). Note that the input range for the Multimeter click's voltage sensor is 8 mV minimum to +17,068 mV maximum. Additional conditions can be added by clicking on the **+Condition** icon

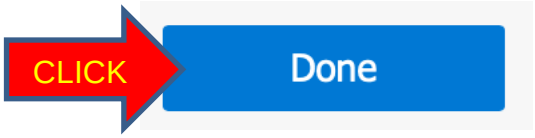


Step 3f: Set the appropriate action to take when triggered (click on the **+Email** icon)

Step 3g: Enter the information for the e-mail message to be automatically generated. Note that "Emails will only be sent to users who have been added to this application and have signed-in at least once" so use the e-mail address that is associated with your Azure Portal account



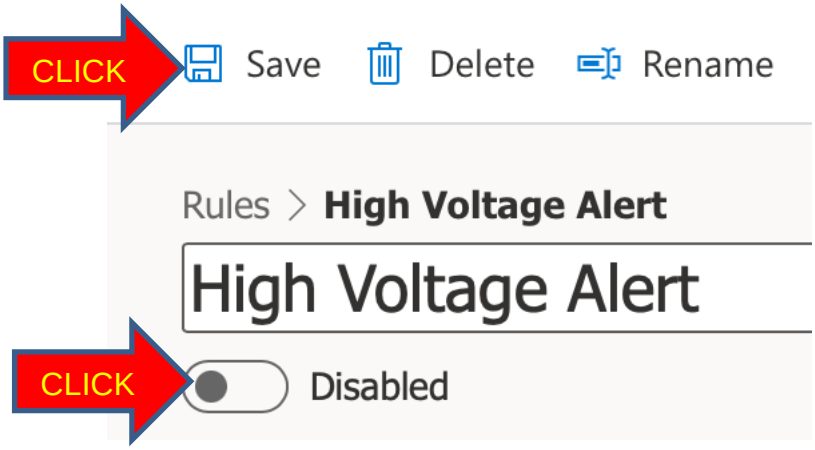
Step 3h: Click the **Done** button



Step 3i: Click the **Save** icon at the top of the page

Step 3j: Increase the voltage across the Multimeter click's voltage measurement terminals to go above the trigger threshold and wait for the e-mail alert. Use the device's **Overview** or **Raw Data** tab to visually confirm that the last known value of **MULTIMETER_voltage** has gone above the trigger threshold prior to checking for reception of the e-mail message.

Step 3k: Once you start receiving the e-mail alerts, you will receive a new e-mail alert every time a new telemetry message is sent that satisfies the rule. Adjust the power supply voltage to below the trigger condition and the e-mail alerts should cease. To permanently stop the incoming flurry of e-mail alerts, configure the rule for **Disabled** and then click on the **Save** icon



Step 3k: For additional guidance on configuring rules, consult the Microsoft Azure IoT Central documentation: https://docs.microsoft.com/en-us/azure/iot-central/core/howto-configure-rules

Results:

In this exercise, we extended the functionality of the IoT Central application by creating an analytics query, running a job, and setting a rule.

Summary:

Upon conclusion of the lab exercises, we have created a successful connection between an IoT device and the Microsoft Azure cloud services. We have seen how telemetry, properties, and commands can be implemented in a sensor node and how we can harness the power of using IoT Central to monitor and manage groups of devices at scale using analytics, jobs, and rules.