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:

- USB-to-UART Serial Adapter
- 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)

Required Software Tools:

- MPLAB IPE v6.05
- MPLAB X IDE v6.05
- MPLAB® XC32 Compiler v4.20
- Terminal emulator software (e.g. Tera Term)

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 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

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 device that has already successfully connected to Azure IoT Central

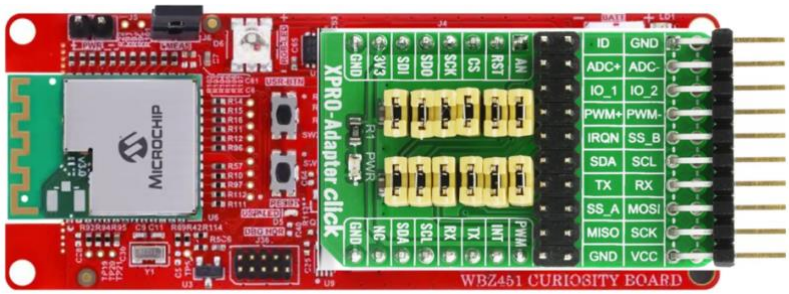
Overview:

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.

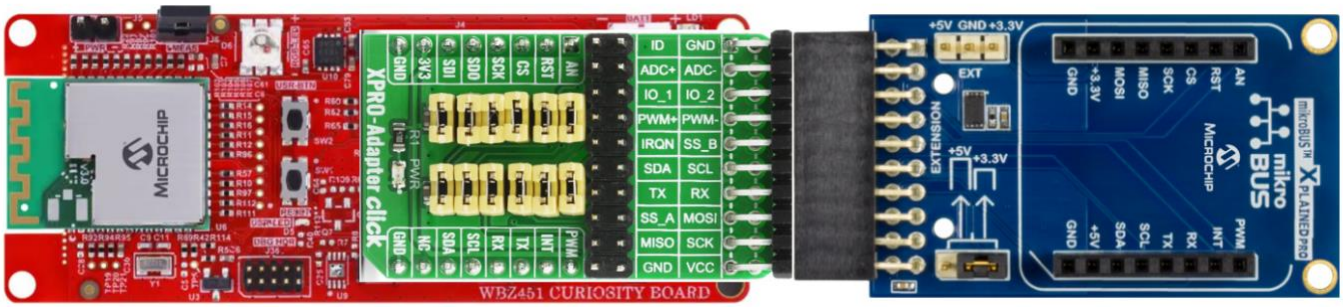
Procedure:

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

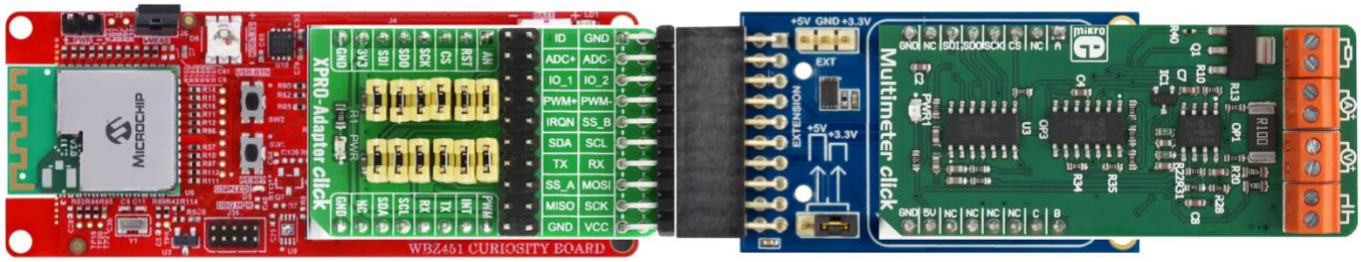
Step 1a: Install the XPRO-Adapter Click onto the WBZ451 Curiosity Board’s mikroBUS socket



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

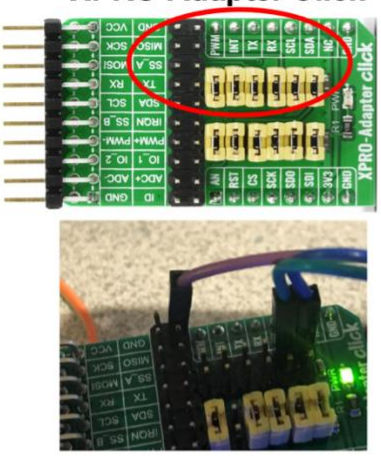


Step 1c: Install the Multimeter click onto the mikroBUS Xplained Pro

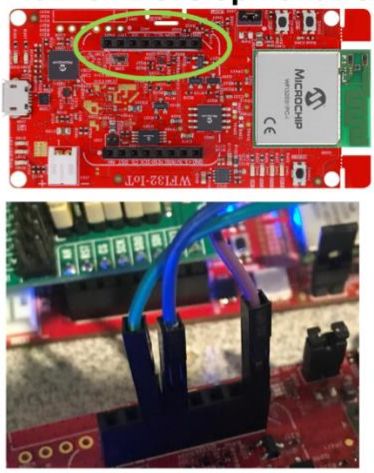


Step 1d: Connect 3 jumper wires between the XPRO-Adapter Click & WFI32-IoT boards as shown in the following wiring diagrams

XPRO-Adapter Click



WFI32-IoT Development Board



mikroBUS Pins

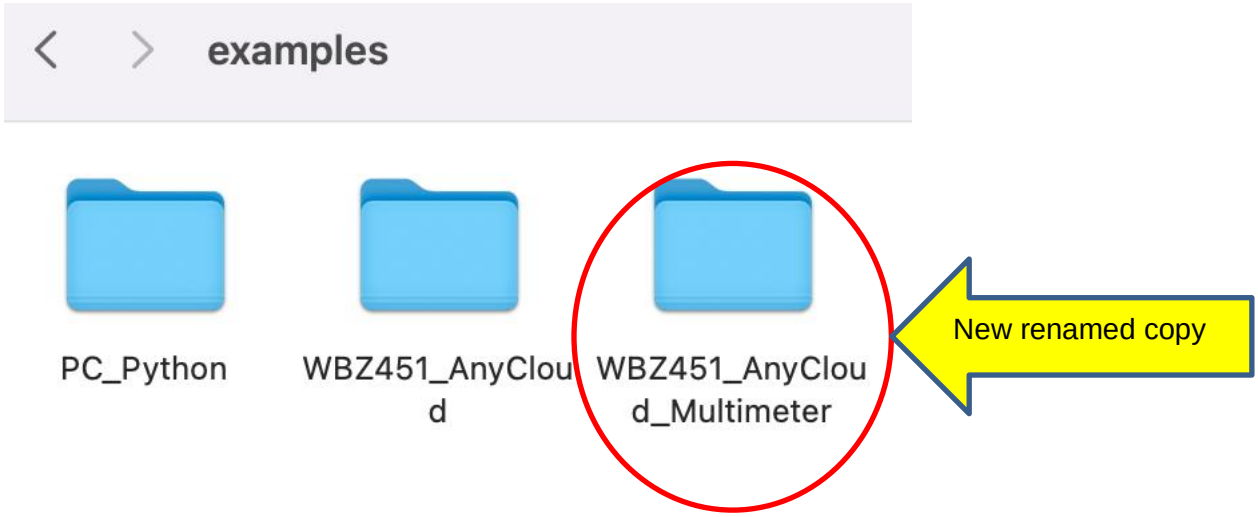
Pin Name	Pin Number
GND	9
SDA	11
SCL	12

mikroBUS Socket (J402)

Pin Number	Pin Name
9	GND
13	TX
14	RX

STEP 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

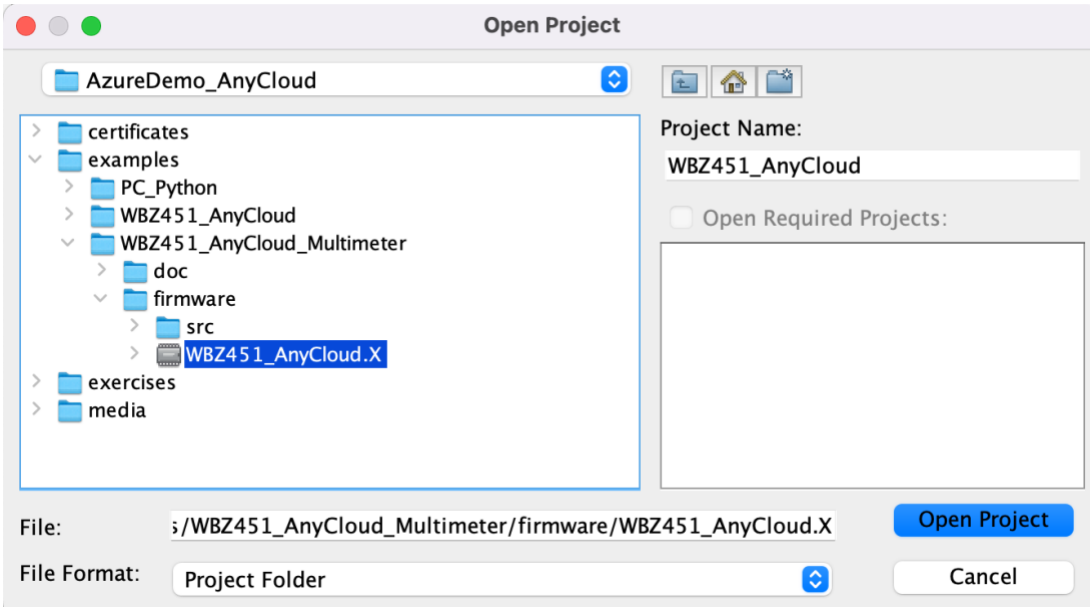
STEP 4: Open the Newly Created Project

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

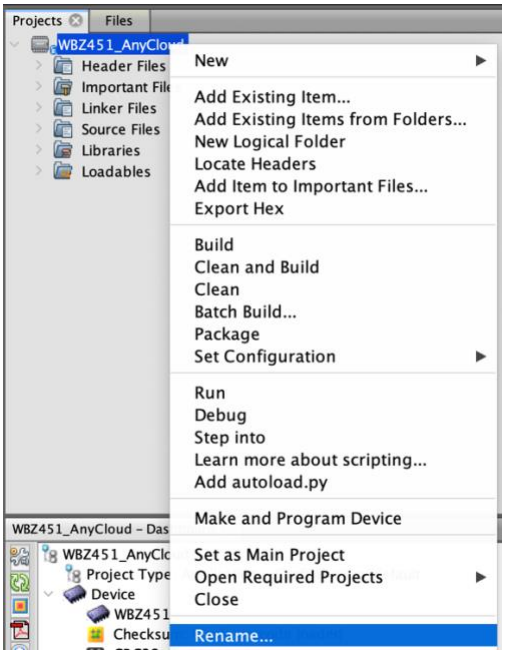
Step 4c: Select (click on) the **WBZ451_AnyCloud.X** project folder

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

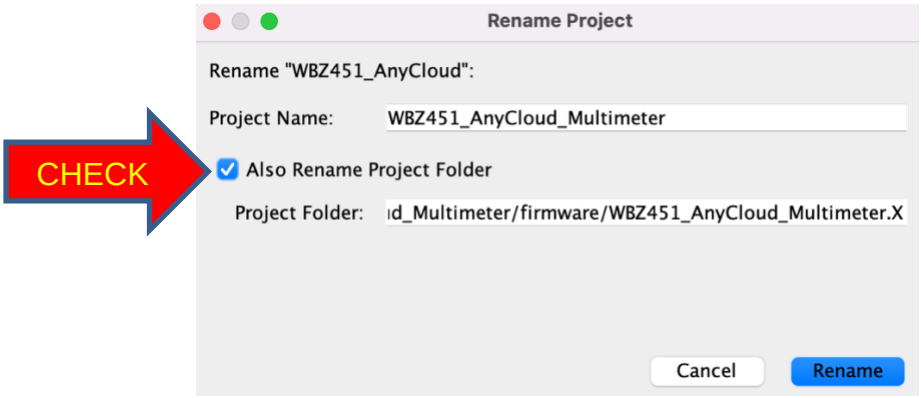


STEP 5: Rename the Project to *WBZ451_AnyCloud_Multimeter*

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



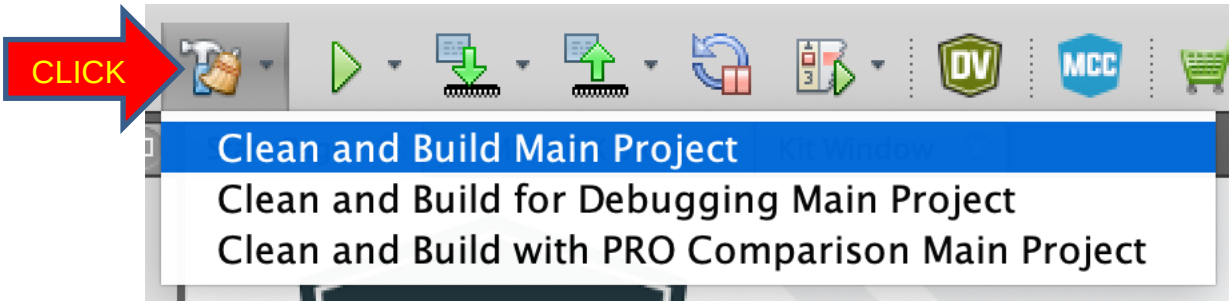
Step 5b: Check the box "**Also Rename Project Folder"** and then type **WBZ451_AnyCloud_Multimeter** for the **Project Name**



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

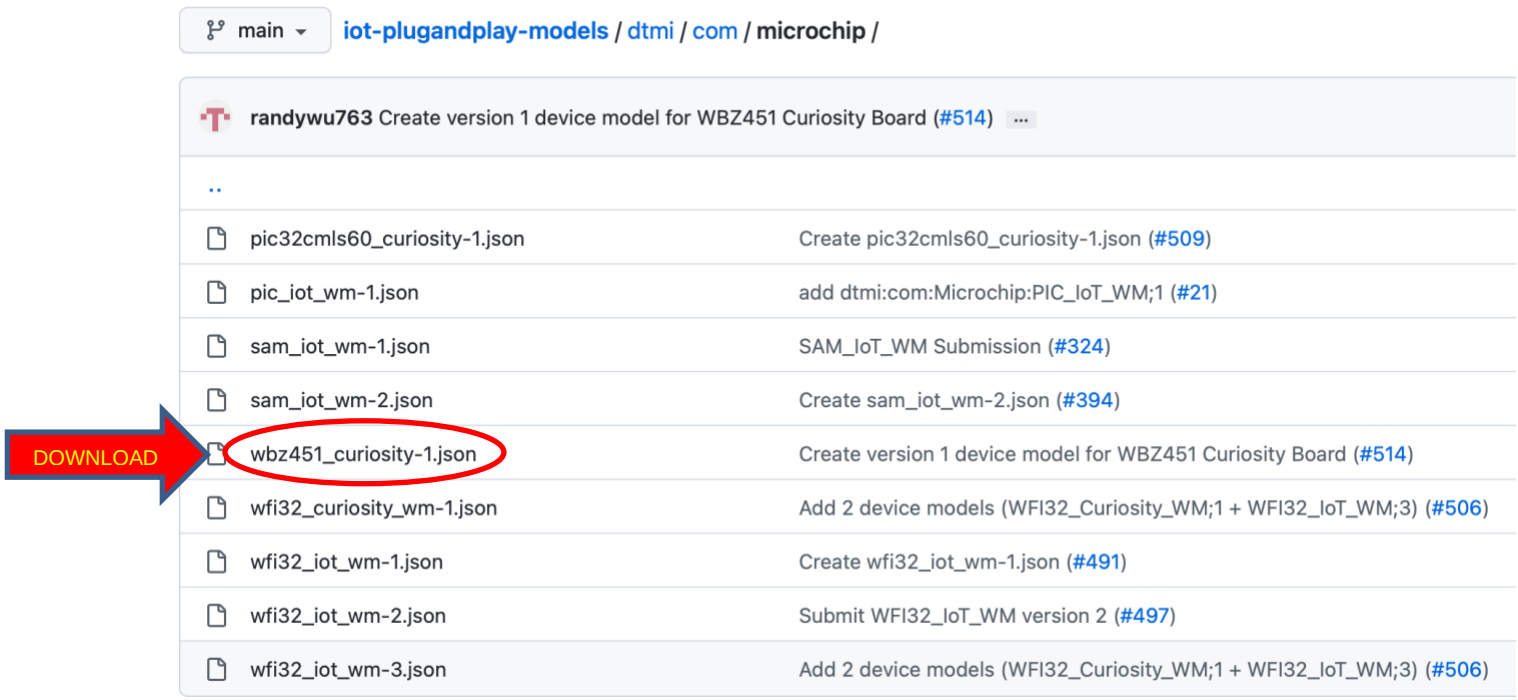


Step 6b: Verify that the build is successful before proceeding; otherwise review the previous steps to make sure each one has been implemented correctly. Do not proceed until the project builds successfully. If you need to refer to a working copy of the project, the solution can be found in the **\exercises\solutions\WBZ451_AnyCloud_Multimeter** 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**. Access this 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 7b: Determine the name of the new DTMI to be created. For the purposes of this exercise, let's just expand the model ID name to identify the addition of the Multimeter click: **dtmi:com:Microchip:WBZ451_Curiosity_Multimeter;1** (where '1' denotes version 1)

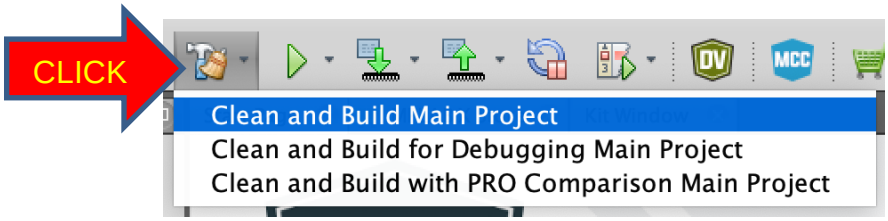
Step 7c: Open (double-click) the **app_rio2_config.h** header file located under [WBZ451_AnyCloud_Multimeter → Source Files]

Step 7d: Edit this header file by enabling the definition for including Multimeter click support. Notice that a different Digital Twins Model Interface (DTMI) is selected to represent the additional telemetry in the interface

```
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 programming the board until the project builds successfully.

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 sensor)
2. Enabled the code to send telemetry messages on behalf of the Multimeter click's voltage sensor

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.

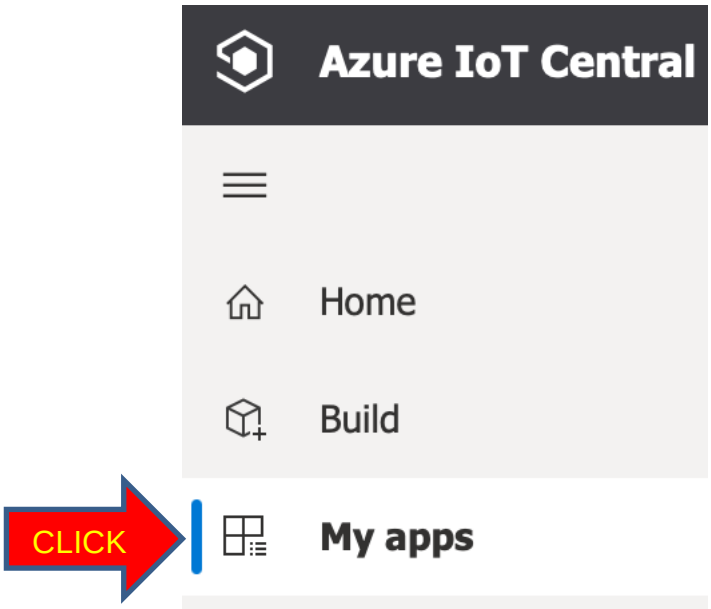
Overview:

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.

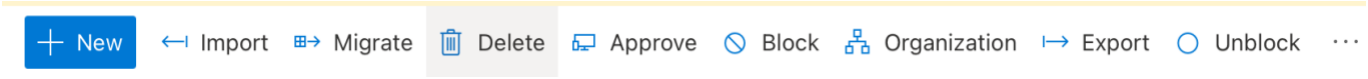
Procedure:

STEP 1: Access the existing IoT Central Demo Application

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



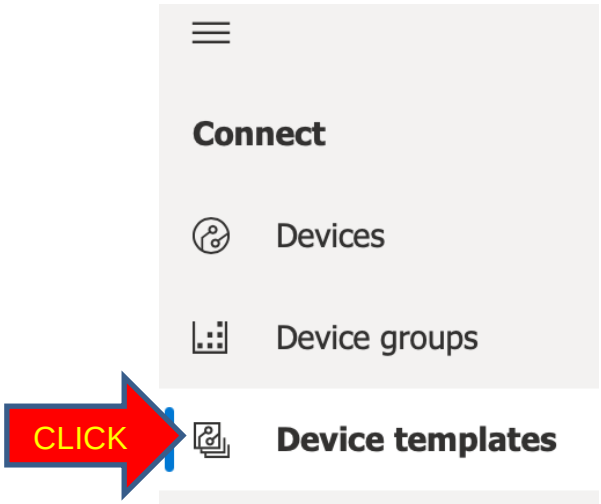
Step 1b: Click on the name of the demo application which was created/used back in Lab 1



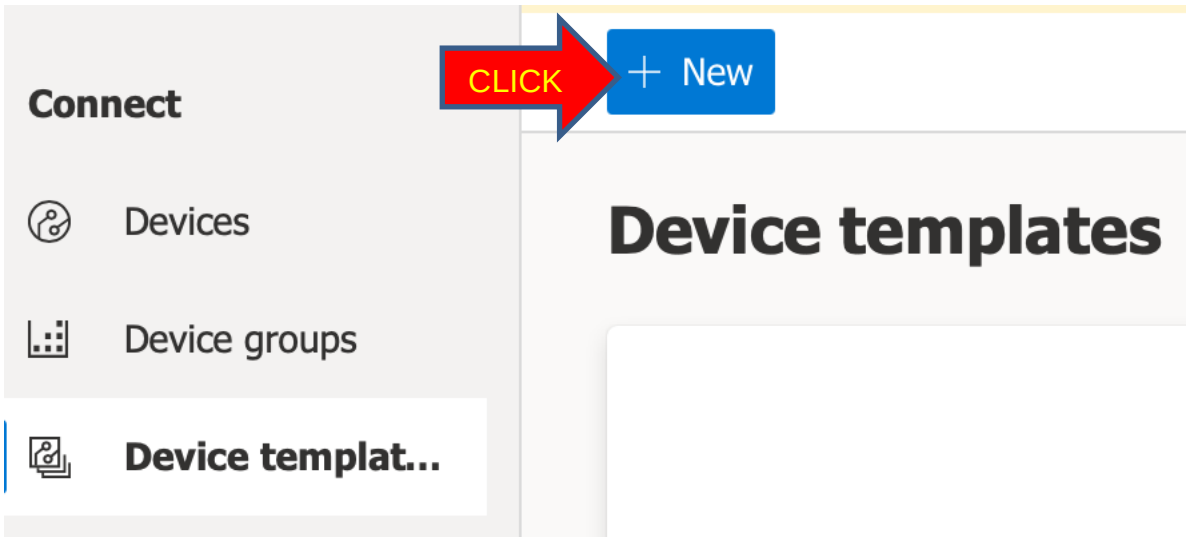
STEP 2: Create the Device Template for the New Multimeter Click DTMI

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

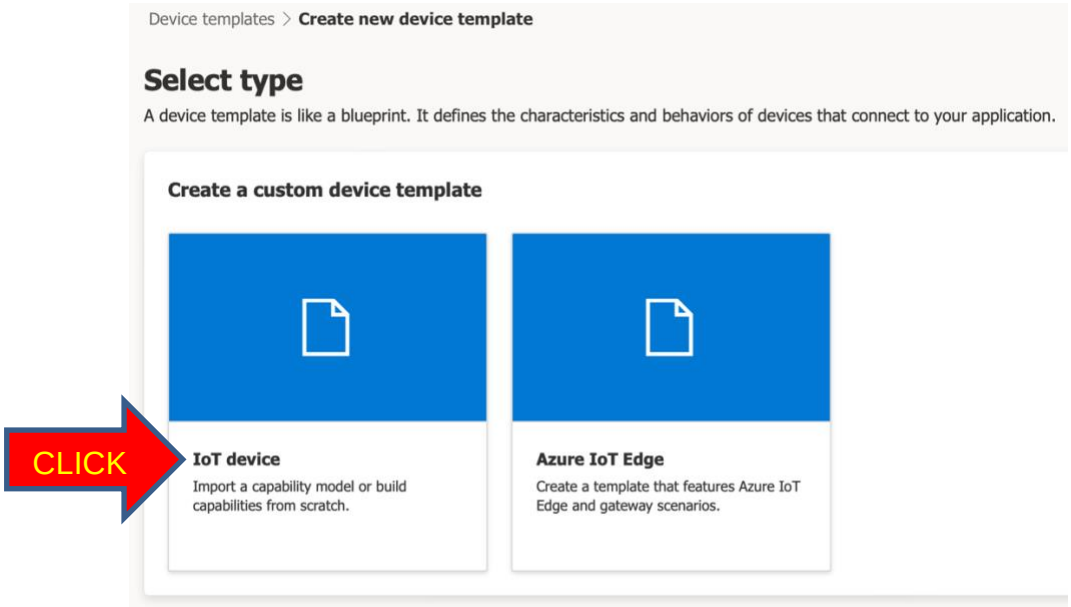
Step 2a: Using the navigation pane on the left-hand side, select [**Connect → Device templates**]



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

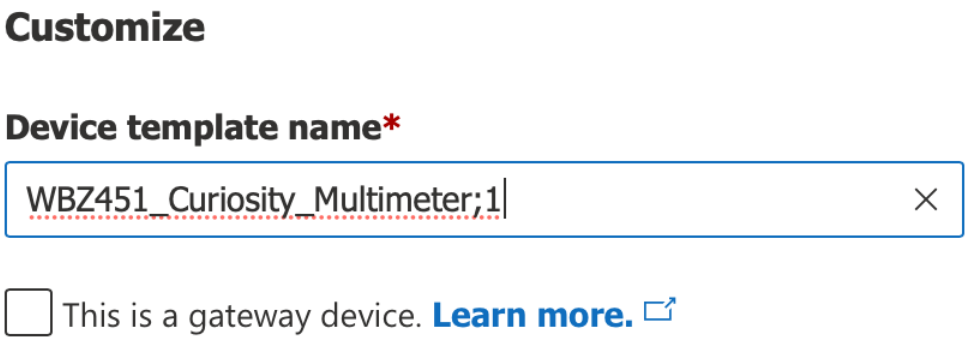


Step 2c: Click the **IoT device** tile



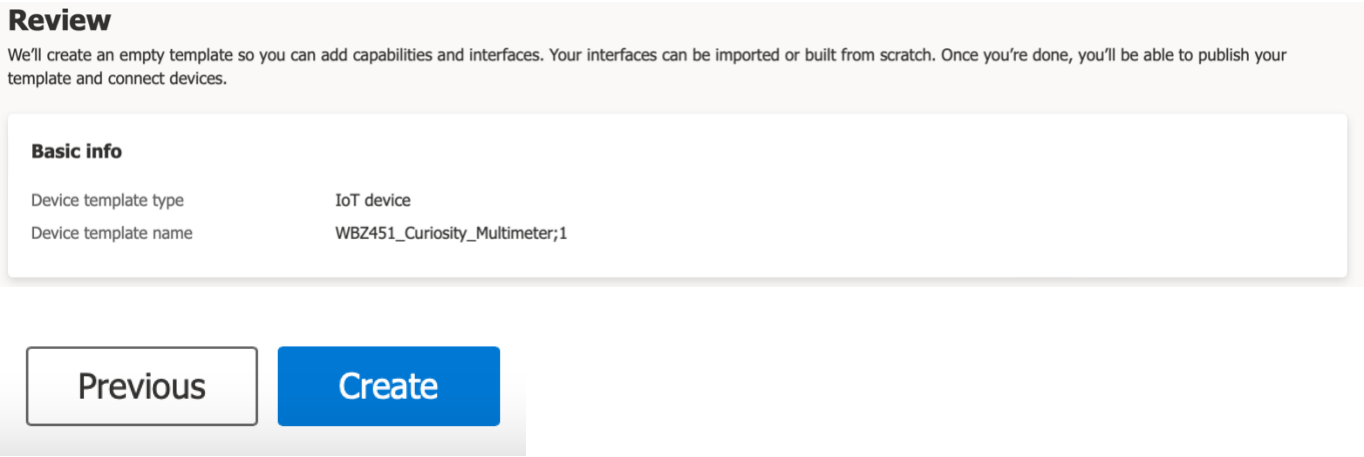
Step 2d: Click on the **Next: Customize** button

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

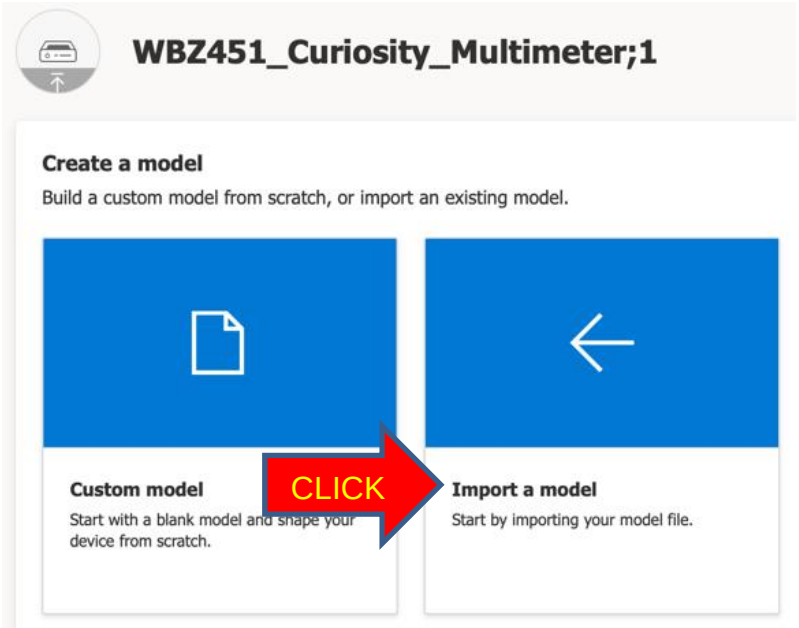
```
266     "description": {
267         "en": "WBZ451 Curiosity Board & Multimeter Click"
268     },
269     "displayName": {
270         "en": "WBZ451 Curiosity Board & Multimeter Click"
271     },
```

- Copy and paste the following lines just below the line beginning with “contents” (line #4). This new object specifies an extra telemetry value called “MULTIMETER_voltage” (in mV)

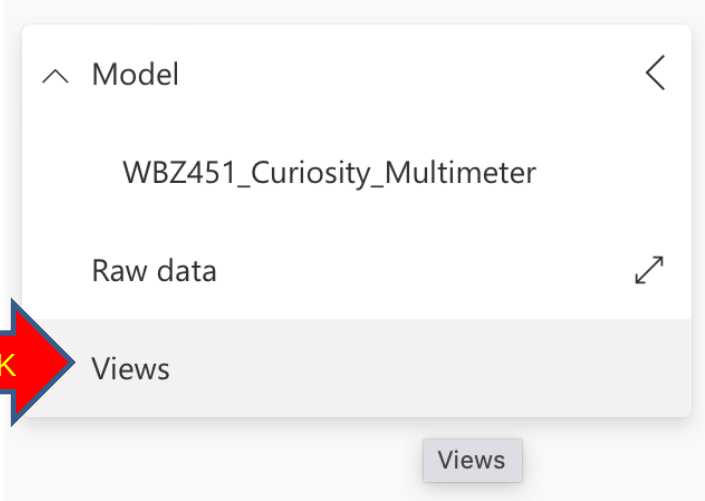
```
{
    "@type": [
        "Telemetry",
        "Voltage",
        "NumberValue"
    ],
    "description": {
        "en": "Voltage reading in mV from the Multimeter Click"
    },
    "displayName": {
        "en": "MULTIMETER_voltage"
    },
    "name": "MULTIMETER_voltage",
    "schema": "double",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

- Save changes to the file

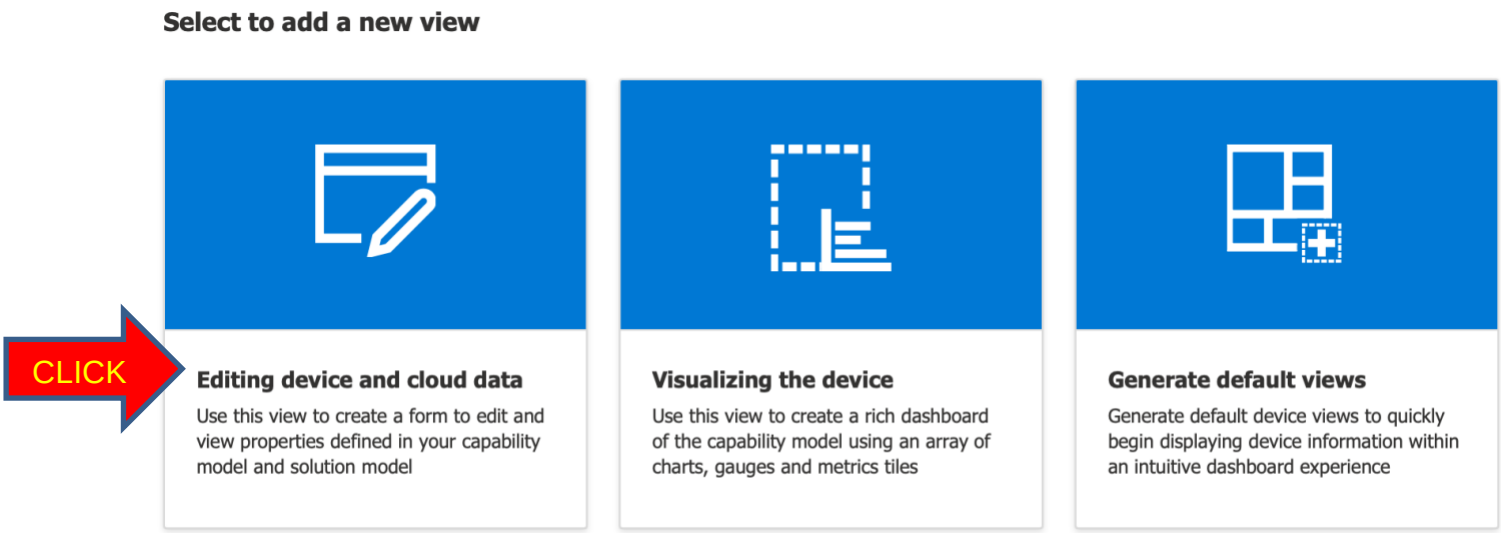
Step 2i: Click the **Import a model** tile



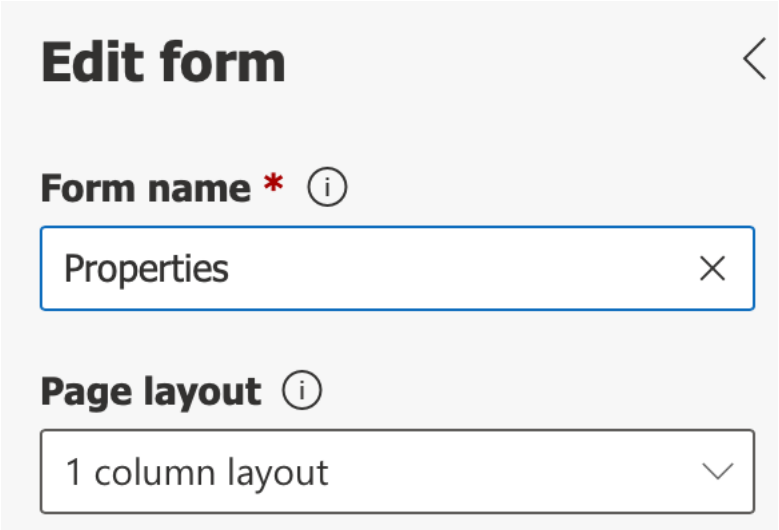
Step 2j: Click on the **Views** category underneath the model name



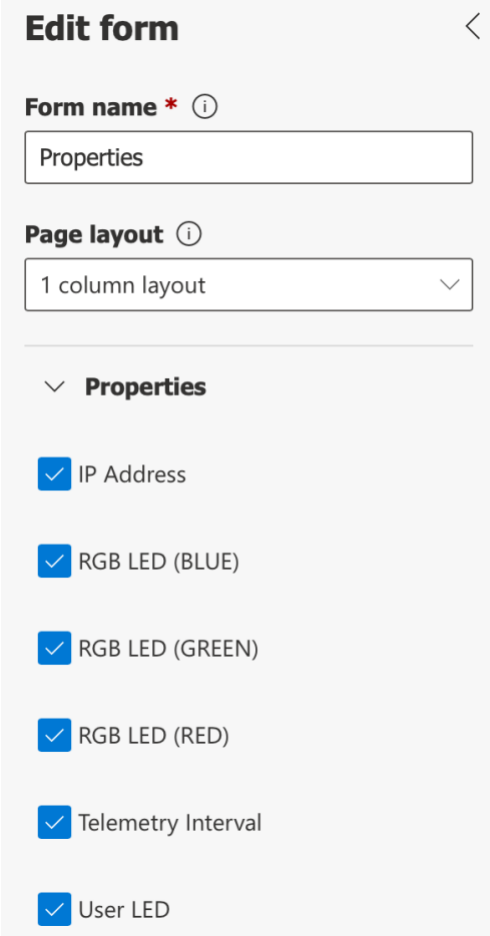
Step 2k: Click on the **Editing device and cloud data** tile



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



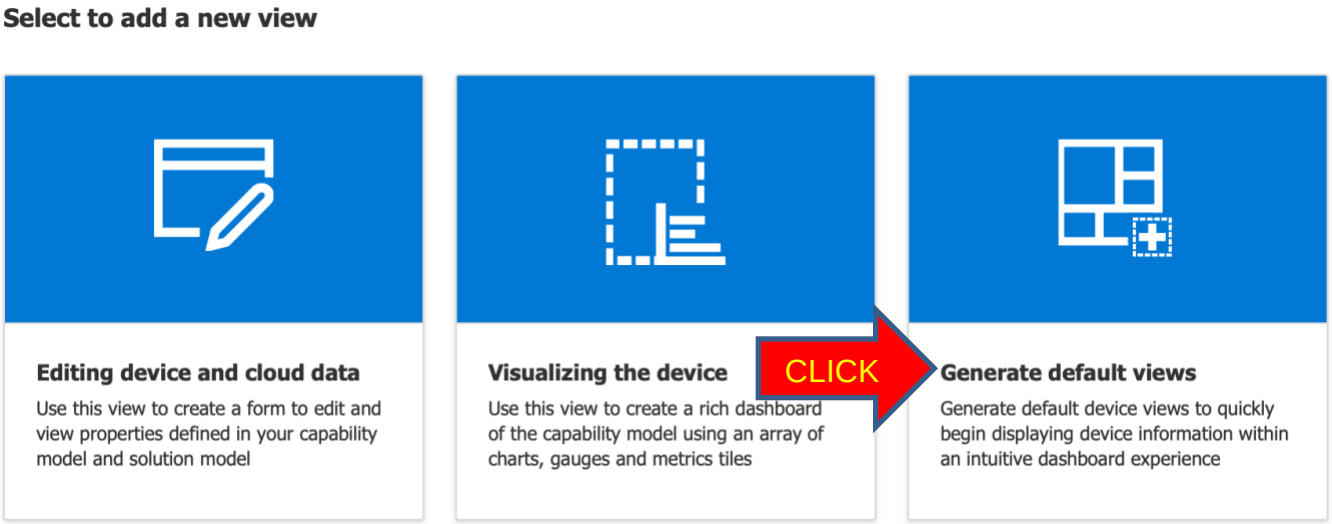
Step 2m: Click on the **Properties** category and check all the boxes



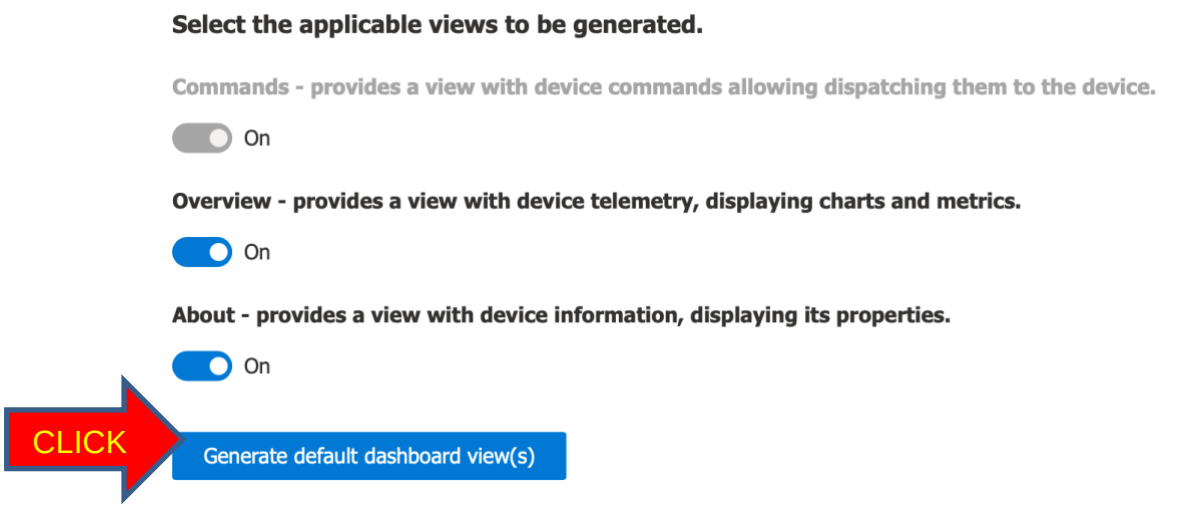
Step 2n: Click on the **Add section** button (bottom of the page), click on the **Save** icon (top of the page), and then click on the **Back** icon

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

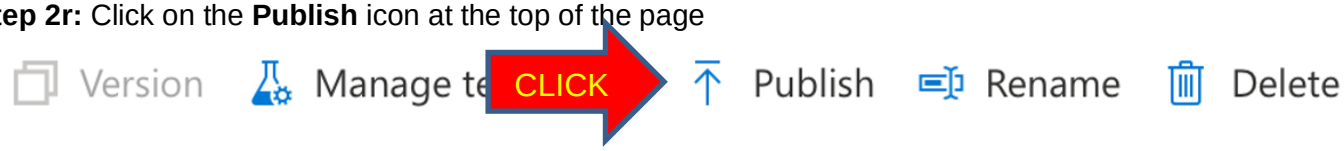
Step 2p: Click on the **Generate default views** tile



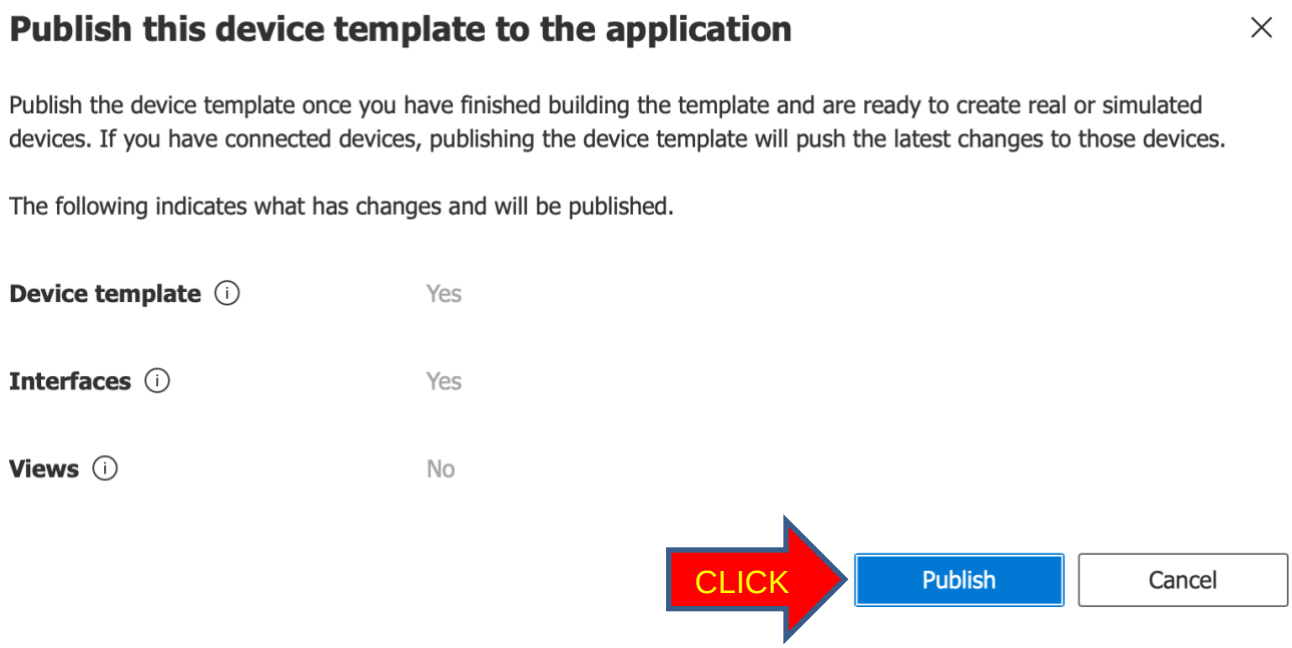
Step 2q: Click on the **Generate default dashboard view(s)** button



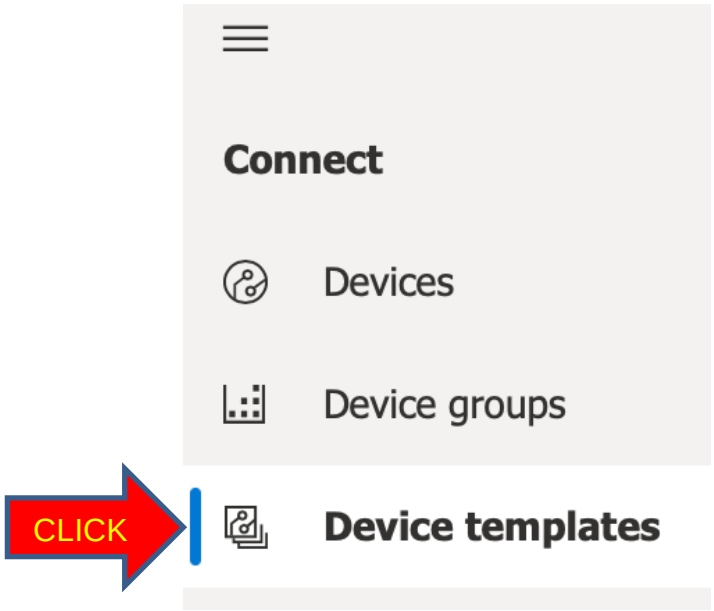
Step 2r: Click on the **Publish** icon at the top of the page



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



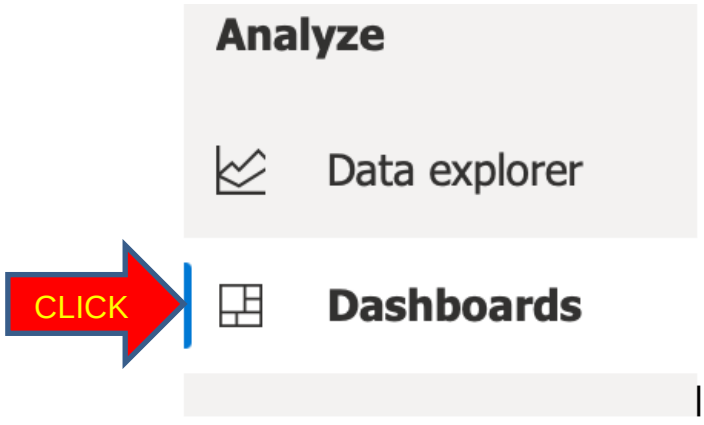
Step 2t: Using the navigation pane on the left-hand side, select [**Connect → Device templates**]



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

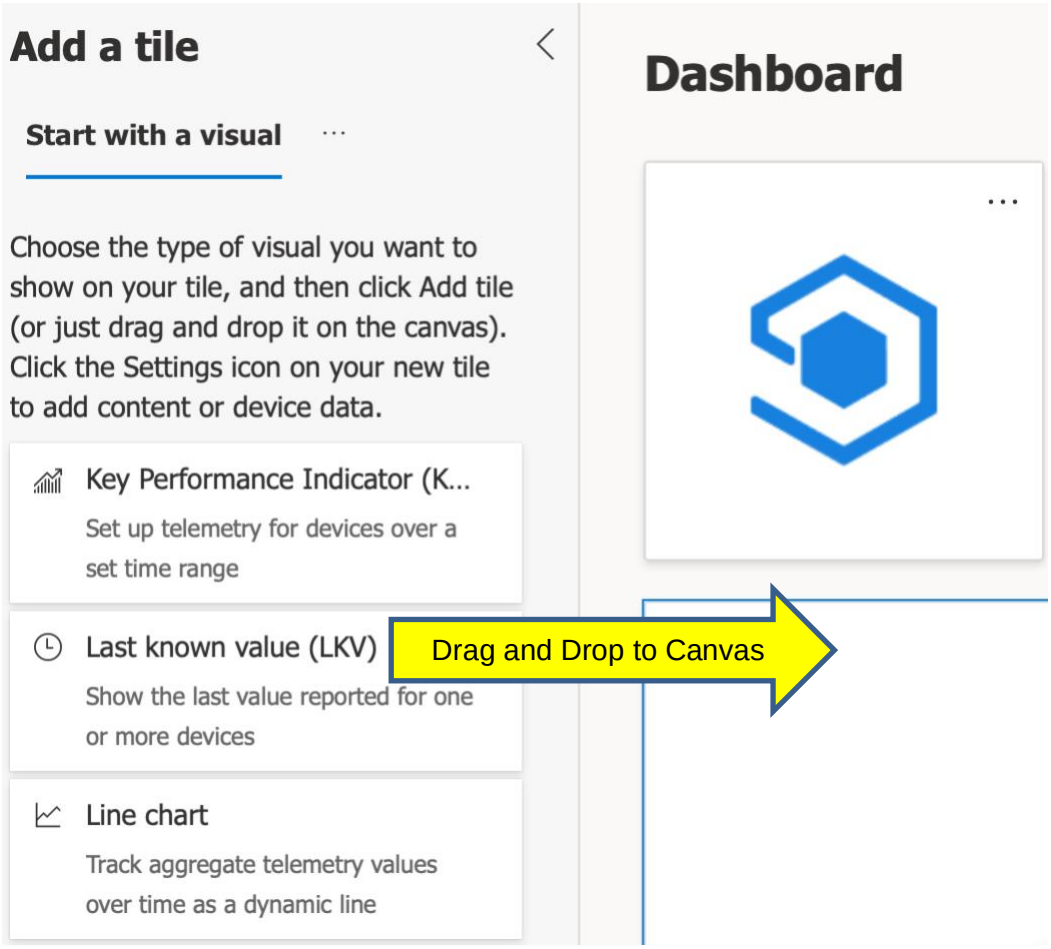
STEP 3: Create a New Dashboard to Display the Multimeter Click Sensor Values

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

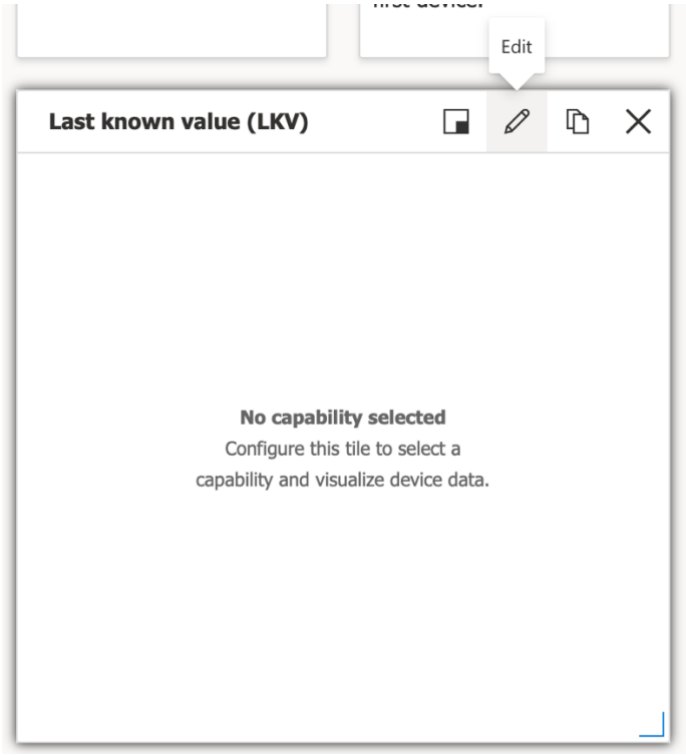


Step 3b: Click on the **Edit** icon

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



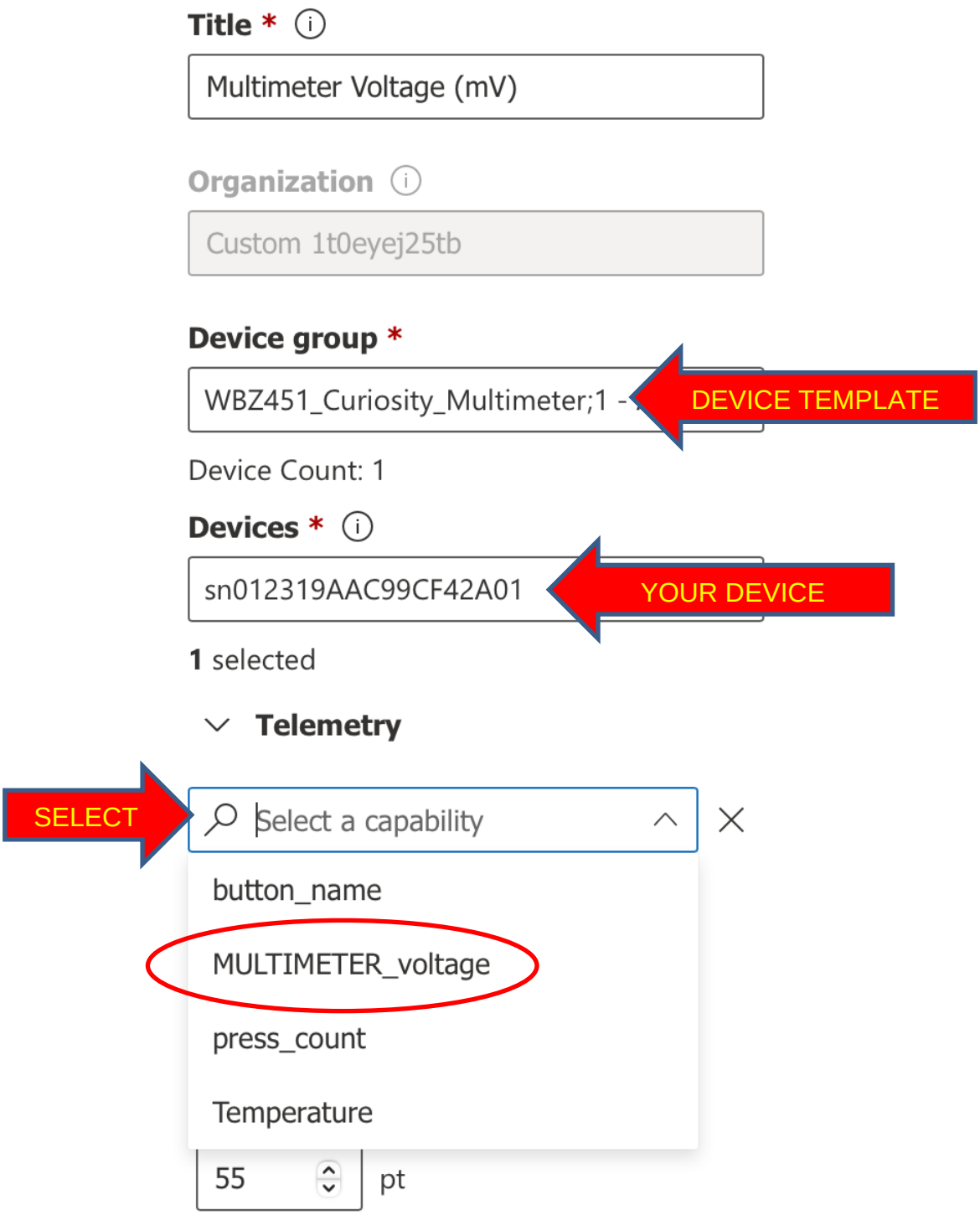
Step 3e: Change the Title of the tile to something meaningful (e.g. **Multimeter Voltage (mV)**)

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

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

Step 3h: Click on **+Capability** to add the telemetry for the Multimeter Click voltage

Step 3i: Increase the **Text size** to something above 50 pt. Click the **Update** button to finalize the edits to the tile

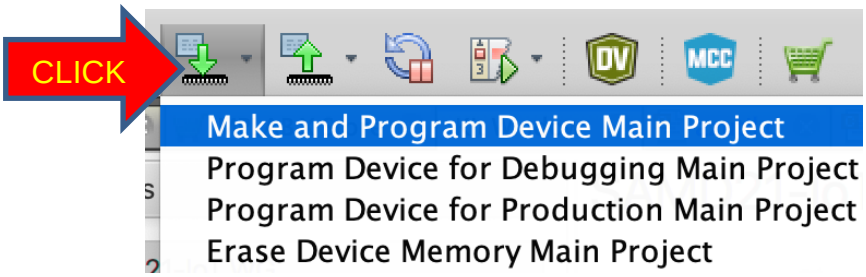


Step 3j: Click the **Save** icon

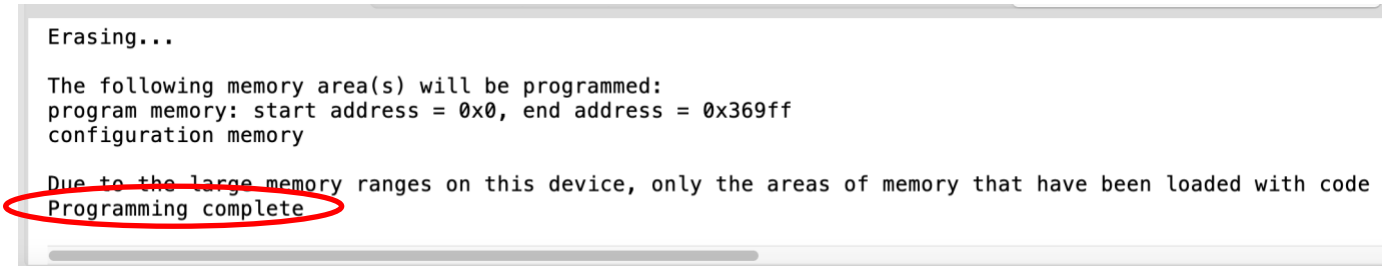
STEP 4: Press the RESET button on the WFI32-IoT AnyCloud™ bridge board

STEP 5: Program the WBZ451 Curiosity Board

Step 5a: Click on the **Make and Program Device Main Project** icon in the MPLAB X main toolbar

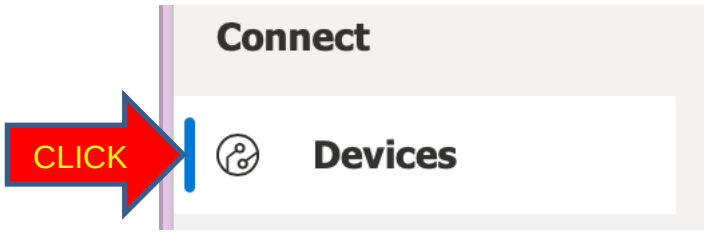


Step 5b: Verify that the programming phase was successful

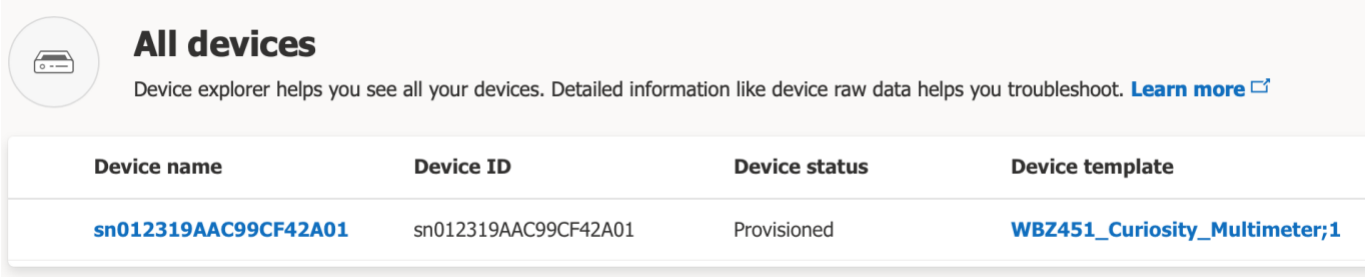


STEP 6: Confirm Device Telemetry Received by the Cloud

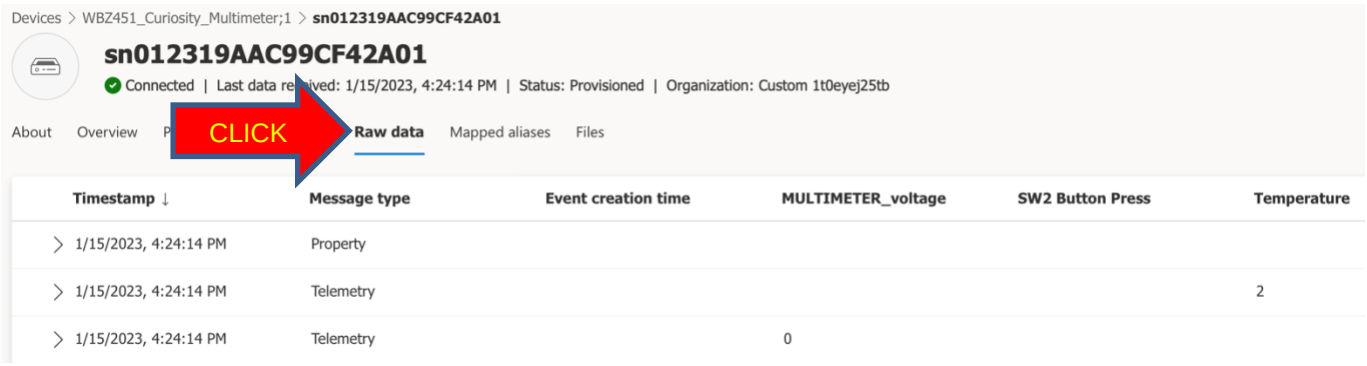
Step 6a: Using the navigation pane on the far left-hand side of the page, click on [**Connect** → **Devices**]



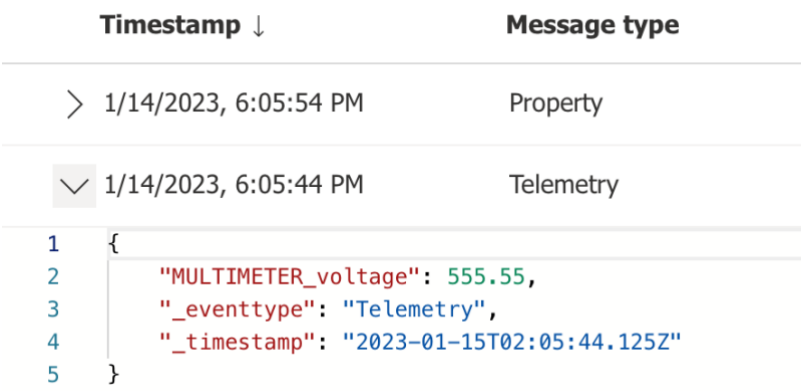
Step 6b: Click on your device name



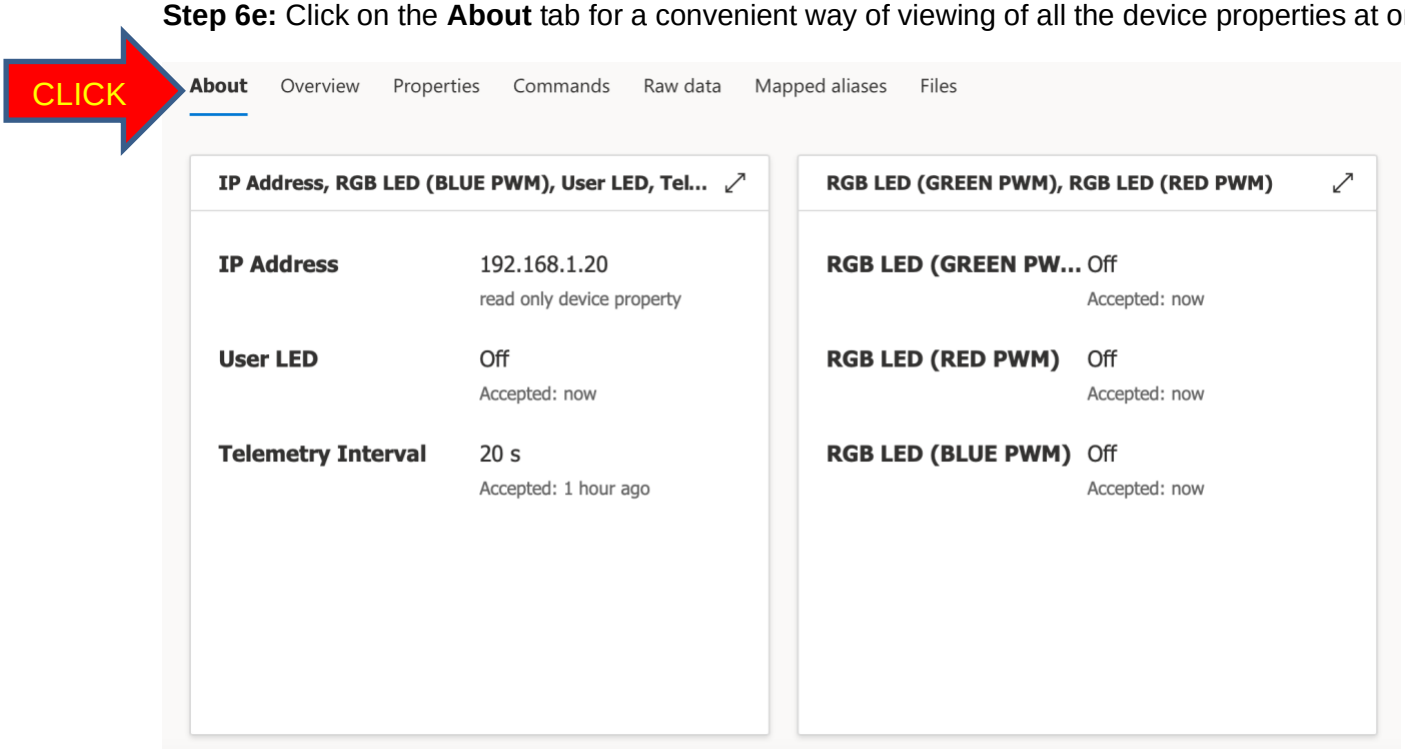
Step 6c: Click on the **Raw data** tab and confirm that property and telemetry messages are currently being received



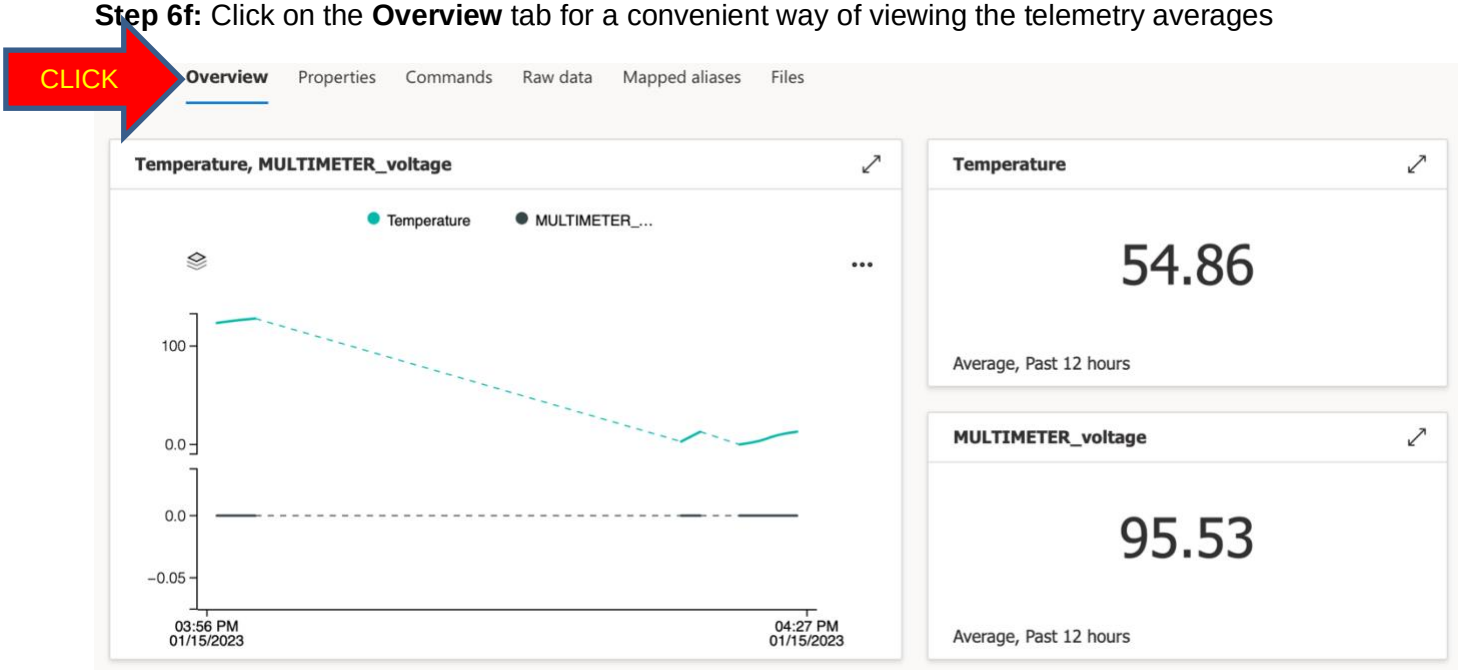
Step 6d: Expand one of the telemetry messages and observe the Multimeter voltage value being reported



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

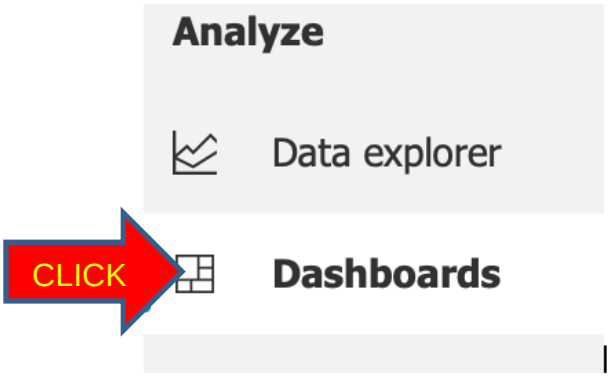


Step 6f: Click on the **Overview** tab for a convenient way of viewing the telemetry averages



STEP 7: Access the Dashboard to Visualize the Multimeter Voltage Data

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



Results:

The last reported value for the Multimeter Click voltage will be displayed on the Dashboard. The Dashboard can be easily edited to add more tiles to visualize other telemetry in various formats.

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.

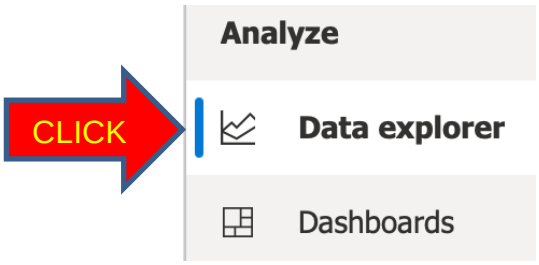
Procedure:

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**

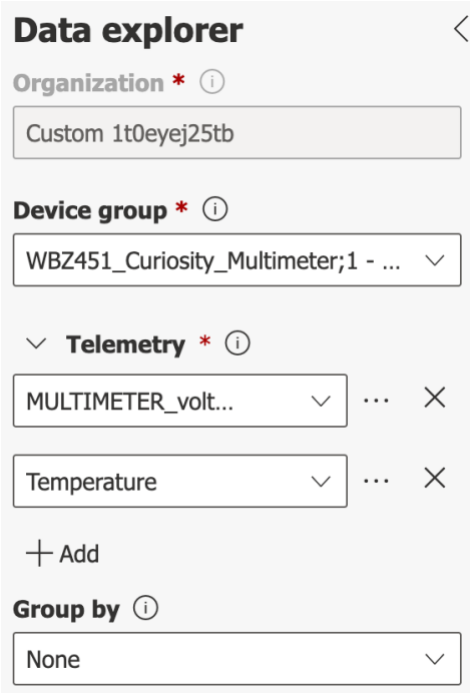


Step 1b: Click on the **+ New query** button

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

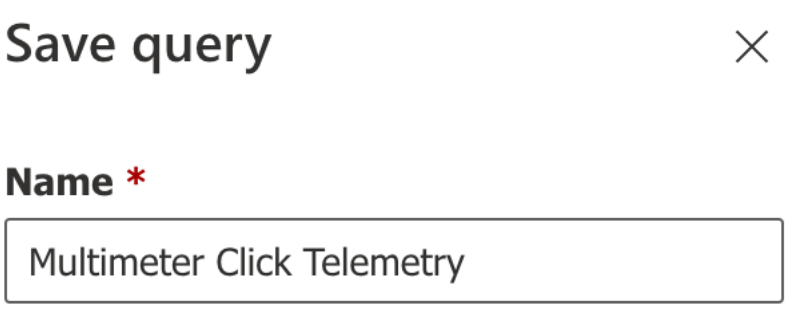
Step 1d: Under **Telemetry**, select **Temperature**

Step 1e: Click **+Add** and select **MULTIMETER_voltage**

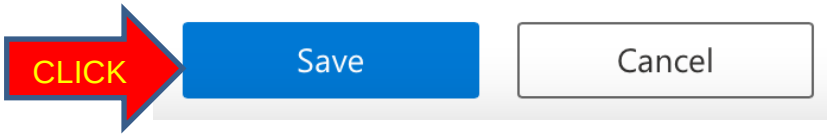


Step 1f: Click on the **Save** icon

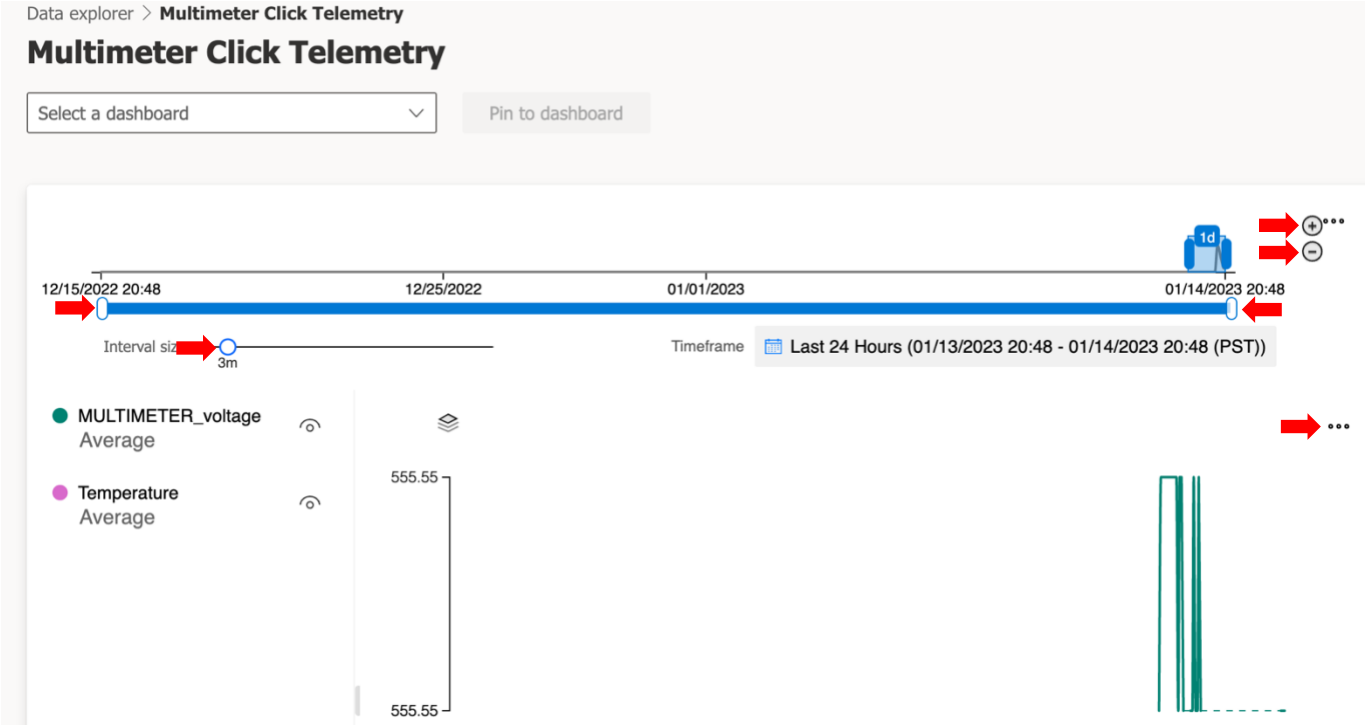
Step 1g: Type in a name for the query (e.g. Multimeter Click Telemetry)



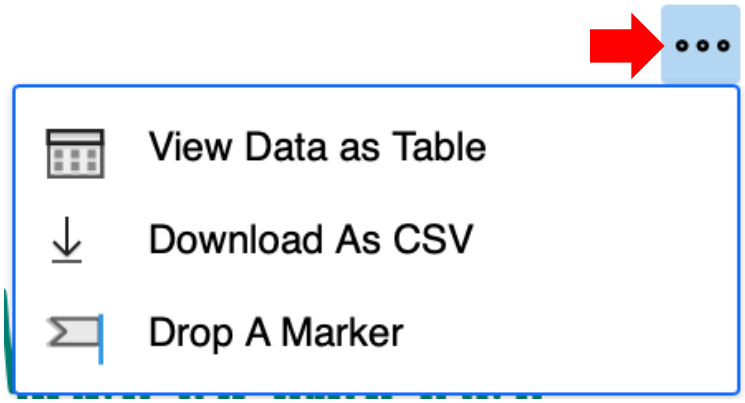
Step 1h: Click on the **Save** button



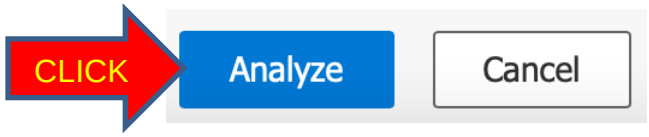
Step 1i: Click on the various sliders and buttons to interact with your data



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

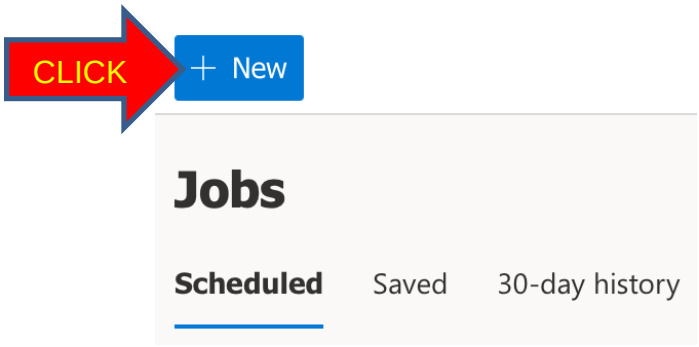
STEP 2: Run a Job

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.

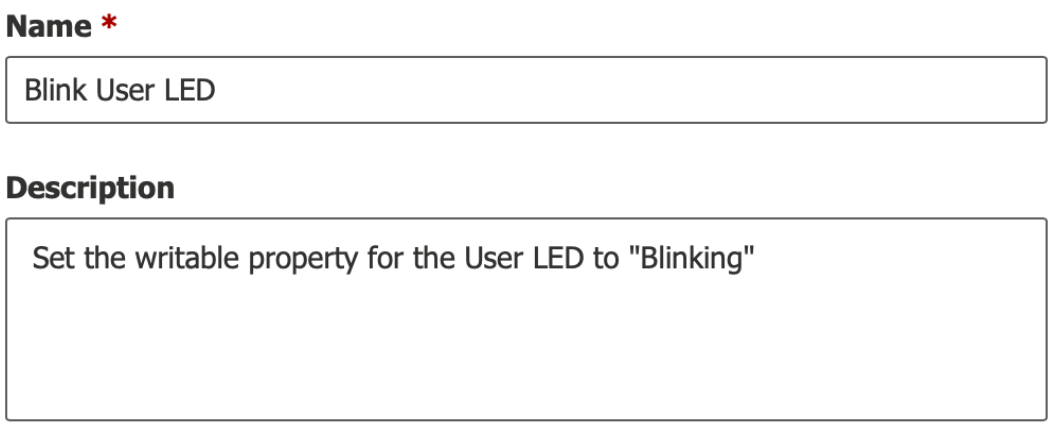
We will create a job and then run the job to set one of the writable properties (e.g. Yellow LED).

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

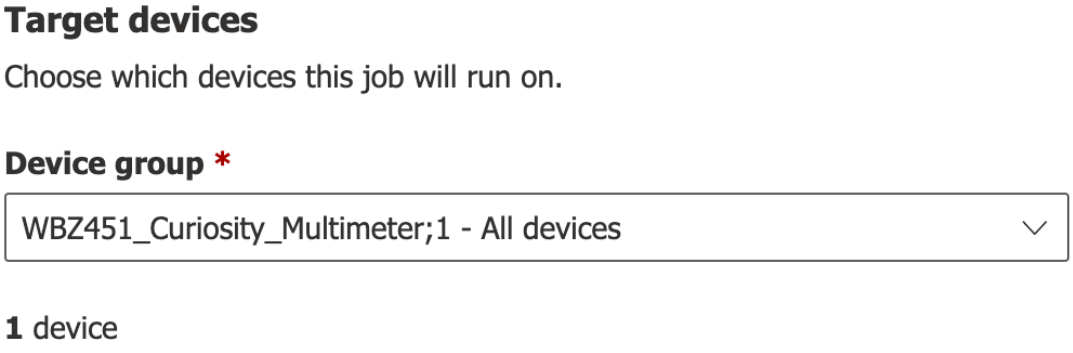
Step 2b: Click the **+ New** button



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")

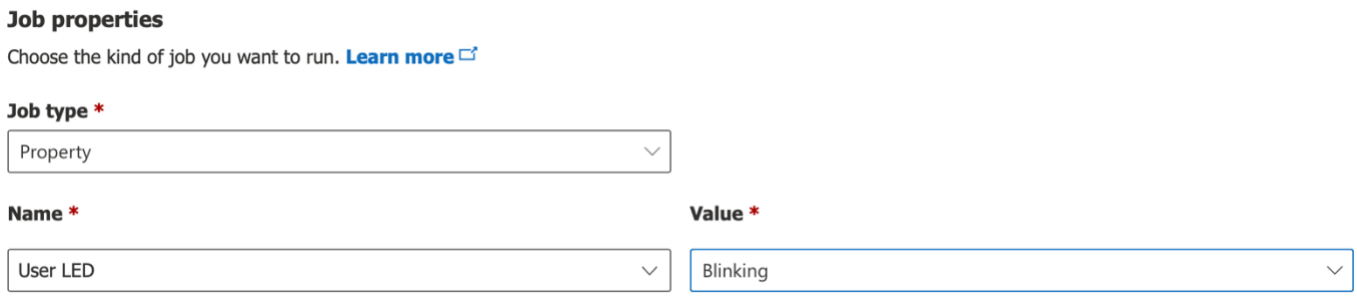


Step 2d: Select the target device group that you want your job to apply to (e.g. WBZ451_Curiosity_Multimeter;1). You can see how many devices your job configuration applies to just below the **Device group** selection



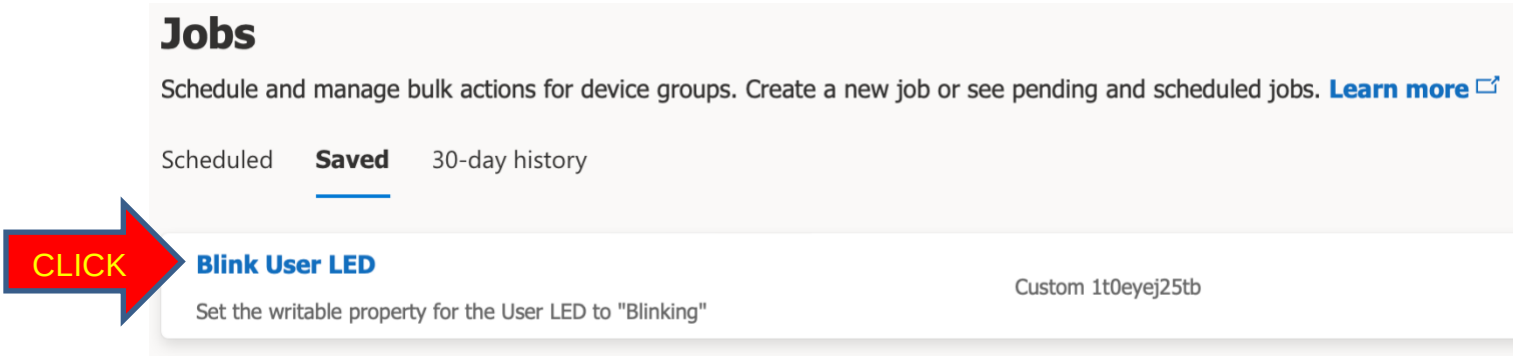
Step 2e: In the Job properties box, select the following:

- 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  }
```

Delivery options Edit

Batches ⓘ	Off
Cancellation threshold ⓘ	Off

Schedule Edit

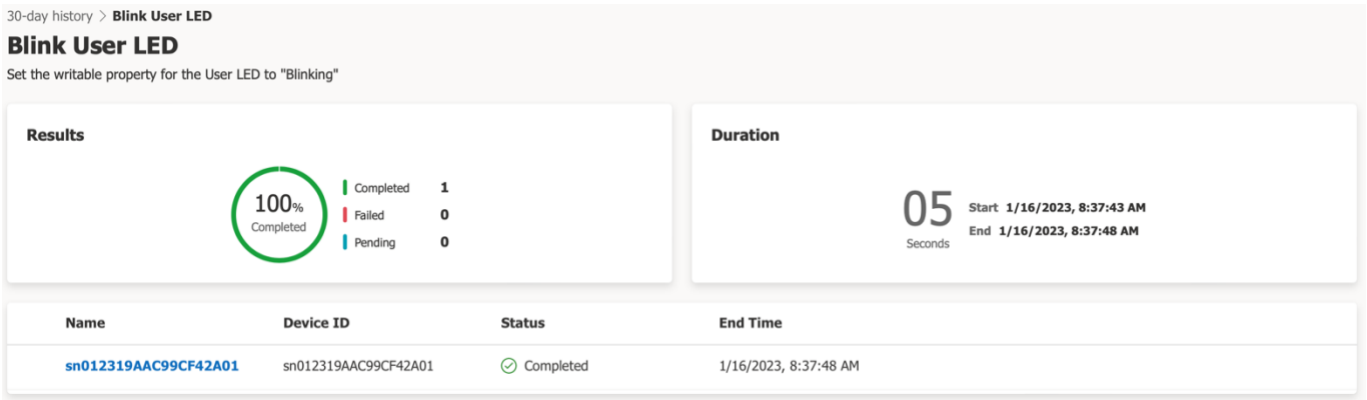
One-time ⓘ	Immediately

Previous Run

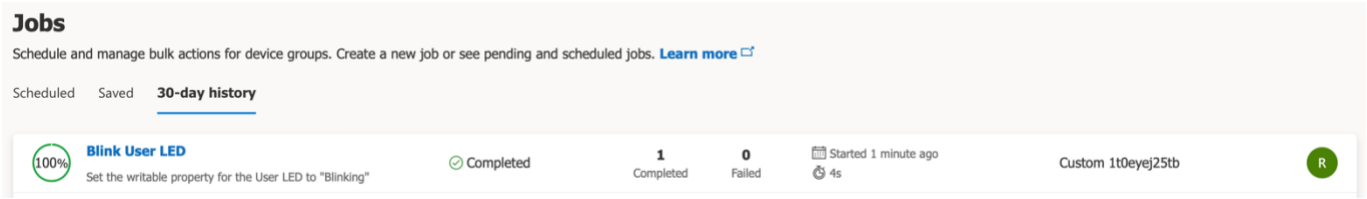
Step 2k: The job details page shows information about scheduled jobs. When the scheduled job executes, you see a list of the job instances. The scheduled job execution should also part of the **30-day history** job list. On this page, you can **Unschedule** the job or **Edit** the scheduled job. You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

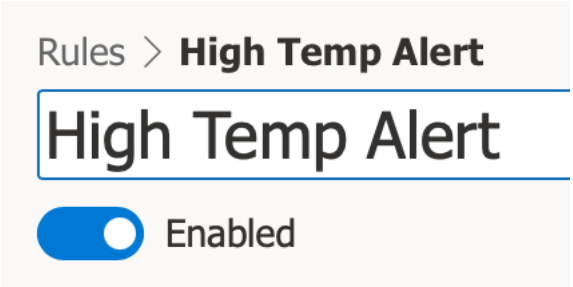
STEP 3: Set a Rule

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 temperature sensor of the WBZ451 Curiosity board surpasses the 30 degrees Celsius mark.

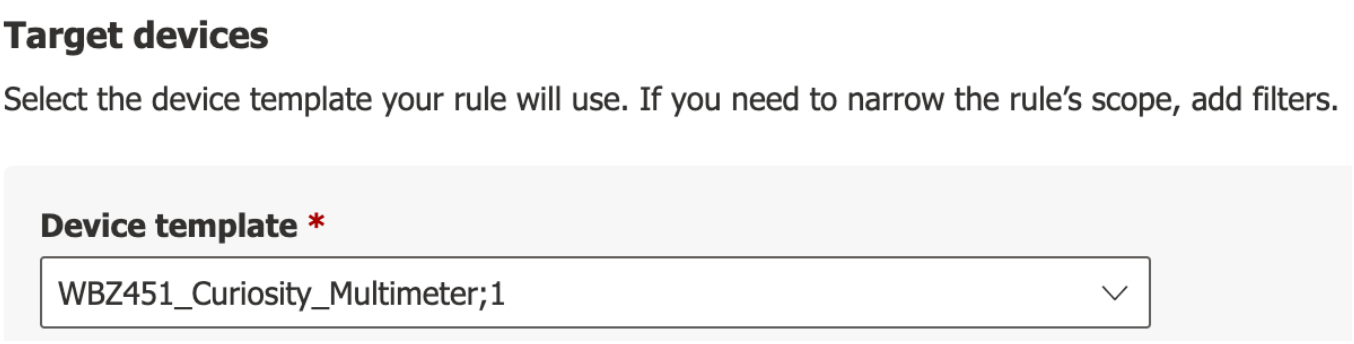
Step 3a: On the left-hand navigation pane under **Extend**, select **Rules**

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

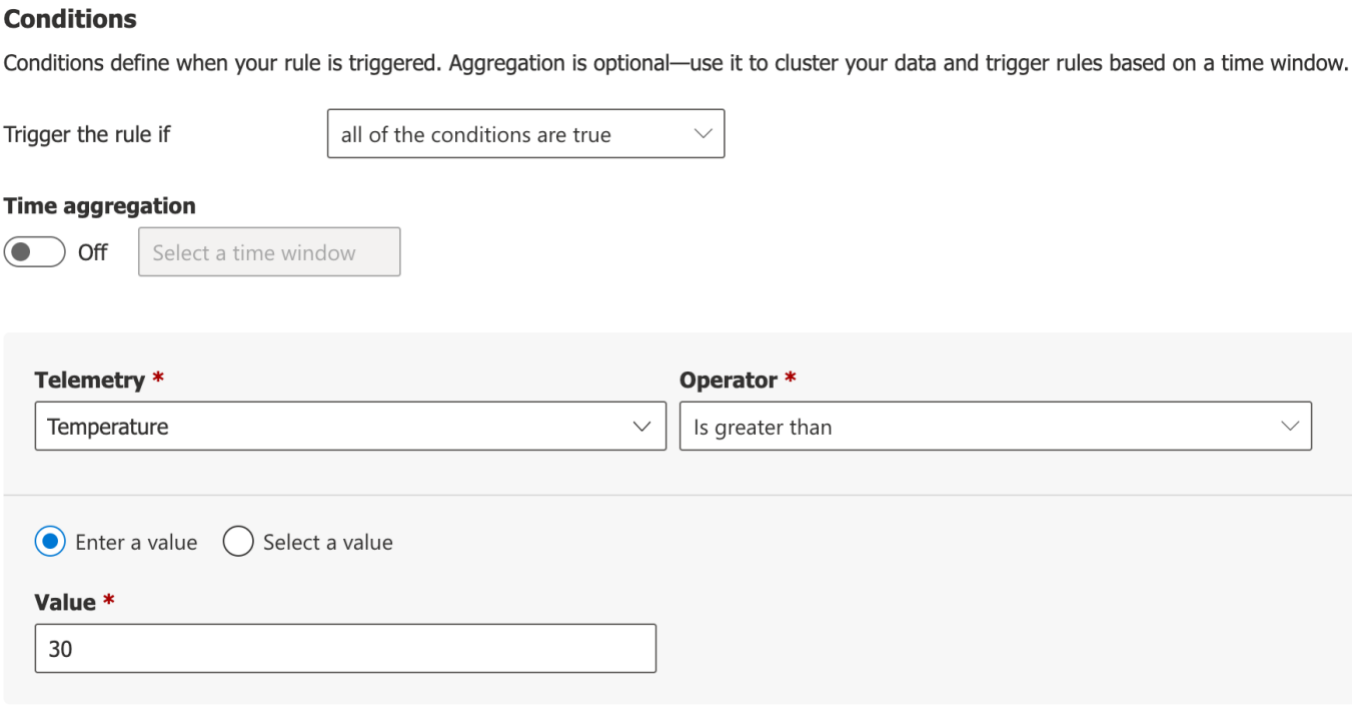
Step 3c: Enter a name for the new rule (e.g. “High Temp Alert”)



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

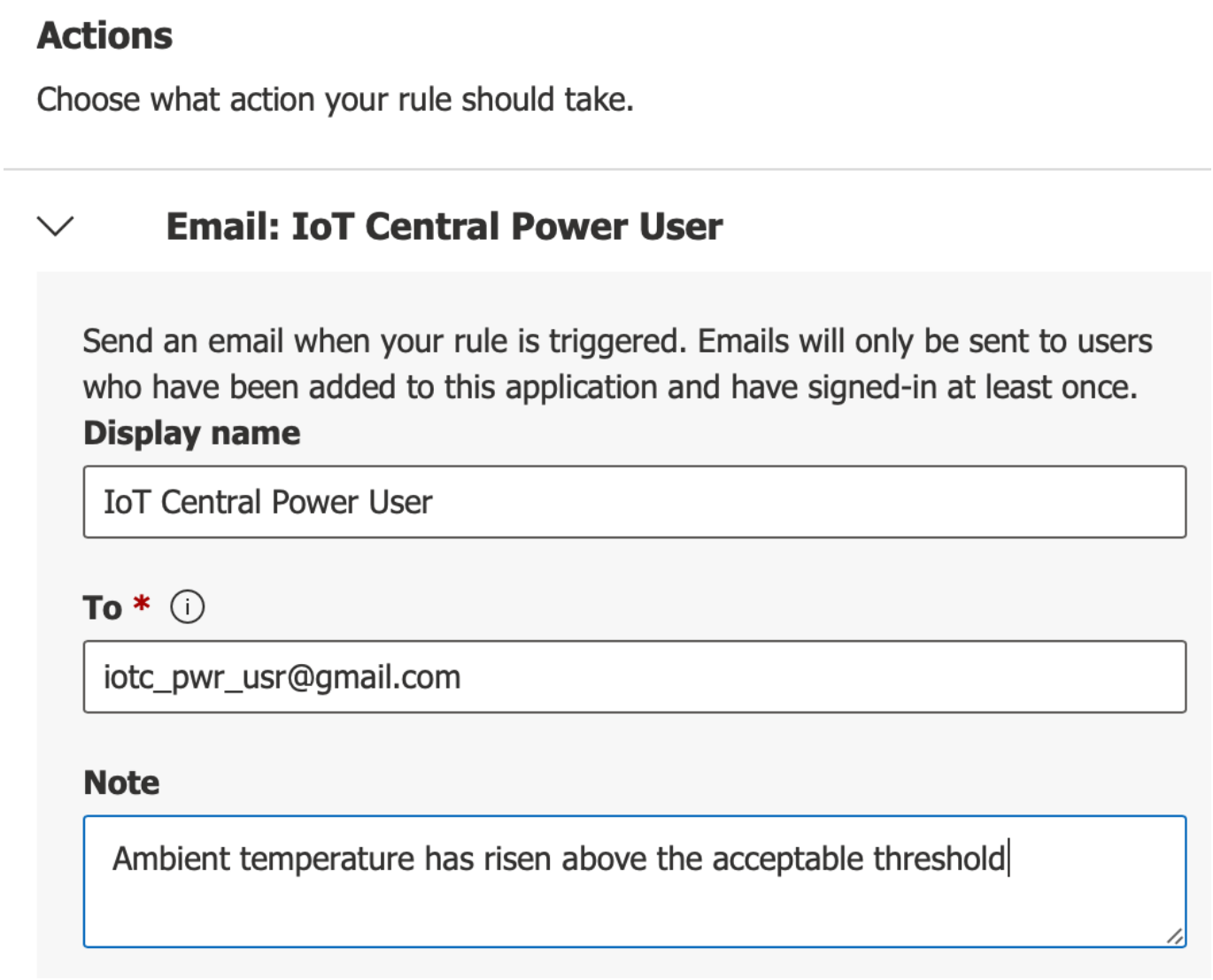


Step 3e: Set the conditions for generating the trigger (e.g. temperature sensor value, ***Temperature***, goes above 30 degrees Celsius)

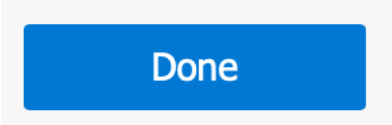


Step 3f: Set the appropriate action to take when triggered (click on the **+ Email** icon)

Step 3g: Enter the information for the e-mail message to be automatically generated



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



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

Step 3j: Using a "safe" heat source (e.g. hair dryer), warm up the WBZ451 Curiosity board and wait for the e-mail alert. Use the device's **Overview** tab to visually confirm that the last known value of **Temperature** has gone above 30 degrees Celsius prior to checking for reception of the e-mail message.

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.