IoT Central - Hands-on Lab

Scenario

Suppose you run a company that operates a fleet of refrigerated trucks. You have many customers within a city, and you operate from a base. You command each truck to deliver its contents to a customer.

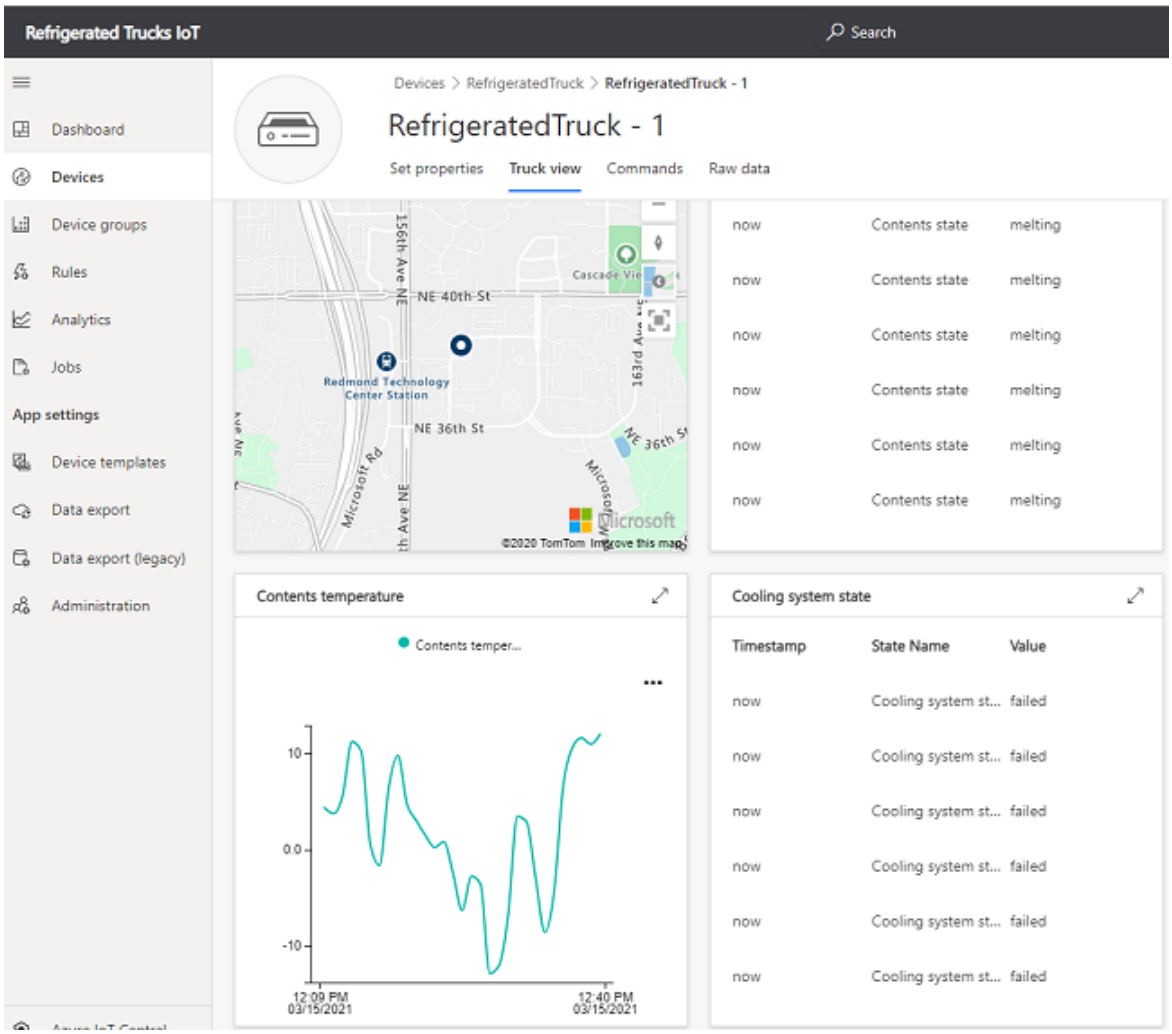
If the cooling system fails on a truck and the contents start to melt, you need to instruct the truck to return to base and unload the contents. Or you can instead deliver the contents to a customer who's nearby when the cooling system fails.

To make these decisions, you need an up-to-date picture of all that happens with your trucks. You need to know the location of each truck on a map, the status of the cooling system, and the status of the contents.

IoT Central provides all you need to handle this scenario. In the following image, for example, the colored circles show the location of a truck on its way to a customer.

Final Application

At the end of this hands-on lab, your newly-created application should look like the one below.



Tools Needed

- Azure subscription
- Visual Studio Code

Content - Hands-on Lab:

- Exercise 1: Create a Custom IoT Central app
  - Task 1: Creating an Application
  - Task 2: Add Capabilities - Telemetry
  - Task 3: Add Capabilities - Properties
  - Task 4: Add Capabilities - Commands

- Exercise 2: Create a Dashboard
  - Task 1: Visualizing the device
  - Task 2: Writable Properties View

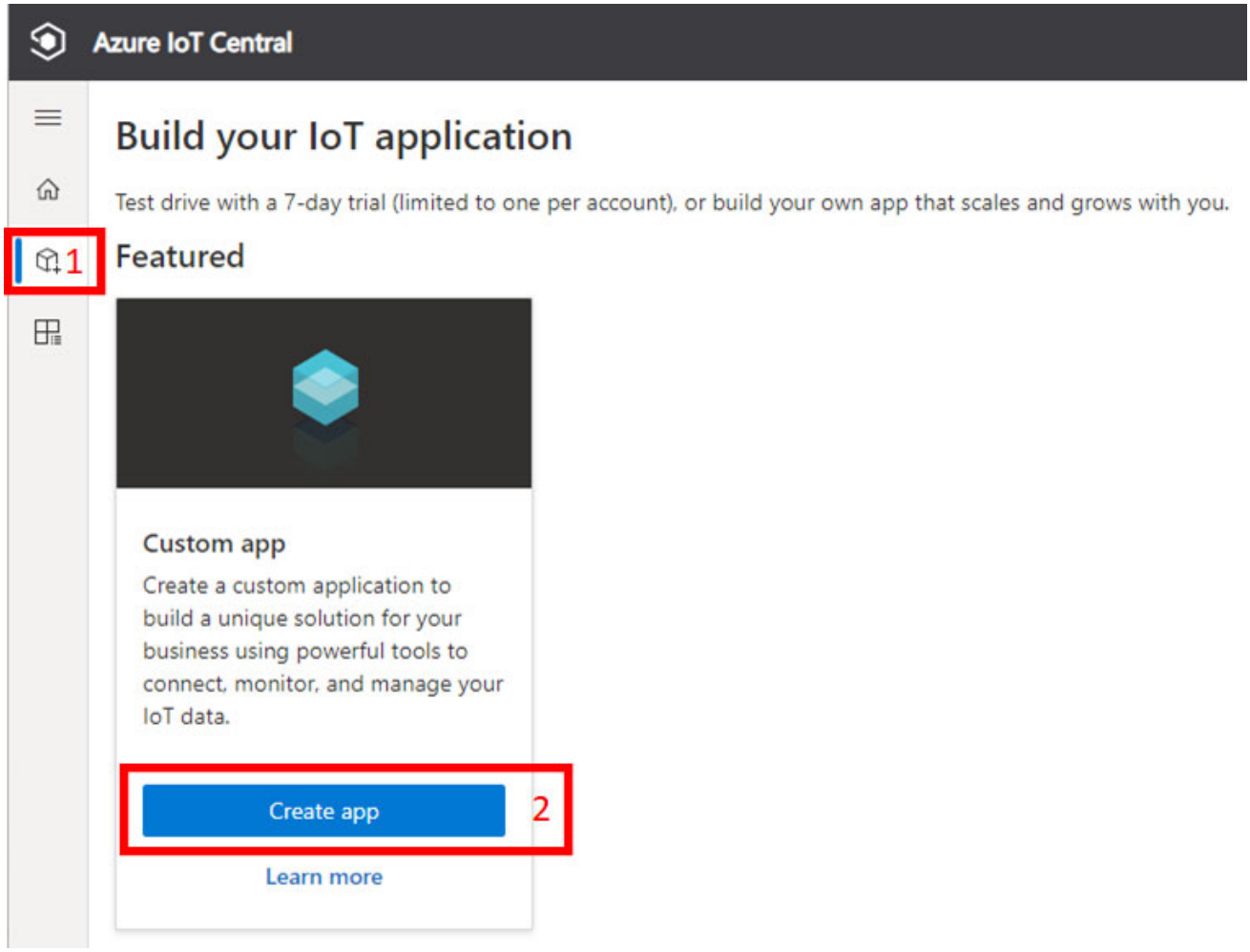
Exercise 1: Create a Custom IoT Central app

Task 1: Creating an Application

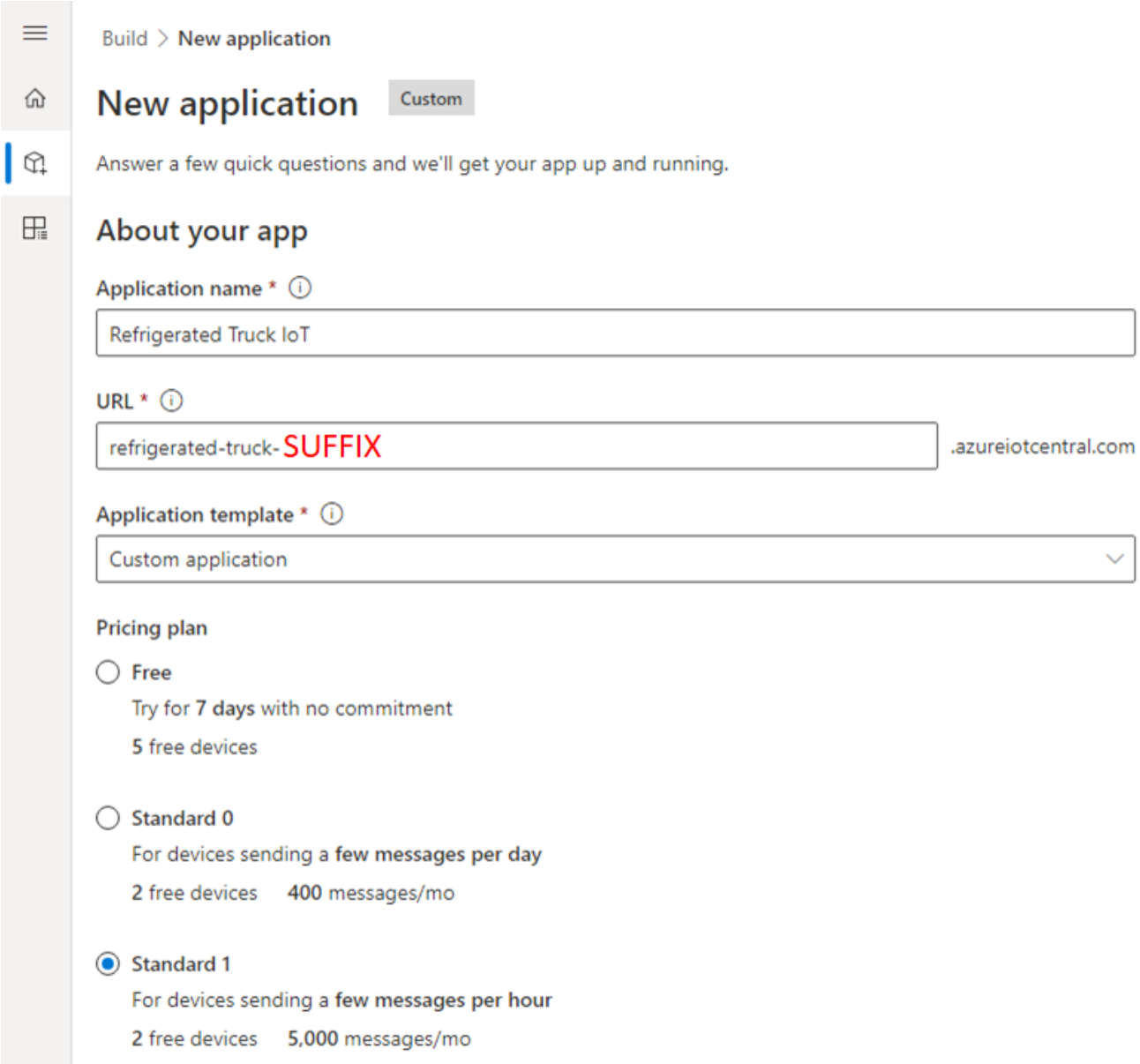
In your browser:

Log in to your Azure Portal using the same credentials you used to create your subscription with the Azure Pass: https://portal.azure.com/

Open Azure IoT Central: https://apps.azureiotcentral.com/



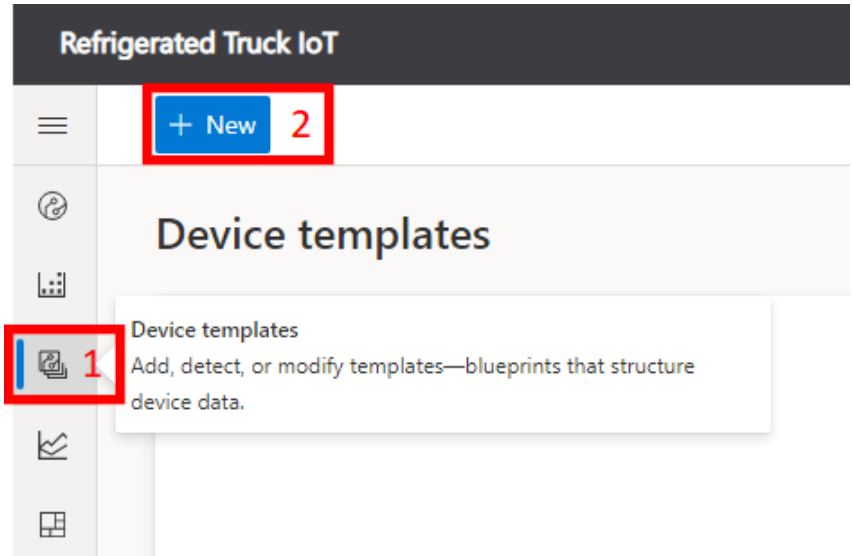
Selct the build icon on the left, then select Create app within Custom app. You should fill the fields in the Application Form:



- **Application Name:** Refrigerated Trucks
- **URL:** refrigerated-trucks-**SUFFIX** must be a unique URL
- **Application Template:** Custom application, default.
- **Pricing Plan:** Standard 1
- **Directory:** Directory where your subscription is located, typically "Default Directory" associated with your login.
- **Azure Subscription:** Your subscription
- **Location** Select the region you are using for this training.

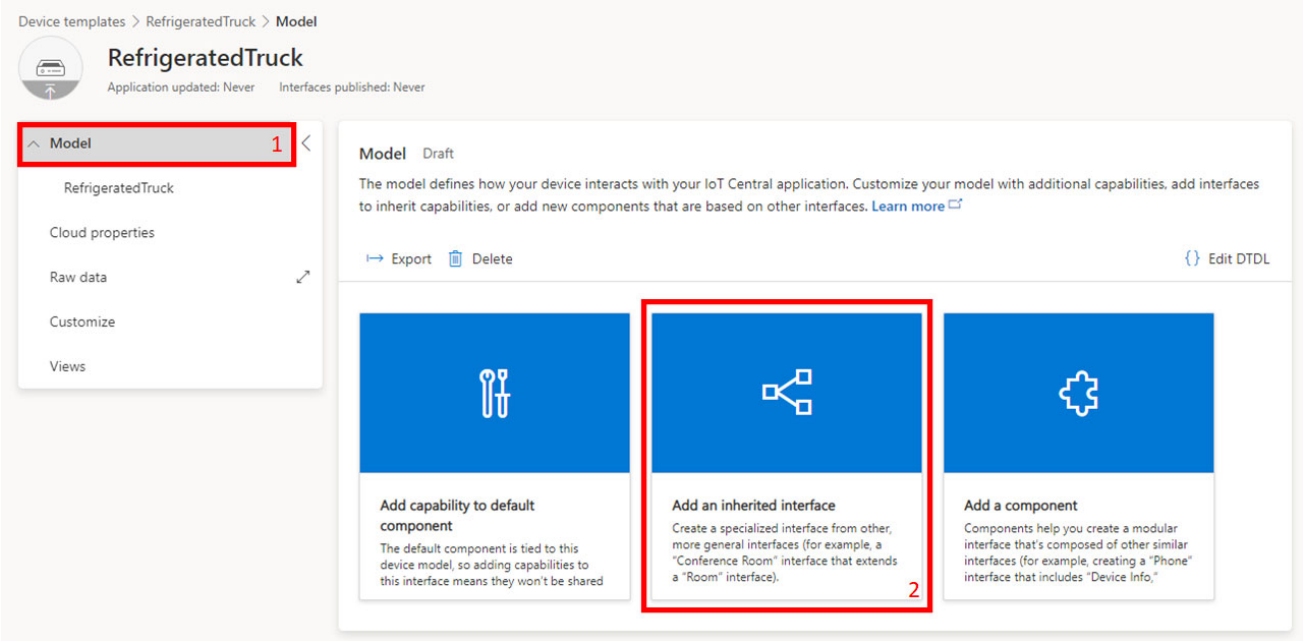
Then select **Create**

Once your Application is available the next step will be to **Create a device template**. On your left menu click on **Device Templates** and then in **New**

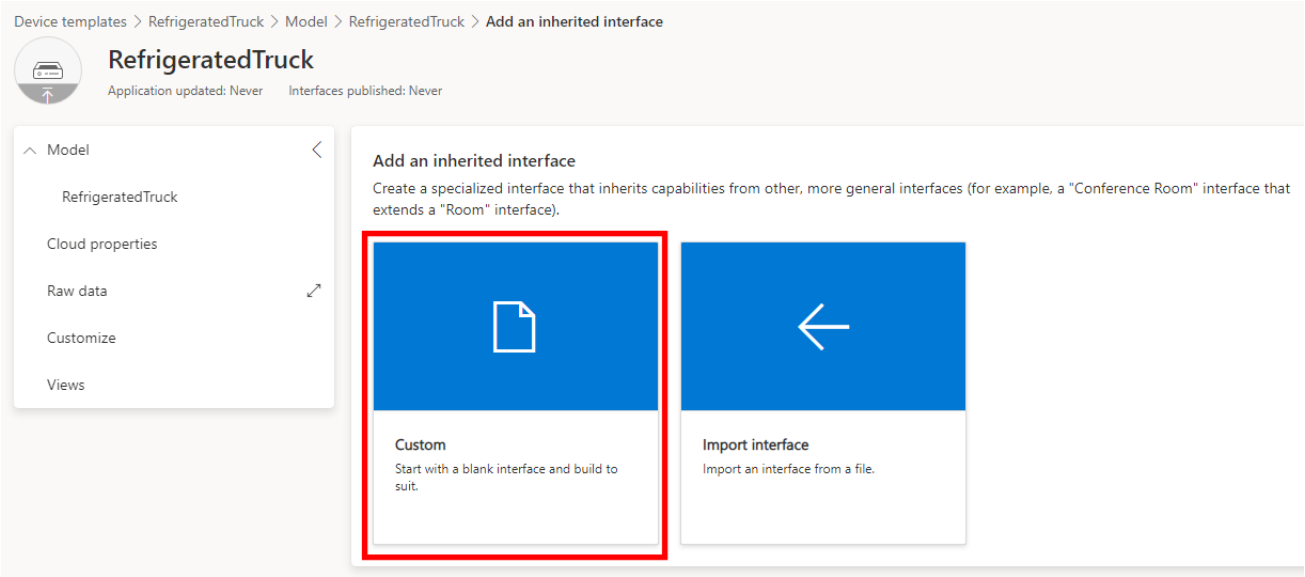


1. Select **IoT Device** then **Next: Customize**
2. In the customize screen assign a **Device Template name** RefrigeratedTruck
3. Don't select **Gateway device** box
4. Select **Next: Review**. Then select **Create**.
5. In the Create a model area, select Custom model. Your view should now look similar to the following image.

6. Click **Model**, then **Add an inherited interface**.



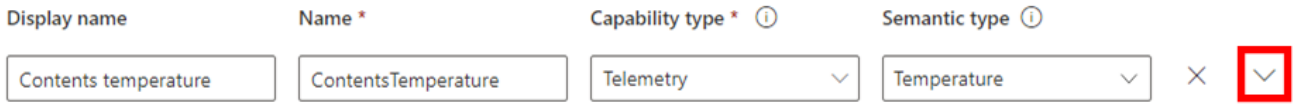
8. Then select **Custom** to start building from a blank interface



Task 2: Add Capabilities - Telemetry

Note: The interface names must be entered exactly as shown in this unit. The names and entries must exactly match in the code you'll add later in this module.

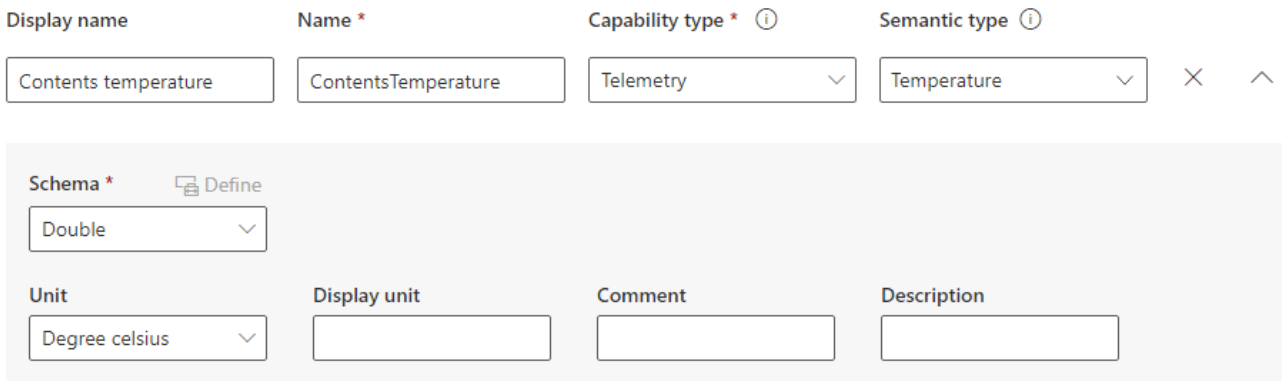
1. To get started, select **Add capability** and click the carat to show all the fields.



2. Enter the following values

Entry Summary	Value
Display Name	Contents temperature
Name	ContentsTemperature
Capability type	Telemetry
Semantic Type	Temperature
Schema	Double
Unit	Degree celsius

Ensure your capability looks like the following image:

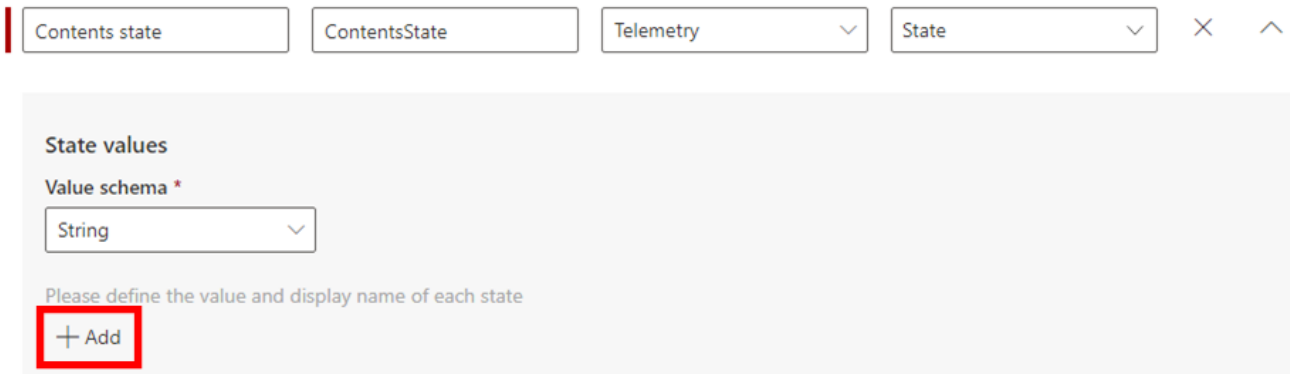


3. States are important. They let the operator know what's happening. A state in IoT Central is a name associated with a range of values. Later you'll choose a color to associate with each value.

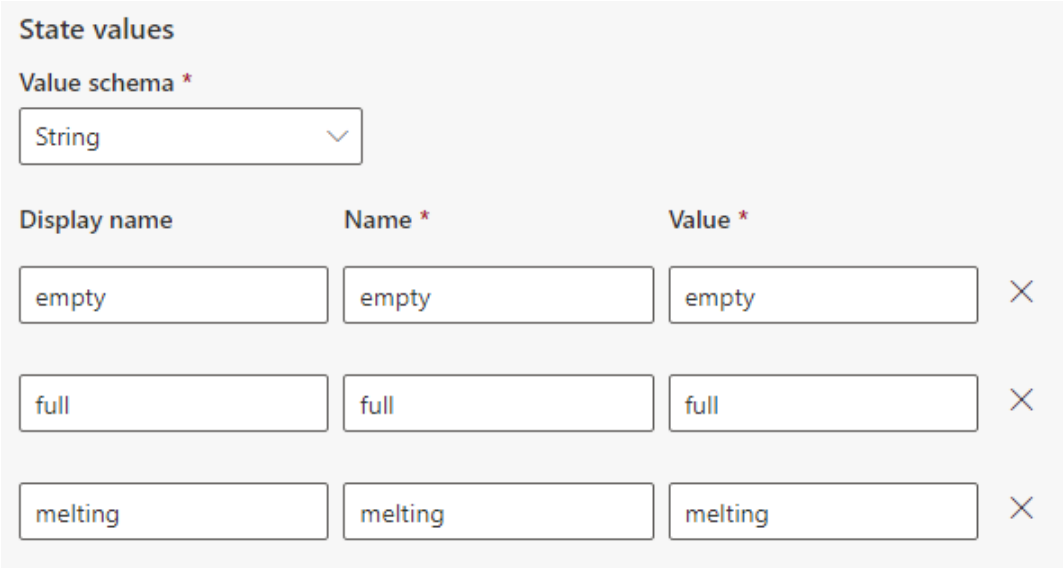
Use the **Add capability** control to add a state for the truck's refrigerated contents: **empty**, **full**, or **melting**.

Entry Summary	Value
Display Name	Contents state
Name	ContentsState
Capability type	Telemetry
Semantic Type	State
Schema	String

Select **Add**.



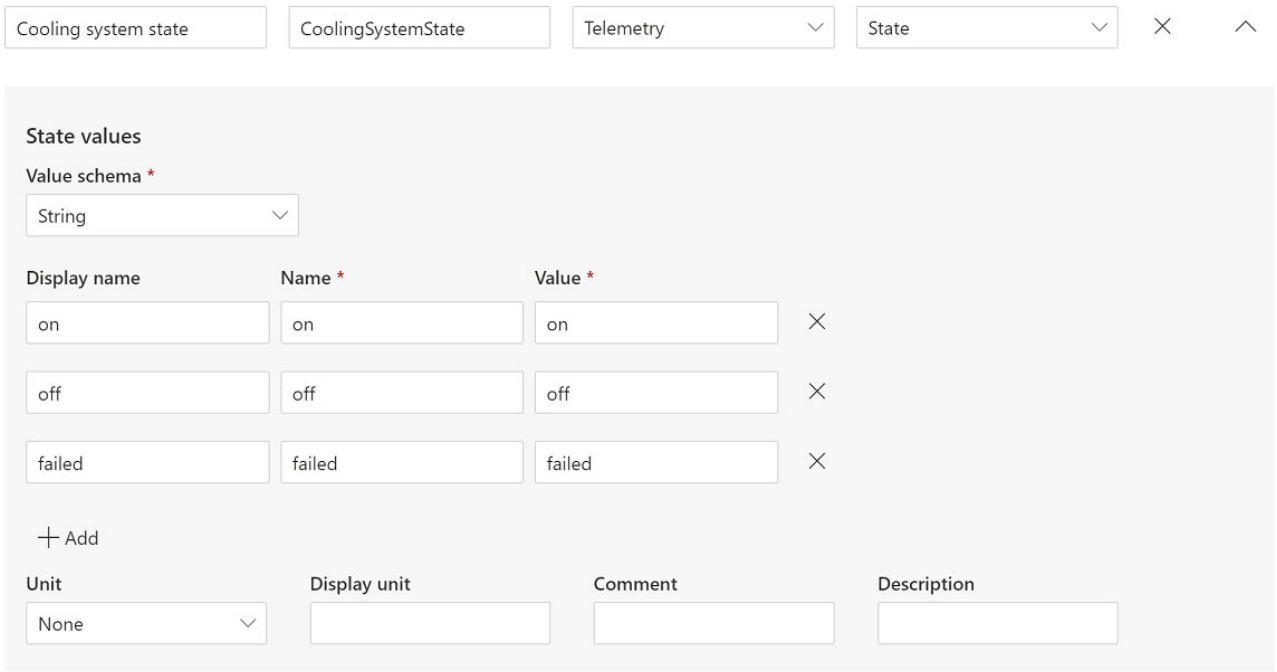
For Display name and Value, enter empty. The Name field should be populated automatically with empty. So all three fields are identical, containing **empty**. Add two more state values: **full** and **melting**. Again, the same text should appear in the fields for Display name, Name, and Value.



4. If the cooling system fails, as you'll see in the following units, the chances of the contents melting increase considerably.

Entry Summary	Value
Display Name	Cooling system state
Name	CoolingSystemState
Capability type	Telemetry
Semantic Type	State
Schema	String

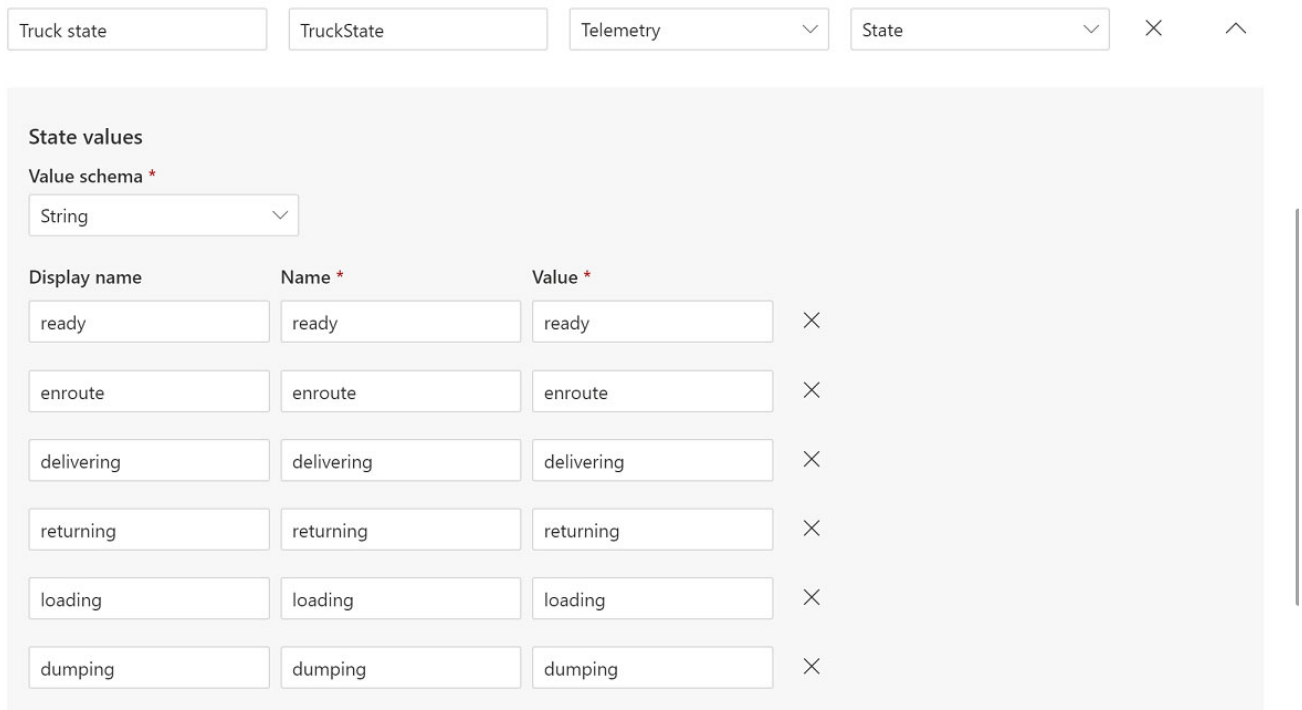
Add **on**, **off**, and **failed** entries for the cooling system. Start by selecting Add capability. Then add another state:



4. A more complex state is the state of the truck itself. If all goes well, a truck's normal routing might be ready, enroute, delivering, returning, loading, and back to ready again. Also add the dumping state to account for the disposal of melted contents! To create the new state, use the same process as for the last two steps.

Entry Summary	Value
Display Name	Truck state
Name	TruckState
Capability type	Telemetry
Semantic Type	State
Schema	String

Now add: **ready**, **enroute**, **delivering**, **returning**, **loading**, and **dumping** as shown below:



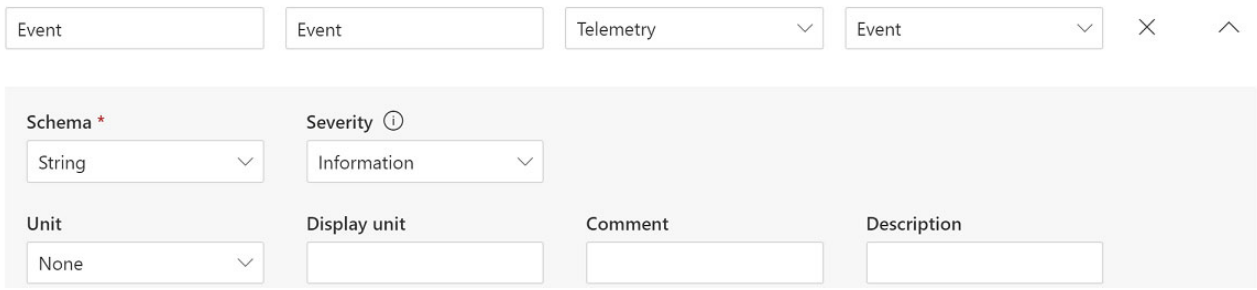
5. Add and Event Capability. One event a device might trigger is a conflicting command. An example might be when an empty truck that's returning from a customer receives a command to deliver its contents to another customer. If a conflict occurs, the device should trigger an event to warn the operator of the IoT Central app.

Another event might just acknowledge and record the customer ID that a truck is to deliver to.

To create an event, select **Add capability**. Then fill in the following information.

Entry Summary	Value
Display Name	Event
Name	Event
Capability type	Telemetry
Semantic Type	Event
Schema	String
Severity	Information

Your settings should look like the image below:



6. Add a Location capability with the following information:

Entry Summary	Value
Display Name	Location
Name	Location
Capability type	Telemetry
Semantic Type	Location
Schema	Geopoint

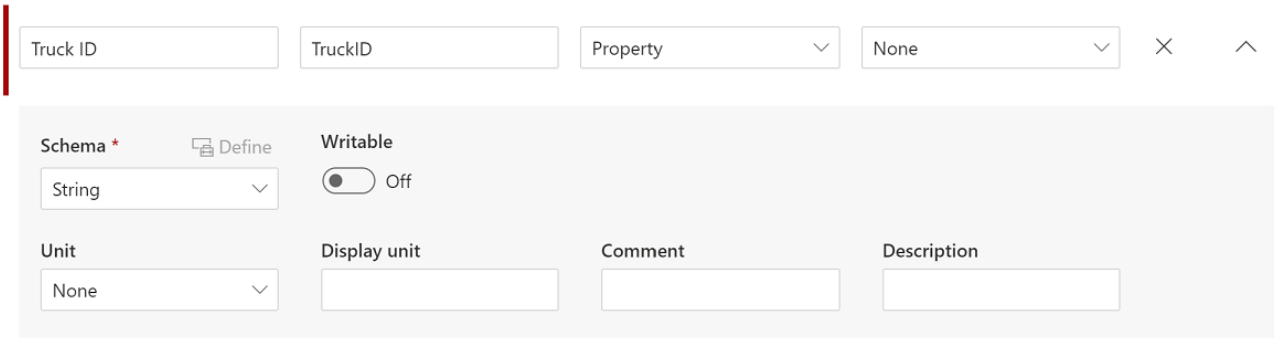
Task 3: Add Capabilities - Properties

You'll define an optimal temperature for the truck contents as a property.

1. Select Add capability. Then add the truck ID property.

Entry Summary	Value
Display Name	Truck ID
Name	TruckID
Capability type	Property
Semantic Type	None
Schema	String
Writable	Off
Unit	None

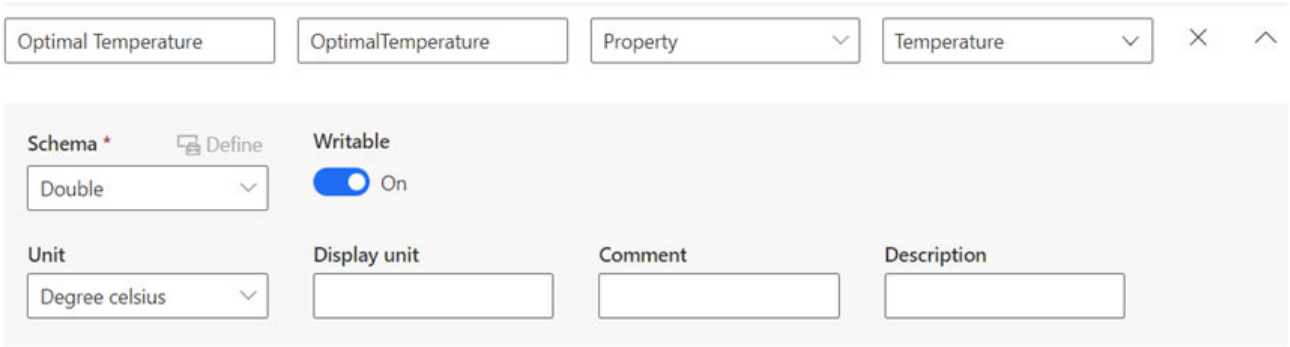
You should see your property set up as this one below:



2. Add the optimal temperature property.

Entry Summary	Value
Display Name	Optimal Temperature
Name	OptimalTemperature
Capability type	Property
Semantic Type	Temperature
Schema	Double
Writable	On
Unit	Degree celsius

Now, should look like the below image:



Task 4: Add Capabilities - Commands

For refrigerated trucks, you should add two commands:

A command to deliver the contents to a customer A command to recall the truck to base

1. To add the commands, select **Add capability**. Then add the first command.

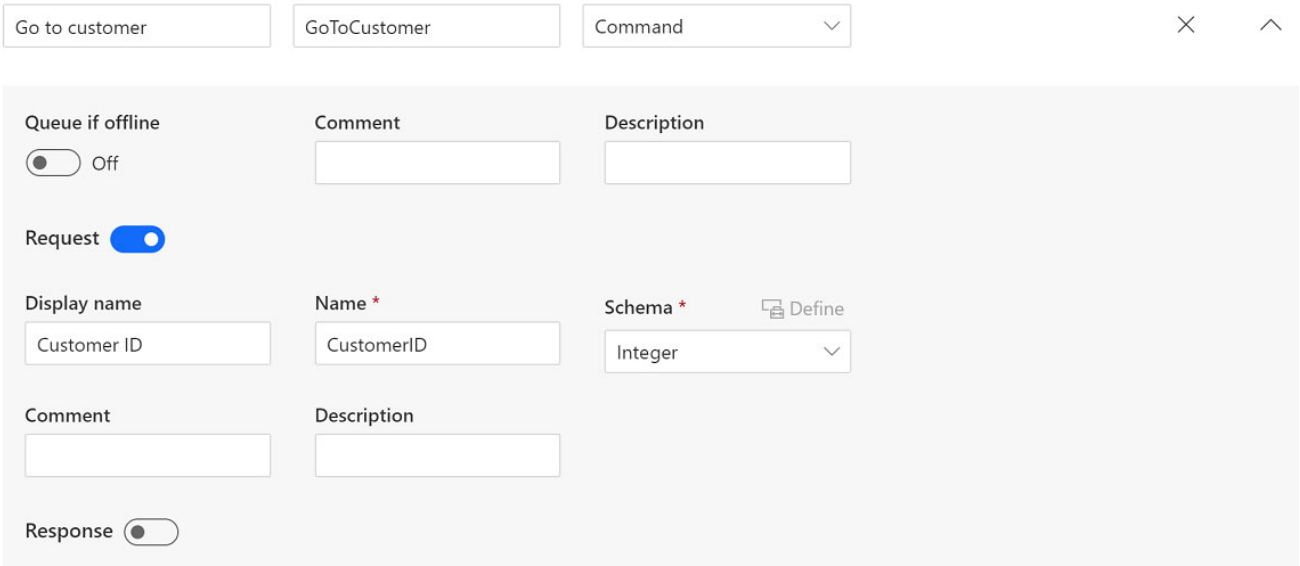
Entry Summary	Value

Entry Summary	Value
Display Name	Go to customer
Name	GoToCustomer
Capability type	Command

Turn on the **Request** option to enter more command details.

Entry Summary	Value
Request	On
Display Name	Customer ID
Name	Customer ID
Schema	Integer

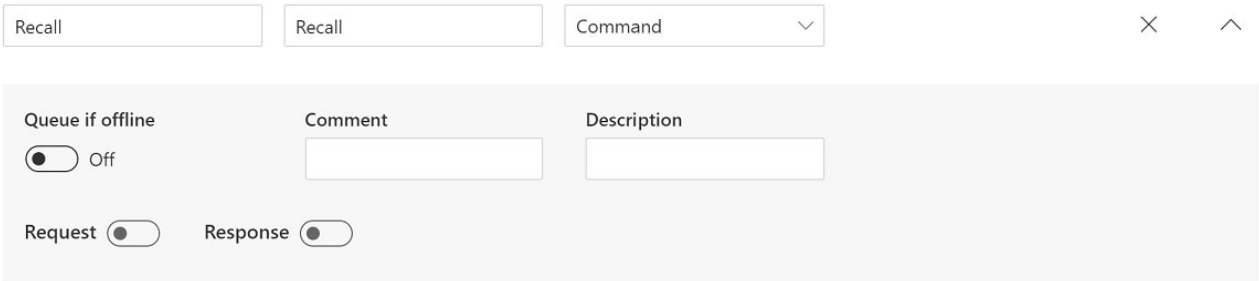
Validate your inputs with the below image:



2. Create a command to recall the truck.

Entry Summary	Value
Display Name	Recall
Name	Recall
Capability type	Command

Your recall property should look like the below one:

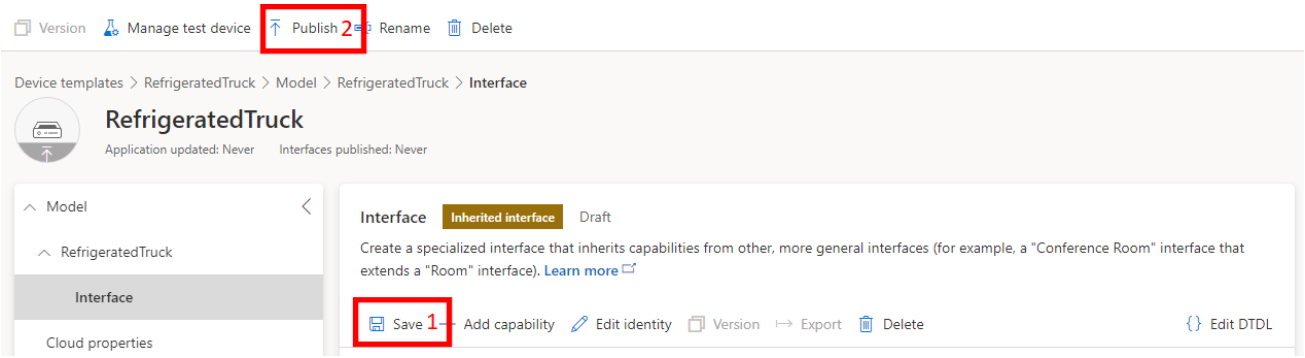


3. Before you go any further, carefully double-check your interface. After an interface is published, editing options are limited. So you should get it right before publishing.

When you select the name of the device template, the menu that ends with the Views option summarizes the capabilities, 6 Telemetry based, 2 Properties and 2 Commands:

Display name	Name *	Capability type *	Semantic type
Contents temperature	ContentsTemperature	Telemetry	Temperature
Contents state	ContentsState	Telemetry	State
Cooling system state	CoolingSystemState	Telemetry	State
Truck state	TruckState	Telemetry	State
Event	Event	Telemetry	Event
Location	Location	Telemetry	Location
Truck ID	TruckID	Property	None
Optimal Temperature	OptimalTemperature	Property	Temperature
Go to customer	GoToCustomer	Command	
Recall	Recall	Command	

4. Select **Save**, then **Publish**.

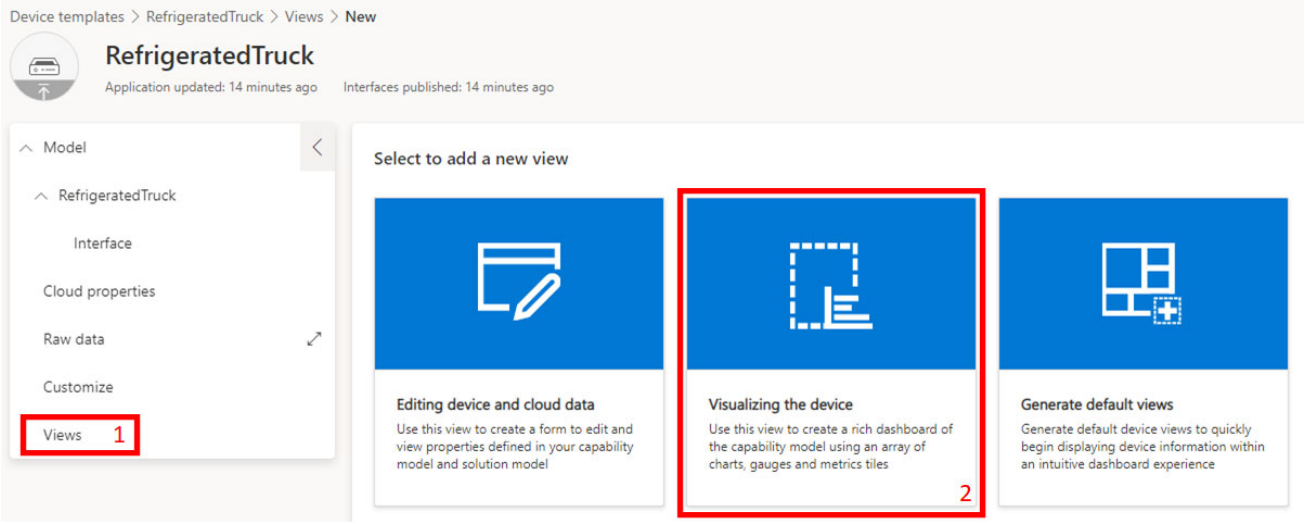


5. In the dialog box, select **Publish** again. The annotation should change from ***Draft*** to ***Published***.

Exercise 2: Create a Dashboard

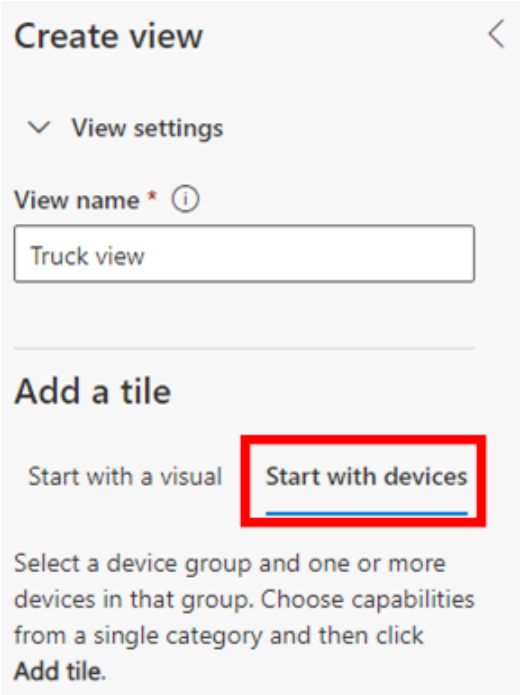
Task 1: Visualizing the device

1. Select **Views**. Then select **Visualizing the device**.

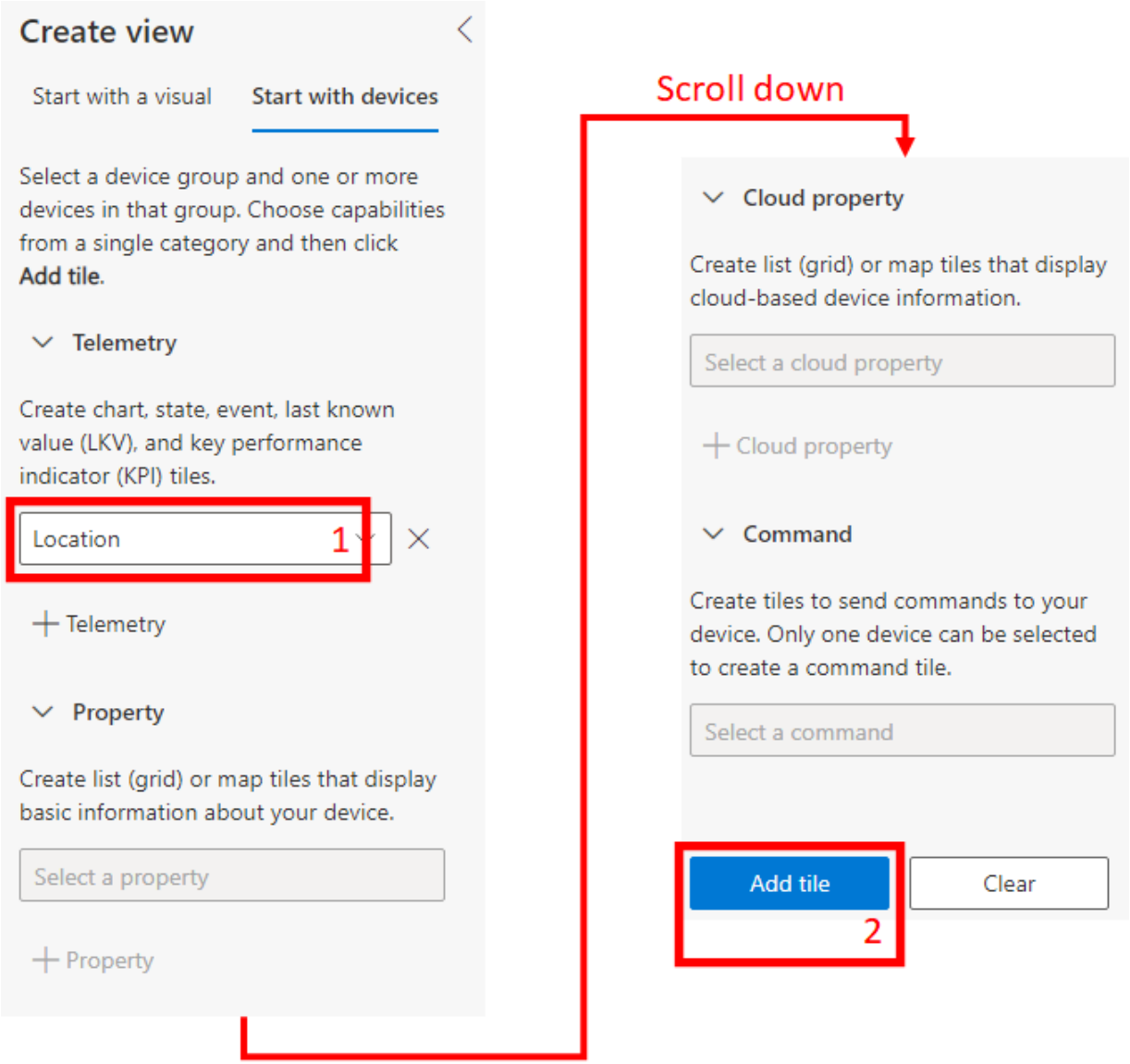


2. Change the View name to something more specific, for example, **Truck view**

3. Click **Start with devices** under **Add a tile**.



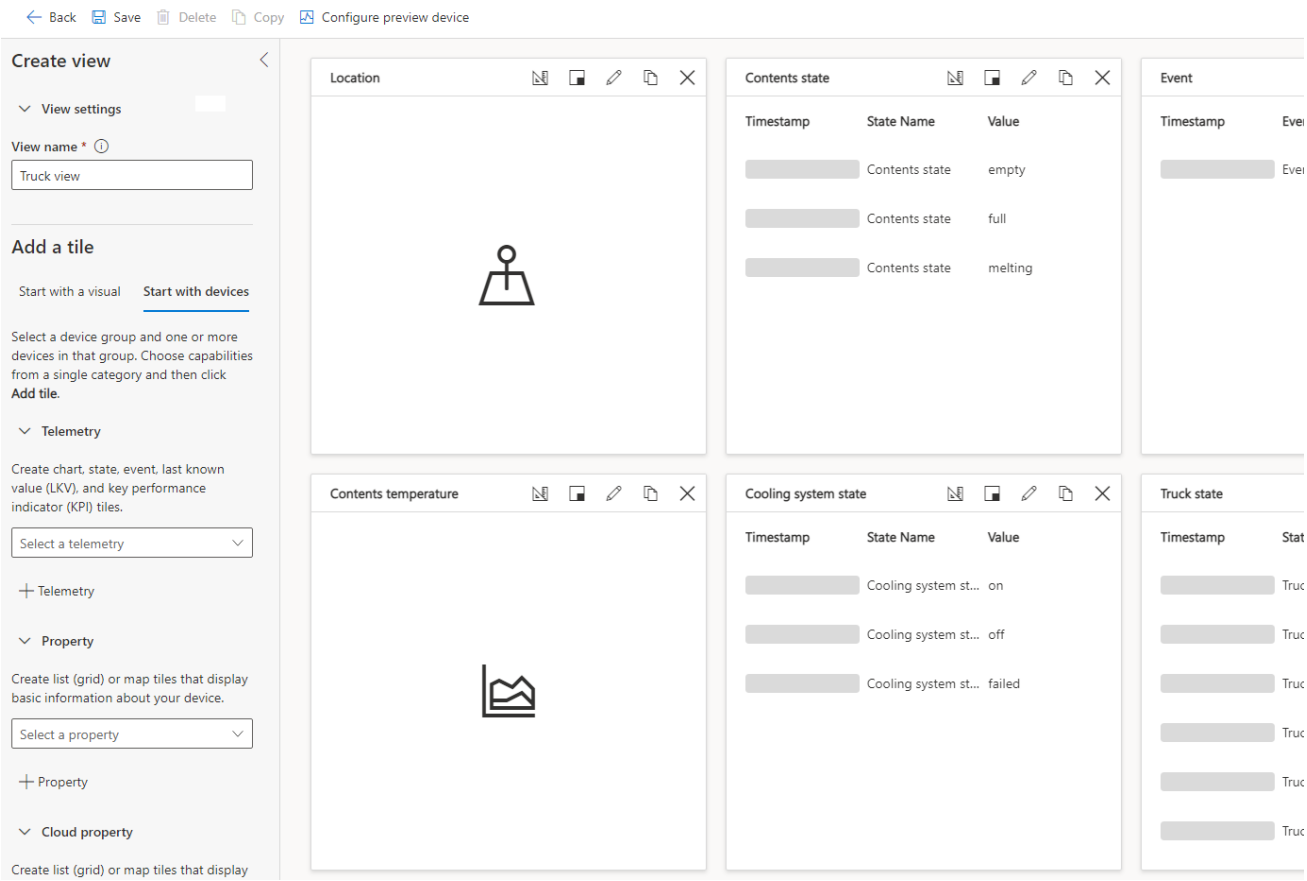
3. Under **Telemetry**, select **Location** in the dropdown, then scroll to the bottom to click **Add tile**. Dashboards are made of tiles. We choose the location tile first because we want to expand it.



4. Select each of the rest of the telemetry and property capabilities that you created, starting at the top. For each capability, select **Add tile** at the bottom.

Telemetry: Location, Contents state, Contents temperature, Cooling system state, Event, Truck state
Property: Optimal Temperature, Truck ID

Your new Dashboard should look like this one:

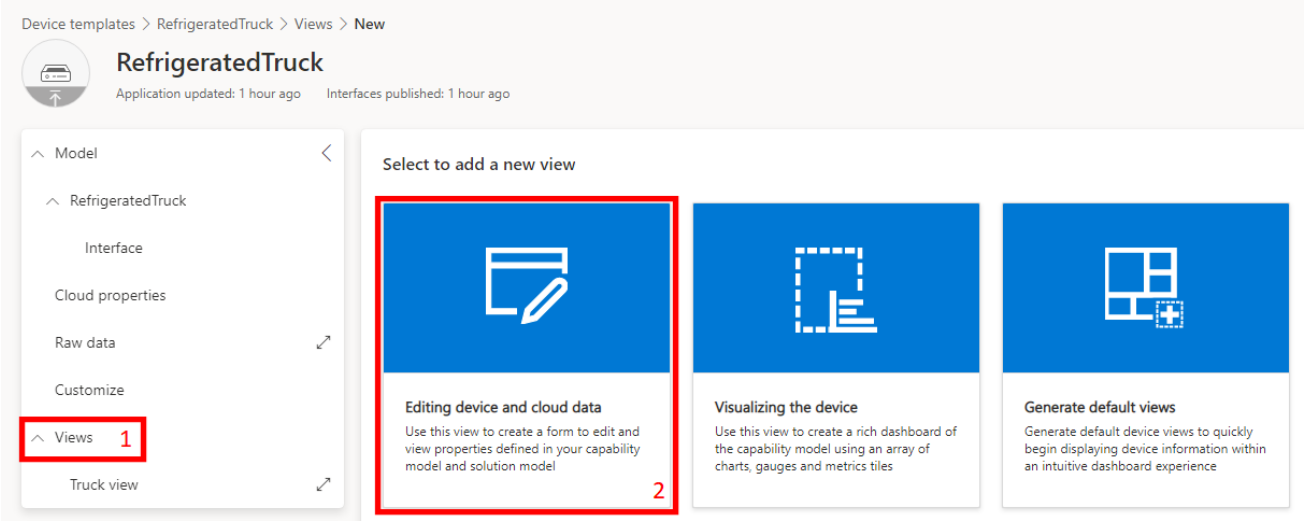


5. Click **Save** to save this view and **Back** to return to the device template.

Task 2: Writable Properties View

We need to create a separate view. Its sole purpose will be to set writable properties.

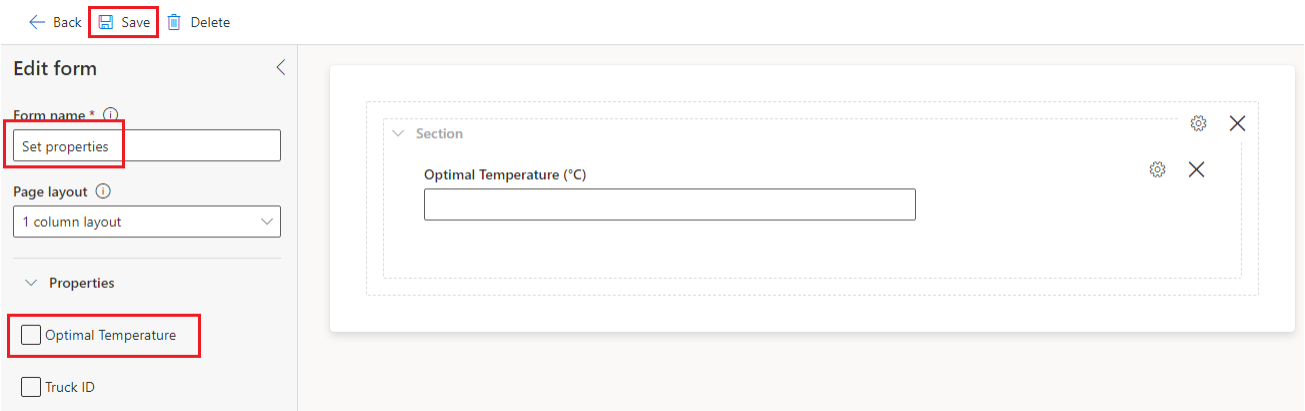
1. Select **Views**, and then select the **Editing device and cloud data** tile.



2. Change the form name to something like **Set properties**.

3. Select the **Optimal temperature** property check box. Then click **Add section**.

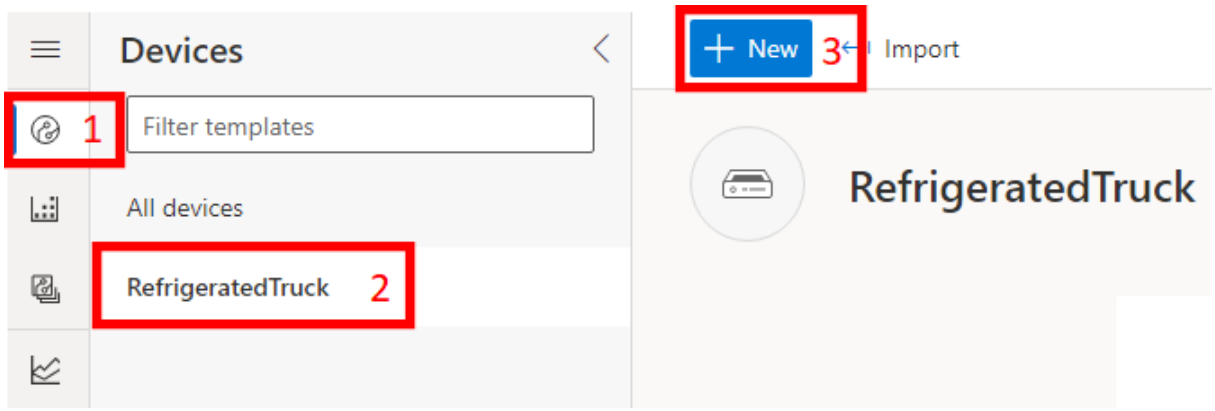
4. Verify that your view looks similar to the following image. Then click **Save** to save this view and **Back** to return to the device template.



5. Select **Publish**. Then in the dialog box, select **Publish** again.

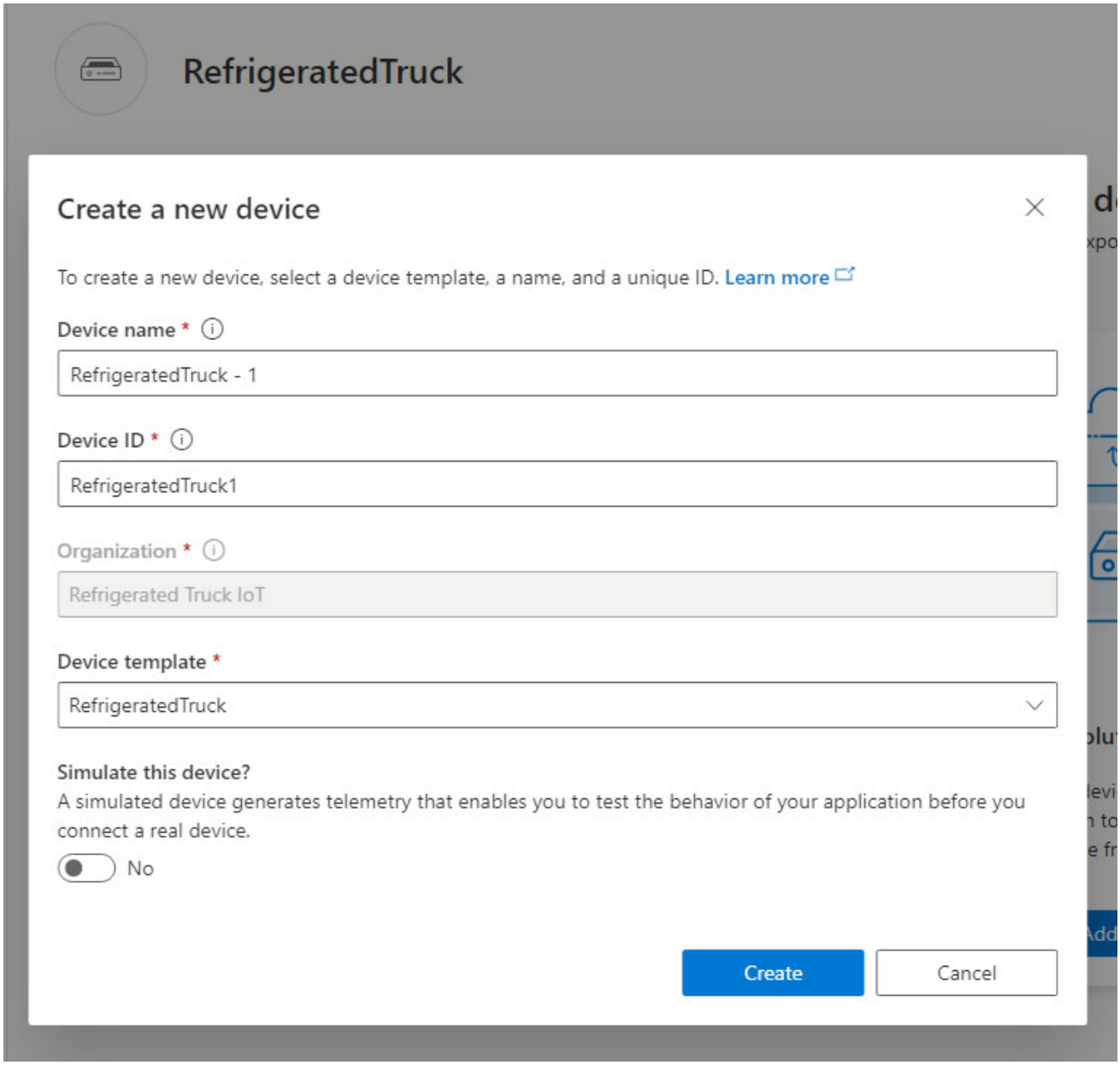
Task 3: Create a Device

1. On the menu on the left, click **Devices**, select **RefrigeratedTruck**, and click **New**.



2. In the Create a new device dialog box, verify that the device template is **RefrigeratedTruck**.

- **Device name**: RefrigeratedTruck - 1
- **Device ID**: RefrigeratedTruck1
- **Device template**: RefrigeratedTruck (default)
- **Simulate this device?**: No (default)



4. Click **Create**. Notice that the Device status is **Registered**. Only after the device status is **Provisioned** will the IoT Central app accept a connection to the device. The coding unit that follows shows how to provision a device.

5. Click **RefrigeratedTruck - 1**, then **Truck view** to see the live dashboard, where all the tiles will show *No data found* because we don't have any telemetry yet. On the bar that includes **Truck view**, click **Commands** where you will see the two commands you entered are ready to run.

6. In the upper-right, click **Connect**.

In the Device connection dialog box that opens, carefully copy the **ID scope**, **Device ID**, and **Primary key**. The ID scope identifies the app. The device ID identifies the real device. And the primary key gives you permission for the connection.

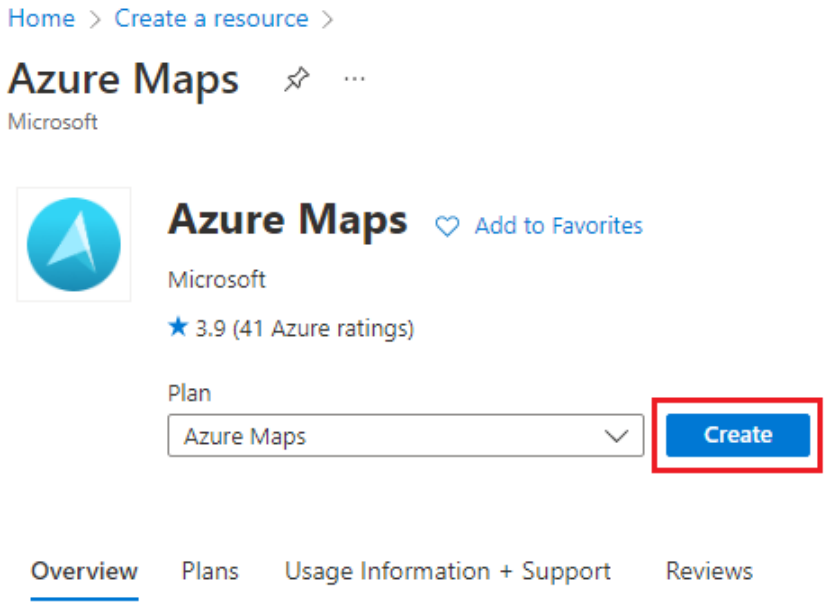
Paste this information in a text file.

Leave the Authentication type setting as **Shared access signature (SAS)**.

After you save the IDs and the key, select Close on the dialog box.

Exercise 3: Azure Maps

1. Go to Azure Portal: https://portal.azure.com/
2. Click **Create a Resource** then in the search box type **Azure Maps**. Open the Azure Maps service page and click **Create**.



Complete the creation form:

- **Subscription**: Select the subscription you are using for this training if it is not currently selected.
- **Resource group**: IOTC
- **Name**: mytrucksacademy**SUFFIX**
- **Region**: Select the region you are using for this training.
- **Pricing tier**: Gen1 (S1)
- **License and Privacy Statement** Checked

Then click **Review + create** at the bottom of the page, then click **Create** at the bottom of the review page.

Home > Create a resource > Marketplace > Azure Maps >

Create an Azure Maps Account resource ...

Basics | Identities | Tags | Review + create

Azure Maps is a collection of geospatial services and SDKs that use fresh mapping data to provide geographic context to web and mobile applications. Learn more

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * : Azure subscription 1

Resource group * : IOTC

Create new

Instance details

Name * : mytruckacademy SUFFIX

Region * : West US 2

Pricing tier * : Gen1 (S1)

View full pricing details

Terms

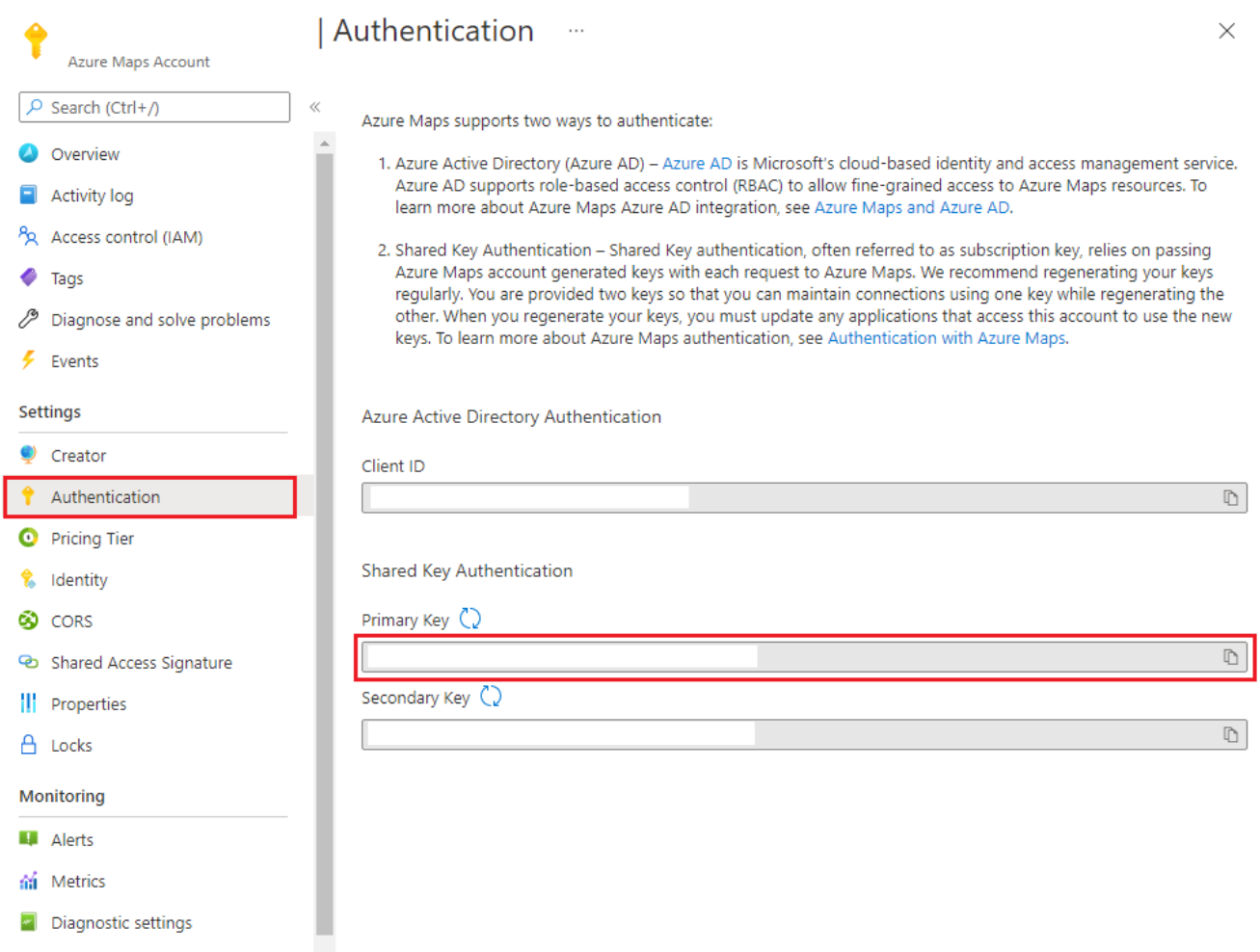
Azure Maps shares customer-provided address/location queries ("Queries") with third party TomTom for mapping functionality purposes. Queries are not linked to any customer or end-user when shared with TomTom and cannot be used to identify individuals. Microsoft is currently in the process of adding TomTom to the Online Services Subcontractor List.

Learn more about Preview

I confirm that I have read and agree to the License and Privacy Statement. ☑

Review + create | < Previous | Next : Identities >

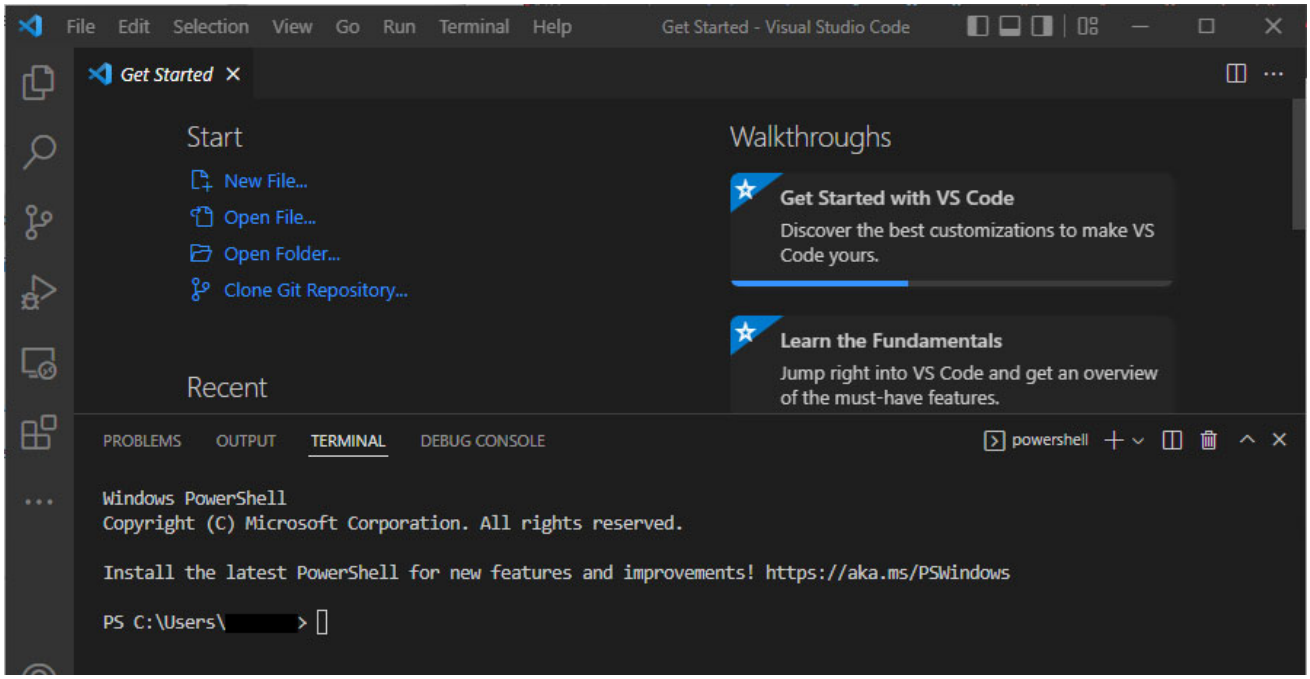
Once Azure Maps resource is created, click **Go to resource** then find the key on the **Authentication** blade. Copy the **Primary key** and paste it into your notepad.



Exercise 4: Create the device app

Task 1: Set up your environment

1. Open Visual Studio Code locally
2. On the top bar select **Terminal** and then **New Terminal** in Visual Studio Code.
3. Please make sure you are in the local directory where you want to create a new directory. (cd.. to change directory location locally)



637 lines (375 sloc) | 23 KB

4. Run the following commands to create a directory, set up a dotnet environment, and install required libraries:

```
mkdir RefrigeratedTruck
cd RefrigeratedTruck
dotnet new console
dotnet restore
dotnet add package AzureMapsRestToolkit
dotnet add package Microsoft.Azure.Devices.Client
dotnet add package Microsoft.Azure.Devices.Provisioning.Client
dotnet add package Microsoft.Azure.Devices.Provisioning.Transport.Mqtt
dotnet add package System.Text.Json
```

5. From the File menu, open the Program.cs file just created. On the github repo month 1/day 1 there is a folder titled Code-Sample: https://github.com/AzureIoTGBB/iot-academy/tree/main/Month_1/Day_1/Code_sample Copy this content from the Prgram.cs file and paste into your Visual Studio code Program.cs file. This will replace the whole content.

6. Once you replace the content of the files, we need to add our keys to connect with our services. Look for lines **123** to **126**. Replace accordingly based on the keys you were adding to your notepad in previous exercises.

```
119        static string GlobalDeviceEndpoint = "global.azure-devices-provisioning.net";
120        static TwinCollection reportedProperties = new TwinCollection();
121
122        // User IDs.
123        static string ScopeID = "<your Scope ID>";
124        static string DeviceID = "<your Device ID>";
125        static string PrimaryKey = "<your device Primary Key>";
126        static string AzureMapsKey = "<your Azure Maps key>";
127
128        static double Degrees2Radians(double deg)
```

After the changes are made, save the file. Click **File - Save **

Task 2: Launch your device

1. To begin testing, first open the Azure IoT Central app in a browser: https://app.azureiotcentral.com/ Click **My apps** on the left. Click the **Refrigerated Truck IoT** tile.

2. In the VS Code terminal, start the device app using the following command:

```
dotnet run
```

A console screen opens with the message Starting Truck number 1.

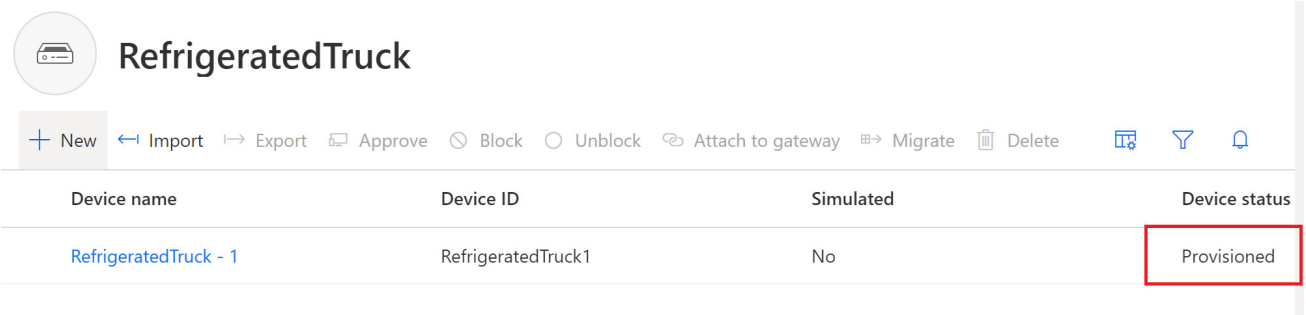
```
TERMINAL   PROBLEMS   OUTPUT   DEBUG CONSOLE                                  2: dotnet

Register device...
RegistrationID = RefrigeratedTruck1
ProvisioningClient RegisterAsync...Assigned
Device successfully connected to Azure IoT Central

Sent device properties: Truck number 1

Register settings changed handler...Done
Telemetry data: {"ContentsTemperature":-3.05,"TruckState":"ready","CoolingSystemState":"on","ContentsState":"full","Location":{"lon":-122.130137,"la
t":47.644702} "Event":"none"}
```

Once your device in registered through VS Code, you should see in your IoT Central an status change to **Provisioned**:



At this point in the Truck View dashboard in IoT Central you should see data flowing thorught it, the map should show a blue dot with your truck and the chart receiving telemetry data should show some data points already.

3. Select the device's **Commands** tab. This control should be under the truck name, to the right of the Truck view control.

4. Enter a customer ID, say **1**. (Numerals 0 through 9 are valid customer IDs.) Then select **Run**.

In the console for the device app, you see both a New customer event and a Route found message

```
Telemetry sent 3:02 PM

Telemetry data: {"ContentsTemperature":12,"TruckState":"ready","C
,"lat":47.644702},"Event":"none"}
Telemetry sent 3:02 PM

Route found. Number of points = 673

Telemetry data: {"ContentsTemperature":12,"TruckState":"enroute",
0048344247,"lat":47.64604},"Event":"New customer: 1"}
Telemetry sent 3:02 PM

Telemetry data: {"ContentsTemperature":12,"TruckState":"enroute",
0997706499,"lat":47.641170152900116},"Event":"none"}
Telemetry sent 3:02 PM
```

5. On the dashboard's Location tile, check to see whether your truck is on its way. You might have to wait a short time for the apps to sync.

6. Verify the event text on the Event tile. You should see a new Customer Event.

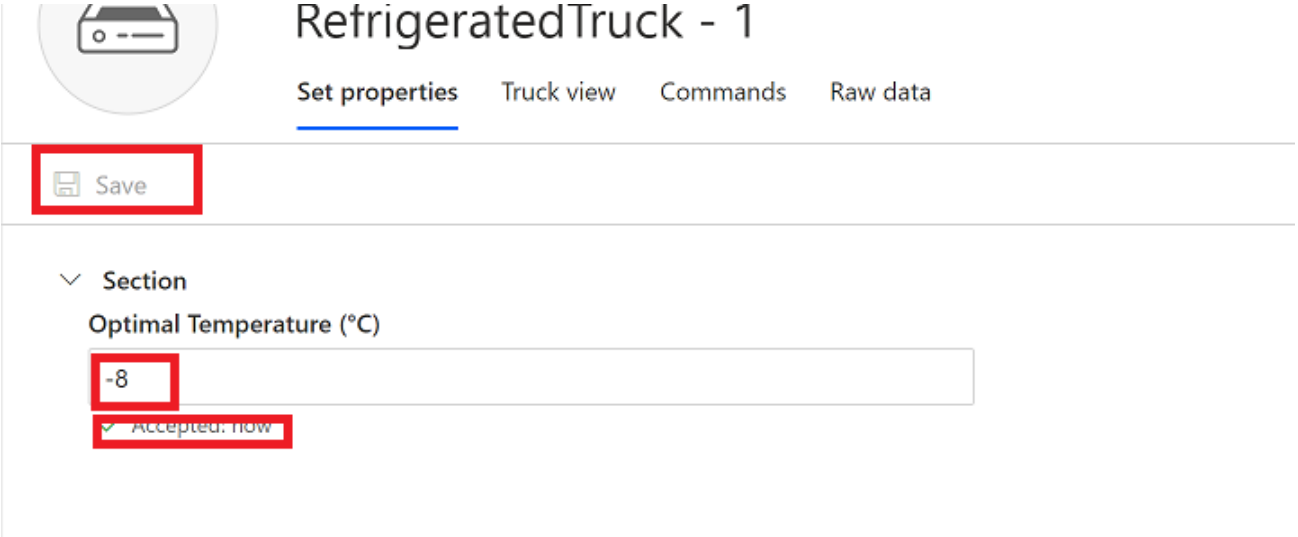
7. When the truck returns to base and is reloaded with contents, its state is ready. Try issuing another delivery command. Choose another customer ID.

8. Before the truck reaches the customer, make a recall command to check whether the truck responds.

Task 3: Set up Properties

The next test is to check the writable property, **OptimalTemperature**. To change this value, select the **Set properties** view.

Set the optimal temperature to any value, say **-8**. Select **Save** and then notice the Pending status.

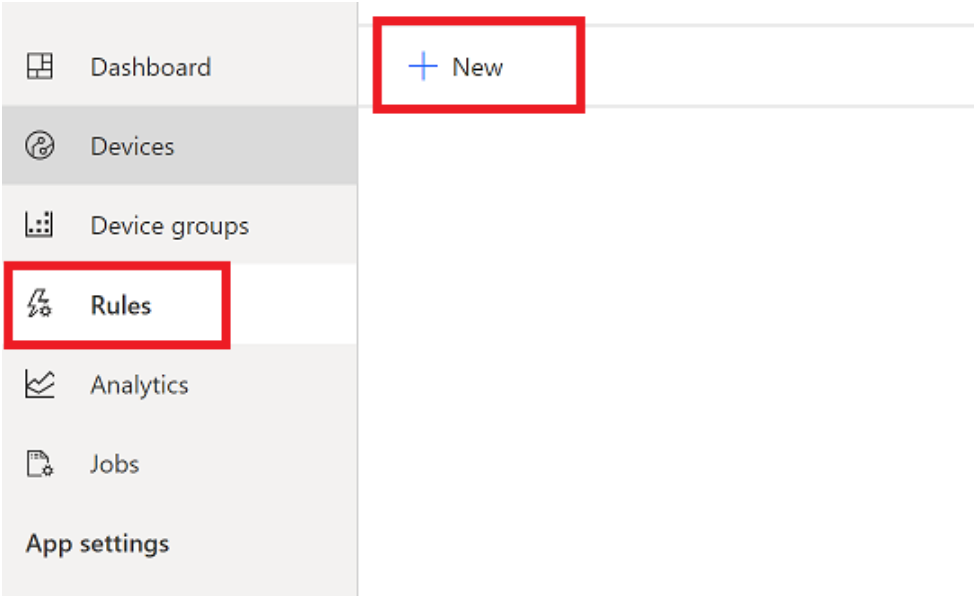


Now you should see the new Optimal temperature is set to -8. in the **Optimal Temperature** Tile.

Exercise 5: Create Rules

Task 1: Cooling system state

1. In the IoT portal, select **Rules** in the left-hand menu, then + **New**. Enter a meaningful name for the rule, such as "**Cooling system failed**". Press Enter.



2. Select **RefrigeratedTruck** for the **device template**.

3. Under **Conditions** notice that all the telemetry elements of the device template are available. Select **Cooling system state**.

For Operator, select **Equals**.

For value, type the word "**failed**", then click on Select: "failed".

Leave Time aggregation as Off.

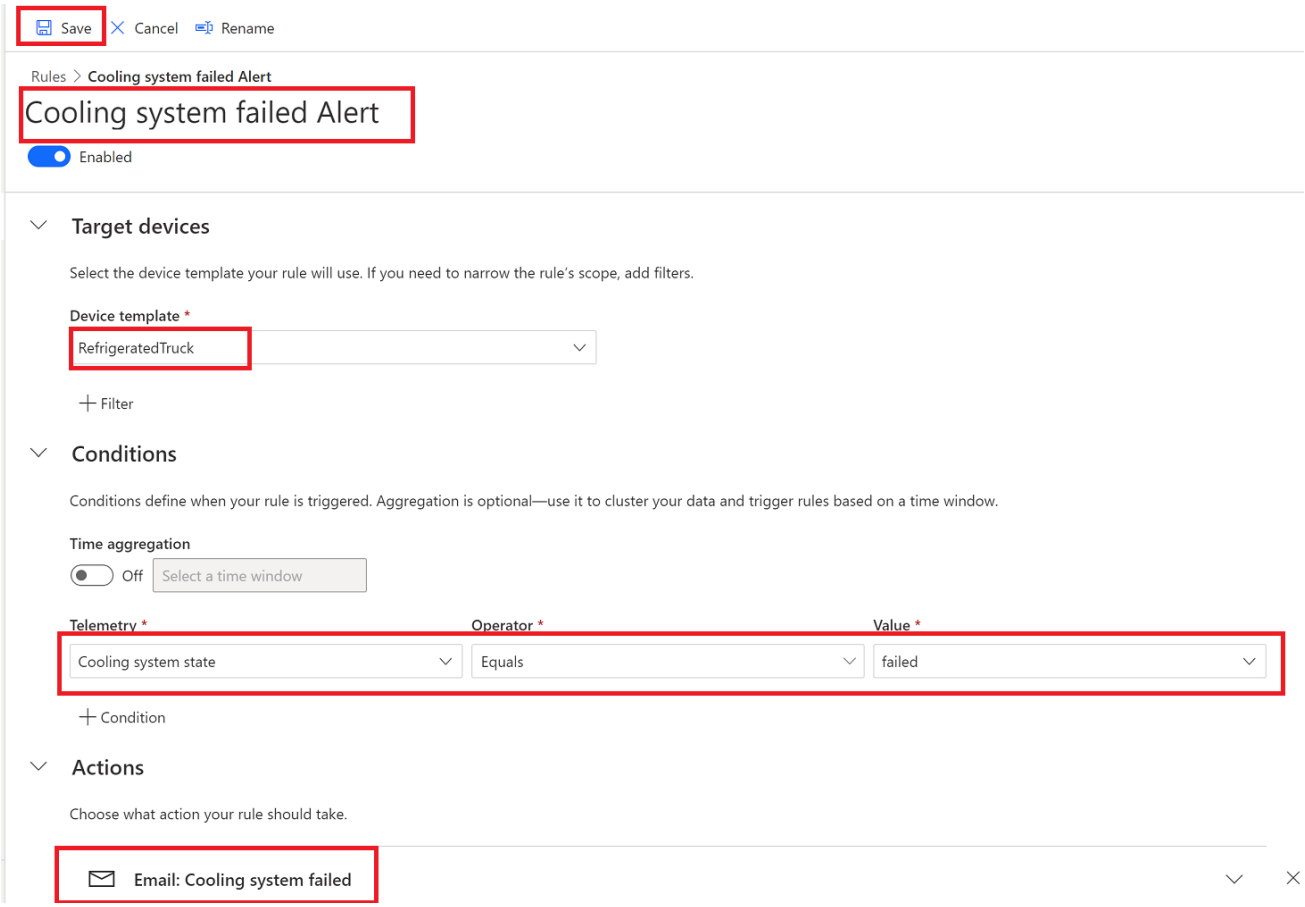
For **Actions**, click on + **Email**.

In Display name, enter a title for the email, perhaps "Cooling system failed!"

For To, enter the email you've used for your IoT Central account. And for Note enter some descriptive text that will form the body of the email.

Note: To receive emails the account you select has to be login to IoT central at least one time, otherwise you will not receive any emails.

Your new rule should look like the below image.

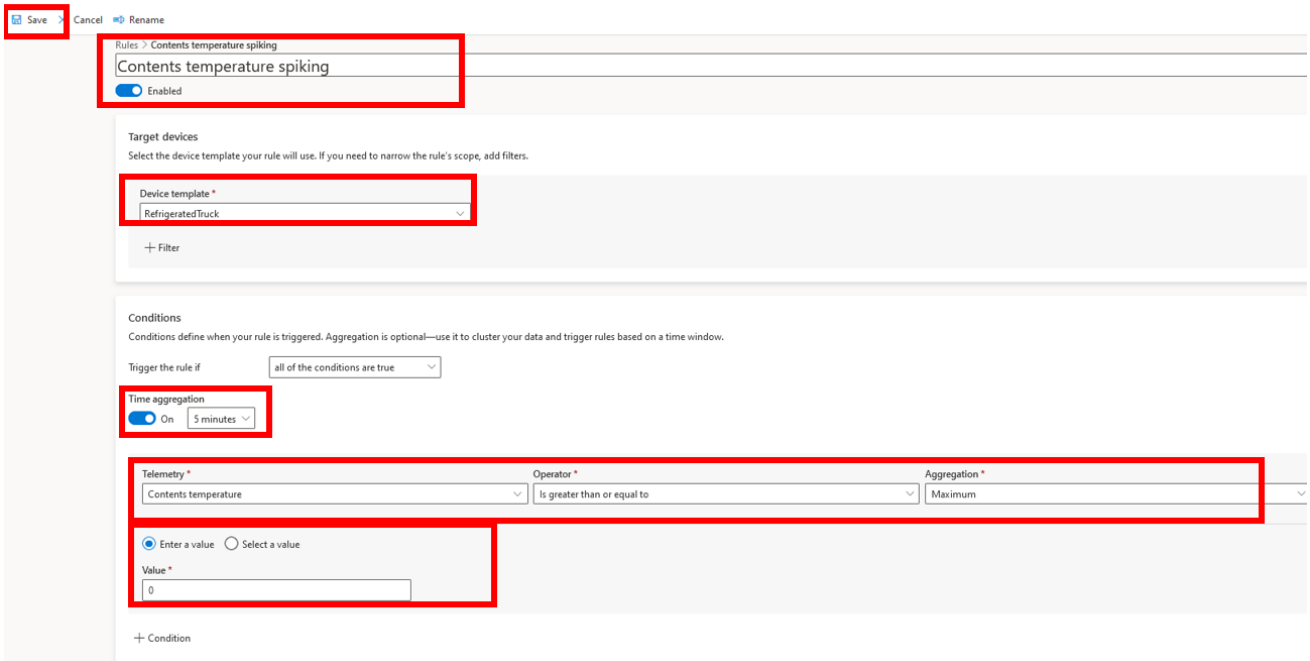


Task 2: Temperature spiking

1. Create a new rule with a name such as **"Contents temperature spiking"**.
2. Turn on **Time aggregation**, and select an interval of **5 minutes**.
3. Select **Contents Temperature** for Telemetry.
4. In the range Aggregation values, select **Maximum.**
5. For Operator. select Is greater than or equal to. Then enter **"0"** for the value, and select that as the value.

6. For Actions, fire off another email. Give the email an appropriate title and note.

7. Make sure to click Save, to save off this rule.



Task 3: Truck leaves base

1. Select **Rules** in the left-hand menu, then **+ New**. Enter a meaningful name for the rule, such as **"Truck leaving base"**. Press Enter.

Now, enter the following conditions.

- Location / Latitude: doesn't equal => **47.644702**
- Location / Longitude: doesn't equal => **-122.130137**
- Truck state: Equals => **enroute**

2. Again, enter an appropriate **email action**, and click **Save**.

Task 4: Temperature of the contents

1. Enter a rule with a name such as **"Truck contents OK"**.

2. Turn on Time aggregation, with a period of **five minutes**.

3. Enter conditions that fire if the average Contents Temperature is less than **-1** degrees Celsius, and greater than **-18** degrees Celsius.

4. Again, enter an appropriate **email action**, and click **Save**.

At this point you should see all the rules listed as below:

Rules

+ New

Name	Status
Cooling system failed Alert	Enabled
Contents temperature spiking	Enabled
Truck leaving base	Enabled
Truck contents OK	Enabled

At this point it is time to test your Rules Go to your Device Dashboard, sent a Command to trigger a new Customer trip, remember use numbers from 1 to 9. In a few minutes you should start receiving emails.

Note: To receive emails the account you select has to be login to IoT central at least one time, otherwise you will not receive any emails.

Exercise 6: Clean up

Once you completed all the exercises, go to Azure Portal, look for the azure IoT Central Application and delete resource.