



PI World 2021 Lab

OCS Remote Operations Monitoring

© 2021 AVEVA Group plc and its subsidiaries. All rights reserved.

AVEVA, the AVEVA logos and AVEVA product names are trademarks or registered trademarks of aveva group plc or its subsidiaries in the United Kingdom and other countries. Other brands and products names are the trademarks of their respective companies.

AVEVA Group plc
High Cross, Madingley Road
Cambridge CB3 0HB, UK
Tel +44 (0)1223 556655
Fax +44 (0)1223 556666

[aveva.com](https://www.aveva.com)

Table of contents

1. LEARNING OBJECTIVES	4
1.1 OSISOFT MESSAGE FORMAT (OMF).....	5
1.2 CONFIGURATION OF PI ADAPTERS	5
2. REQUIRED PI SYSTEM SOFTWARE COMPONENTS	5
3. BUSINESS USE CASE- REMOTE MONITORING OF MINING TRUCKS	5
4. INTRODUCTION TO PI ADAPTERS.....	7
5. GETTING TO KNOW YOUR DATA SOURCE (MQTT BROKER).....	8
6. CONFIGURATION OF PI ADAPTER FOR MQTT.....	10
6.1 VERIFY THAT THE PI ADAPTER FOR MQTT SERVICE IS RUNNING:.....	10
6.2 CONFIGURE ONE OR MORE MQTT SYSTEM COMPONENTS.....	11
6.3 CONFIGURE AN MQTT DATA SOURCE FOR THE MQTT DEVICE	12
6.4 CONFIGURE MQTT DATA SELECTION	14
6.5 CLIENT-CREDENTIALS CLIENTS IN OCS	15
6.6 CONFIGURE EGRESS ENDPOINTS	16
7. CREATING METADATA RULES AND ASSET MANAGEMENT	17
7.1 VERIFY STREAMS WERE CREATED AND RECEIVING DATA IN SEQUENTIAL DATA STORE	17
7.2 CREATE METADATA RULES FOR YOUR STREAMS	19
7.3 CREATE AND CONFIGURE AN ASSET TYPE FOR OUR TRUCK ASSETS	23
7.4 USE ASSET RULE BUILDER TO BULK CREATE MINING TRUCK ASSETS	26
8. REMOTE OPERATIONS MONITORING USING OCS	32
8.1 MONITOR AND MANAGE YOUR ASSETS	32

- 8.2 FILTER ASSETS IN THE ASSET EXPLORER 35
- 8.3 SHARE A VIEW OF YOUR FLEET 36
- 9. VISUALIZE SDS DATA37
 - 9.1 UTILIZING TREND FEATURE OF OSISOFT CLOUD SERVICES 37
 - 9.2 UTILIZING GRAFANA 38
- 10. MISCELLANEOUS REMARKS (APPENDIX).....42

1. Learning Objectives

In this lab, you will be introduced to the key concepts of MQTT (Message Queuing Telemetry Transport), a messaging protocol created for Machine-to-Machine (M2M)/Internet of Things (IOT) communication. You will also be introduced on how to create assets within OSIsoft Cloud Services (OCS) and how one can monitor these assets in the cloud. If you are a PI professional and you are already familiar with the basic PI Tools, this lab will get you one-step further in the process of monitoring remote assets and deriving value from these assets.

This lab will go through all the steps from exploring data within a MQTT Broker, to configuring a PI Adapter for MQTT, which will create streams in OCS, and finally, creating assets in a templated manner through the asset rule builder inside the OCS platform. The topics will also include the process of visualizing the assets you have created with the 'Trend' functionality within OCS and in Grafana. At the end of this lab, you will have gained knowledge on how to properly configure a PI Adapter for MQTT and how to visualize the data from this PI Adapter through the creation of assets in OCS.

More specifically, the lab is structured in the following sections:

Part 1 – Introduction

- Introduction to MQTT (Message Queuing Telemetry Transport).
- Introduction to the PI Adapter for MQTT.
- Introduction to OSIsoft Cloud Services – with a focus on Asset building and Visualization of Assets.
- Introduce the business objective and the available data.

Part 2 – Understand the Data

- Explore the available data in the Mosquitto MQTT Broker.

Part 3 – Configuration of the PI Adapter for MQTT

- Walk through the necessary steps to configure PI Adapter for MQTT (data Ingress of the MQTT Broker).
- Walk through the necessary steps for data Egress to OCS (verification of stream data flows into OCS).

Part 4 – Creation of Contextualized Streams and the Creation of Contextualized Assets in OCS

- Creation of Streams and Contextualization of Streams (through Metadata Rules).
- Creating Asset Types and utilizing Asset Rule Building for automation of Asset creation.

Part 5 – Remote Asset Visualization

- Visualizing the assets in OCS.
- Visualizing the streams in Grafana.

1.1 OSIsoft Message Format (OMF)

In general, when dealing with PI Adapters, they will send data to their target endpoints as OSIsoft Message Format messages. OSIsoft Message Format (OMF) can also be used for programmatic access to data. OMF defines a set of message headers and bodies you can use to transfer data to OSIsoft Cloud Services. Because it does not depend on a particular protocol, such as HTTP, you can use OMF to develop data-acquisition applications on platforms and in languages for which there are no supported OSIsoft libraries.

1.2 Configuration of PI Adapters

PI Adapters are configured and administered using REST APIs and HTTP requests. For customers who wish to use a client tool to accomplish those tasks, we provide the EdgeCmd Utility (OSIsoft's proprietary tool for configuring PI Adapters and EDS), but other REST client tools, such as Postman, can be used and are also very straightforward to use. Using REST API, you can configure all functions of the Adapter. The REST API interface is very powerful and flexible, allowing you to both edit any single component or facet of the system individually as well as configuring the entire PI Adapter as a whole with a single REST API HTTP request call.

2. Required PI System Software Components

The VM (virtual machine) used for this lab has the following PI System software components installed:

Software	Version
PI Adapter for MQTT	1.0.0.96
OSIsoft EdgeCmd Utility	1.2.0.100

3. Business Use case- Remote Monitoring of Mining trucks

Monitoring a fleet of large assets can be difficult for organizations as this requires a comparison of the previous and current performance of each asset against benchmarks provided by the manufacturer. This issue is compounded as organizations typically have silos of data and integrating all of these data sources can be difficult. This scenario is not uncommon in the mining industry, especially when discussing mine trucks.



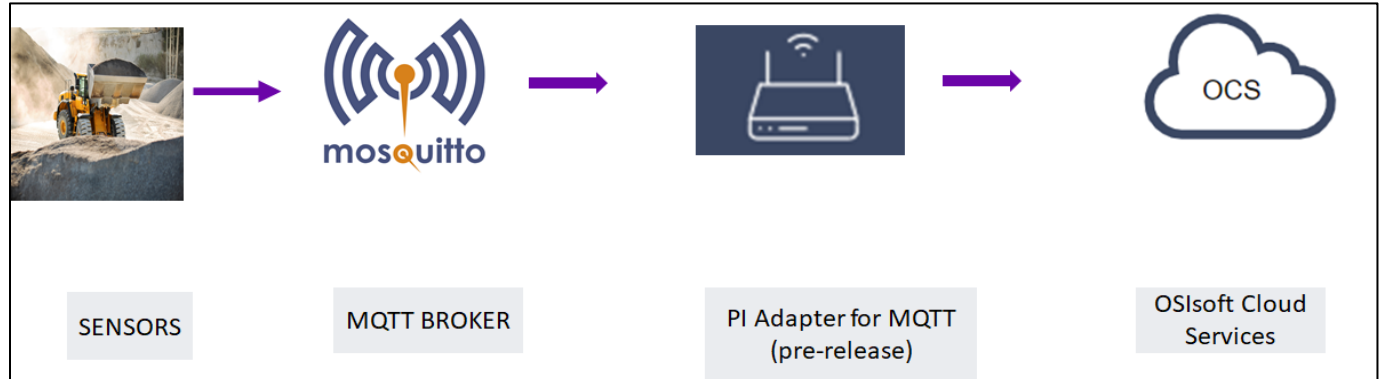
Mine trucks are a fairly large capital expense in the order of millions of dollars per vehicle. Extending the life of these vehicles, improving their reliability, and minimizing costly unplanned maintenance expenditures, can lead to significant savings. However, mining organizations are challenged in maintaining a fleet of trucks that may originate from a variety of vendors. Data from these trucks reside in different systems such as the Vehicle Information Management System (VIMS), Lab Information Management System (LIMS), and Fleet Management System (FMS). Therefore, integration with these systems is necessary to obtain a comprehensive view of fleet performance.

PI System can help mining organizations monitor in real-time their fleet of trucks by integrating with these various systems. Furthermore, the data retrieved from these systems can be aggregated, organized, and enhanced allowing mining organizations to develop analyses that compare current truck performance against manufacturer benchmarks. This provides situational awareness of the fleet of vehicles, can help identify leading indicators for System failures, and eventually can lead to a fully integrated conditioned-based maintenance solution.

What will you do in this lab specific to this use case?

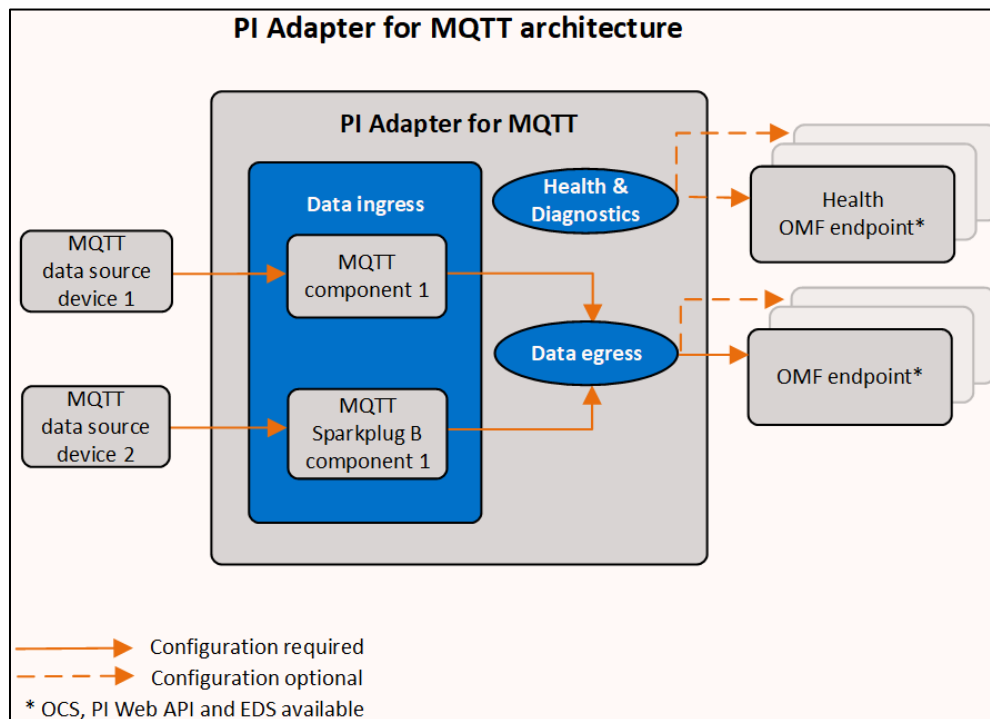
In this lab, you will be presented with an opportunity to configure data collection from simulated mining trucks and sending them to OSIsoft Cloud Services as part of your remote monitoring tasks. You will specifically configure PI Adapter for MQTT to collect simulated truck data from a MQTT Server using topics and send them to OCS. Then, you will configure metadata rules, create an asset type, and use Asset Management functionality to build the assets in bulk. Finally, you will visualize the data using OCS Trend functionality and in Grafana.

You will establish the flow of data between the various components aligning with the below data architecture:



4.Introduction to PI Adapters

PI Adapter for MQTT is a data-collection component that transfers time-series data from source devices to OMF endpoints in OSIsoft Cloud Services or PI Servers. MQTT (Message Queuing Telemetry Transport) is a messaging protocol created for Machine-to-Machine (M2M)/Internet of Things (IOT) communication. The adapter can connect to any device that uses the MQTT protocol for communication with constrained devices and server applications for data exchange.



OSIsoft Cloud Services (OCS) is a database platform as a service (dbPaaS) designed for storing, retrieving, and analyzing real-time operations data. With OCS, people in and out of your organization have flexible and secure

access operations data. OCS can collect operations data from within a primary control network to the edge of the industrial network. Use OCS to add context to time-based operations data, to enable process engineers and systems operators to make decisions and take corrective or preemptive actions. From data collection to data access to data delivery, OCS provides flexibility and control with its REST API.

OSIsoft Cloud Services (OCS) makes it possible to monitor remote assets in real time. The immediate access to data about the status of assets gives you the ability to anticipate problems and proactively perform preventative maintenance.

5. Getting to know your data source (MQTT Broker)

This section discusses how we will explore the input data to be used for the data ingress portion of the PI Adapter for MQTT. The data pertaining to the 10 mining trucks resides within the Mosquitto MQTT Broker.

In order to view this data, we must connect a MQTT Client to the MQTT broker and explore.

1. Verify the Mosquitto Broker service is Running:
 - Open 'Run' and Type Services.msc
 - Verify that the 'Mosquitto Broker' service is running. If it is not running, Right click on the service and click 'Start'.
2. Open MQTT Explorer, shortcut will be on the desktop.

Enter the following:

- Name: OCS ROM Lab.
- Validate certificate: No encryption.
- Protocol: mqtt://
- host: localhost.
- Port: 1883.
- Username and Password are left empty.
- Click the Connect Button.

+

Connections

OCS ROM 2021

mqtt://localhost:1883/

test.mosquitto.org

mqtt://test.mosquitto.org:1883/

MQTT Connection

mqtt://localhost:1883/

Name

OCS ROM 2021|

Validate certificate

Encryption (tls)

Protocol

mqtt://

Host

localhost

Port

1883

Username

Password

DELETE

ADVANCED

SAVE

CONNECT

3. Click on fleet option when it appears, and verify that the topics are created and the number of messages are starting to increase.
 - This is an example of what the following should look like:

- This is an example of what the following should look like:

Truck	Status	Engine_Coolant_Temperature	Engine_Fuel_Rate	Engine_Load	Engine_Oil_Pressure
Truck 103	'Under Maintenance'	86.02219	326.3284	91.3984	
Truck 104	'Running'	87.8709	323.8319	55.20742	
Truck 105	'Under Maintenance'	90.74877	317.5281	62.73914	
Truck 106	'Stopped'	90.51813	318.8409	91.36404	
Truck 107	'Under Maintenance'	90.29766	323.7773	72.4159	
Truck 108	'Stopped'	89.68406	328.0084	55.20647	
Truck 109	'Under Maintenance'	89.1134	320.3191	51.0472	
Truck 110	'Running'	90.30286	320.925	82.54514	
Truck 101	'Running'	90.81158	326.0575	68.46484	
Truck 102	'Stopped'	90.91887	320.1272	55.18041	

Note: Certain Key-Value pairs within the JSON payloads seen in the image above contain single quotes. Valid JSON structure uses double quotes for key's and double quotes for type String values. The reason for seeing single quotes is simply due to the MQTT Explorer client displaying the data in a single quote fashion. In reality, the PI Adapter for MQTT treats the JSON payloads as if they had the appropriate double quotes key and values.

4. You can see the distinctive features for each respective mining truck. For the purpose of this lab, 10 features were selected to be used as the data properties pertaining to each mining truck.

The data features for this lab are the following:

- “Aftercooler Temperature”,
 - “Engine Coolant Temperature”,
 - “Engine Fuel Rate”,
 - “Engine Load”,
 - “Engine Oil Pressure”,
 - “Engine RPM”,
 - “Ground Speed”,
 - “Longitude”,
 - “Latitude”,
 - “Status”
5. Click on ‘Truck 104’ and analyze the ‘Value’ section to the right of the MQTT Explorer client to see the JSON payload of the data structure. It is important to note that for the PI Adapter for MQTT, input data must be structured as a JSON payload. If the input data is not specified as a JSON payload, data ingress for the PI Adapter for MQTT will not function correctly.

Note: JavaScript Object Notation (JSON) is a standardized format commonly used to transfer data as text that can be sent over a network. JSON represents objects as name/value pairs.

6. Configuration of PI Adapter for MQTT

In this section, we will look at the configuration of the PI Adapter for MQTT. The first step is to verify that the PI Adapter for MQTT service is running.

6.1 Verify that the PI Adapter for MQTT service is running:

1. Open ‘Run’ and type ‘Services.msc’
2. Verify that the ‘PI Adapter for MQTT’ service is running. If it is not running, Right click on the service and click ‘Start’.

Once we confirmed that the PI Adapter for MQTT service is running, the next steps will be to configure the PI Adapter for MQTT. A high-level summary of the steps required for configuring the PI Adapter for MQTT can be seen below:

1. Configure one or more MQTT system components.
2. Configure an MQTT data source for the MQTT device.
3. Configure an MQTT data selection for each MQTT data source.

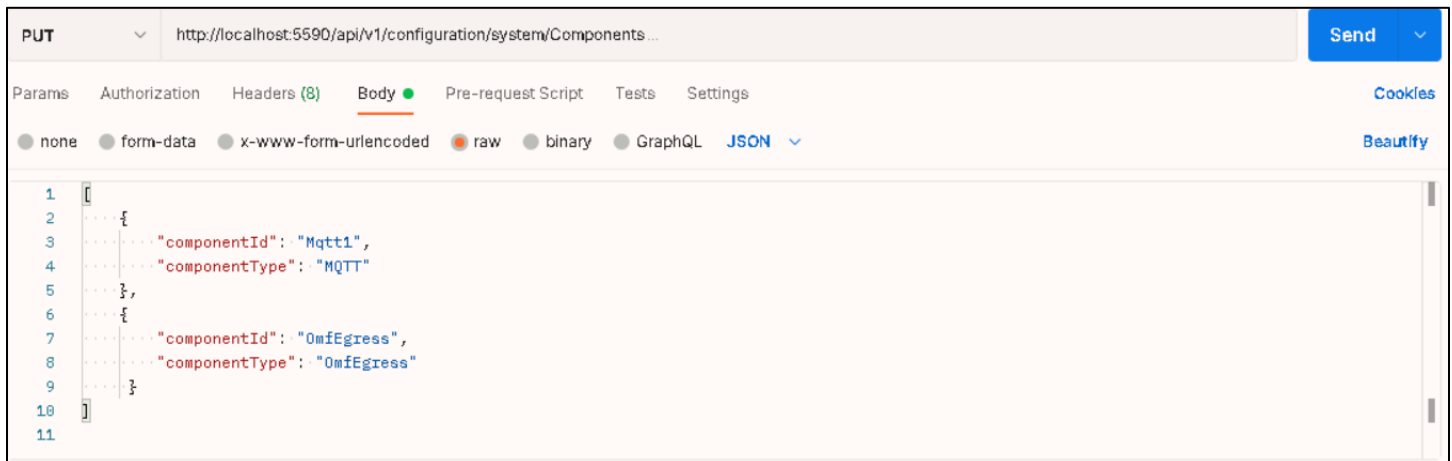
Optional:

- Data filters
 - Diagnostics and metadata
 - Buffering
 - Logging
 - Configure a network proxy
4. Configure one or more egress endpoints (Data Egress to OCS endpoint).

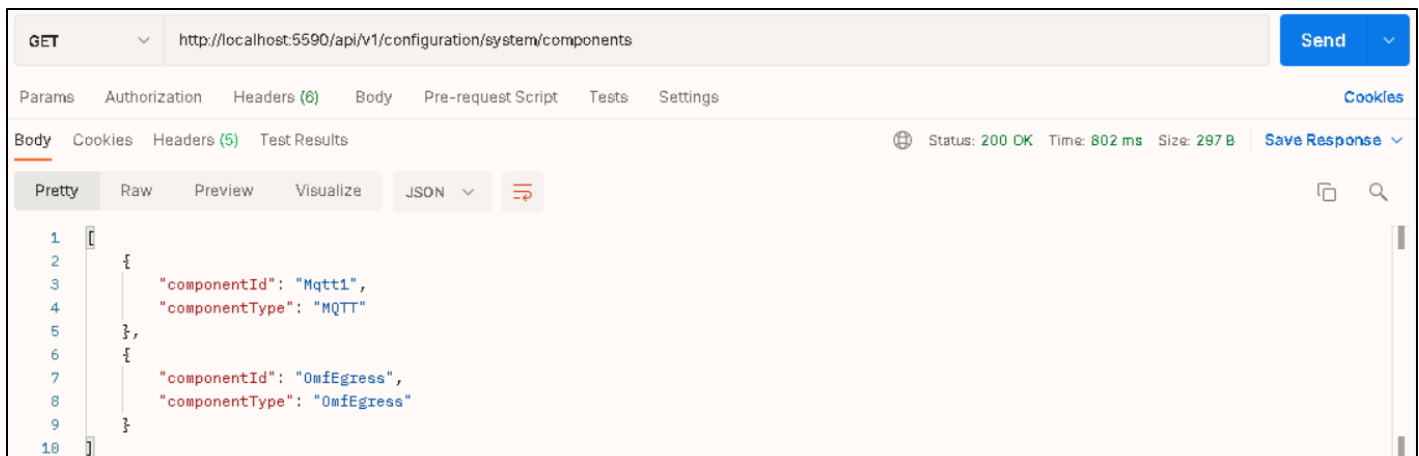
6.2 Configure one or more MQTT System Components

Complete the following steps to configure system components. Use the PUT method in conjunction with the **<http://localhost:5590/api/v1/configuration/system/components>** REST endpoint to initialize the configuration.

1. Open the Postman application on the desktop.
2. Click PI Adapter for MQTT Configuration dropdown.
3. Click 'PUT Components'.
4. Click 'Send' button.



5. Ensure that a 204 No Content status code is returned
6. Verify that the system components have propagated to PI Adapter REST endpoint
7. Click 'GET Components'
8. Verify that the following is returned in Postman:



6.3 Configure an MQTT Data Source for the MQTT device

Complete the following steps to configure an MQTT data source. Use the PUT method in conjunction with the **http://localhost:5590/api/v1/configuration/<ComponentId>/DataSource** REST endpoint to initialize the configuration.

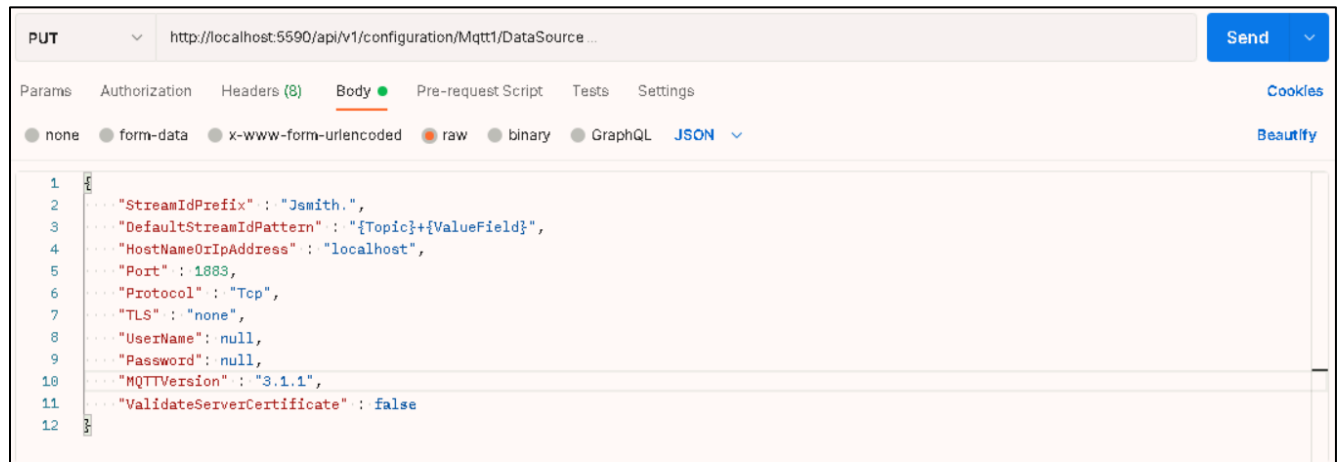
Note: The ComponentId can be seen in the previous step. The ComponentId for this lab is 'Mqtt1'.

1. Open the Postman application on the desktop:
2. Click PI Adapter for MQTT Configuration dropdown.

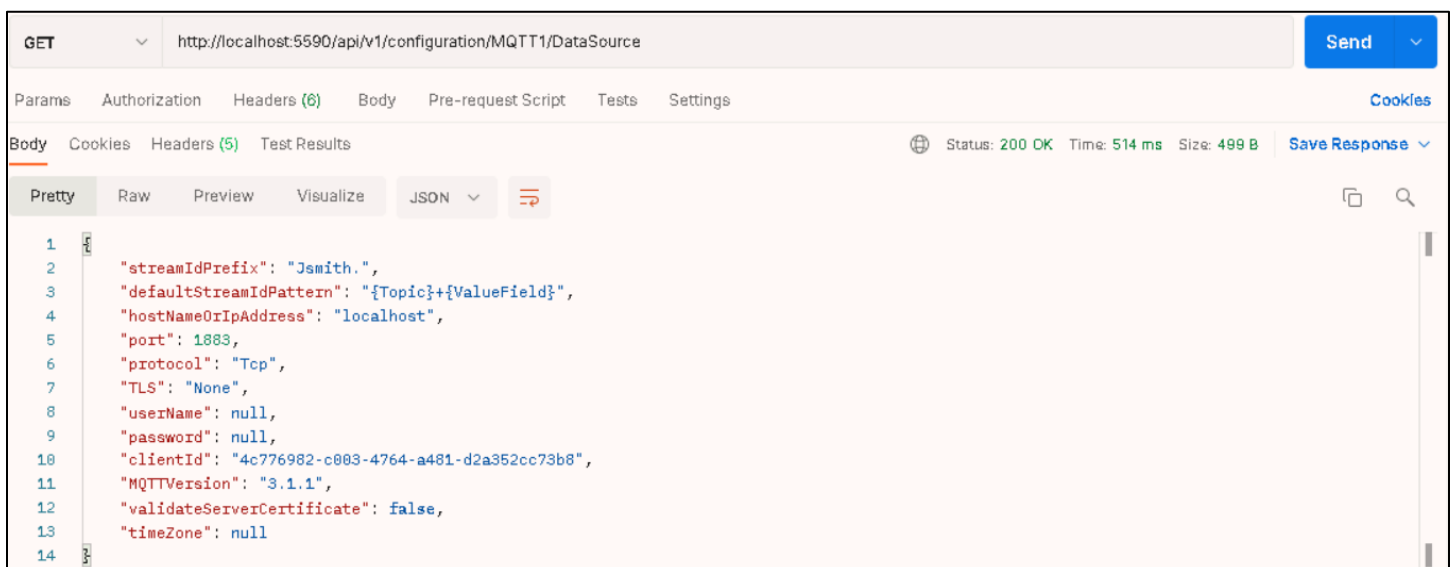
3. Click 'PUT DataSource'.
4. Replace 'StreamIdPrefix' key from 'Jsmith' to your studentId (first letter of first name and last name).

Example: John Smith, would be: jsmith. The value for the "StreamIdPrefix" key should also end with a period "." Which will act as the delimiter for our streams.

5. Click 'Send' button.



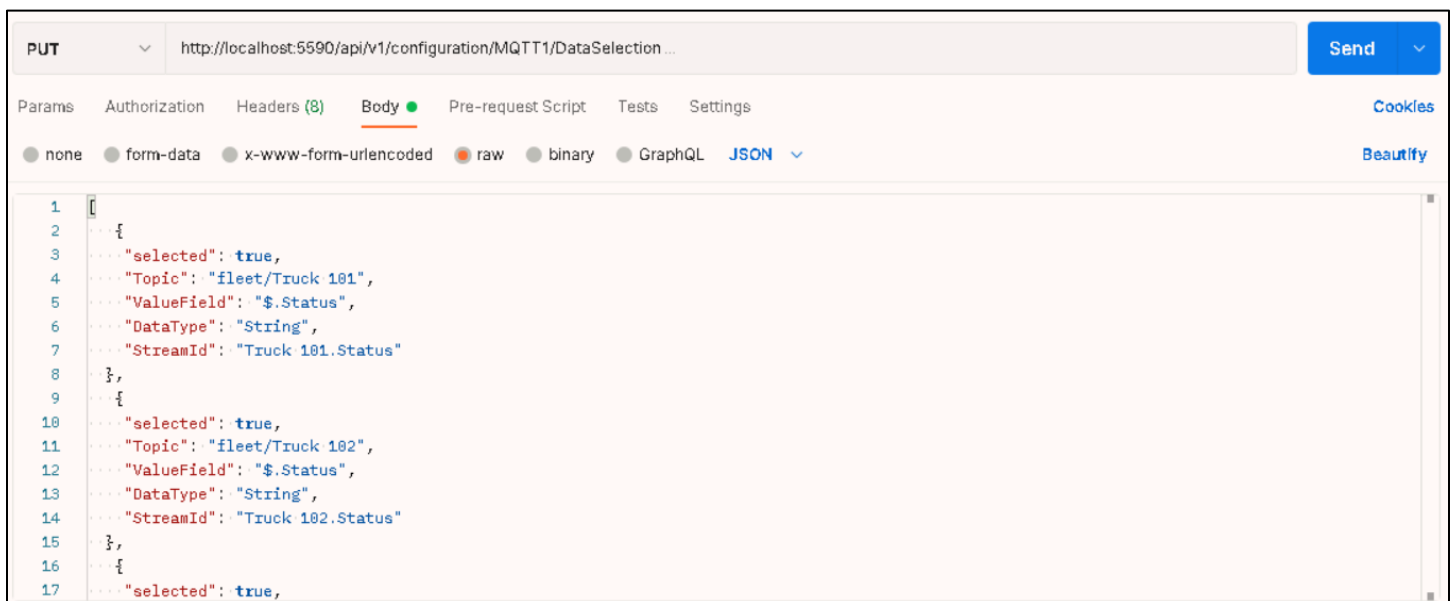
6. Ensure that a 204 No Content status code is returned.
7. Verify that the datasource components have propagated to PI Adapter REST endpoint.
8. Click 'GET DataSource'.
9. Verify that the following is returned in Postman:



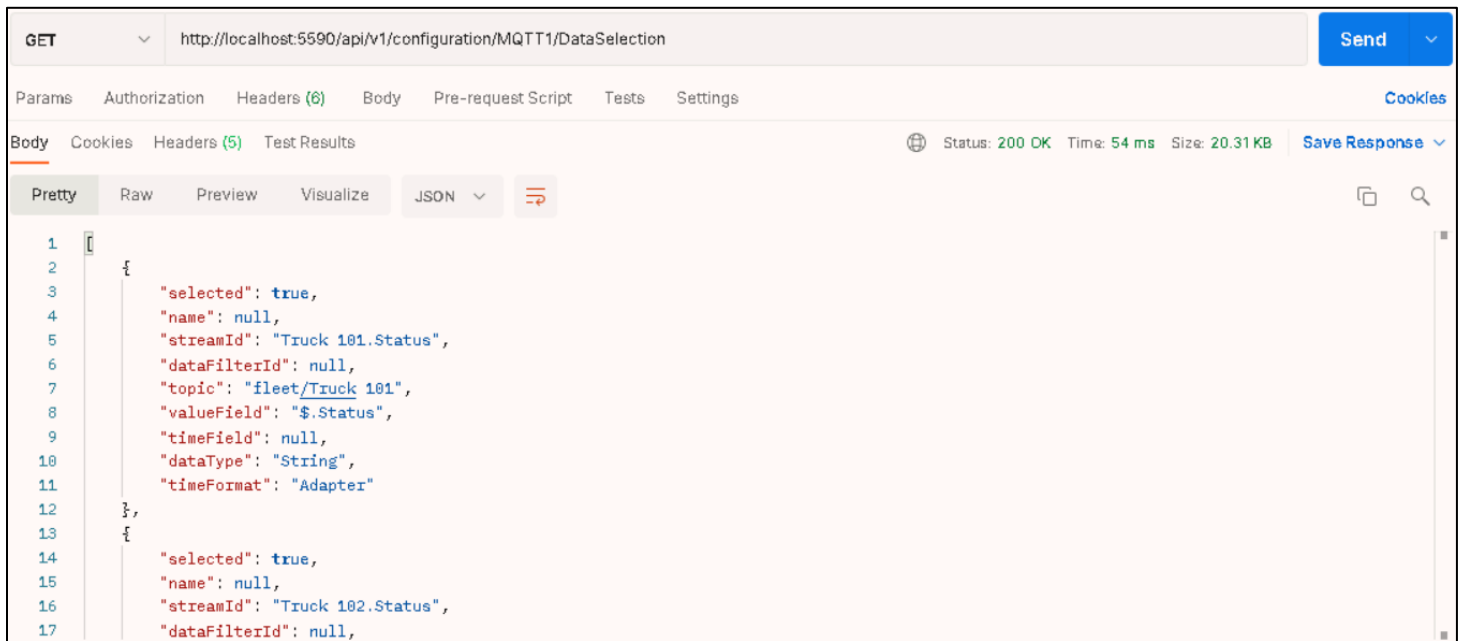
6.4 Configure MQTT Data Selection

Complete the following steps to configure an MQTT data selection. Use the PUT method in conjunction with the **http://localhost:5590/api/v1/configuration/MQTT1/DataSelection** REST endpoint to initialize the configuration.

1. Open the Postman application on the desktop:
2. Click PI Adapter for MQTT Configuration dropdown.
3. Click 'PUT DataSelection'.
4. Click 'Send' button.



5. Ensure that a 204 No Content status code is returned.
6. Verify that the data selection components have propagated to PI Adapter REST endpoint.
7. Click 'GET DataSelection'.
8. Verify that the following is returned in Postman:



GET <http://localhost:5590/api/v1/configuration/MQTT1/DataSelection> Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (5) Test Results Status: 200 OK Time: 54 ms Size: 20.31 KB Save Response

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "selected": true,
4     "name": null,
5     "streamId": "Truck 101.Status",
6     "dataFilterId": null,
7     "topic": "fleet/Truck 101",
8     "valueField": "$.Status",
9     "timeField": null,
10    "dataType": "String",
11    "timeFormat": "Adapter"
12  },
13  {
14    "selected": true,
15    "name": null,
16    "streamId": "Truck 102.Status",
17    "dataFilterId": null,
```

6.5 Client-Credentials Clients in OCS

Client-credentials clients are typically used for server-to-server communication that does not require user interaction. The client typically authenticates with the token endpoint using its client ID and secret. A secret is a unique key generated for each client to connect to OSIsoft assets, resources, and services for a time-limited period. Because secrets allow access to data, you need to keep them secure.

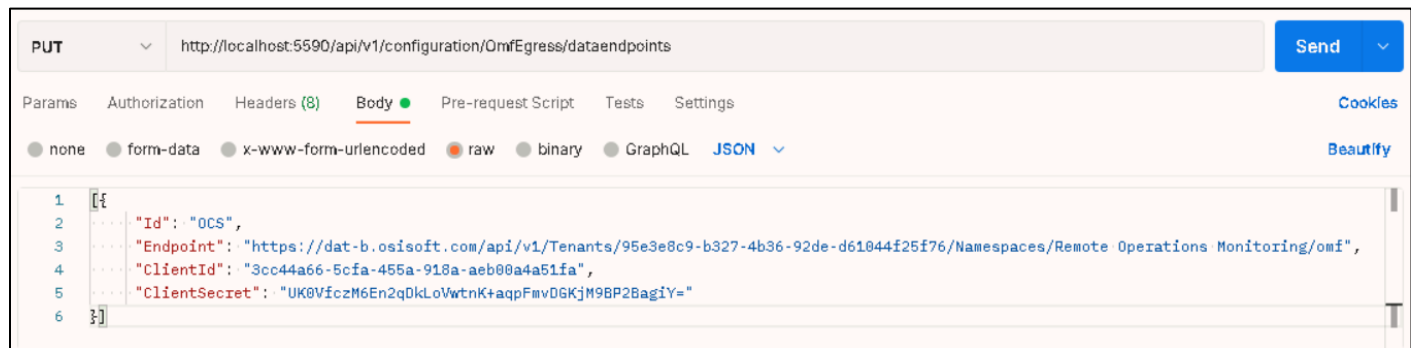
In this lab, we have precreated the client-credentials client needed for our PI Adapter for MQTT to send data to OCS. We have also precreated the OMF Connection that assigns the client to our given namespace, 'Remote Operations Monitoring'.



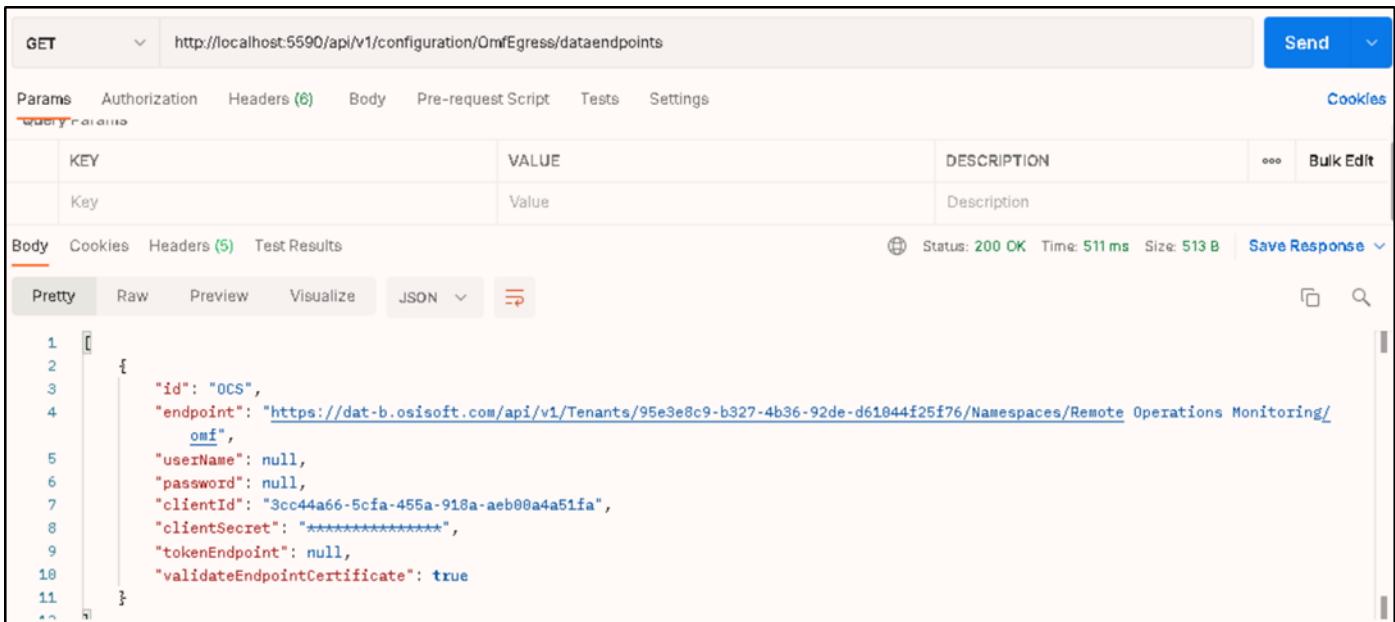
6.6 Configure egress endpoints

Complete the following steps to configure egress endpoints. Use the PUT method in conjunction with the **`http://localhost:5590/api/v1/configuration/OmfEgress/dataendpoints`** REST endpoint to initialize the configuration.

1. Open the Postman application on the desktop:
2. Click PI Adapter for MQTT Configuration dropdown.
3. Click 'PUT EgressEndpoints'.
4. Click 'Send' button.



5. Ensure that a 204 No Content status code is returned.
6. Verify that the egress endpoints components have propagated to PI Adapter REST endpoint.
7. Click 'GET EgressEndpoints'.
8. Verify that the following is returned in Postman:



7. Creating metadata rules and asset management

7.1 Verify Streams were created and Receiving Data in Sequential Data Store

In this step, we will verify if the PI Adapter for MQTT was successfully able to create streams in the Sequential Data Store (SDS). The Sequential Data Store (SDS) is used to store, retrieve, and organize any type of streaming data in OCS. We will also verify that these streams are properly receiving data from the PI Adapter for MQTT.

1. On the left panel in the OCS portal, click on the 'Data Management' option.
2. Click on Sequential Data Store.
3. At the top middle of the window, in the search box, enter the following:
Example: *Jsmith* (your student ID, which is the first letter of your first name concatenated with your full last name).*

The following will return streams that begin with jsmith, which is the streamIdPrefix that was specified in the previous section.

Streams ▾

jsmith*

×





4. The results of the search should return all 100 streams created for your respective studentId.

<input type="checkbox"/>	Id	Name ↑	Description	Type Id
<input type="checkbox"/>	jsmith.Truck 101.Aftercooler_Temperature	jsmith.Truck 101.Aftercooler_Temperature		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 101.Engine_Coolant_Temperature	jsmith.Truck 101.Engine_Coolant_Temperature		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 101.Engine_Fuel_Rate	jsmith.Truck 101.Engine_Fuel_Rate		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 101.Engine_Load	jsmith.Truck 101.Engine_Load		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 101.Engine_Oil_Pressure	jsmith.Truck 101.Engine_Oil_Pressure		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 101.Engine_RPM	jsmith.Truck 101.Engine_RPM		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 101.Ground_Speed	jsmith.Truck 101.Ground_Speed		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 101.Latitude	jsmith.Truck 101.Latitude		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 101.Longitude	jsmith.Truck 101.Longitude		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 101.Status	jsmith.Truck 101.Status		TimeIndexed.String
<input type="checkbox"/>	jsmith.Truck 102.Aftercooler_Temperature	jsmith.Truck 102.Aftercooler_Temperature		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 102.Engine_Coolant_Temperature	jsmith.Truck 102.Engine_Coolant_Temperature		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 102.Engine_Fuel_Rate	jsmith.Truck 102.Engine_Fuel_Rate		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 102.Engine_Load	jsmith.Truck 102.Engine_Load		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 102.Engine_Oil_Pressure	jsmith.Truck 102.Engine_Oil_Pressure		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 102.Engine_RPM	jsmith.Truck 102.Engine_RPM		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 102.Ground_Speed	jsmith.Truck 102.Ground_Speed		TimeIndexed.Double
<input type="checkbox"/>	jsmith.Truck 102.Latitude	jsmith.Truck 102.Latitude		TimeIndexed.Double

5. Click on 'StudentId.Truck 101.Aftercooler_Temperature' Stream and click on 'Manage Data'.


Details

Metadata and Tags

Stream Details

 Manage Data

6. Ensure that the timestamp is recent, and that there is a valid value present for this stream.

Timestamp 	Value
Oct 7, 2021, 3:28:35 PM	65.3712

7.2 Create Metadata Rules for your Streams

Aveva recommends that you explicitly include metadata, when possible, when you create streams. Metadata rules are used to assign metadata to individual streams. Metadata rules assign defined metadata to all streams in a given namespace matching the stream name pattern defined in the rule. OCS parses each stream and builds out the metadata following their defined metadata rules. This is important as it contextualizes the streams that we have created.

1. On the left panel in the OCS portal, click on the 'Data Management' option.
2. Click on Metadata Rules.
3. Click on 'Add Metadata'.
4. When the Metadata Rules window appears, in the search box, enter: Jsmth*
Note: The following will return all streams pertaining to your individual StudentId.
5. Select the Jsmith.Truck 101.Aftercooler_Temperature stream as the stream to be used to construct this metadata rule.
6. Click on the first "+" sign and define the key as "String Literal"
Note: A String Literal is not a variable, and only stream names displaying its literal value will be captured.
7. Click on the "+" sign after the Truck and define the key as "Asset Type".
8. Click on the "+" sign after the 101, and define the key as "Asset ID".
9. Define the last key, which encompasses "Aftercooler_Temperature" as Measurement.
The final output should look like this:

The screenshot shows the OCS Metadata Rules configuration window. At the top, a green checkmark indicates the rule is valid. The rule is defined as `jsmith.{AssetType} {AssetID}. {Measurement}`. Below the rule, the stream `jsmith.Truck 101.Aftercooler_Temperature` is displayed with blue vertical bars and plus signs indicating where metadata is applied. The configuration table below shows the metadata assignments for each part of the stream name:

Stream Part	Metadata Type	Metadata Key
"jsmith"	String Literal	
"Truck"	Metadata	Asset Type
"101"	Metadata	Asset ID
"Aftercooler_Temperature"	Metadata	Measurement




10. Click on Next

11. We must now define the Metadata Mappings from the previous section. We must define how to extract the metadata value from its section of the stream name.

There are two options available for the Mapping Type, 'Copy Values' and 'Map Values'.

- To display the raw stream data for the specified metadata key, select Copy Values.
- To assign a label to data values, select Map Values and click Generate Mappings.
- We will keep the Asset mapping and Asset Number mapping defined with the 'Copy Values' mapping type.

jsmith.{AssetType}{AssetID}.{Measurement}

Asset Type  Asset ID  Measurement 

Values for "Asset Type"

For the metadata key "Asset Type", choose whether values should be copied literally from the section (Copy Values) or if the section should be mapped to user-defined values (Map Values). Typically, Copy Values is helpful for creating unique values such as IDs. Select "Map Values" to generate a list of all matches for this section. Add an entry for each mapping. Click the "x" button on a generated mapping if you do not want to create metadata for it. Map Values are helpful for creating multiples of the same value, such as mapping "Temperature" to all sections with "Temp".

Key




Asset Type

Mapping Type

☒ Copy Values

☐ Map Values

jsmith.{AssetType}{AssetID}{Measurement}

Asset Type  Asset ID  Measurement 

Values for "Asset ID"

For the metadata key "Asset ID", choose whether values should be copied literally from the section (Copy Values) or if the section should be mapped to user-defined values (Map Values). Typically, Copy Values is helpful for creating unique values such as IDs. Select "Map Values" to generate a list of all matches for this section. Add an entry for each mapping. Click the "x" button on a generated mapping if you do not want to create metadata for it. Map Values are helpful for creating multiples of the same value, such as mapping "Temperature" to all sections with "Temp".

Key

Asset ID

Mapping Type

☒ Copy Values

☐ Map Values

12. We will define the Measure mapping with the 'Map Values' mapping type. Once you click on 'Map Values' and 'Generate Mappings', this should be the following output:

Asset

Asset Number

Measure

Values for "Measure"

For the metadata key "Measure", choose whether values should be copied literally from the section (Copy Values) or if the section should be mapped to user-defined values (Map Values). Typically, Copy Values is helpful for creating unique values such as IDs. Select "Map Values" to generate a list of all matches for this section. Add an entry for each mapping. Click the "x" button on a generated mapping if you do not want to create metadata for it. Map Values are helpful for creating multiples of the same value, such as mapping "Temperature" to all sections with "Temp".

Key

Measure

Mapping Type

☐ Copy Values

☒ Map Values

Generate Mappings Remove All

Mappings

Aftercooler_Tempere	→	Map To...		Engine_Coolant_Terr	→	Map To...		Engine_Fuel_Rate	→	Map To...	
Engine_Load	→	Map To...		Engine_Oil_Pressure	→	Map To...		Engine_RPM	→	Map To...	
Ground_Speed	→	Map To...		Latitude	→	Map To...		Longitude	→	Map To...	
Status	→	Map To...									

Add Mapping

13. We can see that we were able to match multiple patterns for the 'Measure' mapping defined in the previous step. Here we see all the possible measurements found within our streams. We will remap each value to a more presentable manner by removing the underscore '_' symbol for each mapping.

The final result should like this:

Measurement

Values for "Measurement"

For the metadata key "Measurement", choose whether values should be copied literally from the section (Copy Values) or if the section should be mapped to user-defined values (Map Values). Typically, Copy Values is helpful for creating unique values such as IDs. Select "Map Values" to generate a list of all matches for this section. Add an entry for each mapping. Click the "x" button on a generated mapping if you do not want to create metadata for it. Map Values are helpful for creating multiples of the same value, such as mapping "Temperature" to all sections with "Temp".

Key

Measurement

Mapping Type

☐ Copy Values

☒ Map Values

Generate Mappings Remove All

Mappings

Aftercooler_Tempere	→	Aftercooler Tempera		Engine_Coolant_Terr	→	Engine Coolant Temp		Engine_Fuel_Rate	→	Engine Fuel Rate	
Engine_Load	→	Engine Load		Engine_Oil_Pressure	→	Engine Oil Pressure		Engine_RPM	→	Engine RPM	
Ground_Speed	→	Ground Speed		Latitude	→	Latitude		Longitude	→	Longitude	
Status	→	Status									

Add Mapping

14. Click Next.

15. In the Preview and Run Window, Give the metadata rule a name, example: jsmith_metadata_rule, and give a description, example: Metadata rule for streams created by jsmith.
16. Click Save & Execute, at the bottom right of the window to apply this rule to your streams.
17. Return to Data Management > Metadata Rules and confirm that your metadata rule has been created.
18. Click on Data Management.
19. Click on Sequential Data Store.
20. Search for jsmith* in the search box.
21. Select jsmith.Truck 101.Engine_load stream.
22. On the right-hand panel, switch the tab from 'Details' to 'Metadata and Tags'.
23. Ensure under Stream Metadata, we see the Asset, Asset Number and Measurement values.
Note: There may be additional stream metadata provided from the PI Adapter for MQTT.

Details	Metadata and Tags
Stream Tags	
No tags to display.	
Stream Metadata	
Metadata	Value
AdapterType	MQTT
Asset	Truck
Asset Number	101
DataSource	Mqtt1
Measurement	Engine_Load
Topic	fleet/Truck 101
ValueField	\$.Engine_Load

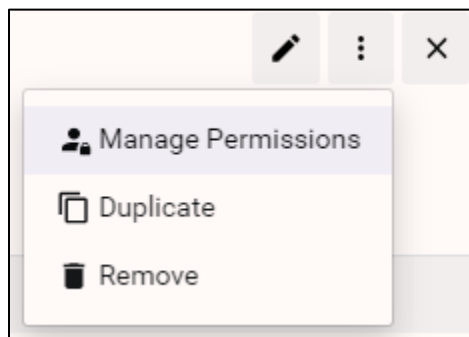
7.3 Create and Configure an Asset Type for our Truck Assets

An asset type is a template for creating assets that share a common structure or type. When you create an asset type, you define the expected metadata and stream references for assets created from that asset type. Using asset types to create assets makes it easier to compare assets of the same type and to ensure consistency across similar assets.

An asset is a container for metadata, or static information about the asset (such as location or company), and properties which are references to streams in the Sequential Data Store. Assets allow you to retrieve all the stream values associated with an asset and to provide context to this dynamic data.

1. Click on Visualization.
2. Click on Asset Explorer - Note: Asset Explorer is used to monitor and manage assets.
3. Change the dropdown at the top left of the window from 'Assets' to 'Asset Types'.
4. To create a new Asset Type, Click on the '+ Add Asset Type' button on the top right of the screen.

For this lab, we will use the incomplete ROM Trucks Asset Type. Click on the 'Incomplete ROM Trucks Asset Type'. At the top right corner of the screen, click on the vertical ellipsis symbol:



5. Click on 'Duplicate' in order to duplicate this Asset Type.

As we can see under the Properties window, we are missing the Latitude, Longitude and Status properties that make up our Truck Assets.

6. Click on 'Pencil' symbol in order to edit the current Asset Type.
7. Click on the Properties tab and Click on the 'Add Stream Type Reference'

In the Select Stream Type window:

- Select 'Double', Id: TimeIndexed.Double

Next, we will assign the Status property to be configured as the property to be used as the Status for these assets.

11. Switch tabs from Properties to Status.
12. Click on 'Add Status Configuration'.
13. Select 'Status' property.
14. Click Continue.

Now define the Value Mappings and assign visualize context to each value.

15. Click on '+ Add Value Mapping'.

The result should look like this:

The screenshot displays the 'Status' configuration interface. At the top, there are tabs for 'Metadata', 'Properties', and 'Status', with 'Status' being the active tab. Below the tabs, there is a 'Remove Status Configuration' button. The 'Property' section shows 'Status' as the selected property. The 'Value' section lists three values: 'Running', 'Under Maintenance', and 'Stopped'. Each value is mapped to a specific status icon: 'Running' is mapped to a green checkmark (Positive Status), 'Under Maintenance' is mapped to a yellow triangle (Cautionary Status), and 'Stopped' is mapped to a red X (Negative Status). Each mapping has a trash icon to its right. A '+ Add Value Mapping' button is located at the bottom of the 'Value' section.

- Positive Status with a value of "Running"
- Cautionary Status with a value of "Under Maintenance"
- Negative Status with a value of "Stopped"

Once the following is completed, let us add Metadata to our Asset to contextualize the Assets.

16. Switch the tab from Status to Metadata.







17. Click on the Add Metadata button, and ensure the following metadata is added.

Metadata

Properties

Status

Add Metadata

Metadata	Value	Type	UOM	
Customer	AvevaGold	String ▾	none ▾	
Engine Model	Cat C175-22	String ▾	none ▾	
Manufacturer	Caterpillar	String ▾	none ▾	
Mine Truck Model	790F BC	String ▾	none ▾	
Region	NAMER	String ▾	none ▾	
Site	USA-Alaska	String ▾	none ▾	

18. Change the Asset Type Name to ROM Trucks Asset Type Jsmith (Your StudentID).

19. Once the following is done, click on 'Save' at the bottom right of the screen to save this Asset Type.

7.4 Use Asset Rule Builder to Bulk Create Mining Truck Assets

An asset rule identifies patterns in a stream name and uses this information to automatically create assets. We will utilize asset rules to bulk create our mining truck assets.

1. Click on Data Management.
2. Click on Asset Management.
3. Click on Add a Rule.

Enter the following:

- Name: Jsmith ROM PI World Lab
- Description: Jsmith Asset Rule for PI World ROM

- Asset Type: Select the Asset Type you defined in the previous step, ROM Trucks Jsmith (your StudentId)

In the 'Select Stream' window, select one of the streams you have created.

For this example, select the 'Truck 101.Aftercooler_Temperature' Stream as the example to be used.

4. Enter in the Search Query box, Search: "jsmith*", which will return all streams starting with jsmith. Replace jsmith with your appropriate studentId.
5. Select the jsmith.Truck 101.Aftercooler_Temperature Stream.

In the next window, we see that when creating an asset rule, there are 4 steps. The first step is Token Extraction.

Step1: Extract tokens from the stream name

In this step, you specify the naming pattern used to find and match the appropriate streams. You isolate each part of the stream name and create a token for it. The rule contains intelligence to recognize special characters in the stream name as delimiters, such as periods, dashes, and underscores. By default, the rule uses any special characters in the name to isolate the stream parts and facilitates the rule-building process. In this step, you also create tokens for the stream metadata.

1. In the Stream Name Pattern pane, move the slider to highlight the first identifiable section of the stream name.
2. In the 'Match' list, select the option that describes how to identify the value in the stream name.
3. In the 'and name it' text box, enter a name for the token.
4. Click Capture.

For our example, we will move the slider until it captures the entire StudentID and the "." Period delimiter.

5. In the 'Match' field, select "the string literal"

Note: The string literal option will create a token for the literal string that was defined.

jsmith.Truck 101.Aftercooler_Temperature

1. Match: ☐ characters of any type (including letters, numbers, and symbols) preceding the delimiter "."

☐ the next 7 characters of any type (including letters, numbers, and symbols)

☐ the next 7 characters of any type (including letters, numbers, and symbols)

☐ the next group of characters of any type (including letters, numbers, and symbols)

☒ the string literal "jsmith."

Capture

6. In the Stream Name Pattern pane, move the slider to highlight the second identifiable section of the stream name.
7. For our example, we will move the slider until it captures the entire Truck 101 portion of the stream name.
8. In the 'Match' field, select "characters of any type (including letters, numbers, and symbols) preceding the delimiter "."
9. In the 'and name it' field, enter: TruckID.
10. Click Capture.

jsmith.Truck 101.Aftercooler_Temperature

1. Match the string literal "jsmith."

2. Match: ☒ characters of any type (including letters, numbers, and symbols) preceding the delimiter "."

☐ the next 9 characters of any type (including letters, numbers, and symbols)

☐ the next 9 characters of any type (including letters, numbers, and symbols)

☐ the next group of characters of any type (including letters, numbers, and symbols)

☐ the string literal "Truck 101"

and name it: TruckID

Undo Capture Capture

11. In the Stream Name Pattern pane, move the slider to highlight the third identifiable section of the stream name.

For our example, we will move the slider until it captures the entire Aftercooler_Temperature portion of the stream name.

12. In the 'Match' field, select "characters of any type (including letters, numbers, and symbols) until the end of the stream name.
13. In the 'and name it' field, enter: Measurement.
14. Click Capture.
15. Click Next.

jsmith.Truck 101.Aftercooler_Temperature

1. Match the string literal "jsmith."
2. Match characters of any type (including letters, numbers, and symbols) preceding the delimiter "." {TruckID} - Truck 101
3. Match the delimiter "."
4. Match: ☒ characters of any type (including letters, numbers, and symbols) until the end of the stream name
☐ the next 23 characters of any type (including letters, numbers, and symbols)
☐ the next 23 characters of any type (including letters, numbers, and symbols)
☐ the next group of characters of any type (including letters, numbers, and symbols)
☐ the string literal "Aftercooler_Temperature"


and name it: Measure

Undo Capture Capture

Step 2: Map the tokens to values

In this step, you specify the token that identifies the stream measurement in the stream. Then you map values for each token.

We are going to select the token defined in the previous step that contains the stream reference name label for the measurement.

1. In the Configure Stream Reference Name Token pane, click the Select token icon  to open the Select Stream Reference Name Token window.

2. Select the token that identifies the stream measurement and click Select. For our example, we will select the 'Measurement' token.
3. Next, make sure the Rename Token Values option is selected and click on Generate Mappings. We are going to Replace the values in the stream name and stream metadata with mapped values.

After clicking on the Generate Mappings button, this is the output that should appear:

i For an Asset Rule linked to an Asset Type, the 'Use Existing Token Values' option is disabled for the Stream Reference Token since the token values won't be guaranteed to match Stream Reference Names from the Asset Type. Instead, use the 'Rename Token Values' option to map token values.

☐ Use Existing Token Values
☒ Rename Token Values

Generate Mappings **+ Add Mapping** **Remove All Mappings**

Mappings

Aftercooler_Temperature	→	Map To...	
Brake_Air_Pressure	→	Map To...	
Engine_Coolant_Temperature	→	Map To...	
Engine_Fuel_Rate	→	Map To...	

We are going to map each of the mappings to the correct mappings defined in our Asset Type. This will allow the asset rule to associate each individual measurement that is found to the correct property defined within the Asset Type used for this asset rule.

4. Click on the 'Map To...' empty field and select the appropriate mappings.

Once the following is completed, you will be able to move to the third step of Asset Configuration.

i For an Asset Rule linked to an Asset Type, the 'Use Existing Token Values' option is disabled for the Stream Reference Token since the token values won't be guaranteed to match Stream Reference Names from the Asset Type. Instead, use the 'Rename Token Values' option to map token values.

☐ Use Existing Token Values
☒ Rename Token Values

Generate Mappings **+ Add Mapping** **Remove All Mappings**

Mappings

Engine_RPM	→	Engine RPM	
Ground_Speed	→	Ground Speed	
Latitude	→	Latitude	
Longitude	→	Longitude	

Cancel **Back** **Next**

Step 3: Configure the Asset


In this step, you specify how the rule builds assets by assigning the tokens to asset fields. When the assets are generated, the tokens are replaced with the value mappings.

1. In the Configure Asset pane, for each of the following asset fields enter the sequence of tokens and characters that resolve to create a value for each asset. To pick from a list of tokens, enter { and select a token.
 - Id - The Id must be unique for each asset. If the ID is not unique, the generated assets may incorrectly reference streams that belong to another asset.
 - Name - Enter the sequence of tokens and characters that resolve to create the name for each asset. To pick from a list of tokens, enter { and select a token.
 - Description - Optional.

For our example, we will enter the following:

- Id: {TruckID}-YourStudentID-piworld.
- Name: {TruckID}-YourStudentID.
- Description: YourStudentID Truck for PI World 2021.

The output should appear like this:

Id	{TruckID}-jsmith-piworld			
Name	{TruckID}-jsmith			
Description	jsmith Truck for PI World 2021			
Stream Reference Na...	{Measurement} 			
Metadata	Name	Value Expression	Value	Type
	Customer	Type '{f' to display tokens	<No matching value>	String ▾
	Engine Model	Type '{f' to display tokens	<No matching value>	String ▾
	Manufacturer	Type '{f' to display tokens	<No matching value>	String ▾
	Mine Truck Model	Type '{f' to display tokens	<No matching value>	String ▾
	Region	Type '{f' to display tokens	<No matching value>	String ▾
	Site	Type '{f' to display tokens	<No matching value>	String ▾

Step 4: Preview the Assets

1. The asset preview displays a list of the assets that will be created using the asset rule. Use the preview to verify that the rule creates all the assets you expect, and they are created correctly.

In this step, we will confirm all 10 mining truck assets have been created properly.

Name	Description
Truck 101-jsmith	jsmith Truck for PI ...
Truck 102-jsmith	jsmith Truck for PI ...
Truck 103-jsmith	jsmith Truck for PI ...
Truck 104-jsmith	jsmith Truck for PI ...
Truck 105-jsmith	jsmith Truck for PI ...
Truck 106-jsmith	jsmith Truck for PI ...
Truck 107-jsmith	jsmith Truck for PI ...
Truck 108-jsmith	jsmith Truck for PI ...
Truck 109-jsmith	jsmith Truck for PI ...
Truck 110-jsmith	jsmith Truck for PI ...











2. You should see only 10 trucks created, where the Trucks have the pattern of Truck ID concatenated with your studentID.
3. Click on Save & Execute in order to go ahead and create these assets.

8. Remote Operations Monitoring using OCS


8.1 Monitor and manage your assets

OSIsoft Cloud Services (OCS) makes it possible to monitor remote assets in real time. The immediate access to data about the status of assets gives you the ability to anticipate problems and proactively perform preventative maintenance. The following procedure describes how to use the OCS Portal to monitor your assets and quickly identify problems.


1. Click Visualization > Asset Explorer.
2. In the search box, enter *YourStudentID*
3. Scan through the assets and identify any assets with a problematic status.

Truck 101-jsmith 	Truck 102-jsmith 	Truck 103-jsmith 	Truck 104-jsmith 	Truck 105-jsmith 
Truck 106-jsmith 	Truck 107-jsmith 	Truck 108-jsmith 	Truck 109-jsmith 	Truck 110-jsmith 

Each asset is identified with one of the following statuses:

Value		Status
Running	→	
Under Maintenance	→	
Stopped	→	

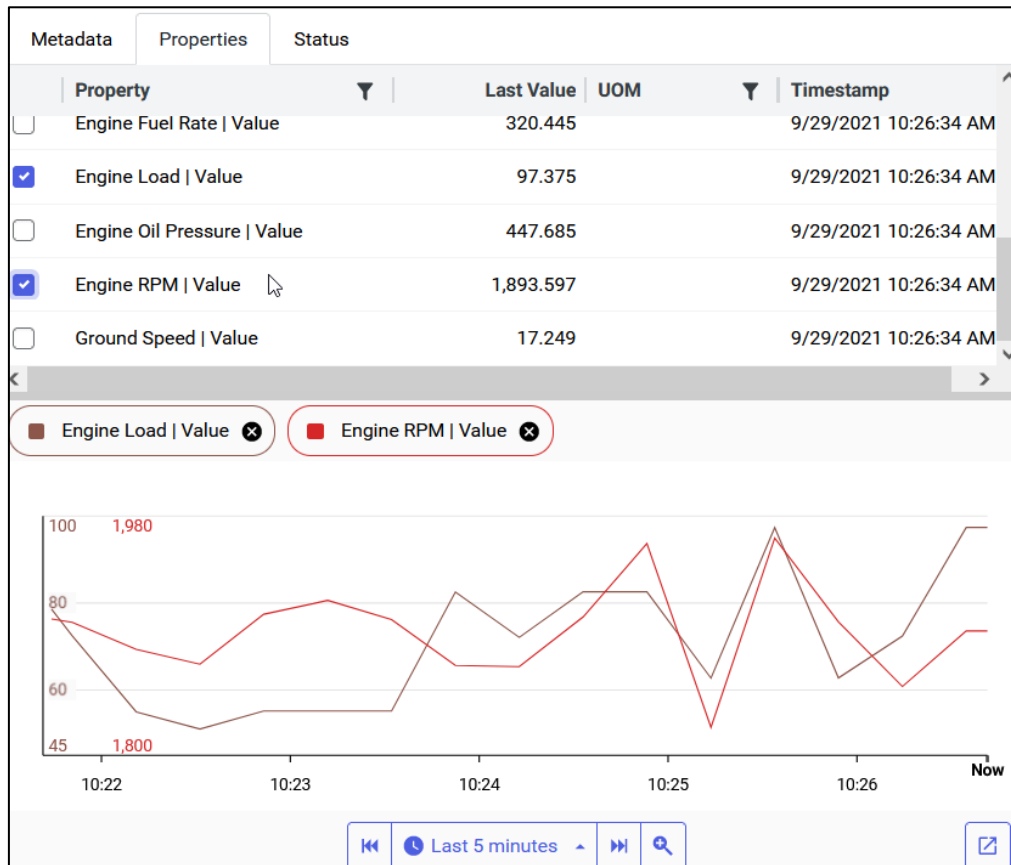
4. Click on a Truck asset to open the Asset details pane. The Asset Details pane provides metadata and property data on the asset that you can use to determine the cause of any problems. The Metadata tab displays metadata associated with the asset.

Truck 101-jsmith 	
jsmith Truck for PI World 2021	
Metadata	Properties Status
Metadata	Value
Customer	AvevaGold
Engine Model	Cat C175-22
Manufacturer	Caterpillar
Mine Truck Model	790F BC
Region	NAMER
Site	USA-Alaska

5. Click the Properties tab.

The Asset Details pane displays the following:

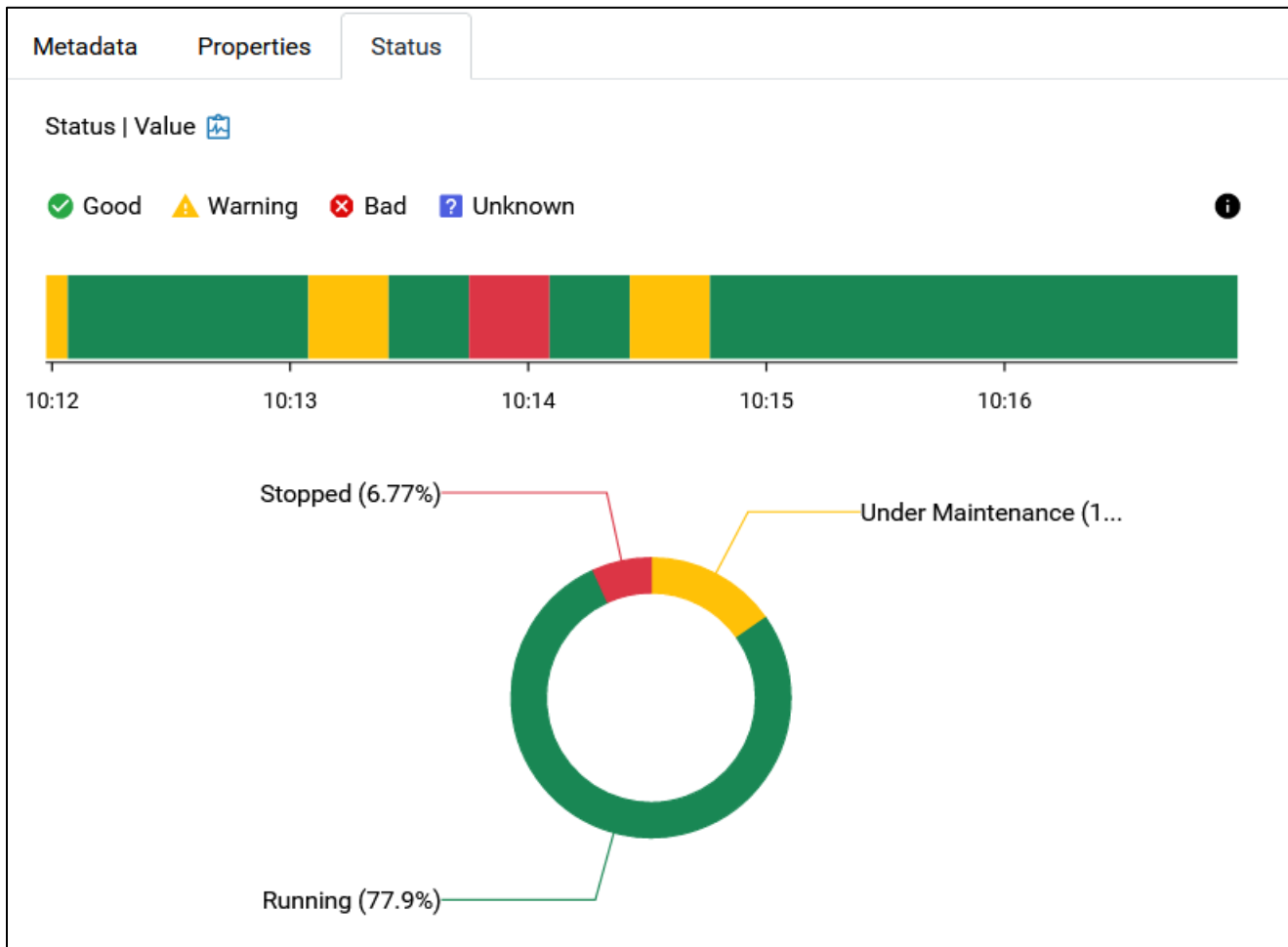
- Data associated with the asset. These values are updated in real time.
- A trend of the selected measurements.



6. Click the Status tab.

The Asset Details pane displays the following:

- Data associated with the asset. These values are updated in real time.
- A trend of the selected measurements.

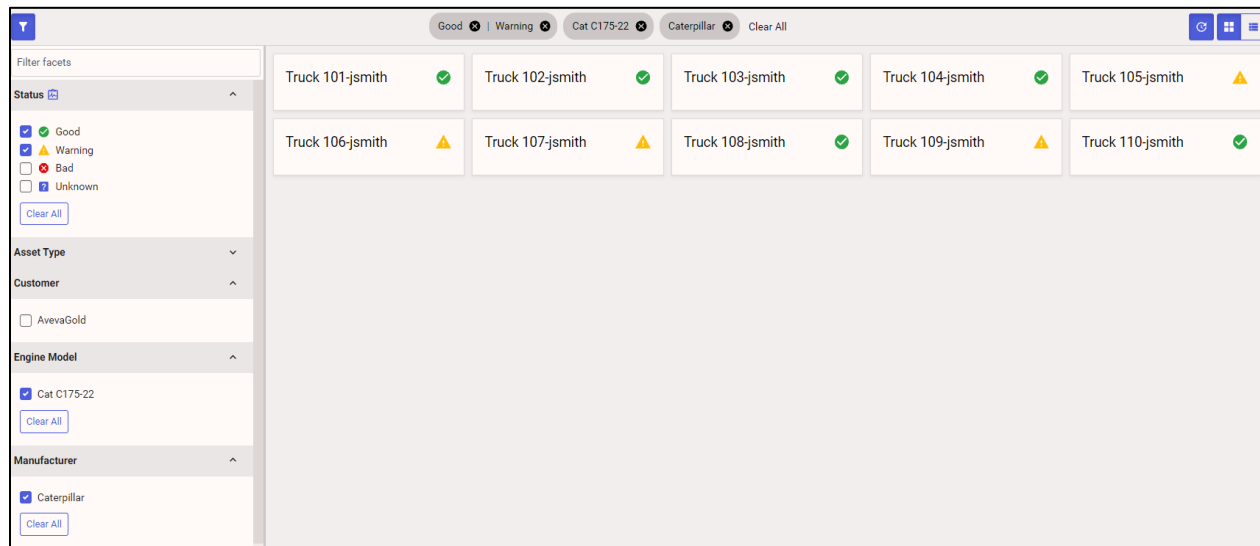


8.2 Filter assets in the Asset Explorer

Filters allow you to find specific assets quickly regardless of how many assets your organization has. Customize and refine your search filter by selecting multiple filter values. Use metadata filters to refine your search using these filter facets:

- Status – Filter for assets that have the same status. For example, filter for assets with a status of Warning or Bad to identify assets that may need attention.
- Asset Type – Filter for assets based on asset type.
- Metadata – Filter for assets based on metadata, such as Location, Manufacturer, Province or State, Region, or Model.

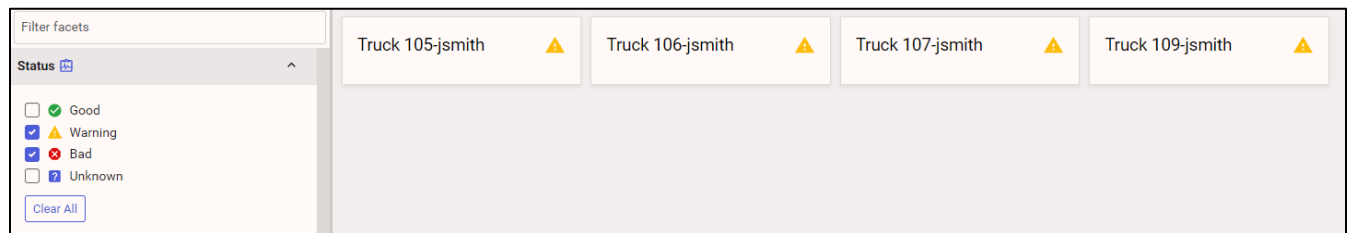
1. Navigate to Visualization > Asset Explorer.
2. Select one or more values from one or more facets to apply the filters to the listed assets.



8.3 Share a view of your fleet

The asset explorer allows you to filter and view a subset of the assets and share this view with others. For instance, you want to obtain a list of all the stopped trucks, and share the assets view to your maintenance personnel.

1. Click Visualization > Asset Explorer.
2. Verify that the Assets/Asset Type selector is set to Assets.
3. Enter a string in the Search box to filter the assets that are displayed. Eg: Status: Stopped.



4. Click the Share icon to copy the URL to the clipboard.
5. This link can now be sent to your maintenance personnel. When they paste this URL into a browser, they will see the fleet view you created.

9. Visualize SDS Data

9.1 Utilizing Trend feature of OSIsoft Cloud Services

Use trace data in a trend to monitor assets, anticipate problems, and proactively perform preventative maintenance. The following procedure describes how to display traces in your trend and glean useful information from your data.

1. Click the navigation icon and click Trend (under Visualization).
2. In the Add Traces blade, click the + sign to add the trace to the trend.
3. Click Step backward or Step forward to move the time range of the data displayed in the trend.
4. Explore *context switch* in visual trending by following the link to the documentation.
 - Asset Switch in visual trending.
 - Asset switch in performance testing.
5. Click on a trace to select it for further analysis.
6. Click the plus sign (+) above the trace to lock the cursors in place. The + turns into an x. To unlock the cursor, click the x.

Note: When two cursors are locked, the Legend table displays summary calculations for the values between the two cursors, known as the Cursor view.

7. Click the **share** icon in the menu bar to copy the URL of the workspace. You can share this URL with colleagues to give them the same view of the trend, which they can use to troubleshoot problems.



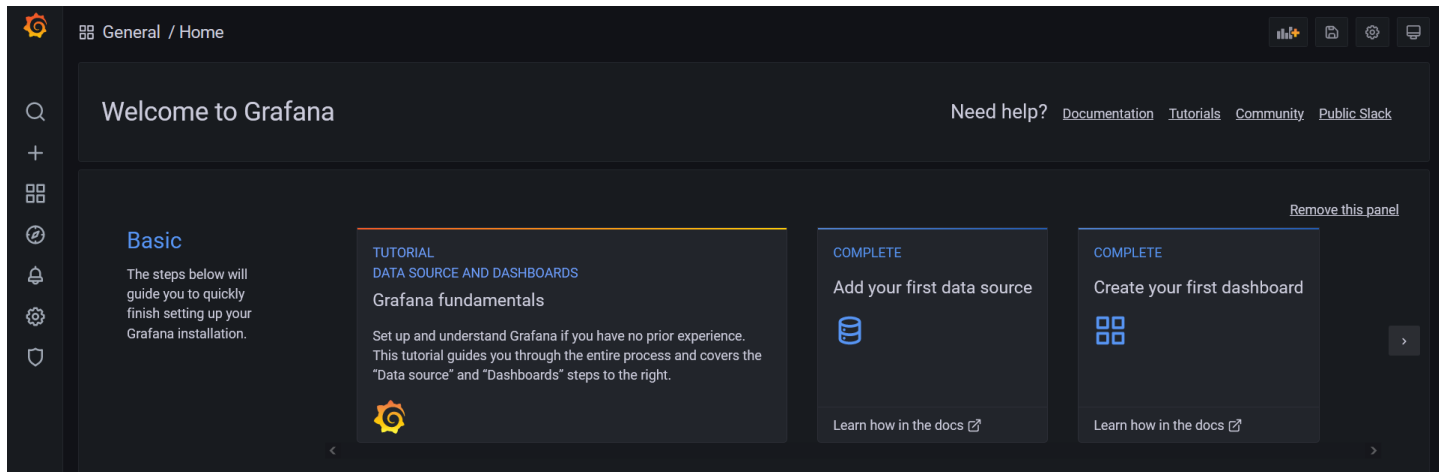
9.2 Utilizing Grafana

OSIsoft's R&D have created a plugin for an open-source visualization application Grafana. In this case, our team created an example that allows Grafana dashboards to use a Sequential Data Store as a data source, for trending real-time data from OSIsoft Cloud Services. Please refer to this github repository that contains resources that you require to build and load a Grafana plugin, so you can visualize SDS data.

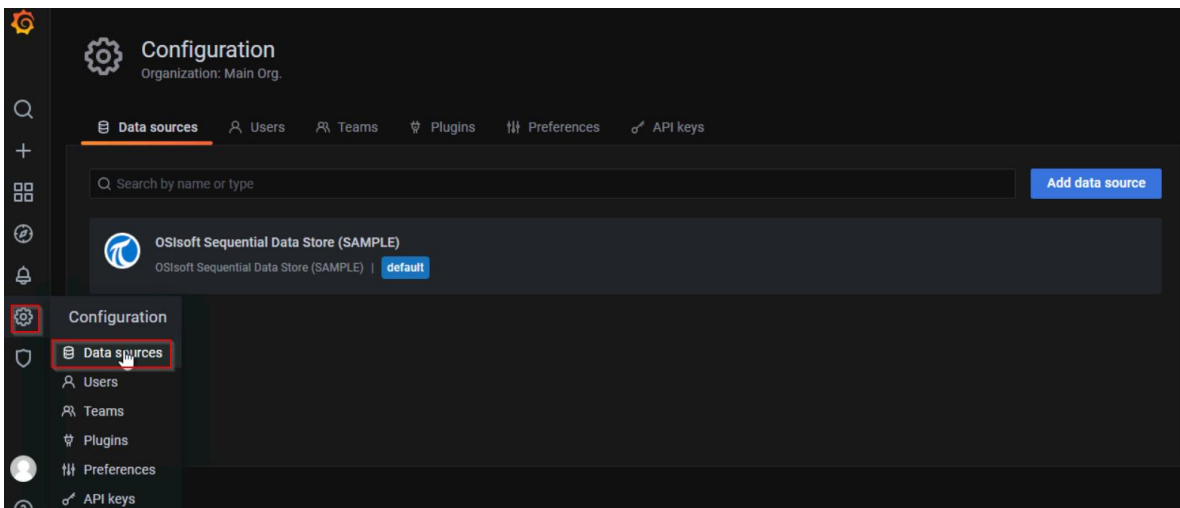
GitHub Link: <https://github.com/osisoft/sample-ocs-grafana-nodejs>

For the sake of simplicity, Grafana has been pre-installed in your VM. In addition, the **Grafana Sequential Data Store** plugin has been built and loaded into the Grafana server. Follow the below steps in order to visualize SDS data in your local Grafana instance:

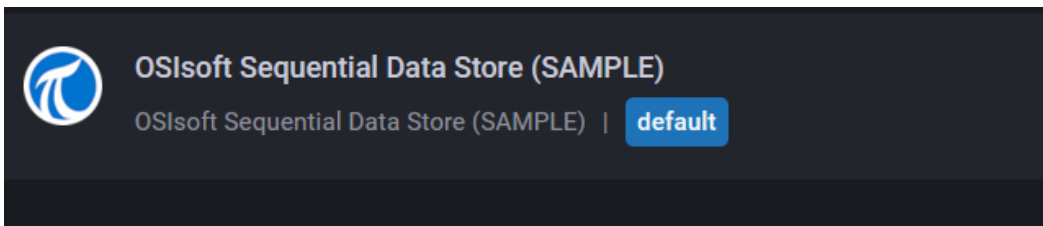
1. Launch Windows Services and verify whether Grafana service is in the running state.
2. From your desktop, launch "Grafana Dashboard". This will take you to your Grafana dashboard.
3. Login using the following credentials, if prompted:
 - Username: admin
 - Password: admin



4. Click the “Configuration” gear icon> Data sources.



5. Under Data sources, you will find OSIsoft Sequential Data Store (SAMPLE) added.

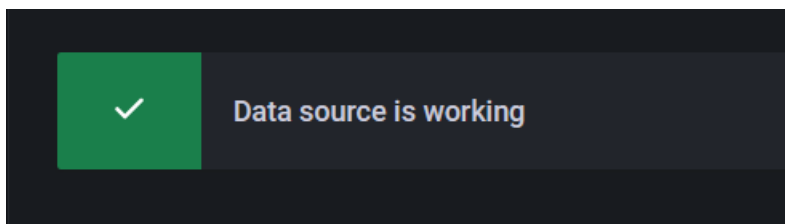


6. Click OSIsoft Sequential Data Store (SAMPLE) and verify the following settings:
- Tenant ID (you can obtain Tenant Id by going to Data Management > Namespaces -> Remote Operations Monitoring > Tenant ID).
 - Client ID : 3cc44a66-5cfa-455a-918a-aeb00a4a51fa (Note: Client Secret is pre-loaded).
 - Namespace: Remote Operations Monitoring.

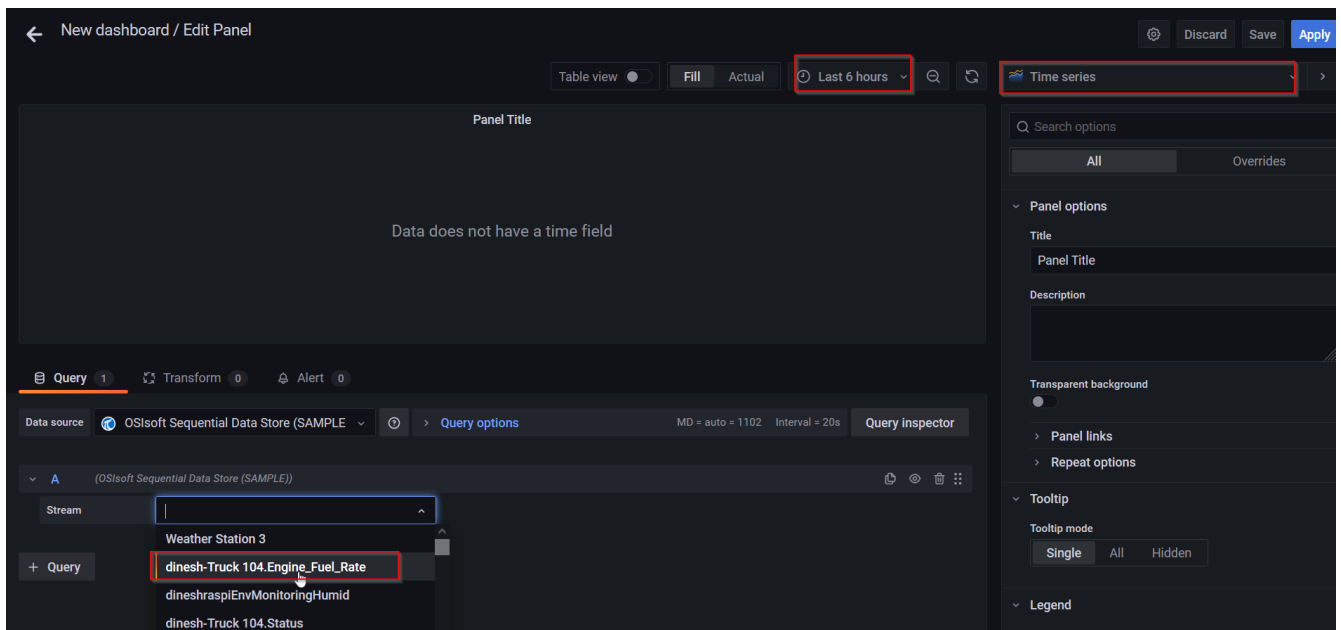
OSIsoft Cloud Services

URL	①	<input type="text" value="https://dat-b.osisoft.com"/>
API Version	①	<input type="text" value="v1"/>
Tenant ID	①	<input type="text" value="95e3e8c9-b327-4b36-92de-d61044f25f76"/>
Community Data	①	<input type="checkbox"/>
Namespace	①	<input type="text" value="Remote Operations Monitoring"/>
Use OAuth token	①	<input type="checkbox"/>
Client ID	①	<input type="text" value="3cc44a66-5cfa-455a-918a-aeb00a4a51fa"/>
Client Secret	①	<input type="password" value="....."/> <input type="button" value="Reset"/>

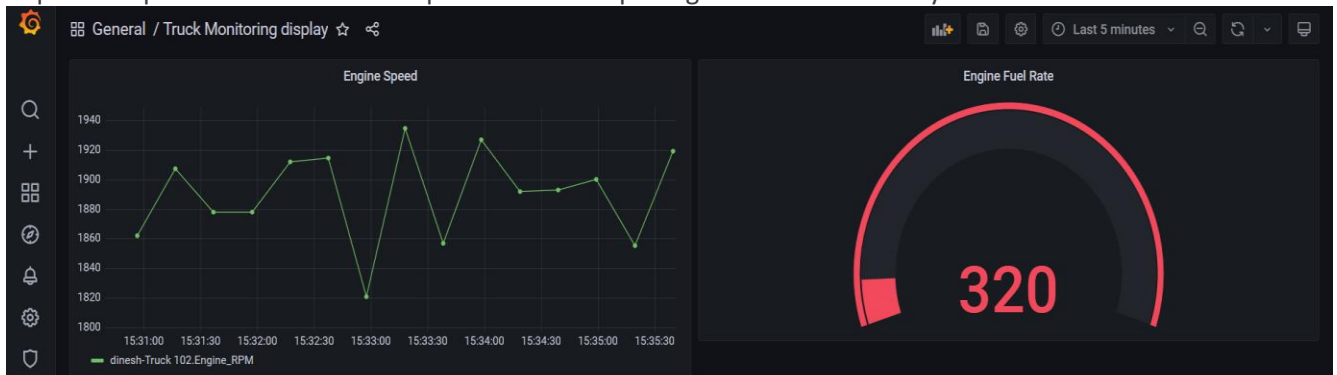
7. Click Save and test. You should receive a confirmation that the data source is working.



8. Navigate to the Dashboards list and add an empty panel. Select one of your streams under the Stream field, select the type of the visualization, and the duration to get your stream visualized.



9. Repeat the process to add another panel- this time pulling another stream of your choice.



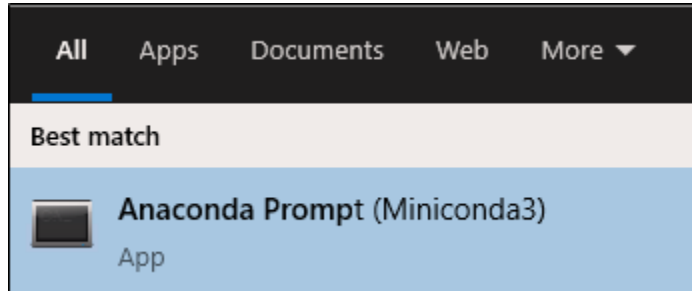
This provides you an idea of how you can visualize SDS data in Grafana allowing you to extend your analysis.

10. Miscellaneous Remarks (Appendix)

Note: During the lab, if the data simulator does not work (i.e. the data in the MQTT Explorer Client is not updating OR if the 'Data Simulator PI World ROM' scheduled task is stuck on a 'Ready' State).

Perform the following tasks:

1. Open Anaconda Prompt in the search bar menu.



2. Enter the following command: `cd "C:\OCS ROM 2021\MQTTSimulator"`.
3. Enter the following command: `python main.py`

The Steps above outline the manual process to initiate and start the data simulator required for this lab. This is to be only used in the case where an impromptu issue occurs with the Windows Task Scheduler.

Once this is done, DO NOT CLOSE the anaconda prompt window, simply minimize it, and continue with the lab.