Critical manufacturing
an ASM PT company

# Event Ingestion in Data Platform

11.2

February 2026

DOCUMENT ACCESS

Public

DISCLAIMER

The contents of this document are under copyright of Critical Manufacturing S.A. it is released on condition that it shall not be copied in whole, in part or otherwise reproduced (whether by photographic, or any other method) and the contents therefore shall not be divulged to any person other than that of the addressee (save to other authorized offices of his organization having need to know such contents, for the purpose for which disclosure is made) without prior written consent of submitting company.

# Event Ingestion in Data Platform

*Estimated time to read: 5 minutes*

Event ingestion is the process of publishing events for immediate consumption and/or storage in the **Data Platform** backend database. Events can be streamed in real time, where each event is processed as the source emits it, or ingested in batches.

This tutorial will demonstrate how to use a basic **Data Platform** event to ingest data.

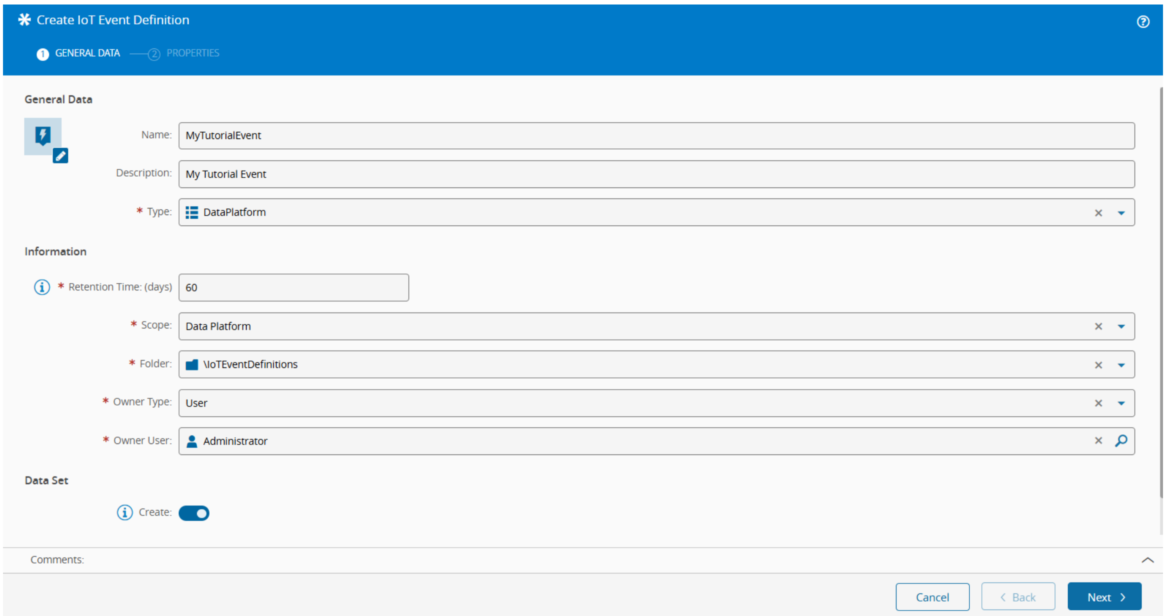
## Overview

In this example we will create an event that receives values from different sensors on a shop floor, and show how that data is made available as it is being published.

## Creating an IoT Event Definition

In the **Data Platform** page group of the main menu, navigate to the **IoT Event Definitions** tile and create a new **IoT Event Definition**:

- Enter a name - bear in mind that the name will identify the **IoT Event** throughout the **Data Platform** infrastructure, so it should be a meaningful name for later reference.
- Set the Type to **Data Platform**.
- Set the Retention Time to `60` days.
- Set the Scope to **Data Platform**.
- Select the desired Owner Type.
- In the **Data Set** section, set the **Create** option to `true` - if set to `true`, a dataset will be created for the data ingested with this event; always set this option to `true` if direct access to the data will be required in the future (example: data collections).
- If the Create Data Set option is enabled, you must define a Retention Time (for example, `60` days).



General Data

Name: MyTutorialEvent

Description: My Tutorial Event

* Type: DataPlatform

Information

* Retention Time: (days) 60

* Scope: Data Platform

* Folder: \IoTEventDefinitions

* Owner Type: User

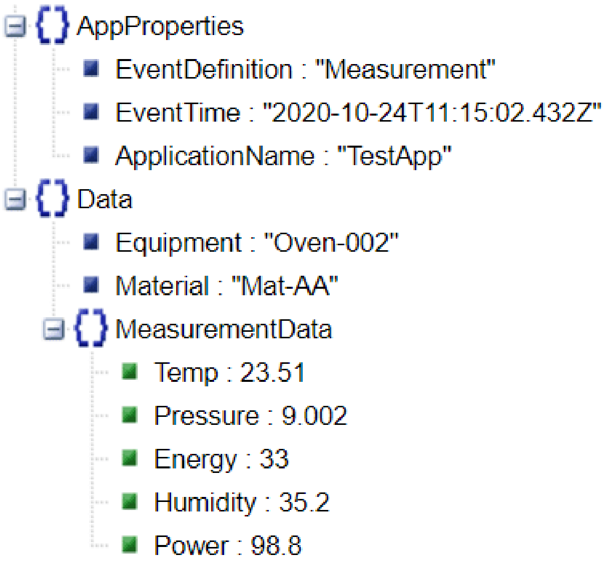
* Owner User: Administrator

Data Set

Create: ●

Comments:

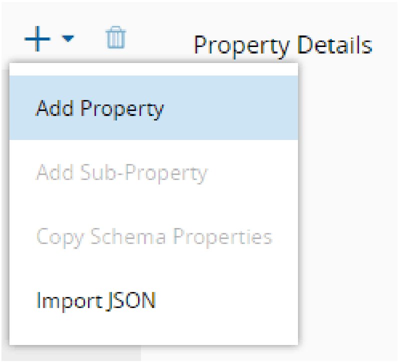
We will create an event with the following structure:



```
{ } AppProperties
    EventDefinition : "Measurement"
    EventTime : "2020-10-24T11:15:02.432Z"
    ApplicationName : "TestApp"
{ } Data
    Equipment : "Oven-002"
    Material : "Mat-AA"
    { } MeasurementData
        Temp : 23.51
        Pressure : 9.002
        Energy : 33
        Humidity : 35.2
        Power : 98.8
```

**AppProperties** is a set of properties that the publisher of the events (example: the client that will produce the events) should add to identify itself, while **Data** is the actual event payload.

Adding properties to the event definition can be done manually or by importing a JSON structure such as the one shown above. The system will infer the data types from the actual values and generate a property for each of the fields.



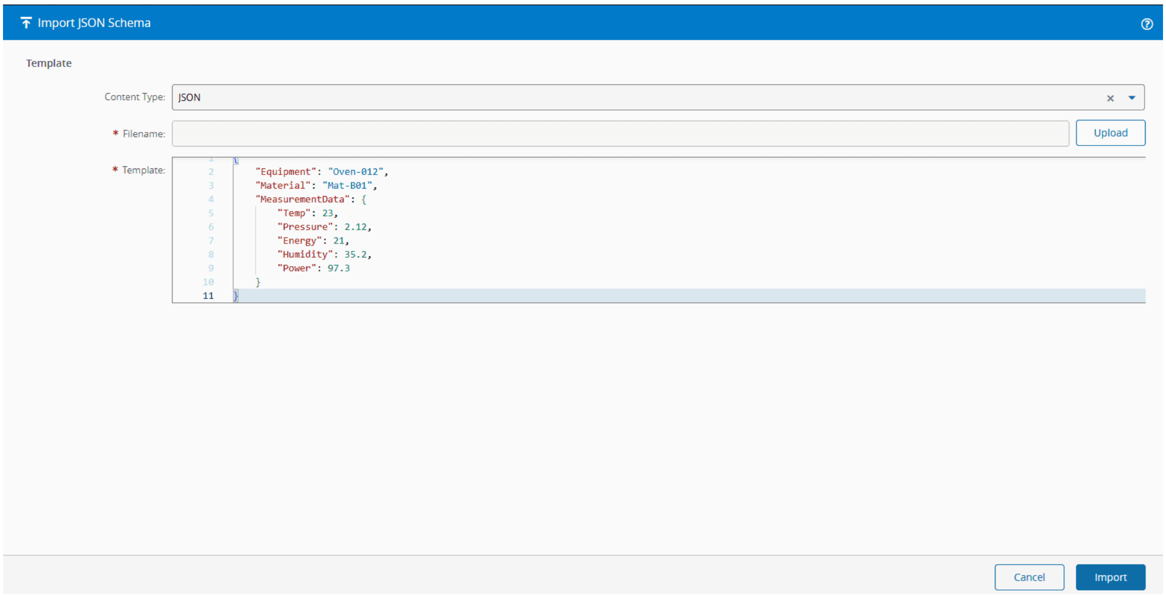
Several properties are essential for the proper functioning of the event ingestion, so specify for each property:

- **Name** - name of the property, that must matches the property in the JSON document that represents the event.
- **Array** - indicates if the property is an array of objects.
- **Mandatory** - indicates if the property is mandatory. If `true`, this information will be used to validate all the arriving events.
- **Indexed** - indicates if the property should be indexed. If `true`, the property will become part of the key that uniquely identifies an event; events with the same key will be automatically deduplicated; if the client sends many events with the same key, then only the event that was sent last will be saved in the data set.
- **Data Type** - the data type of the property. Type **IoTSchema** allows you to create sub-groups of properties.
- **Default Value** - specifies the default value for the property. If the property is not present in the json of the event that was posted, the default value will be set automatically.

**Info**

For more information, see Create IoT Event Definition.

As previously mentioned, rather than adding the properties one by one, it is possible to import the structure by selecting **Import JSON** and providing a sample JSON:



Import JSON Schema

Template

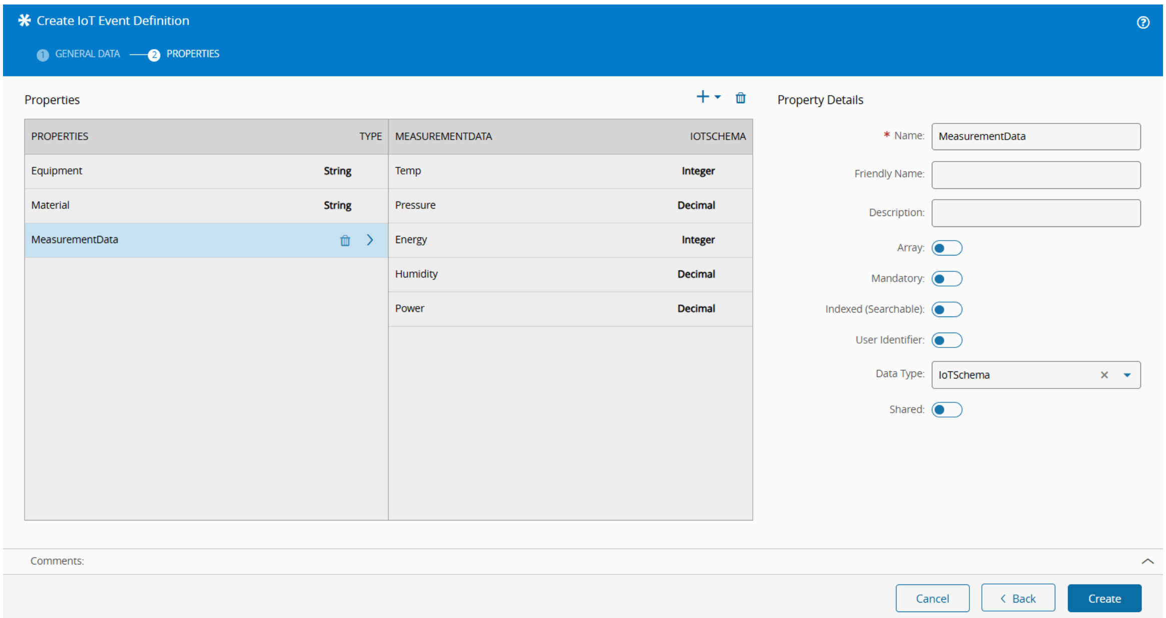
Content Type: JSON

* Filename:

* Template:

```
{
    "Equipment": "Oven-012",
    "Material": "Mat-B01",
    "MeasurementData": {
        "Temp": 23,
        "Pressure": 2.12,
        "Energy": 21,
        "Humidity": 35.2,
        "Power": 97.3
    }
}
```

After creating the properties manually or importing from a JSON file, the **IoT Event Definition** can be created.



Create IoT Event Definition

Properties

| PROPERTIES | TYPE |
|---|---|
| Equipment | String |
| Material | String |
| MeasurementData | |

| MEASUREMENTDATA | IOTSCHEMA |
|---|---|
| Temp | Integer |
| Pressure | Decimal |
| Energy | Integer |
| Humidity | Decimal |
| Power | Decimal |

Property Details

* Name: MeasurementData

Friendly Name:

Description:

Array:

Mandatory:

Indexed (Searchable):

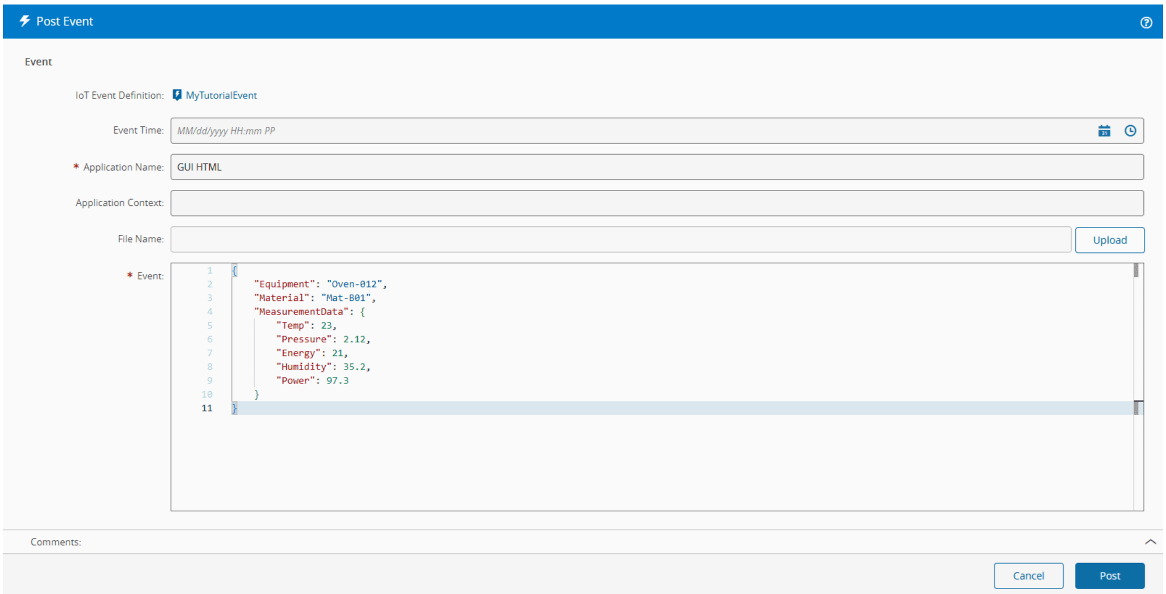
User Identifier:

Data Type: IoTSchema

Shared:

Comments:

When the event is created, it is possible to start posting events using the Post Event host API. We can simulate this by using the **Post** button in the **IoT Events** page. Simply navigate to that page by going to the **IoT Event Definitions** page (which you can access from the **Data Platform** page group), and select the **IoT Event** you created from the list. Then, select the **Post** button on the top ribbon, and type the JSON document for the event you want to post:



Event

IoT Event Definition: MyTutorialEvent

Event Time: MM/dd/yyyy HH:mm PP

* Application Name: GUI HTML

Application Context:

File Name:

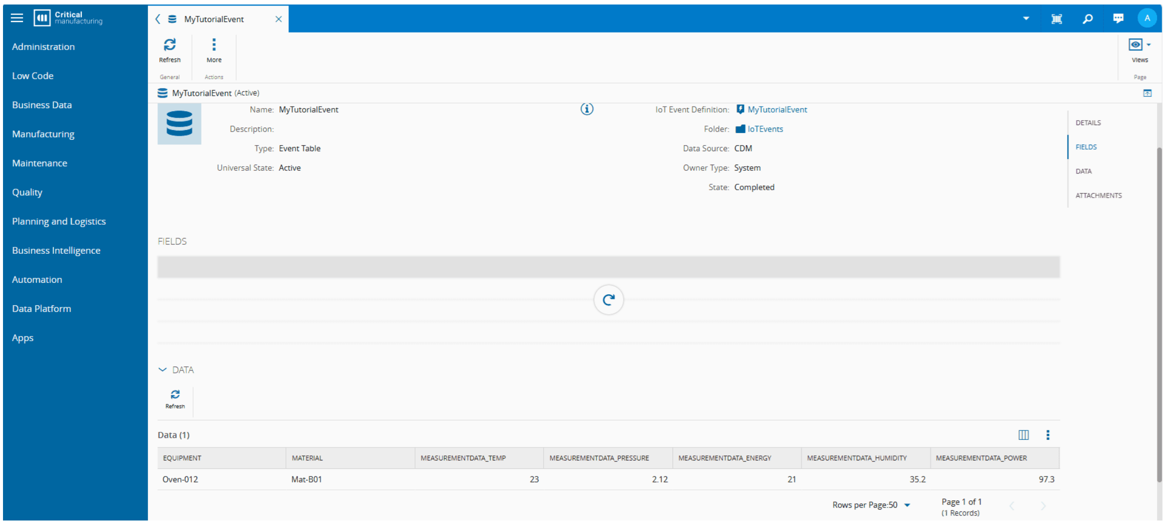
* Event:

```
{
    "Equipment": "Oven-012",
    "Material": "Mat-B01",
    "MeasurementData": {
        "Temp": 23,
        "Pressure": 2.12,
        "Energy": 21,
        "Humidity": 35.2,
        "Power": 97.3
    }
}
```

Comments:

Since the **Create** option was set to `true` when we created the **IoT Event Definition**, once this event is posted we can immediately see the data in the corresponding **Data Set**. Simply select the **Data Sets** tile in the **Data Platform** page group, then navigate the **IoTEvents** folder to find the data set for the event. The data set name and folder will match the name and folder of your iot event.

When you find the data set, simply select the **Data** section and you'll be able to see the data for the event that you just posted:



MyTutorialEvent (Active)

Name: MyTutorialEvent

Description:

Type: Event Table

Universal State: Active

IoT Event Definition: MyTutorialEvent

Folder: IoTEvents

Data Source: CDM

Owner Type: System

State: Completed

Data (1)

| EQUIPMENT | MATERIAL | MEASUREMENTDATA_TEMP | MEASUREMENTDATA_PRESSURE | MEASUREMENTDATA_ENERGY | MEASUREMENTDATA_HUMIDITY | MEASUREMENTDATA_POWER |
|---|---|---|---|---|---|---|
| Oven-012 | Mat-B01 | 23 | 2.12 | 21 | 35.2 | 97.3 |

Rows per Page:50

Page 1 of 1
(1 Records)

This data is now available to be accessed in multiple ways (OData, Grafana Dashboards, etc.). Thus, if you have a client continuously posting events using the Post Event host API, **Data Platform** will make this data immediately available for consumption.

# Legal Information

### Disclaimer

The information contained in this document represents the current view of Critical Manufacturing on the issues discussed as of the date of publication. Because Critical Manufacturing must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Critical Manufacturing, and Critical Manufacturing cannot guarantee the accuracy of any information presented after the date of publication. This document is for informational purposes only.

Critical Manufacturing makes no warranties, express, implied or statutory, as to the information herein contained.

### Confidentiality Notice

All materials and information included herein are being provided by Critical Manufacturing to its Customer solely for Customer internal use for its business purposes. Critical Manufacturing retains all rights, titles, interests in and copyrights to the materials and information herein. The materials and information contained herein constitute confidential information of Critical Manufacturing and the Customer must not disclose or transfer by any means any of these materials or information, whether total or partial, to any third party without the prior explicit consent by Critical Manufacturing.

### Copyright Information

All title and copyrights in and to the Software (including but not limited to any source code, binaries, designs, specifications, models, documents, layouts, images, photographs, animations, video, audio, music, text incorporated into the Software), the accompanying printed materials, and any copies of the Software, and any trademarks or service marks of Critical Manufacturing are owned by Critical Manufacturing unless explicitly stated otherwise. All title and intellectual property rights in and to the content that may be accessed through use of the Software is the property of the respective content owner and is protected by applicable copyright or other intellectual property laws and treaties.

### Trademark Information

Critical Manufacturing is a registered trademark of Critical Manufacturing.

All other trademarks are property of their respective owners.

Copyright © 2023 Critical Manufacturing. All rights reserved.