EVENT AUTOMATION

Exercise: Energy readings

NIKI HEYLIGEN
INTEGRATION DESIGNERS
Veldkant 33B
Kontich 2550

Introduction

Today we will see how we can use IBM Event Automation (EA) to enable users to use real-time data to create new opportunities

EA consists out of 3 products:

- IBM Event Streams
- IBM Event Endpoint Management
- IBM Event Processing

Each platform will have dedicated credentials. You will find the required credentials in the correct subfolder:

- ES Credentials
- EEM Credentials
- EP credentials

The main exercise

Goal

Build a kafka setup to provide your customers with real time pricing updates. The exercise will contain a mix of the following steps:

- Creating topics
- Defining avro contracts
- Publishing topics (for producers and consumers)
- Creating credentials and subscriptions
- Creating Flink processes
- Update camel integrations

Extra challenge

Add a kafka application in Java (which consumes from and produces to one or more kafka topics) to create a customer and link it to an existing meterId or EAN number. Get real-time notifications when the energy prices are favorable to charge your electric vehicle or when to stop charging.

Important: Do not skip any step --> this will result in unexpected behaviour

Let's get started!

Usefull information

Users

Naming convention

For every object you create, use a reference to your ID as prefix. Example give:

- ST01: ST01.READINGS
- ST12: ST12.READINGS

I will use PXL.ID as prefix. Example:

- PXL.ID.READINGS

Vocabulary

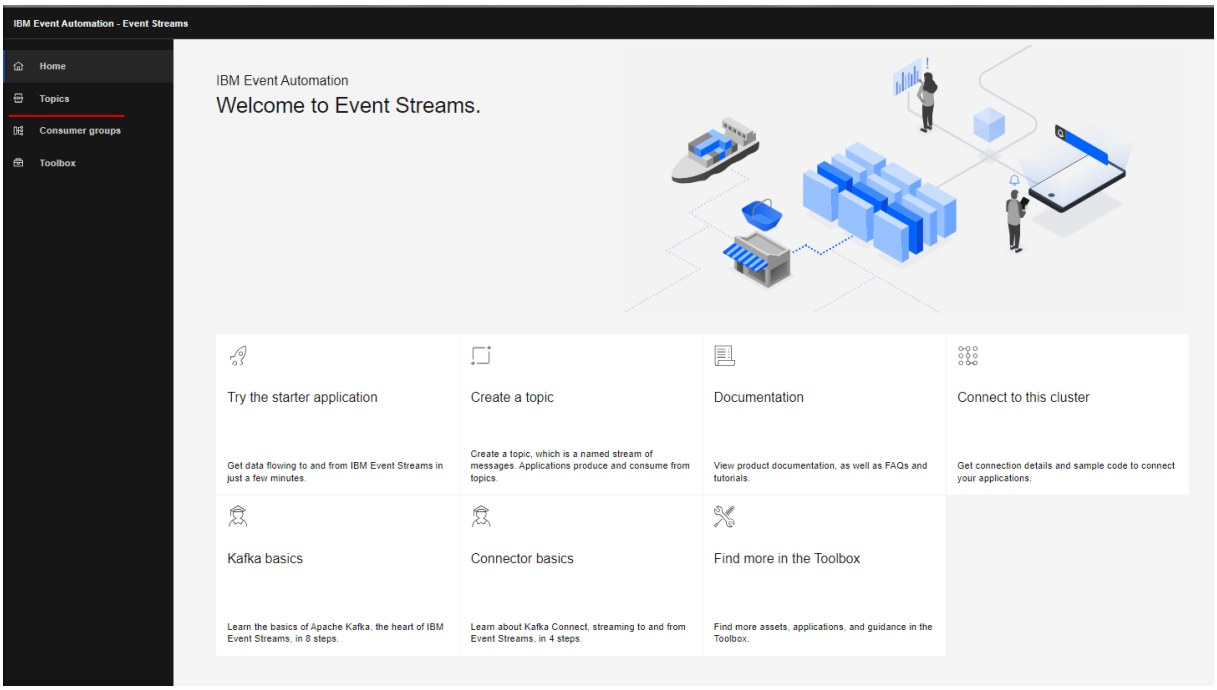
Event Automation	EA	An IBM stack containing the three products ES, EEM, EP
Event Streams	ES	IBM flavoured kafka solution
Event Endpoint Management	EEM	IBM Event Gateway + catalog
Event Processing	EP	IBM flavoured Flink solution
Topic		Historical log
Catalog		A market place for conumers to discover the products/services provided
kafka topic alias		An alias which the consumer or producer will refer to in the connection details

EX.0 Exploration

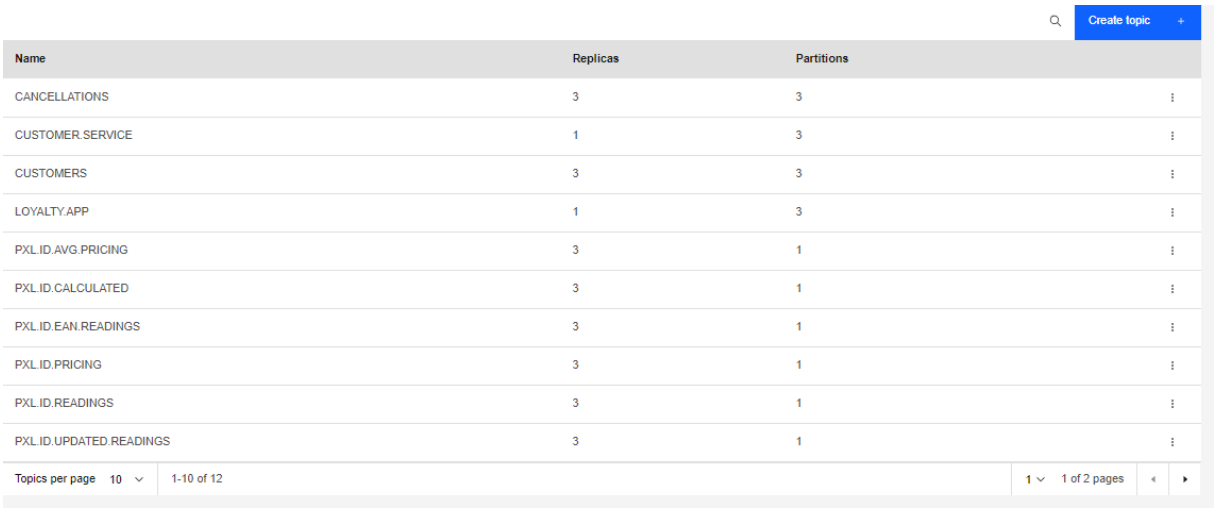
IBM Event Streams

Before we start, lets explore the products we are about to use

- Go to IBM ES: [IBM Event Streams]
- Login using your user.

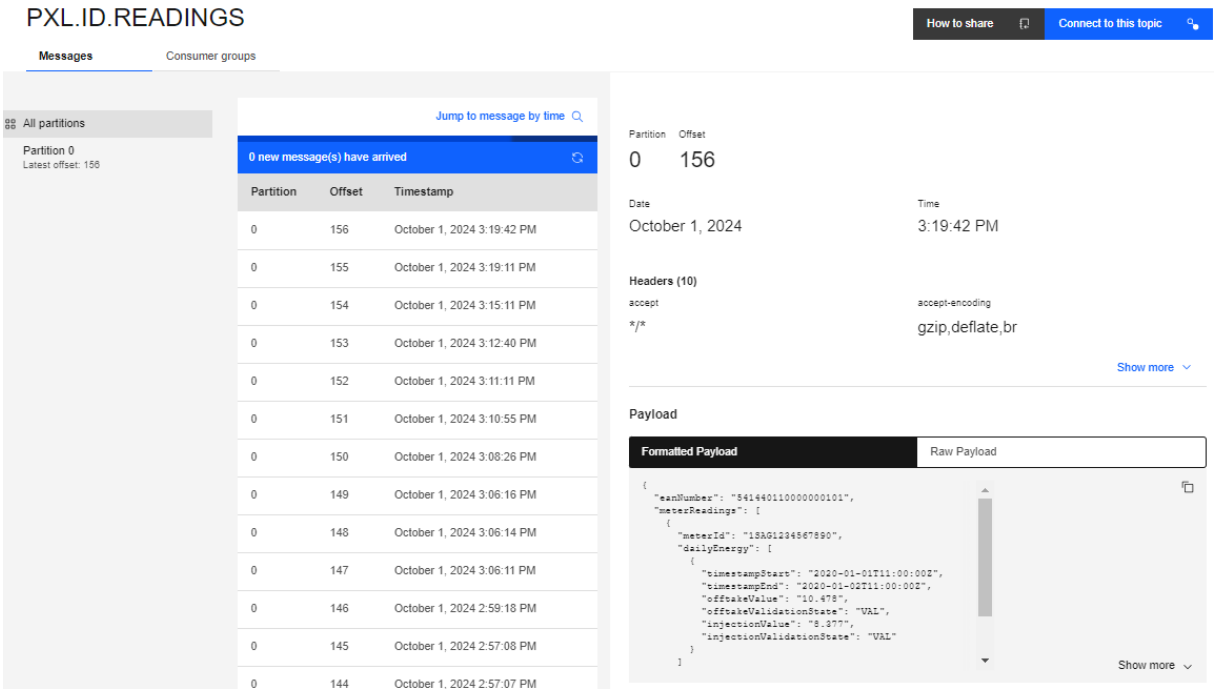


- **The start application** gives you the means to create a basic interface to publish and consume messages to and from a topic.
- **Kafka basics** and **Connector basics** gives you a simple explenation on how kafka and the connectors work.
- **Documentation** is a link to an IBM documentation webpage
- **Connect to this cluster** gives you all info, sample code included, on how to connect to this kafka cluster
- **Create a topic** is an interface to help you build a topic
- **Toolbox** gives you extra info and tools to help you work with kafka. Example:
    - A CLI
    - Java clients
    - Connector info
- **Topics (**On the left top): gives you a view on all topics available



Name	Replicas	Partitions
CANCELLATIONS	3	3
CUSTOMER.SERVICE	1	3
CUSTOMERS	3	3
LOYALTY.APP	1	3
PXL.ID.AVG.PRICING	3	1
PXL.ID.CALCULATED	3	1
PXL.ID.EAN.READINGS	3	1
PXL.ID.PRICING	3	1
PXL.ID.READINGS	3	1
PXL.ID.UPDATED.READINGS	3	1

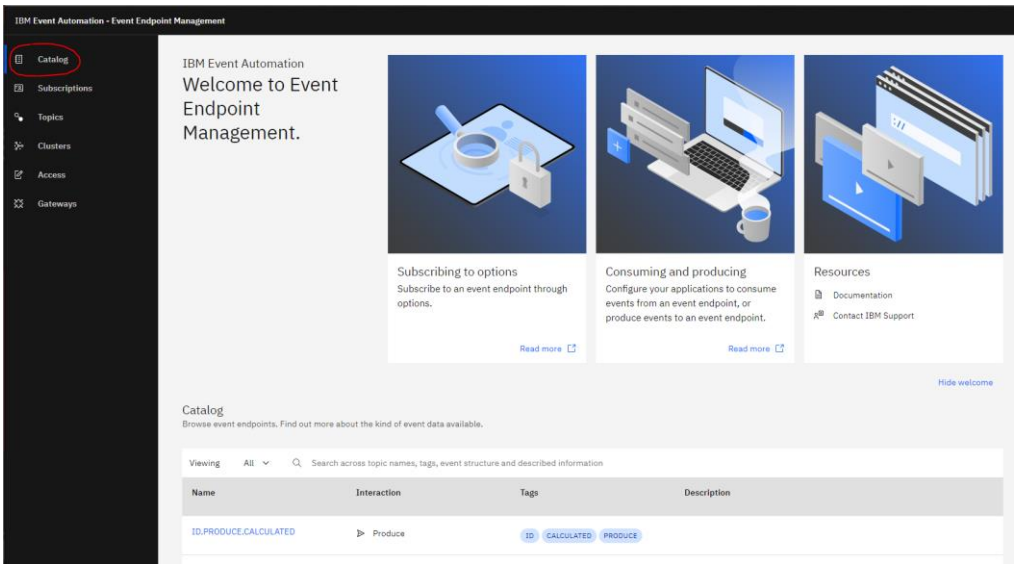
1. When you open one of these topics (example given: PXL.ID.READINGS) you can see the events sent to the topic



PXL.ID.READINGS

IBM Event Endpoint Management

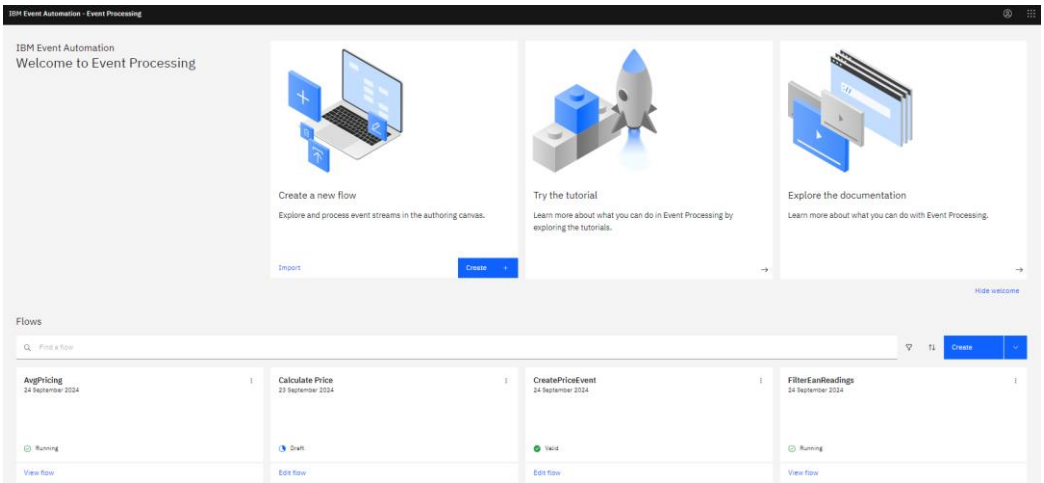
2. Go to EEM: Event Endpoint Management
3. Login using your user.
4. Your user will have administrator permissions. A viewer (client) will only see the catalog page



- **Catalog**: this page shows the available views for the topics available. Either to produce or the consume.
    - When you open a topic, you can see what type of connection it is (produce or consume).
    - It will show you the Avro schema (If available)
    - It will show you an example message (if available)
    - It contains
        - the server info
        - Certificates to download
        - Example codes (Java, Node, kcat)
        - Maven dependencies
- **Subscriptions** helps you to manage the active subscriptions and accept or denie new requests.
- **Topics** allows you to include new topics from ES to produce or consume events

Event Processing

1. Go to EP: Event Processing
2. Login using your user.



Ex1. Digital meter readings

Goal

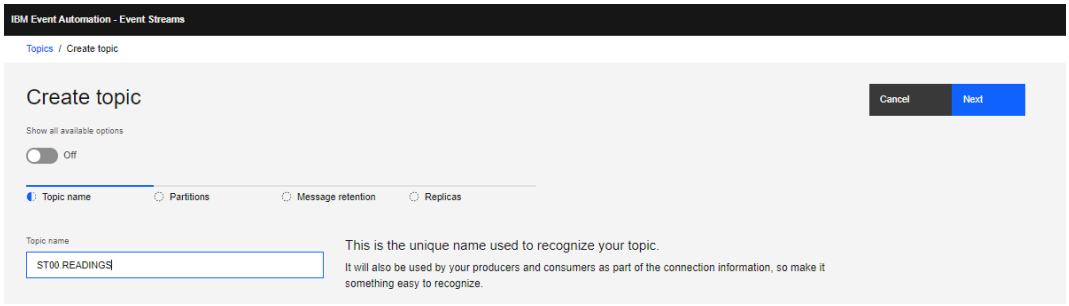
In this initial exercise, you'll create your first ES Topic. This Topic will serve as the destination for your digital meter readings. You'll then integrate this Topic into a shared EEM platform and make it accessible to both data producers and consumers.

Once the Topic is set up, you'll modify your Camel solution to utilize your newly created Topic. This will involve generating new credentials to ensure secure access to your Topic.

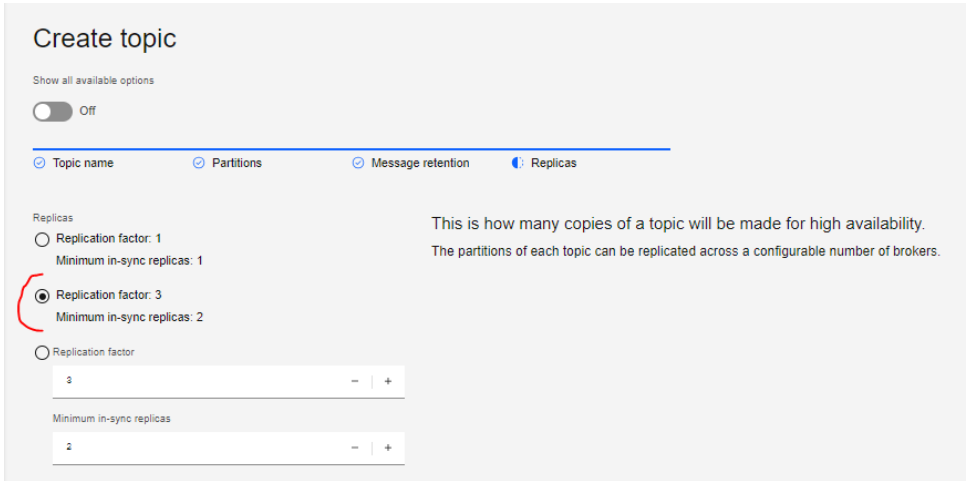
Tasks

Create Topic

1. Go to IBM ES: Event Streams
2. Login using your user.
3. Click on “Create Topic” on the home page or go to the page topics and click “Create Topic”
4. Provide a logical topic name:
    - Syntax:<PREFIX>.READINGS
    - Example: ST01.READINGS (we won’t go into the detailed setup of a topic, standard wizzard will be sufficient, feel free to take a look)



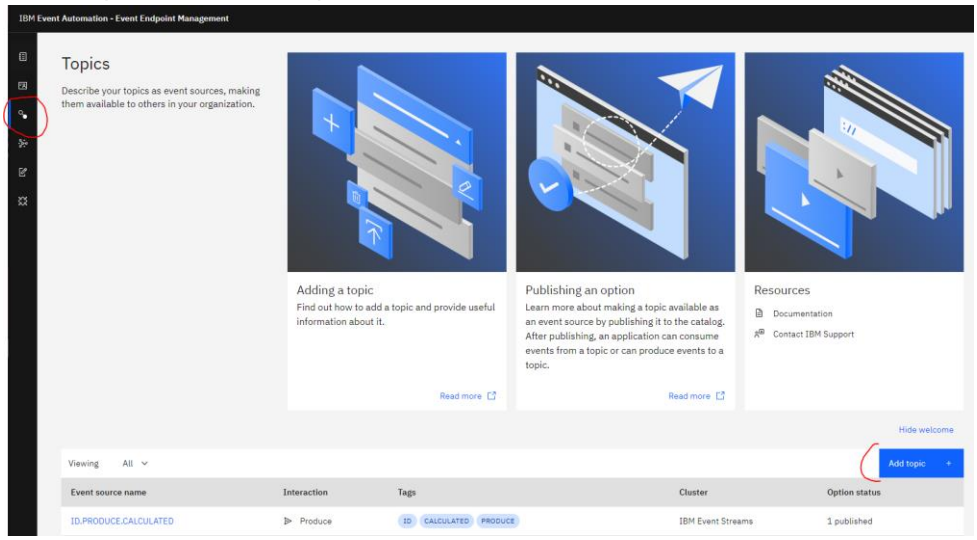
    - Partitions: 1
    - Message retention: a week
    - Replicas: 3



    - Click: “Create Topic”
5. Go see the result

Add the topic to EEM

3. Go to EEM: Event Endpoint Management
4. Login using your user.
5. Go to topics and “Add Topic”



   - Next: Produce events
   - Use the selected cluster “IBM Event Streams”
   - Select your Topic and give it a unique name (<PREFIX>.PRODUCE.READINGS)
   - Add Topic

Update the topic with relevant information

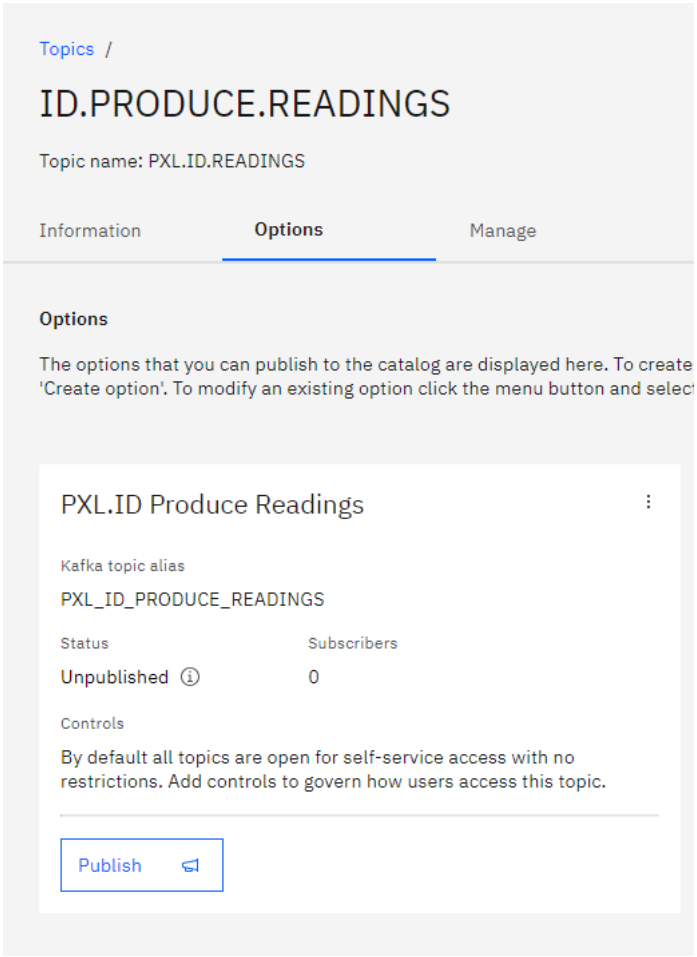
1. Download the contract “energy-readings_V1.avsc” from github: energy-readings_V1.avsc
2. Go to topics and open the topic you have created
3. Click Edit information
4. You can provide a description and contact information for your consumers and add tags for easy discovery in the catalog
5. Go to Event Information and add the contract you have downloaded
6. Add a message example (optional)

Create an option

The option creates an kafka topic alias which allows a client to connect.

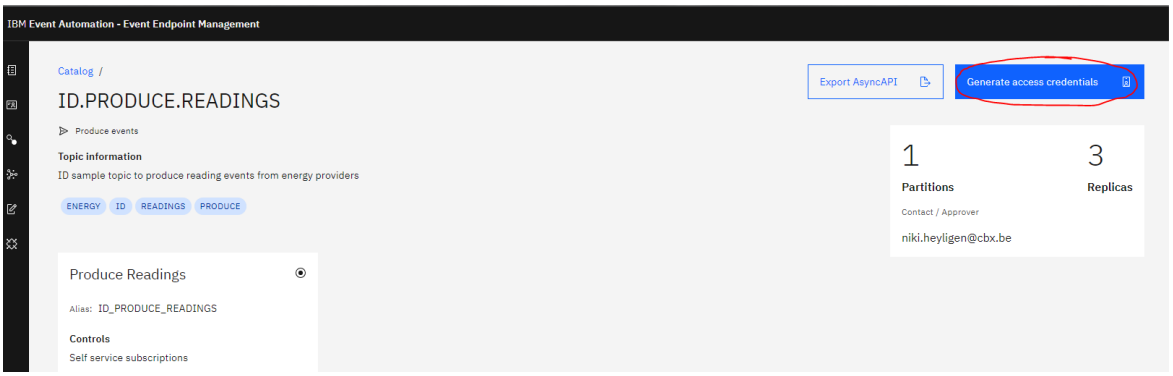
1. Open the tab Options
2. Create option(endpoint)
   - Provide a name <Prefix> Produce Readings
   - and alias <Prefix>_Produce_Readings

3. Click publish



Create credentials

- Go to Catalog
- Open your endpoint
- Overview the result of your work
- Click “Generate access credentials”



- Provide contact details
- Click “Generate”
- Download credentials (if you don’t, you will need to regenerate!)
- Give your file a logical name so you remember what it was for:
    - Exampel given: Cred_prod_readings.yaml

Update Camel

Update your camel integration to route to your own Topic. You can find all information needed in the endpoint page and the credential page you have just downloaded.

Ex 2. Pricing information

Goal

In this second exercise, you'll create your second ES Topic. This Topic will serve as the destination for the energy prices provided by Elia. You'll then integrate this Topic into a shared EEM platform and make it accessible to both data producers and consumers.

Once the Topic is set up, you'll modify your Camel solution to utilize your newly created Topic. This will involve generating new credentials to ensure secure access to your Topic.

Tasks

1. Create a topic for the pricing information
    a. Name: <PREFIX>.PRICINGS
    b. 1 partition
    c. 3 replicas
2. Add the topic to EEM so you can produce information from Camel to it
    a. <PREFIX>.PRODUCE.PRICINGS
3. Create a contract wich supports next message as a result:
   pricing_V1.avsc

```
{
        "ace": -11.865,
        "alpha": 0,
        "alpha_prime": 0,
        "datetime": "2024-10-03T13:26:00+00:00",
        "imbalanceprice": 109.607,
        "marginaldecrementalprice": -60.061,
        "marginalincrementalprice": 109.607,
        "qualitystatus": "NotValidated",
        "quarterhour": "2024-10-03T13:15:00+00:00",
        "resolutioncode": "PT1M",
        "systemimbalance": -71.175
}
```

   (**tip**: You can use the contract you downloaded as a reference; this one should be simpler)
4. Update the topic with relevant information.
5. Create the option (no approval is needed) and publish it
    a. **Keep the naming convention in mind!**
6. Publish your new topic/endpoint
7. Create your credentials and save them
8. Give your file a logical name so you remember what it was for
9. Update your camel integration

Ex 3. Create three more topics

Goal

In this third exercise, you'll create three more topics. Two of these topics will serve as a storage for a refactoring for the energy prices provided by Elia and the readings provided by the digital meters. The third one will the topic that your camel integrations. You'll then integrate these topics into a shared EEM platform and make it accessible to both data producers and consumers.

Once the Topic is set up, you'll modify your Camel solution to utilize your newly created Topic. This will involve generating new credentials to ensure secure access to your Topic.

Tasks

1. In EEM, create consuming options and aliases for your two existing topics
    a. These will be needed for your Flink solution
    b. You can review in Ex 1. How to create an option
    c. <PREFIX>.CONSUME.PRICINGS
    d. <PREFIX>.CONSUME.READINGS
2. Create following new Topics:
   (contracts can be downloaded from github)

Topic 1	
Name	<PREFIX>.AVG.PRICING
Partitions	1
Replicas	3
contract	avgPricing_V1.avs
EEM: Producer	<PREFIX>_PRODUCE_AVG_PRICING **Approval Required**
EEM: Consumer	<PREFIX>_CONSUME_AVG_PRICING
Topic 2	
Name	<PREFIX>.EAN.READINGS
Partitions	1
Replicas	3
Contract	eanReadings_V1.avsc
EEM: Producer	<PREFIX>_PRODUCE_EAN_READINGS **Approval Required**
EEM: Consumer	<PREFIX>_ CONSUME _EAN_READINGS
Topic 3	
Name	<PREFIX>.CALCULATED
Partitions	1
Replicas	3
Contract	calculated.avsc
EEM: Producer	<PREFIX>_PRODUCE_CALCULATED **Approval Required**
EEM: Consumer	<PREFIX>_ CONSUME _CALCULATED

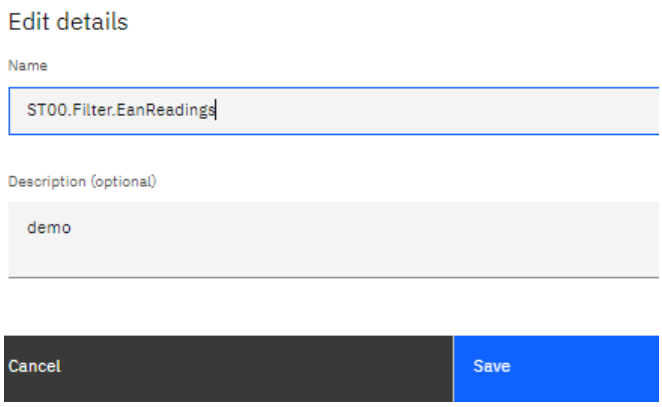
Tip: Don't forget to create credentials and save them for all endpoints!

Ex 4. Flink: Filter EAN readings

The data of the two input streams do not match and are very complex to correlate. Therefore you best create two intermediate Flink flows to simplify the results. You could combine all logic into one flow, but it will be very complex.

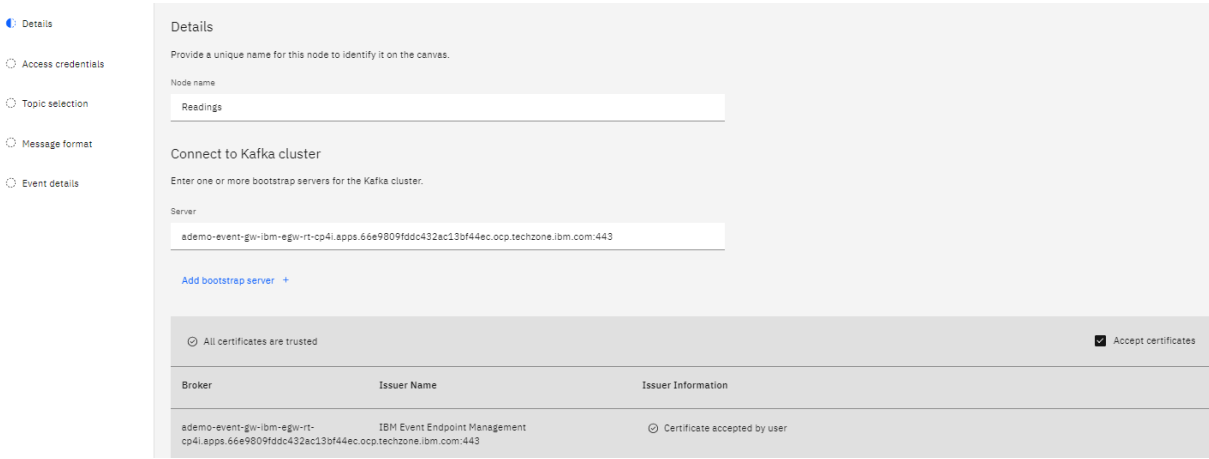
Some Readings are dummy data and not representative. To filter out these events, you best create a Flink flow to filter out all events with no value for us.

1. Go to EP: IBM Event Processing
2. Login using your user.
3. Create a new Flow
    a. <PREFIX>.Filter.EanReadings



Input source: Readings

1. Edit the source already present
2. Add event source:
    a. Node name: Readings
    b. Server: ademo-event-gw-ibm-egw-rt-cp4i.apps.66e9809fddc432ac13bf44ec.ocp.techzone.ibm.com:443
    c. Accept certificates
    d.

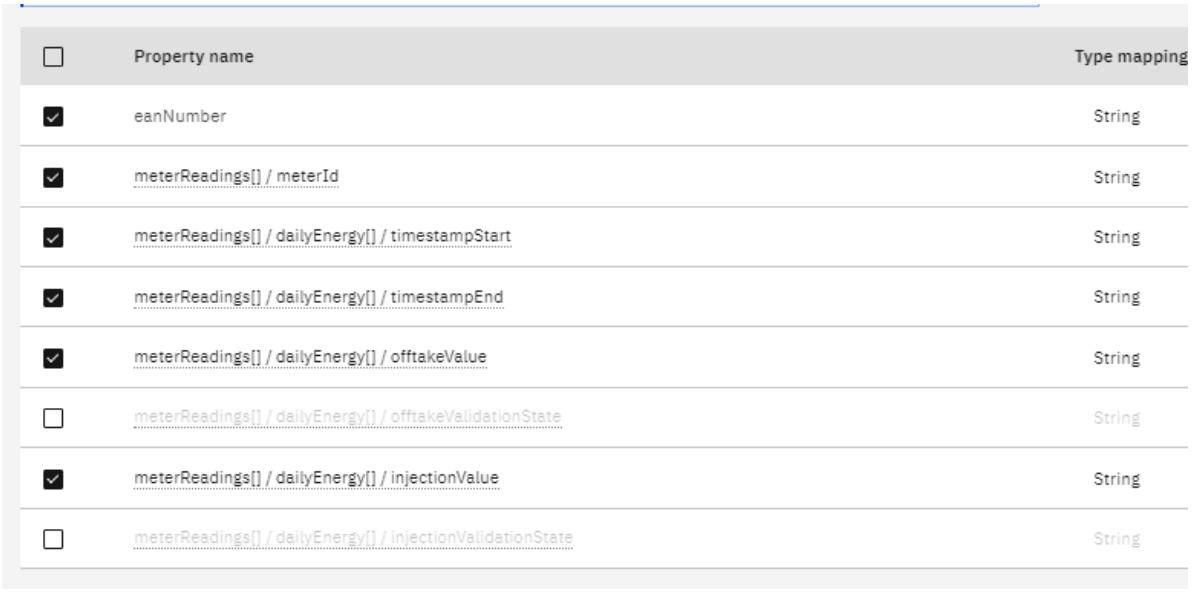


3. Next
    a. Security Mechanism: PLAIN
    b. Username and Password: open the yaml file and copy paste both “username” & “password”

```
{
  "type": "SASL",
  "mechanism": "PLAIN",
  "username": "eem-4635bwww-xxxx-xxxx-xxxx-xxxxxxx873cf",
  "password": "f98xxx4d-xxxx-xxxx-xxxx-xxxx67xxx20",
  "clientSecurity": "SASL_SSL"
}
```

c.

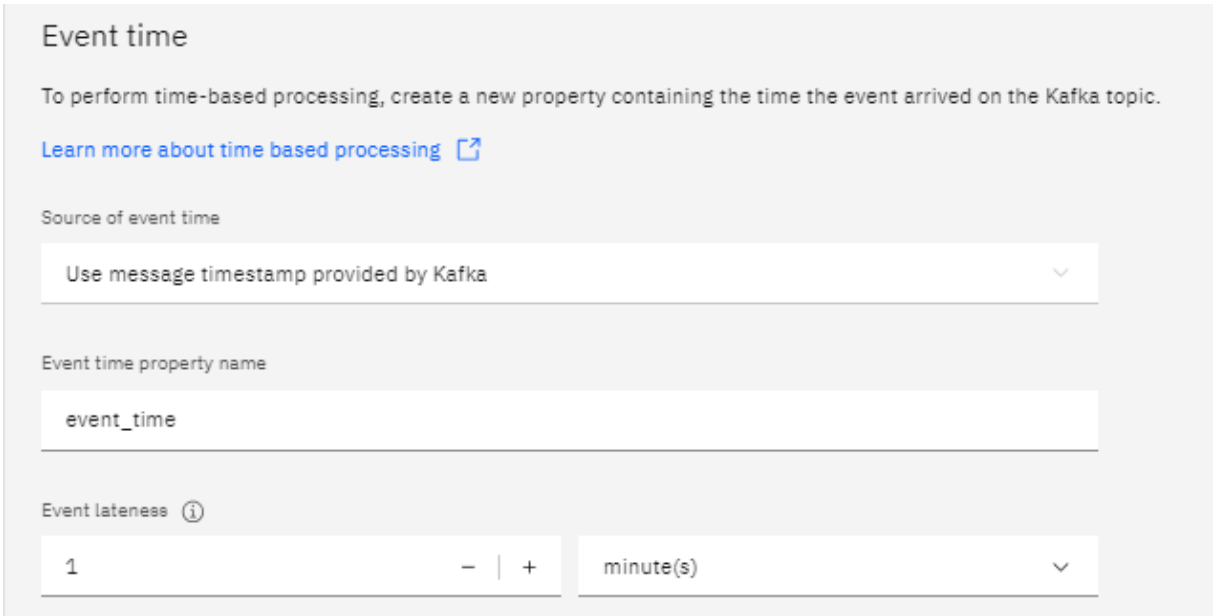
4. Next
    a. Select your topic
5. Next
    a. Json schema
6. Next: Filter out unneeded elements. Remove:
    a. meterReadings[] / dailyEnergy[] / offtakeValidationState
    b. meterReadings[] / dailyEnergy[] / injectionValidationState



	Property name	Type mapping
☑	eanNumber	String
☑	meterReadings[] / meterId	String
☑	meterReadings[] / dailyEnergy[] / timestampStart	String
☑	meterReadings[] / dailyEnergy[] / timestampEnd	String
☑	meterReadings[] / dailyEnergy[] / offtakeValue	String
☐	meterReadings[] / dailyEnergy[] / offtakeValidationState	String
☑	meterReadings[] / dailyEnergy[] / injectionValue	String
☐	meterReadings[] / dailyEnergy[] / injectionValidationState	String

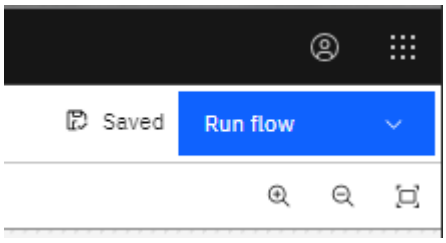
c.

7. Leave the rest as is:

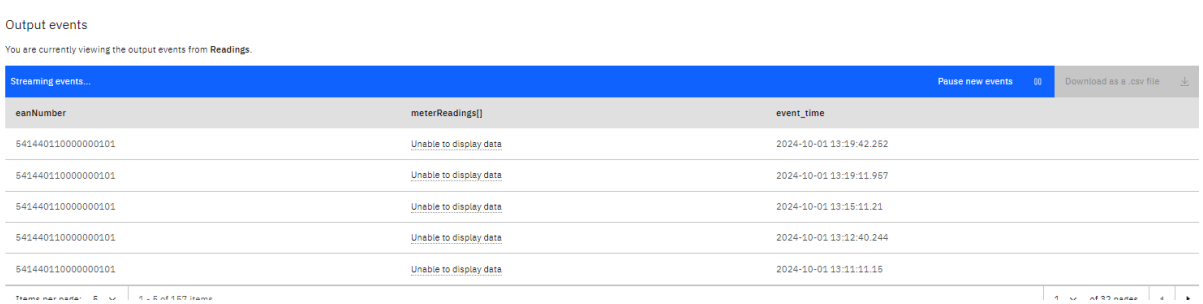


8. Configure

9. Test your result -> Click “Run flow (include historical)” on the top right



(This can take a minute or two before you will see results)

10. 

Tip: *You can test the result after the last node and only the last node*

Continue creating the flow

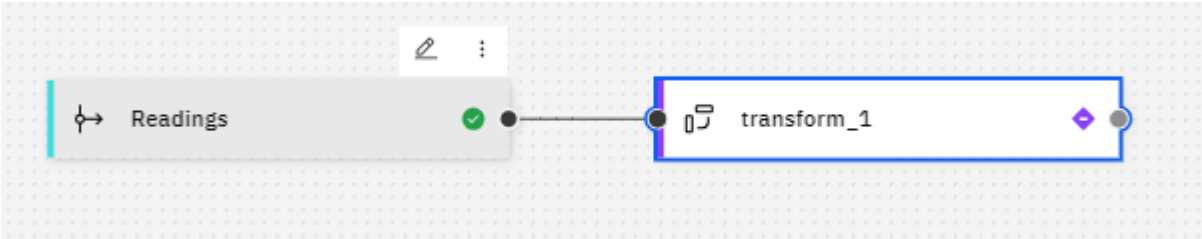
Nodes to create:

1. Transformation: Get Type EAN
2. Filter: FilterEan
3. Unpack array: unpackMeterReadings
4. Unpack array: unpackEnergy
5. Transformation: DateTime
6. Event Destination: Ean Readings

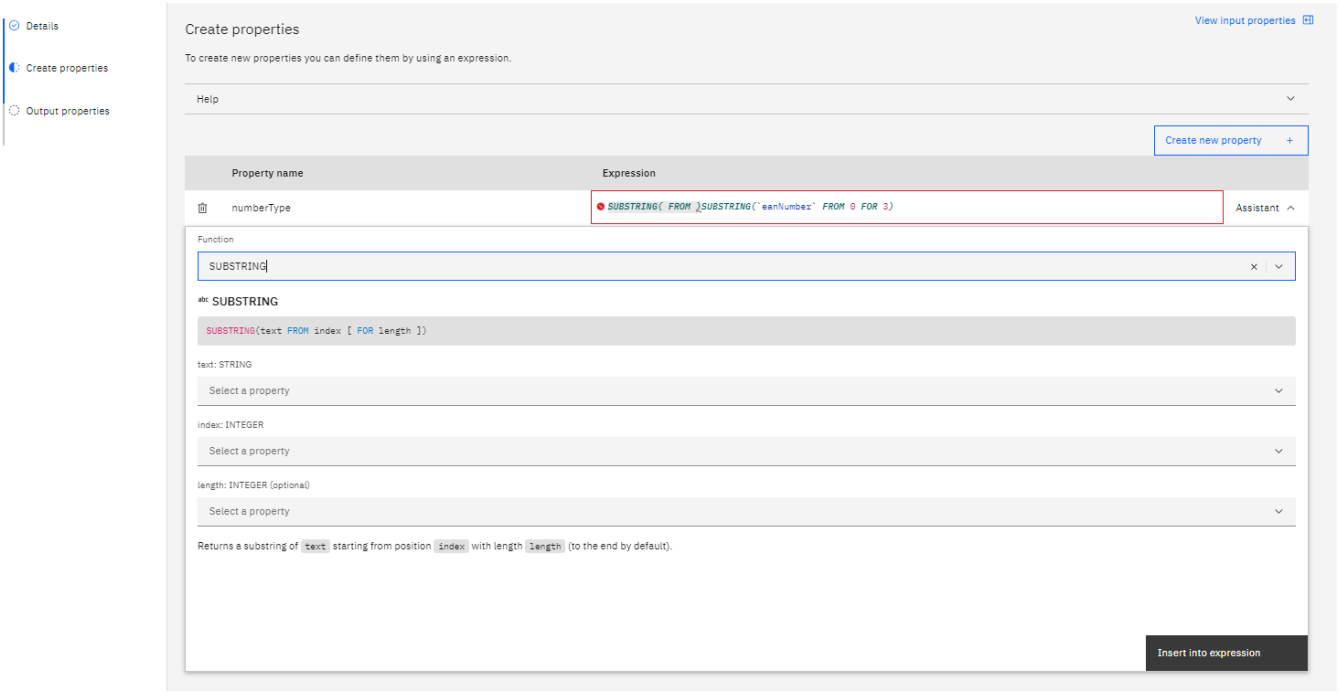
Add a type EAN to the message

Info: this element will be needed to correlate the pricing event with the readings event

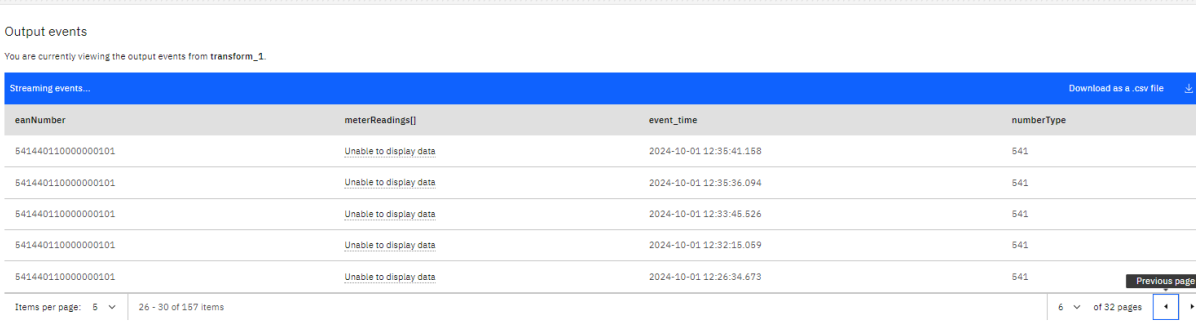
1. Drag a node of type “Tranfsorm” on the canvas and link the Readings node to your new node



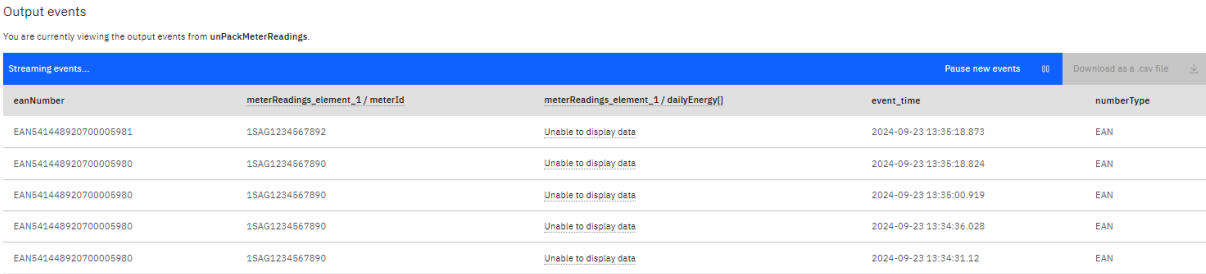
2. Edit node:
   - Node Name: Get Type EAN
   - Create new property
     - numberType
     - Expression: do a substring of elenemt ‘eanNumber’ from 0 for 3)
       Tip: you can use the AI to assist you in making the Expression



- Finish configuration and test your result



3. Add a filter node: FilterEan
   - add an expression so only the numbertypes ‘EAN’ are kept and the rest is skipped.
4. Add an unpack node: unPackMeterReadings
   - Select array: meterReadings[]
   - Unpack into properties
   - Configure



5. Add an unpack node: unpackEnergy
   - Select array: meterReadings_element_1 / dailyEnergy[]

- Unpack into properties

	Property name	Original name	Data type
⊖	eanNumber		STRING
⊖	*meterReadings_element_1* / meterId		STRING
⊖	*meterReadings_element_1 / dailyEnergy_elem...* / timestampStart		STRING
⊖	*meterReadings_element_1 / dailyEnergy_elem...* / timestampEnd		STRING
⊖	*meterReadings_element_1 / dailyEnergy_element...* / offtakeValue		STRING
⊖	*meterReadings_element_1 / dailyEnergy_eleme...* / injectionValue		STRING
⊖	event_time		TIMESTAMP (Event time)
⊖	numberType		STRING

- Configure

6. Add transform node: DateTime
   - Add property: StartDT
   - Expression: TO_TIMESTAMP(REGEXP_EXTRACT(REGEXP_REPLACE(`meterReadings_element_1`.`dailyEnergy_element_1`.`timestampStart`, 'T', ' '), '(\d{4}-\d{2}-\d{2} \d{2}:\d{2})'), 'yyyy-MM-dd HH:mm')
   - Add proerty: EndDT
   - Expression: TO_TIMESTAMP(REGEXP_EXTRACT(REGEXP_REPLACE(`meterReadings_element_1`.`dailyEnergy_element_1`.`timestampEnd`, 'T', ' '), '(\d{4}-\d{2}-\d{2} \d{2}:\d{2})'), 'yyyy-MM-dd HH:mm')
   - Configure
   - Test

Output events

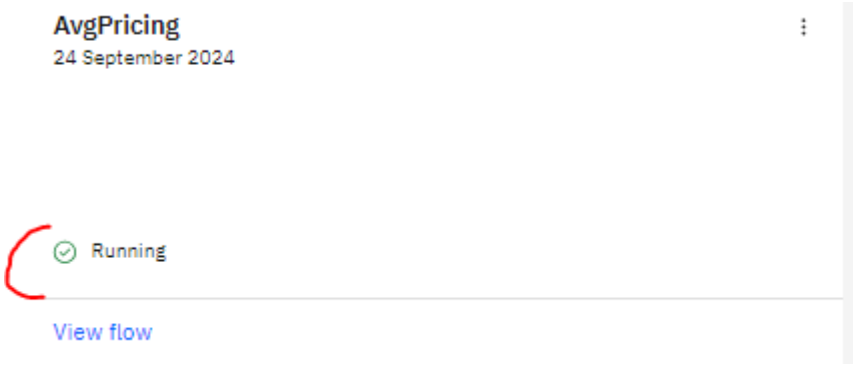
You are currently viewing the output events from **DateTime**.

Streaming events...

eanNumber	meterReadings_element_1 / meterId	meterReadings_elem... / timestampStart	meterReadings_eleme... / timestampEnd	meterReadings_element... / offtakeValue
EAN541448920700005981	1SAG1234567892	2024-09-23T07:00:00Z	2024-09-24T07:00:00Z	5.587
EAN541448920700005980	1SAG1234567890	2024-09-23T09:30:00Z	2024-09-24T09:30:00Z	7.542
EAN541448920700005980	1SAG1234567890	2024-09-23T09:30:00Z	2024-09-24T09:30:00Z	7.542
EAN541448920700005980	1SAG1234567890	2024-09-23T09:30:00Z	2024-09-24T09:30:00Z	7.542
EAN541448920700005980	1SAG1234567890	2024-09-23T09:30:00Z	2024-09-24T09:30:00Z	7.542

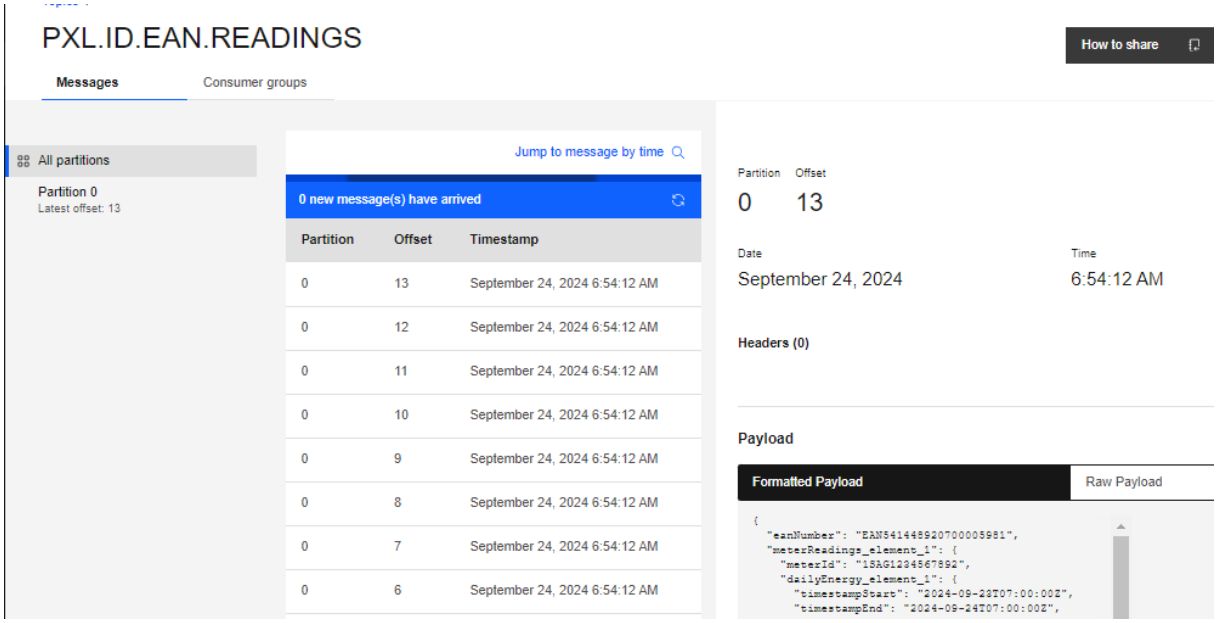
meterReadings_eleme... / injectionValue	event_time	numberType	StartDT	EndDT
3.554	2024-09-23 13:35:18.873	EAN	2024-09-23 07:00:00	2024-09-24 07
5.546	2024-09-23 13:35:18.824	EAN	2024-09-23 09:30:00	2024-09-24 09
5.546	2024-09-23 13:35:00.919	EAN	2024-09-23 09:30:00	2024-09-24 09
5.546	2024-09-23 13:34:36.028	EAN	2024-09-23 09:30:00	2024-09-24 09
5.546	2024-09-23 13:34:31.12	EAN	2024-09-23 09:30:00	2024-09-24 09

7. Create a Event destination: Ean Readings
    - Server: ademo-event-gw-ibm-egw-rt-cp4i.apps.66e9809fddc432ac13bf44ec.ocp.techzone.ibm.com:443
    - Accept certificates
    - Add credentials you have downloaded to produce to <PREFIX>_PRODUCE_EAN_READINGS
    - Configure
8. Click Run flow Include historical
9. When you go back to your home view, you will see that your flow is still running.



Verify result

1. Go to IBM ES: https://ademo-es-ibm-es-ui-cp4i.apps.66e9809fddc432ac13bf44ec.ocp.techzone.ibm.com/#
2. Login using your user.
3. Open your topic <PREFIX>.EAN.READINGS and see the data comming in:



Ex 5. Flink: Calculate average prices

Create a new flow <PREFIX>.Avg.Pricing Containing following nodes:

- Event source: Pricing
- Aggregate: Avg Prices
- Transform: Transform
- Event destination: avg_pricing

Extra information

Some extra details you need to know to correctly configure these nodes:

Pricing

- Add your topic <PREFIX>_CONSUME_PRICING
- Use the correct credentials
- Bootstrap server is the same as previous one
- You will only need folowing elements:
    - Datetime
    - Marginaldecrementalprice
    - Marginalincrementalprice
    - Quarterhour
- Source of event time: quarterhour

Avg Prices

- Time window: quarter hour and 15 min
- Aggregate function = AVG for both marginaldecrementalprice as marginalincremental price
- 

Transform

- Add a property called ‘Type’ with a hardcoded value ‘EAN’

- Properties to keep

	Property name	Original name	Data type
⊖	AVG_marginaldecrementalprice		DOUBLE
⊖	AVG_marginalincrementalprice		DOUBLE
⊖	aggregateStartTime		TIMESTAMP
⊖	aggregateEndTime		TIMESTAMP
⊖	aggregateResultTime		TIMESTAMP_LTZ (Event time)
⊖	Type		STRING

avg_pricing

- Add your topic <PREFIX>_PRODUCDE_AVG_PRICING
- Use the correct credentials

Ex 6. Flink: Calculate Prive Event

Let us join the two new events into one flink application. Create a new flow <PREFIX>.Create.Price.Event Containing following nodes:

Try to join both topics into one message and use the tranform node to map it according to your schema calculated.avsc

Ex 7. Create customers (Java client)

Let us try to create a java client to produce to a topic and consume from a topic

Tasks

1. Download the customer contract: customer.avsc
2. Create a topic for the pricing information
    a. Name: <PREFIX>.PRICINGS
    b. 1 partition
    c. 3 replicas
3. Add the topic to EEM
    - Both to consumer and producer
4. Create credentials for both producer and consume
5. Go to the catalog in Event Endpoint management and open your produce topic
    - Use the code sampe and the maven dependencies to create a Java client to produce events to your topic
6. Go to the catalog in Event Endpoint management and open your consume topic
    - Use the code sampe and the maven dependencies to create a Java client to consume events from your topic