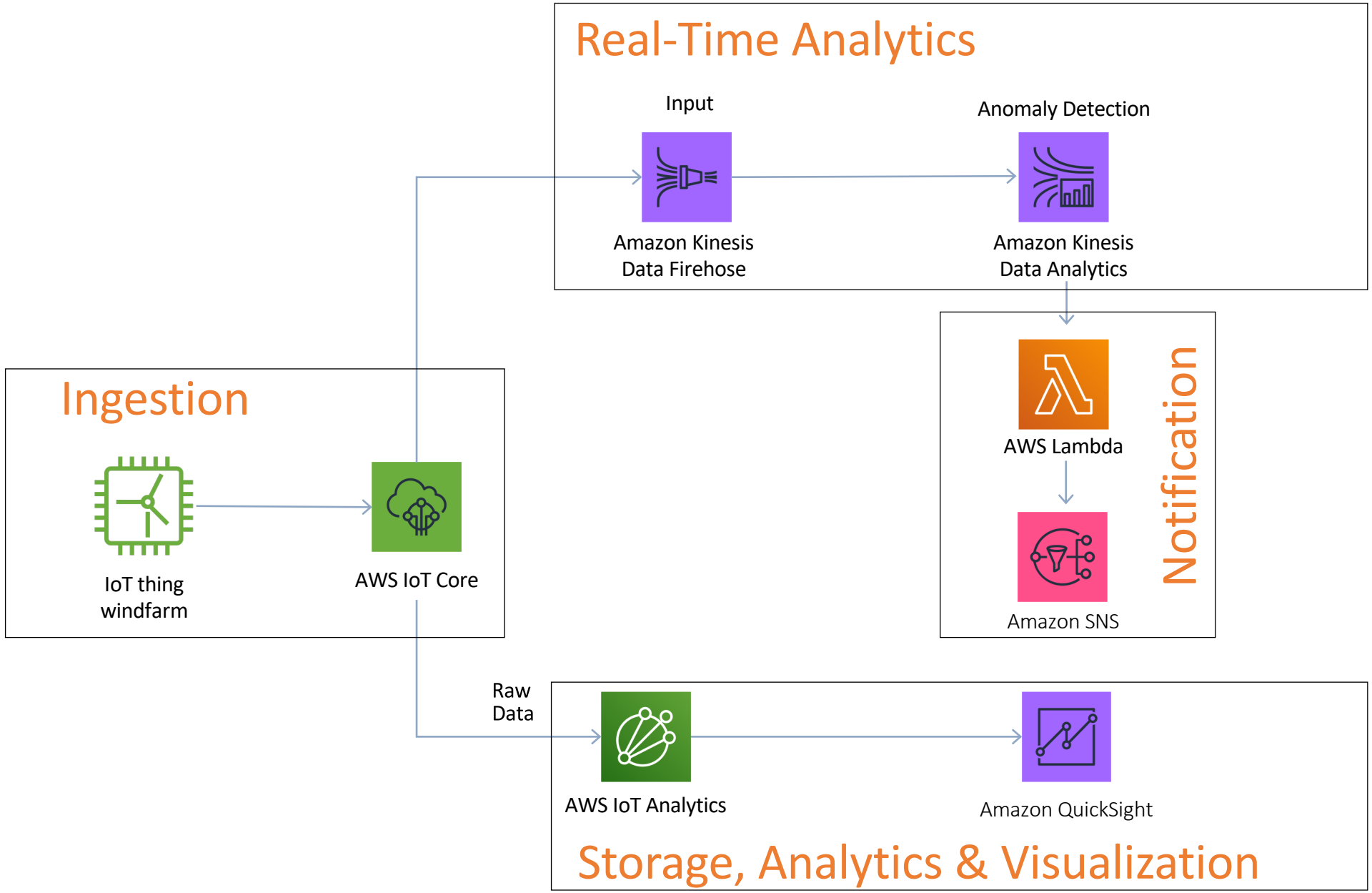
Building Predictive Applications with AWS IoT and Machine Learning

Workshop Guide

What you will build today…



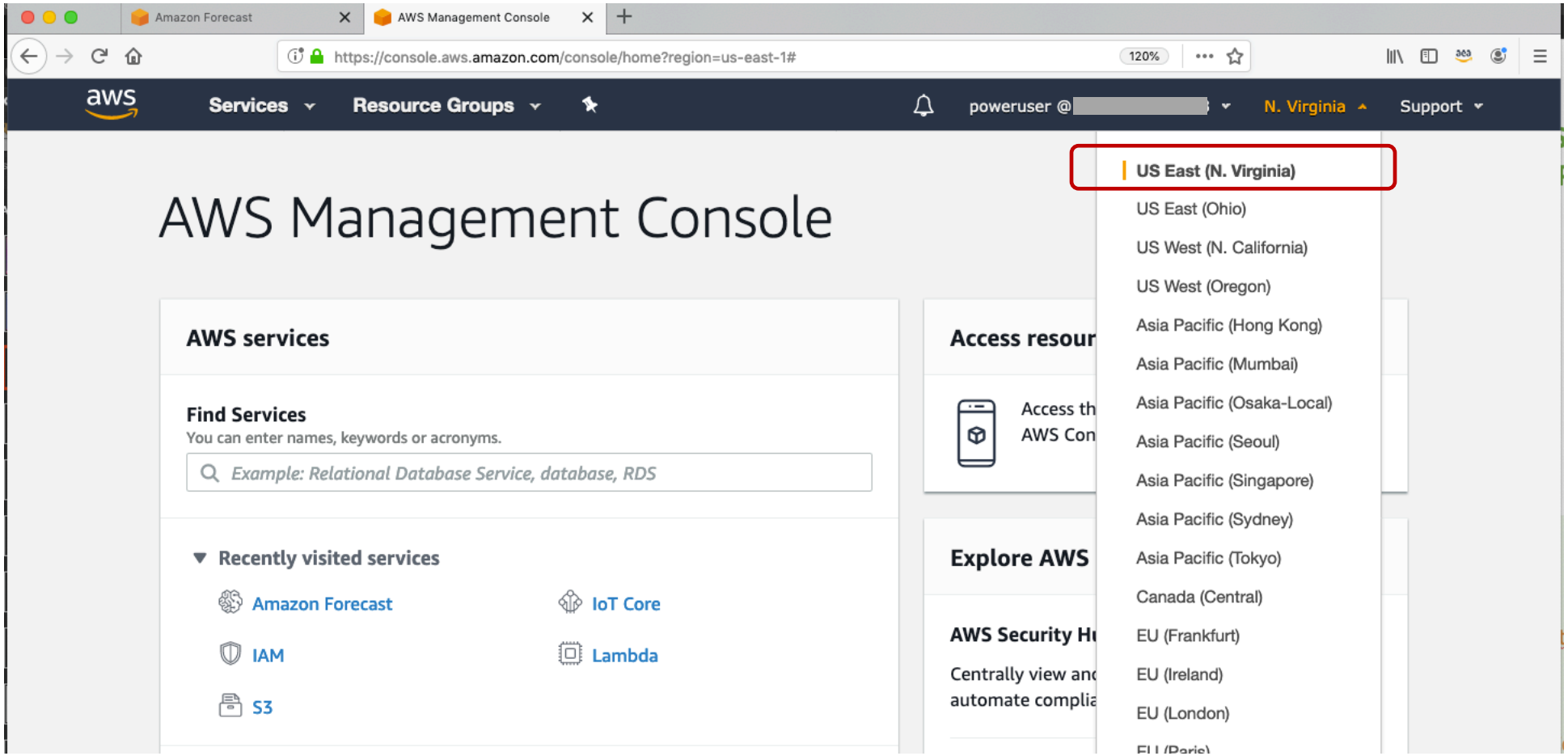
We will go over these labs:

Lab 1 –IoT ‘thing’ & Device Simulation	15 mins
Lab 2 –Kinesis Firehose and Analytics Pipeline	30 mins
Lab 3 –IoT Analytics	30 mins
Lab 4 –Notification (Optional)	30 mins

We will go over these labs:

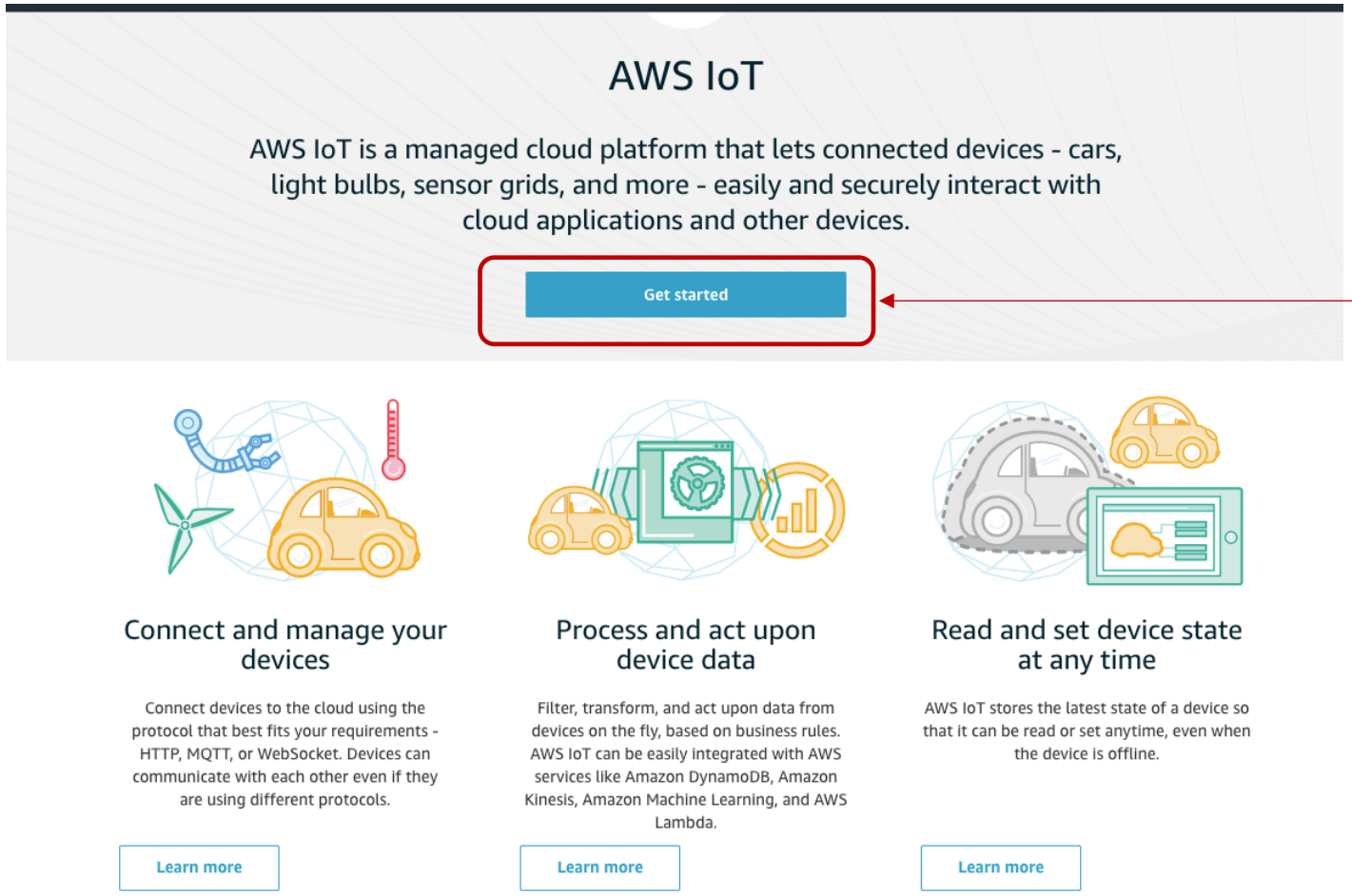
Lab 1 –IoT ‘thing’ & Device Simulation	15 mins
Lab 2 –Kinesis Firehose and Analytics Pipeline	30 mins
Lab 3 –IoT Analytics	30 mins
Lab 4 –Notification (Optional)	30 mins

IMPORTANT - before starting… ensure that you have selected ‘N. Virginia’ as your region. The code sample provided is configured with IoT endpoint for N. Virginia, if you run this in a different region, you may have to tweak this in the code.



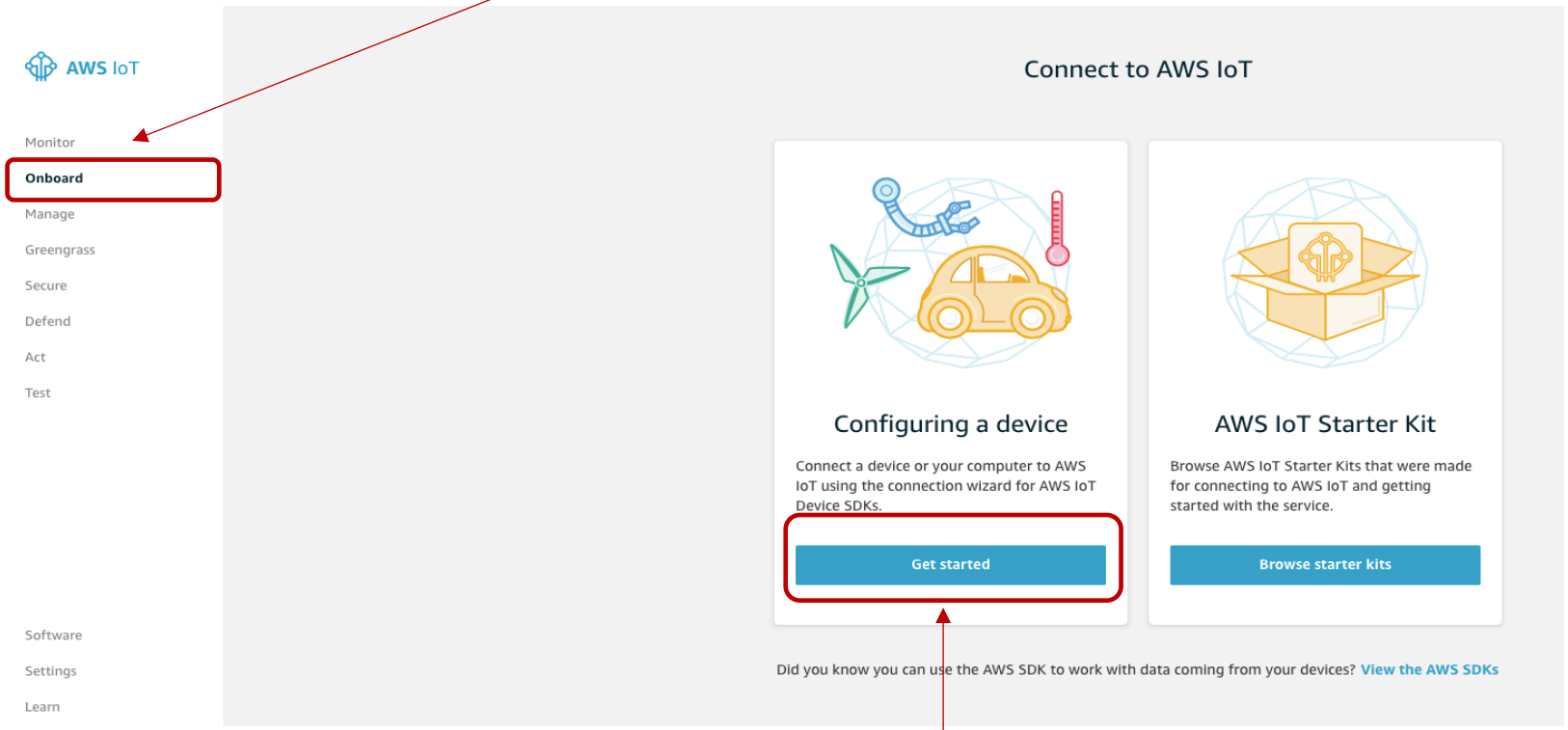
Setup IoT ‘Thing

Select ‘IoT Core’ Service from AWS Console.



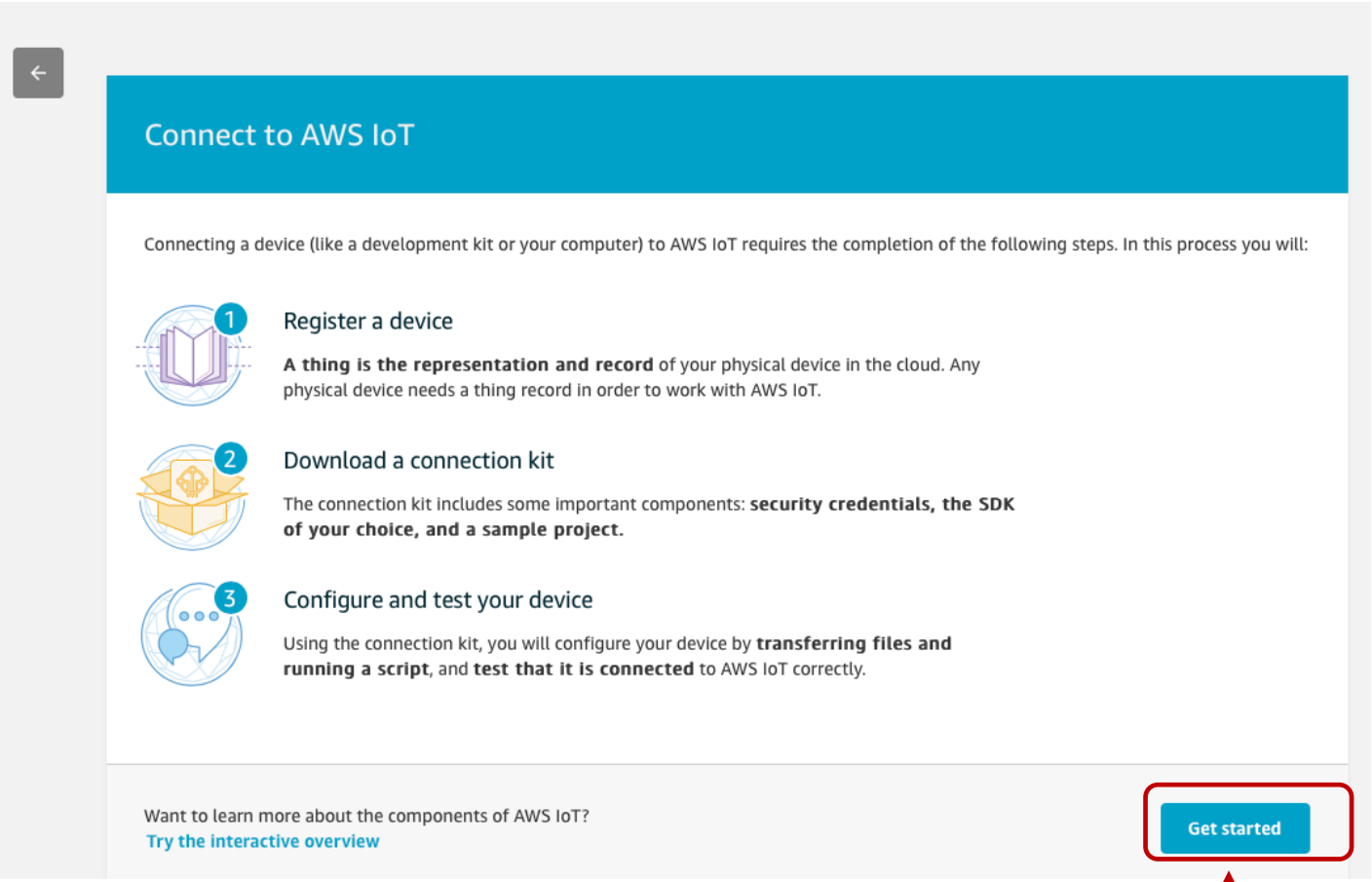
Click Get started

On the IoT landing page, select ‘On Board’ from left navigation panel, then click ‘Get Started’ from ‘Configuring a device’ panel.



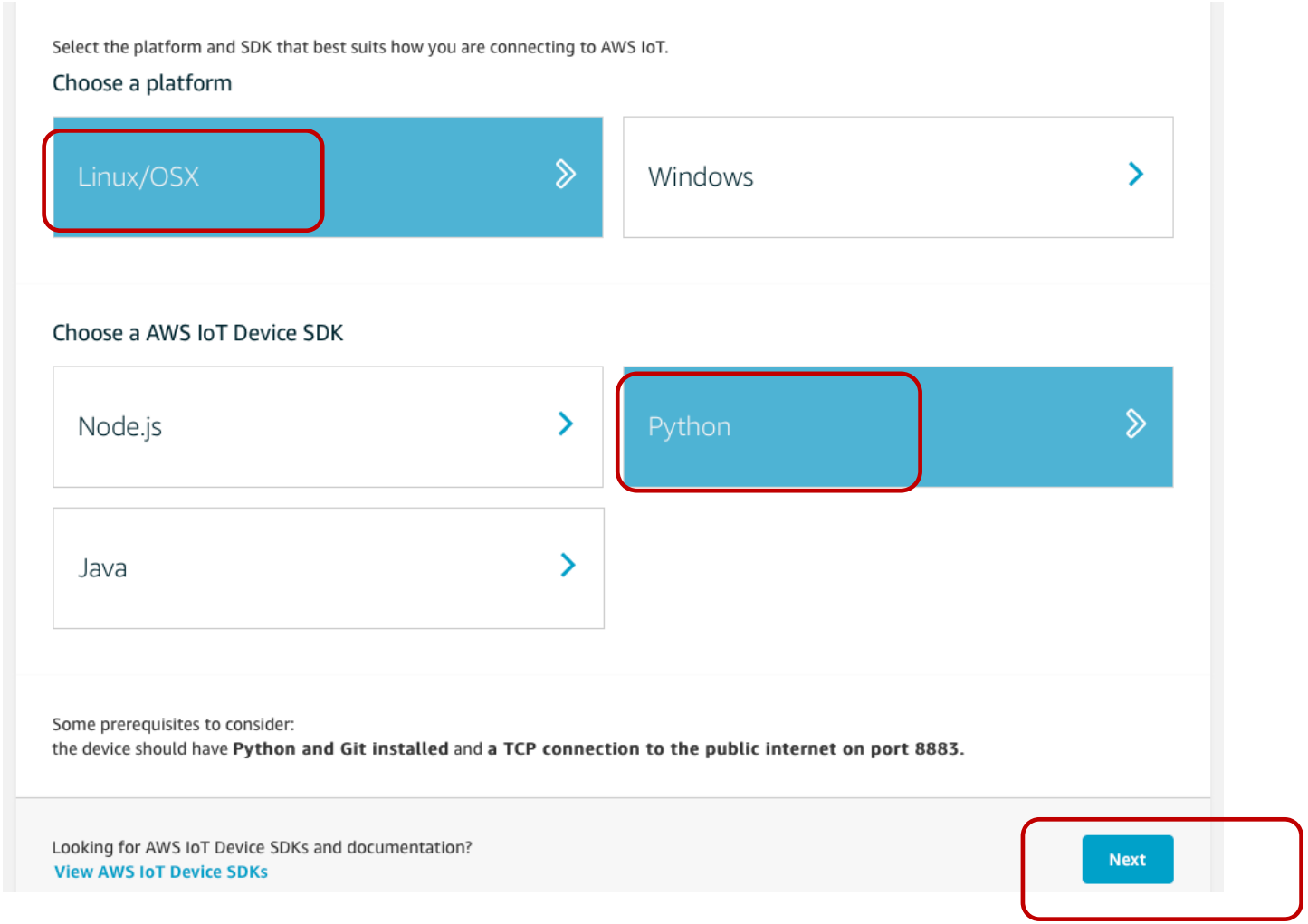
Next, click on ‘Get started’

Select ‘Get Started’ on next page…



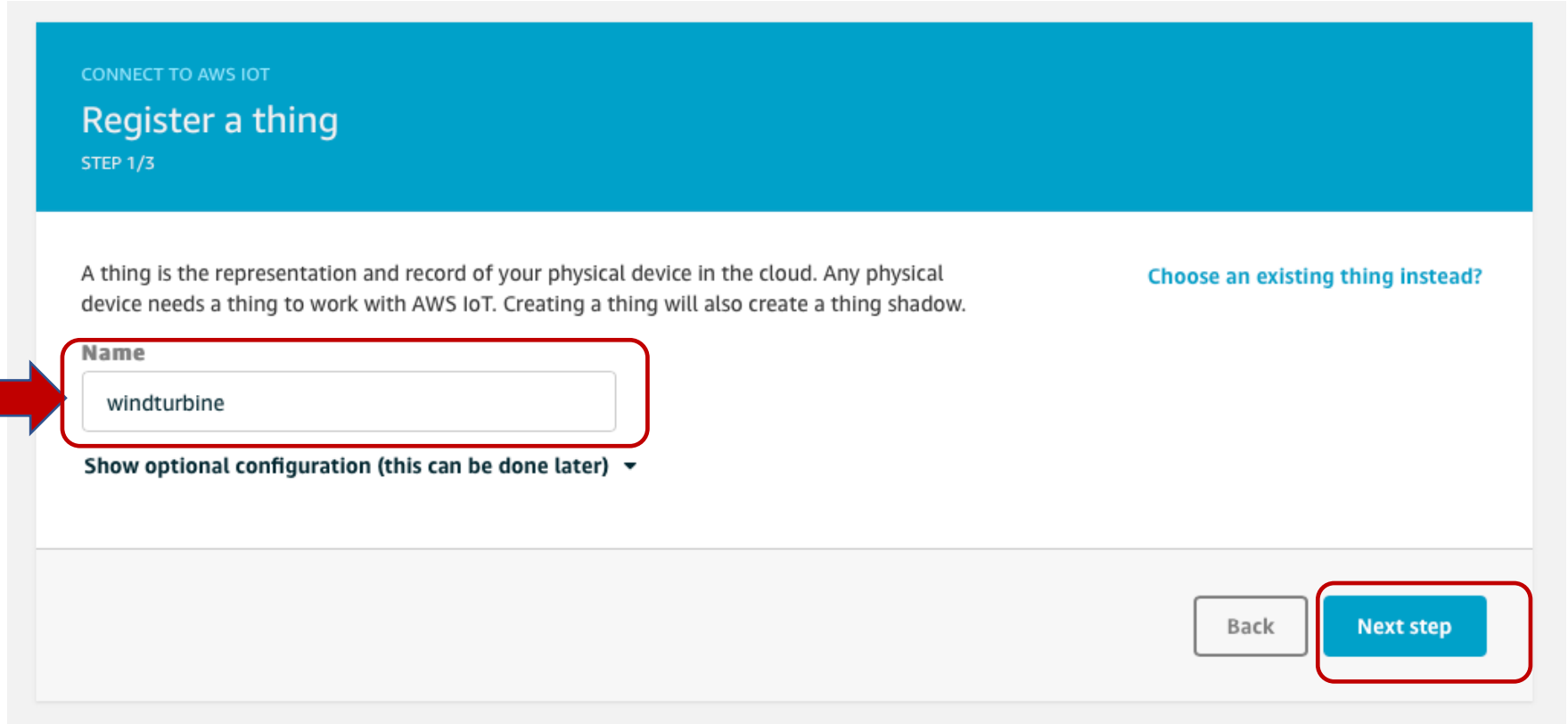
On next screen, click on ‘Get started’

Select ‘Linux/OSX’ from platform and ‘Python’ from Device SDK, then click ‘Next’



Give your device a name – ‘windturbine’ then press ’Next Step’.

Important – you should call your device windturbine (all lowercase, one word) as we refer to the device by this name in the code. Otherwise, you will need to change this in the code.



CONNECT TO AWS IOT

Download a connection kit

STEP 2/3

The following AWS IoT resources will be created:

A thing in the AWS IoT registry	windturbine	
A policy to send and receive messages	windturbine-Policy	Preview policy

The connection kit contains:

A certificate and private key	windturbine.cert.pem, windturbine.private.key
AWS IoT Device SDK	Python SDK
A script to send and receive messages	start.sh

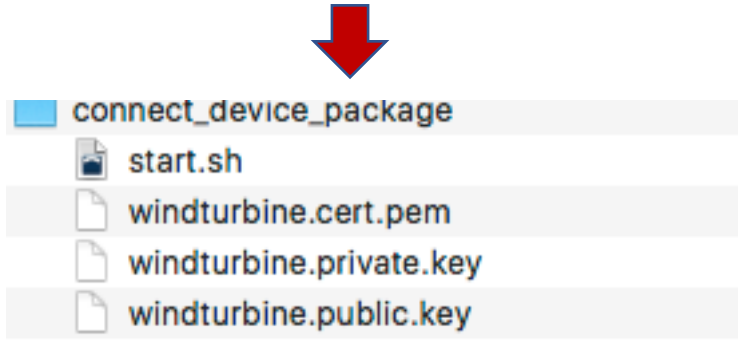
Before your device can connect and publish messages, you will need to download the connection kit.

Download connection kit for

Linux/OSX

Back | Next step

You will see these files in the connection kit



Click here to download the connection kit

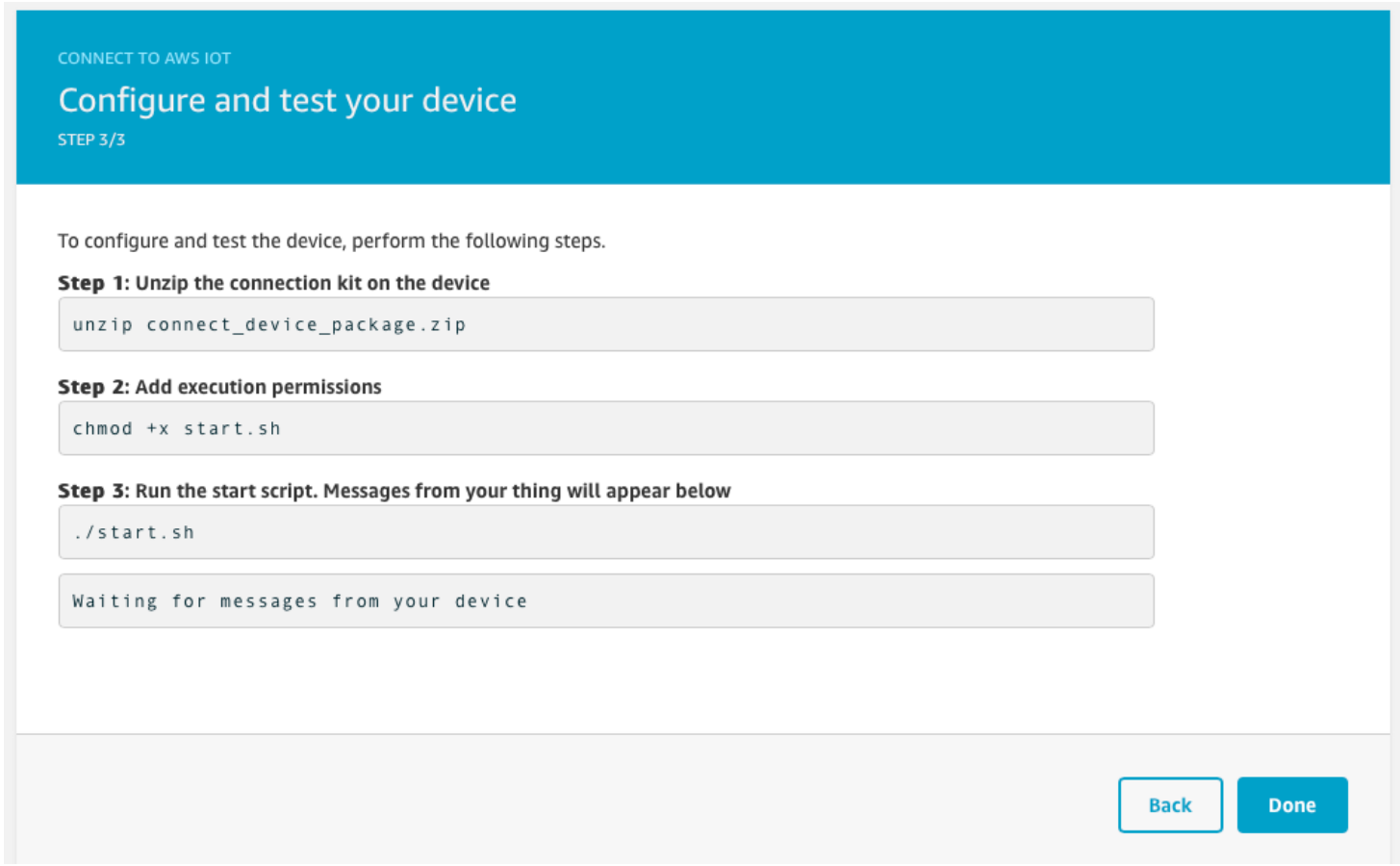
Click 'Next Step'

The next screen will show you the steps to follow, to complete the device setup

Important: Don't run these steps on your laptop. Instead, we will run these steps on Cloud9, which we will use for device simulation. Don't click the 'Done' yet.

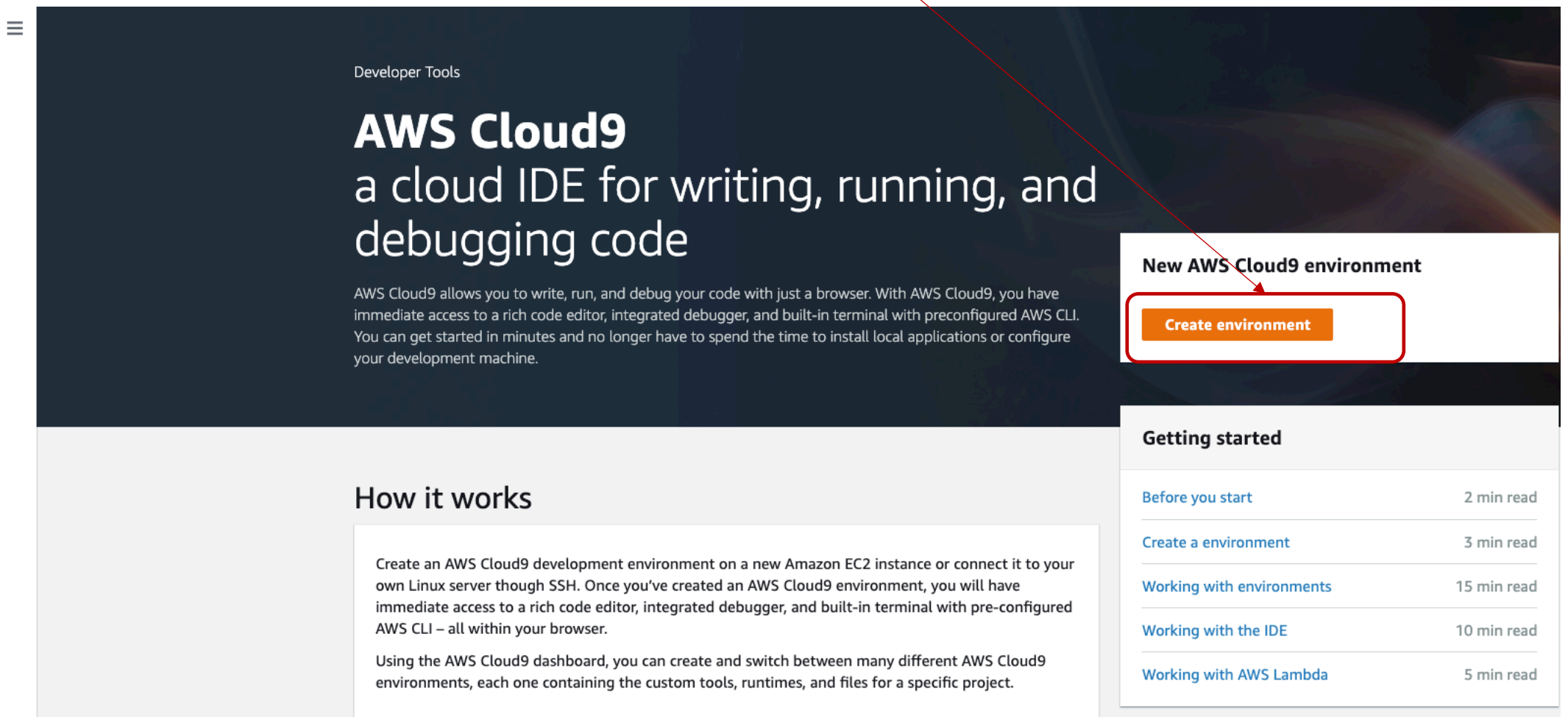
We will setup Cloud9 in the next step.

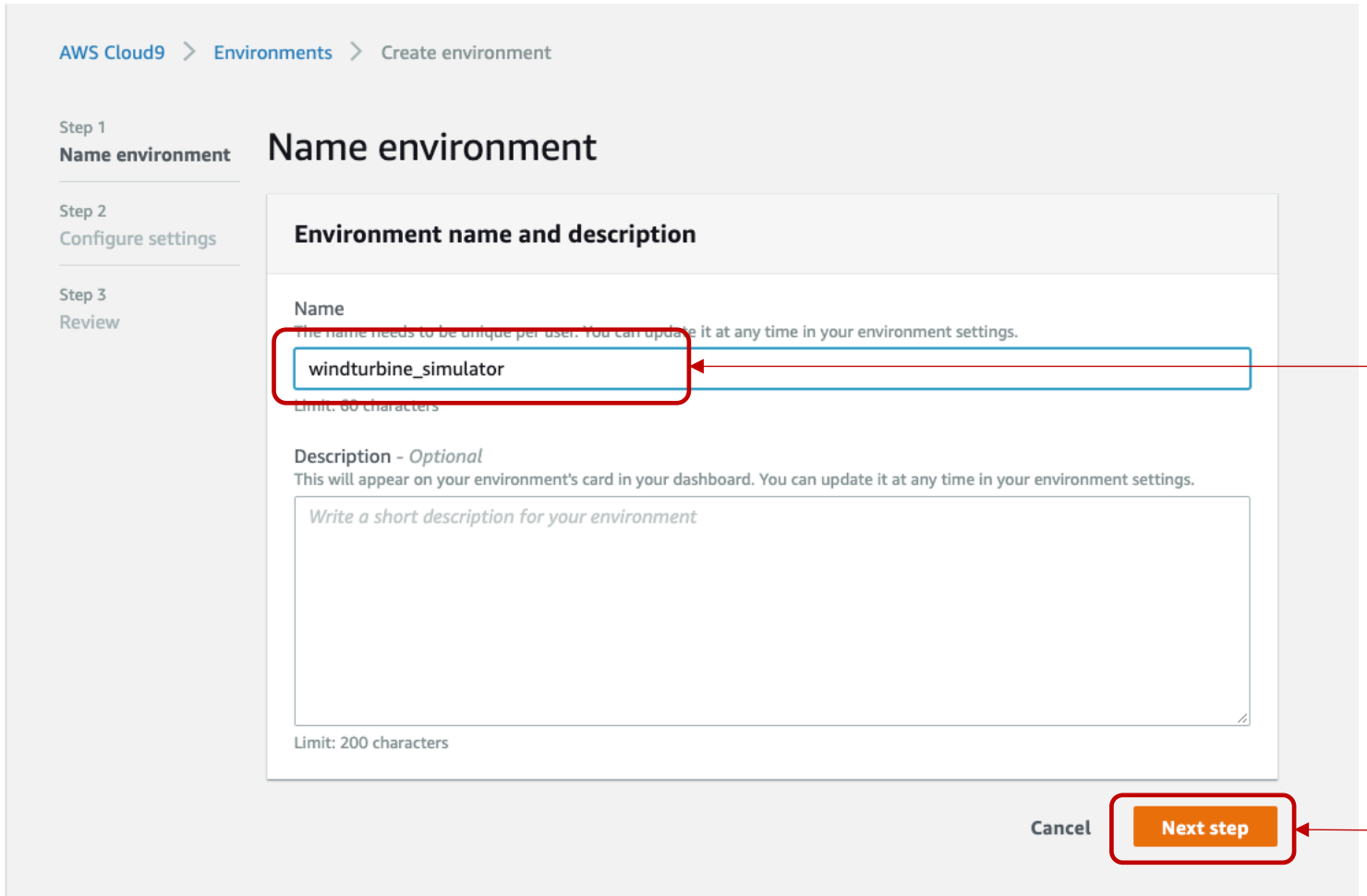
You can leave the IoT console open as we will revisit these steps, after setting up Cloud9 IDE. You can open Cloud9 in a new browser tab or window.



Setup Cloud9

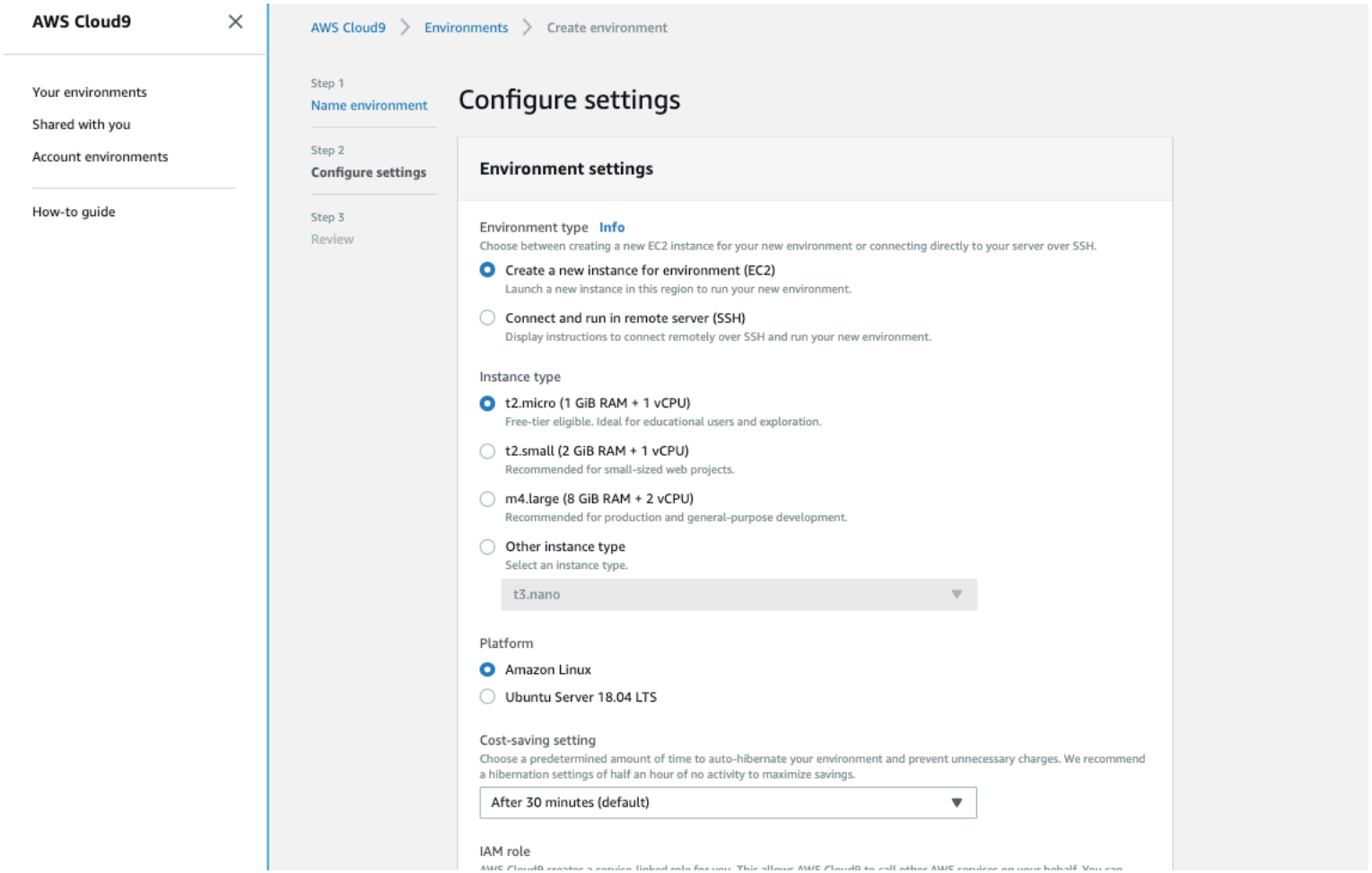
Select ‘Cloud9’ from AWS Console. Then select ‘Create environment’.



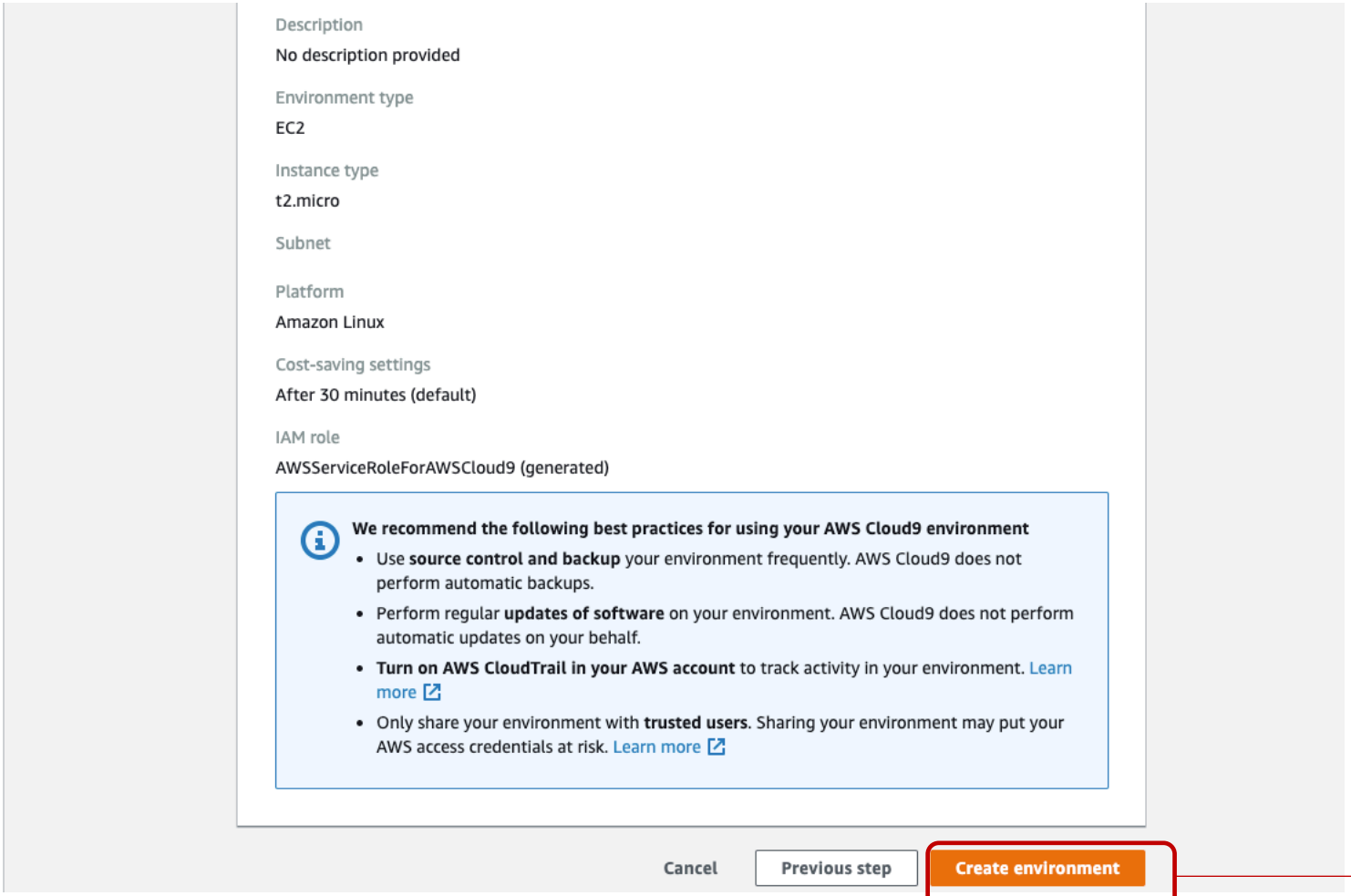


Give a name to this environment

Click Next Step

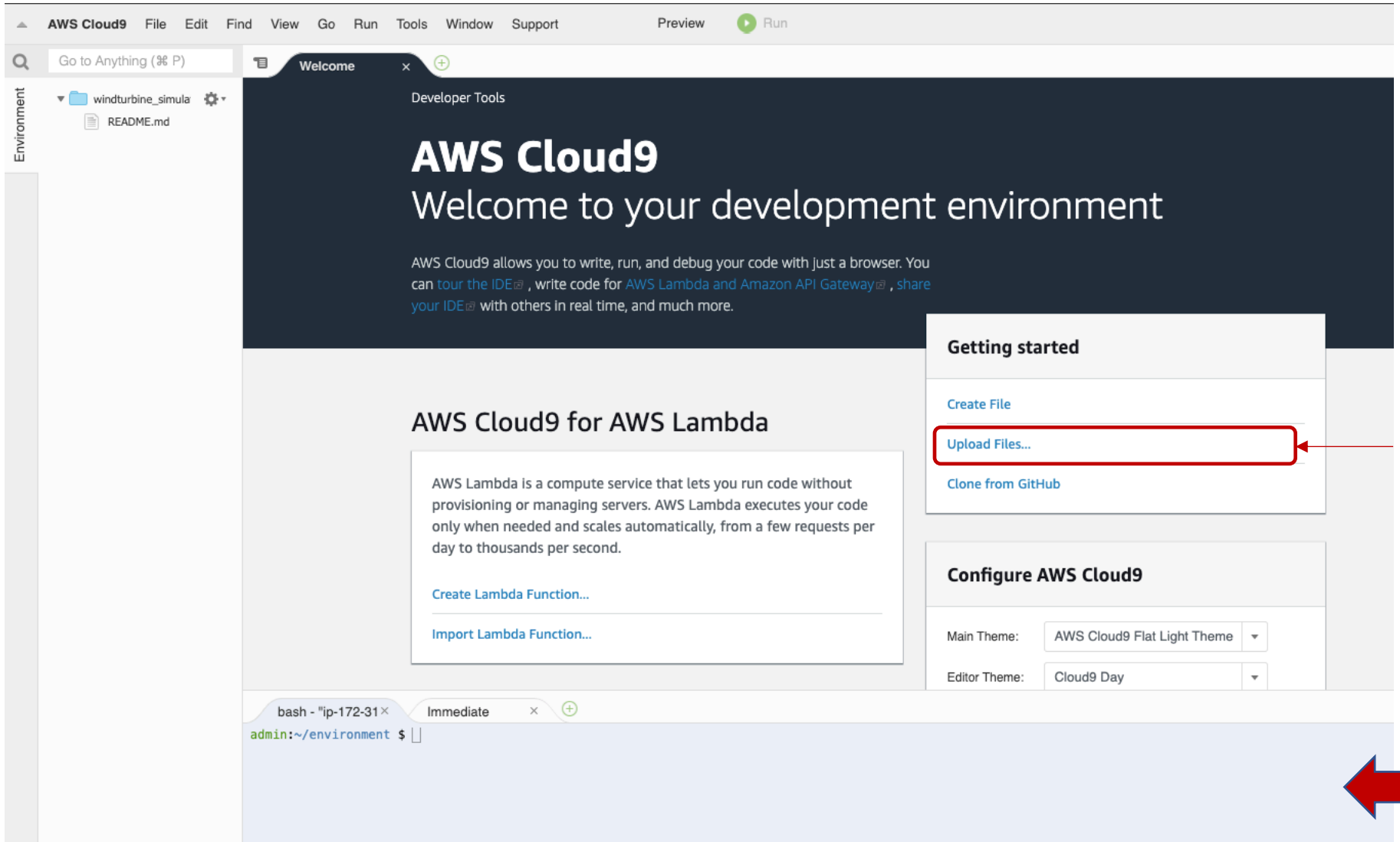


Leave all settings as default and click Next.



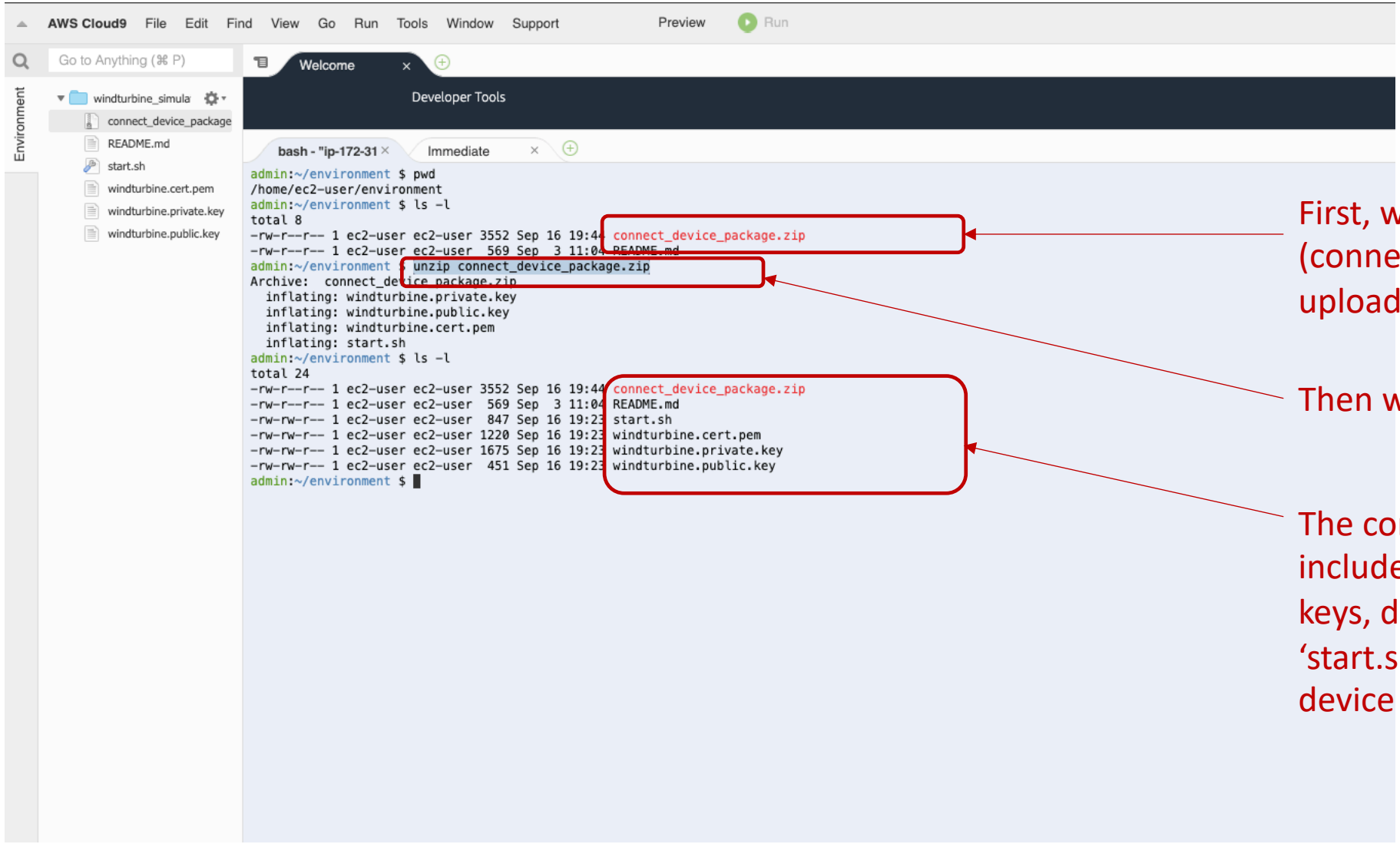
Click ‘Create environment’ on the next screen.

It will take a few minutes before Cloud9 environment is ready…



We will upload the files that we downloaded from IoT Core here

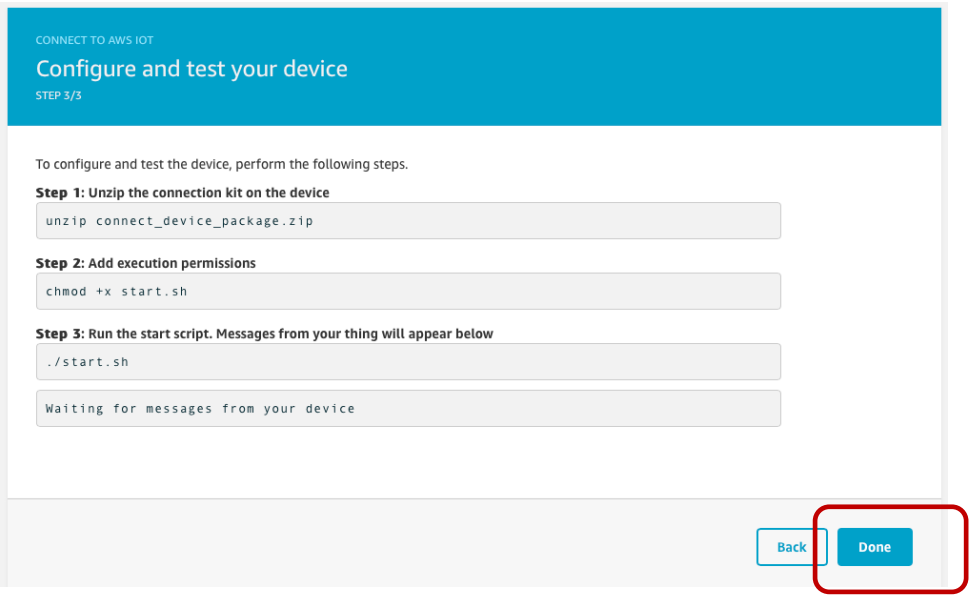
This is terminal window



First, we confirm the package (connect_device_package) is uploaded to cloud9.

Then we unzip the file

The content in this package includes the public and private keys, device certificate and ‘start.sh’ script that will setup the device and get it ready for use.

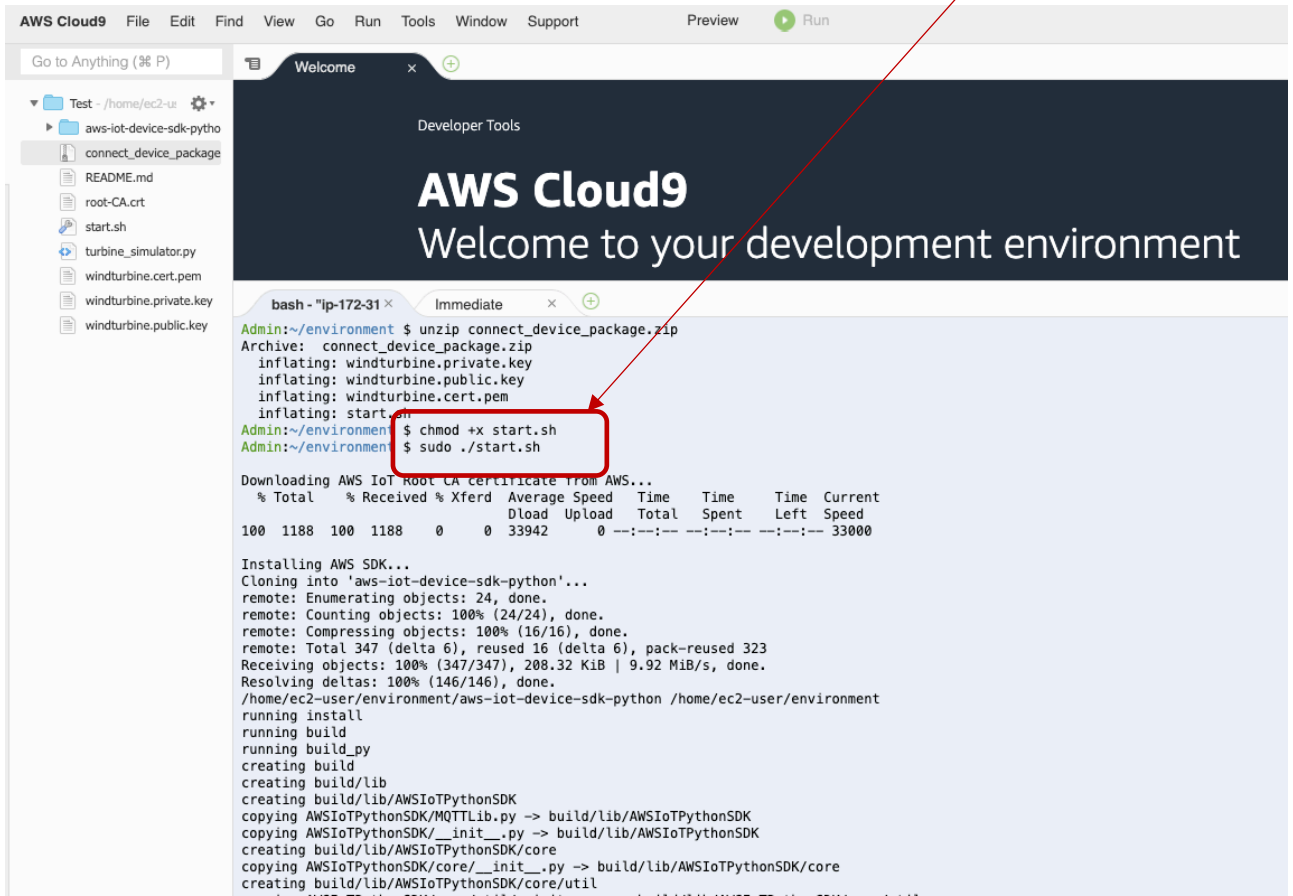


Now go back to the IoT Console, copy step 2 and 3, and run on Cloud9 terminal to finish setting up the IoT device.

Click 'Done', after running this.

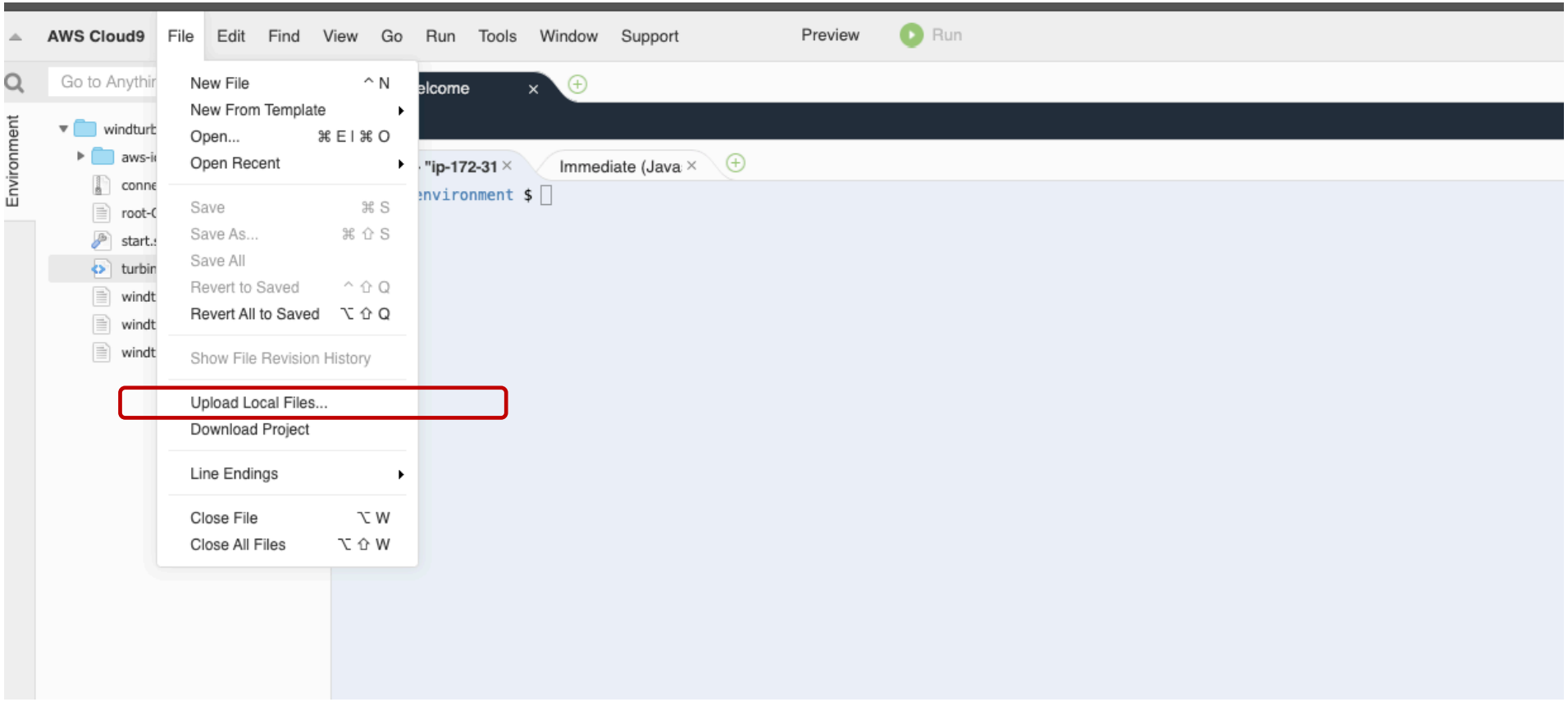
Important: ensure to run this script using sudo

sudo ./start.sh

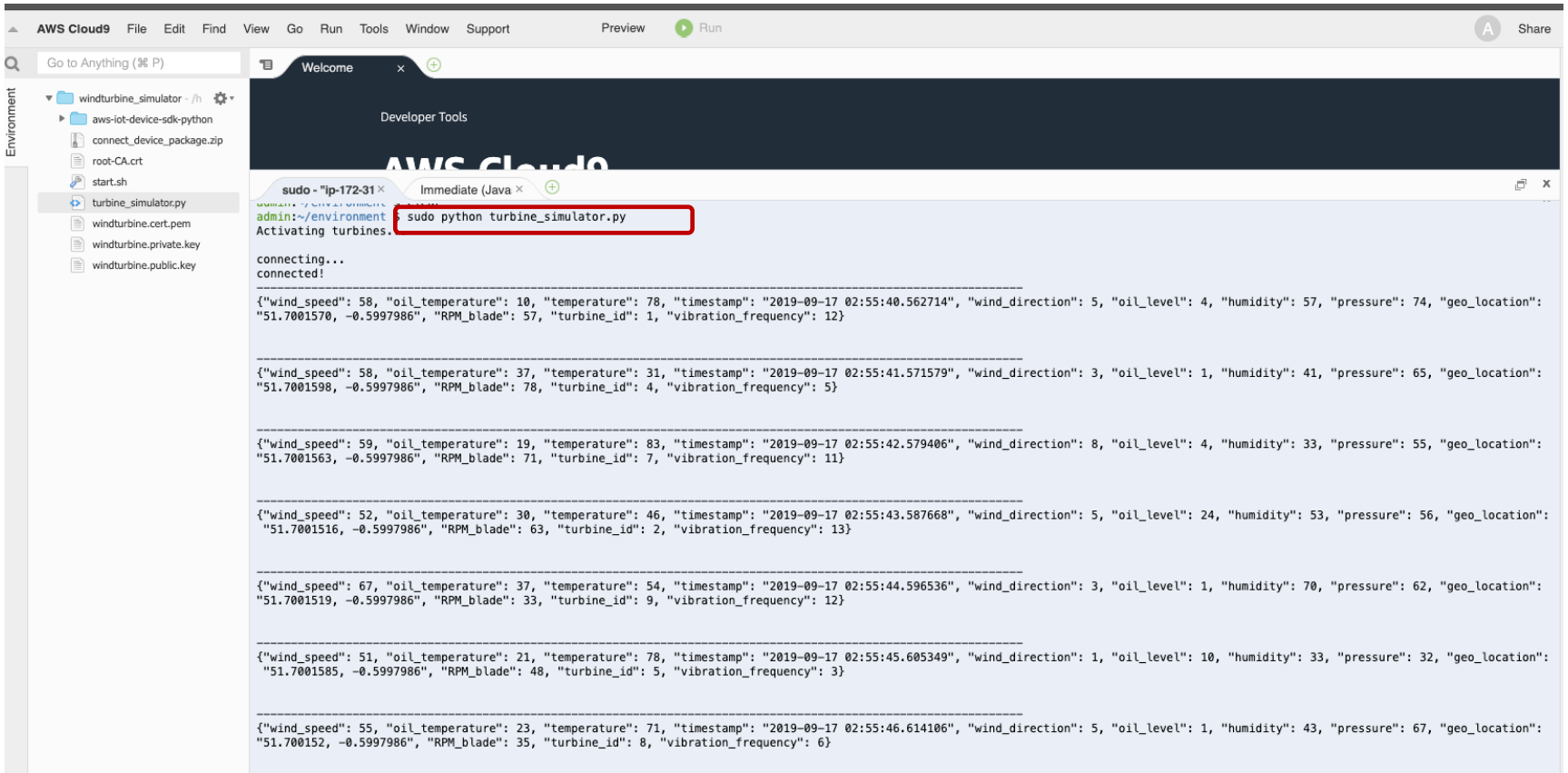


When you run script *start.sh*, it downloads the AWS IoT SDK and starts sending sample messages to AWS IoT core.

Now that we have setup the device on Cloud9, you are ready to upload the device simulation code (*turbine_simulator.py*), that you downloaded with this guide.



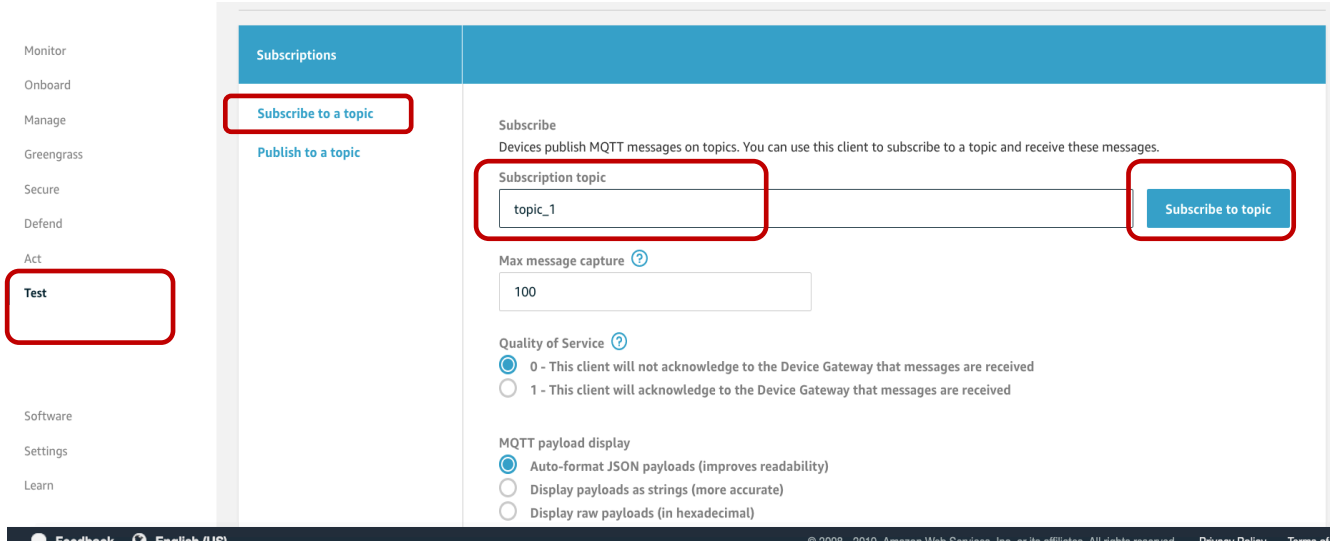
Now, run the python script…
sudo python turbine_simulator.py



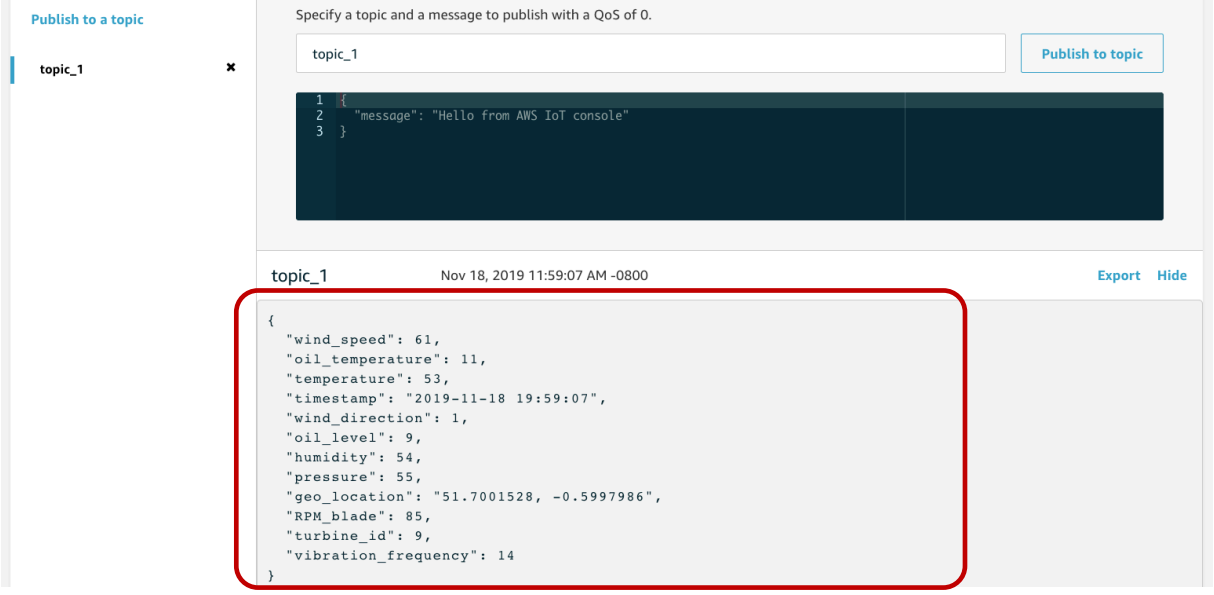
The simulated device is now sending data to AWS IoT Core on the topic “topic_1”. Please note: Let the simulator run for the entire duration of this workshop

This Python script generates telemetry data points using a random function and the anomalies are introduced after few seconds in a loop.

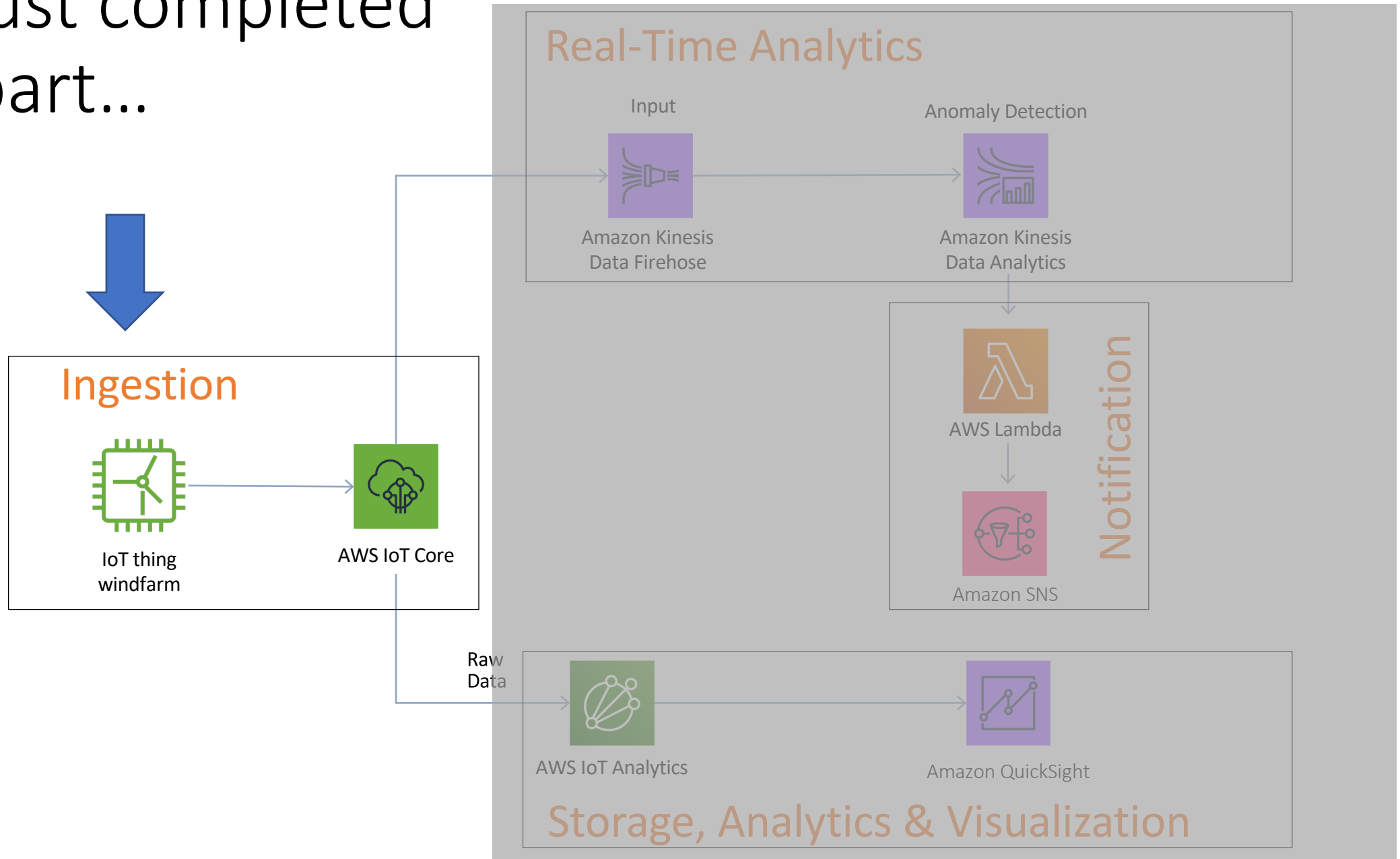
You can test the data is flowing to AWS IoT Core, by selecting ‘Test’ from the left hand panel and subscribing to topic_1 topic



After subscribing, you can see the incoming data in the IoT Core console, as shown below…



You just completed this part…

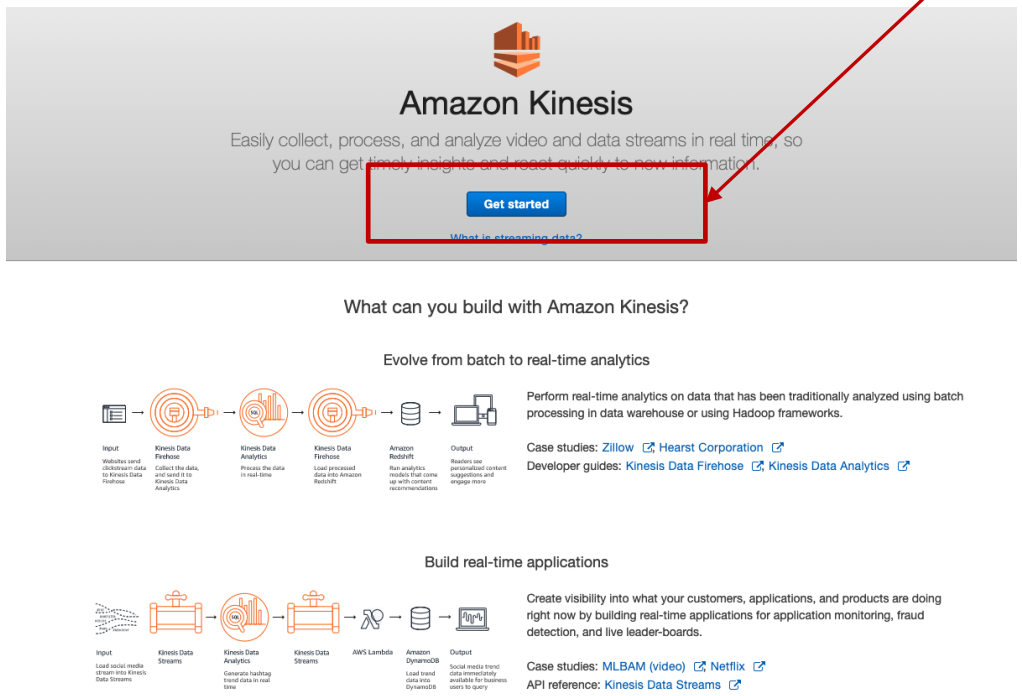


We will go over these labs:

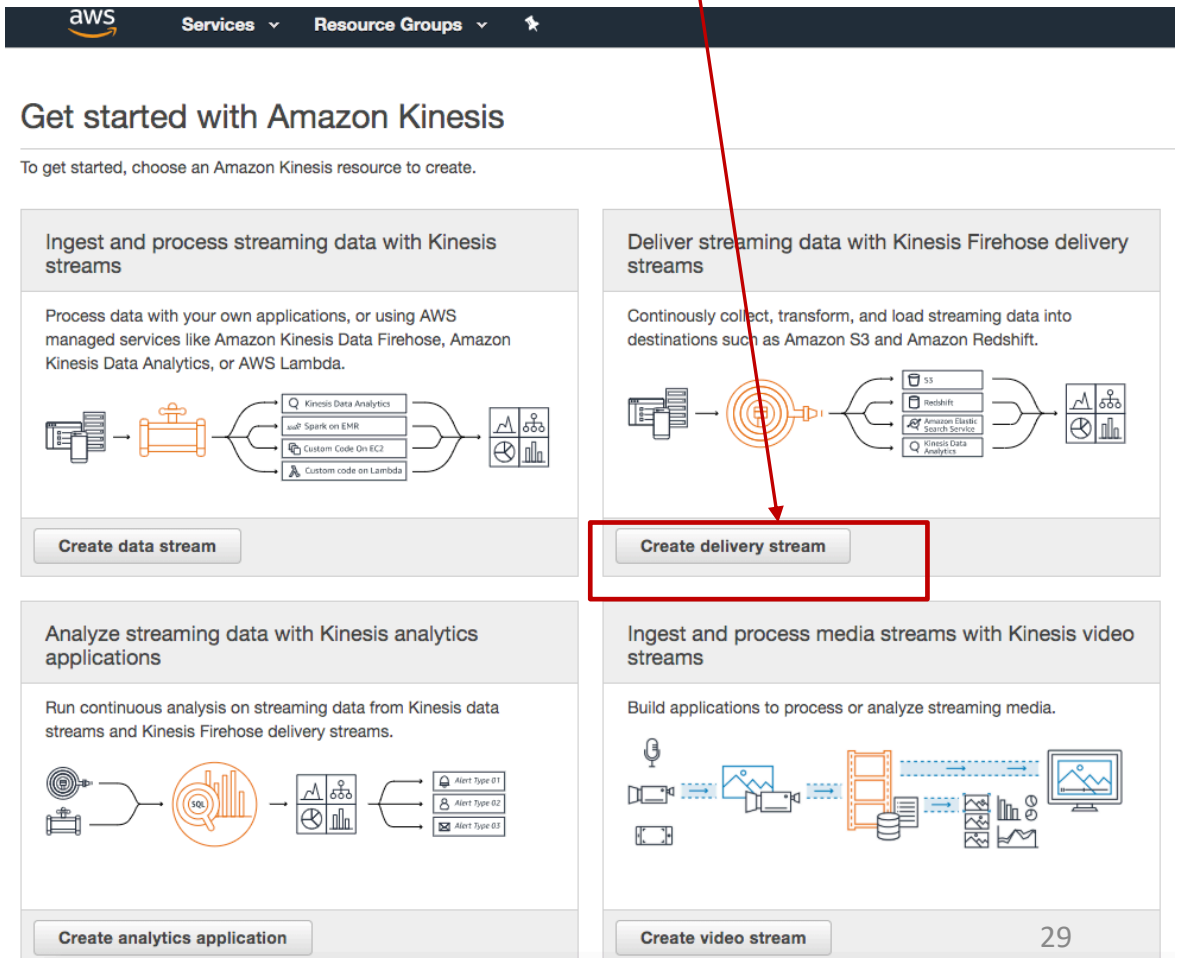
Lab 1 –IoT ‘thing’ & Device Simulation	15 mins
Lab 2 –Kinesis Firehose and Analytics Pipeline	30 mins
Lab 3 –IoT Analytics	30 mins
Lab 4 –Notification (Optional)	30 mins

Set up Kinesis Firehose

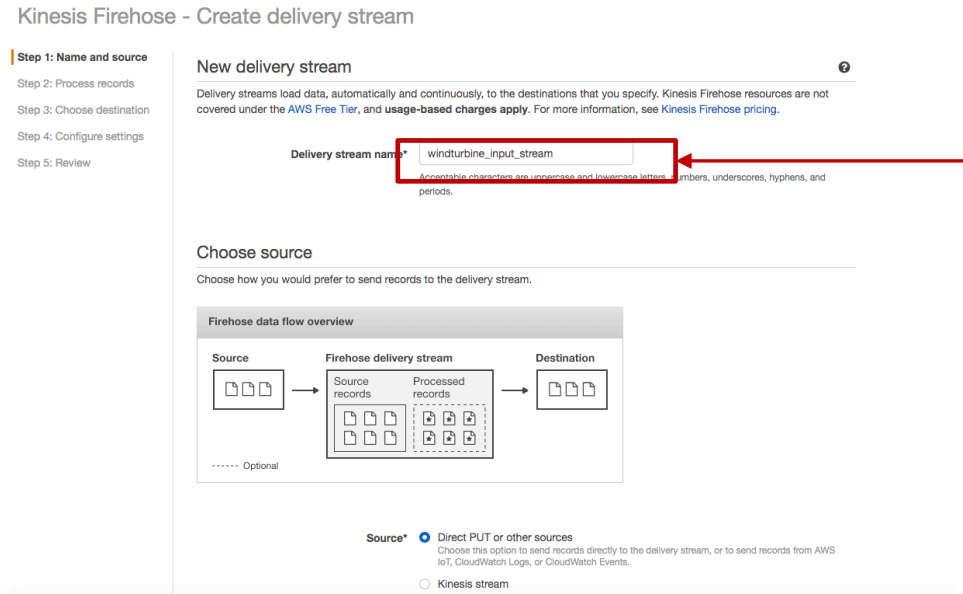
We will first create delivery stream, that will take input from AWS IoT.
Select ‘Kinesis Firehose’ from the Kinesis Console, then click ‘Get started’…



…on the next screen, click ’Create delivery stream’



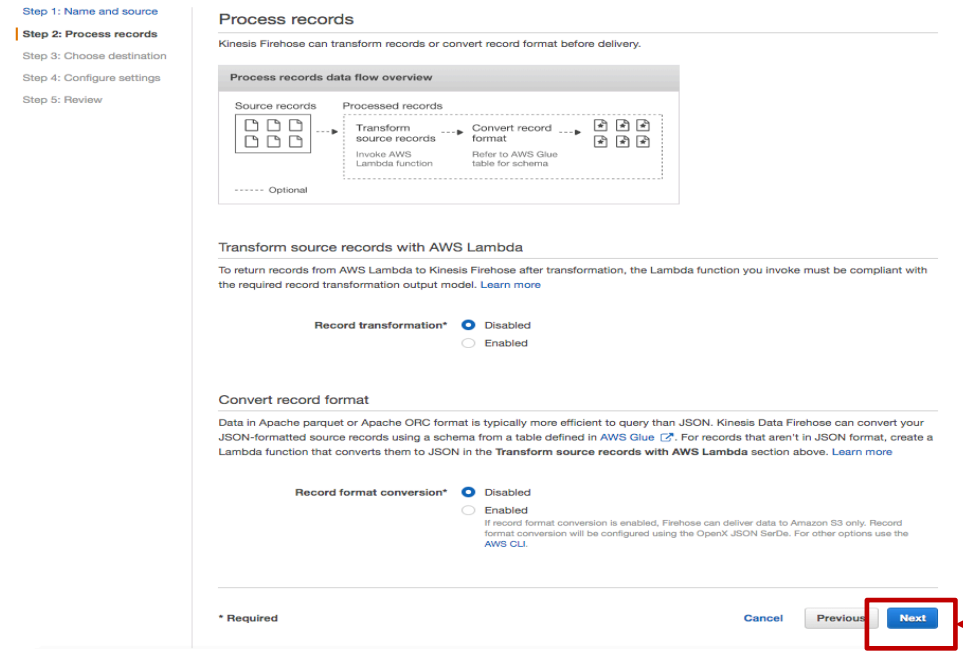
Step1: Name and Source



Give a name to this firehouse, we call it windturbine_input_stream

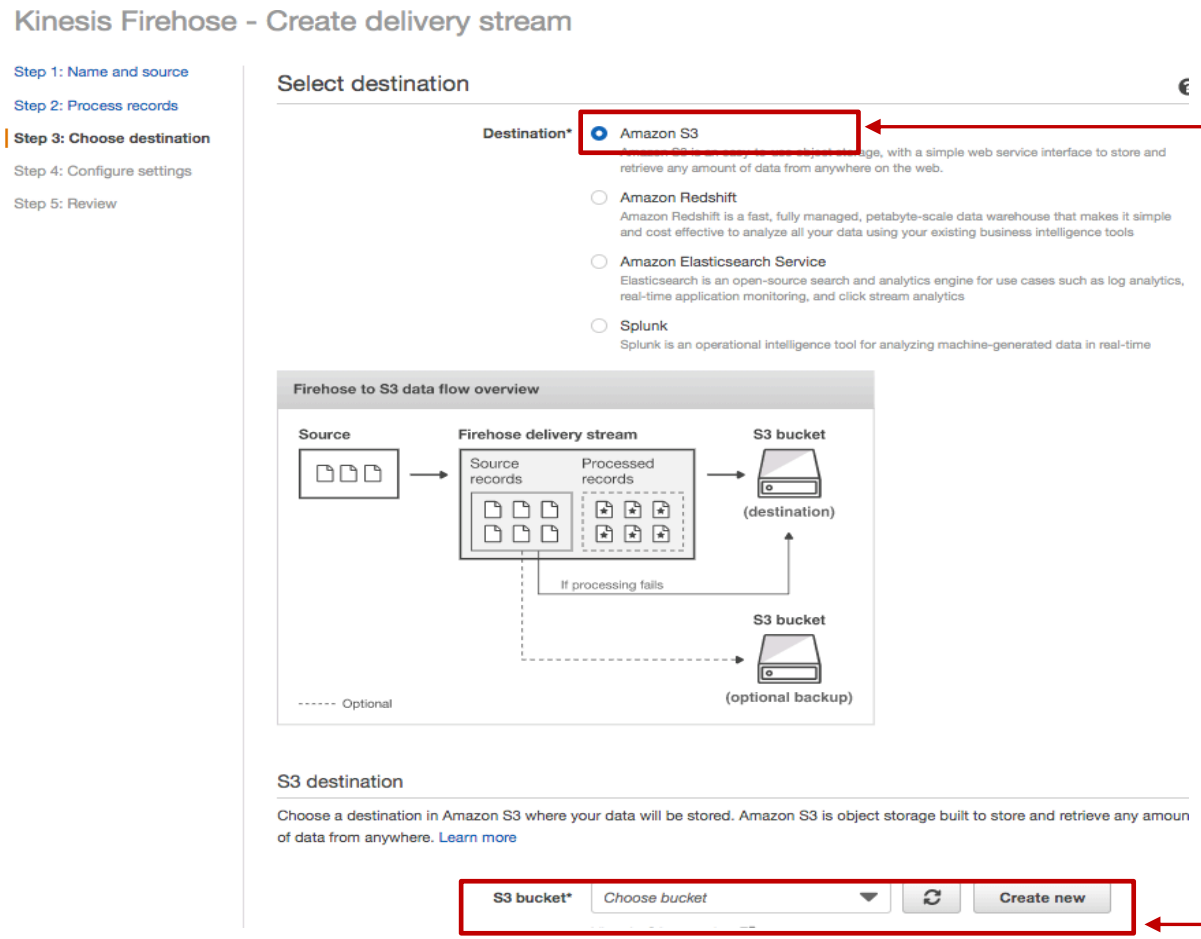
Leave other options default, and click ‘next’….

Step2: Process Records



…on the next page (step 2), leave all settings default and click ‘Next’ again.

Step3: Choose Destination

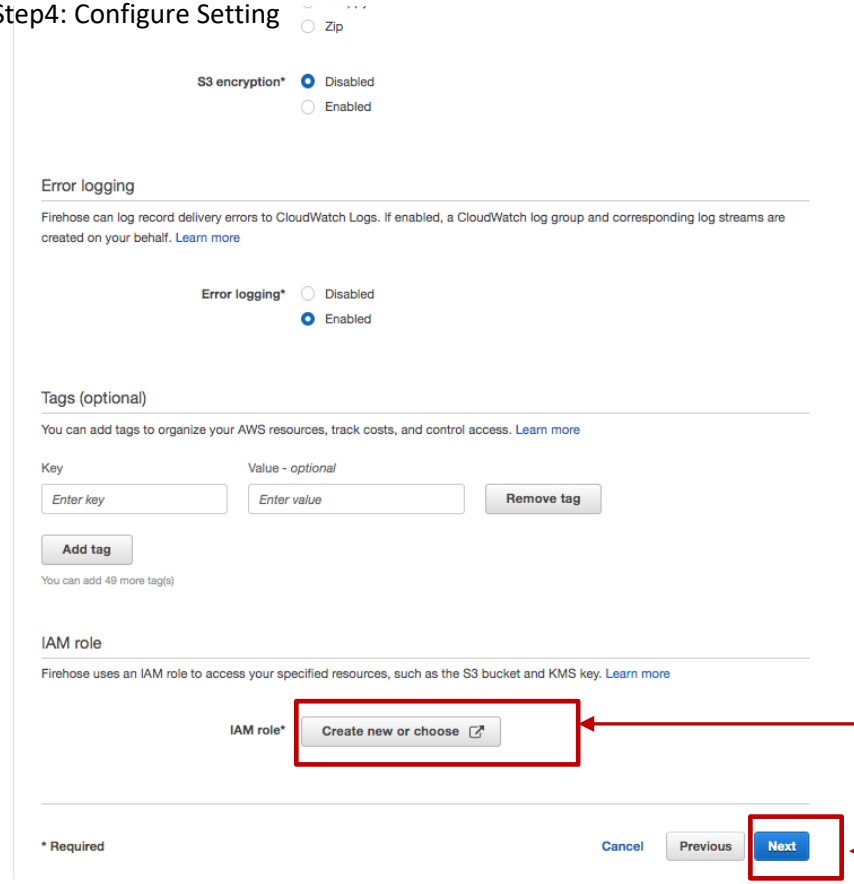


At this step we will choose destination ‘Amazon S3’, to store incoming telemetry as a raw data.

You can choose an existing S3 Buckets or create a new one. We will create a new bucket, we can call it ‘<*yourname123*>-raw-data’. Remember to add your name or a unique prefix or suffix to this bucket name as bucket names are unique.

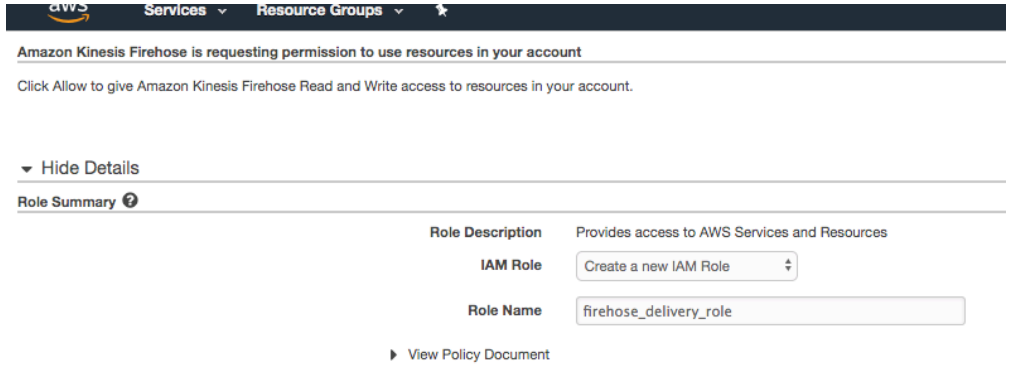
We can leave all other settings default and click next.

Step4: Configure Setting



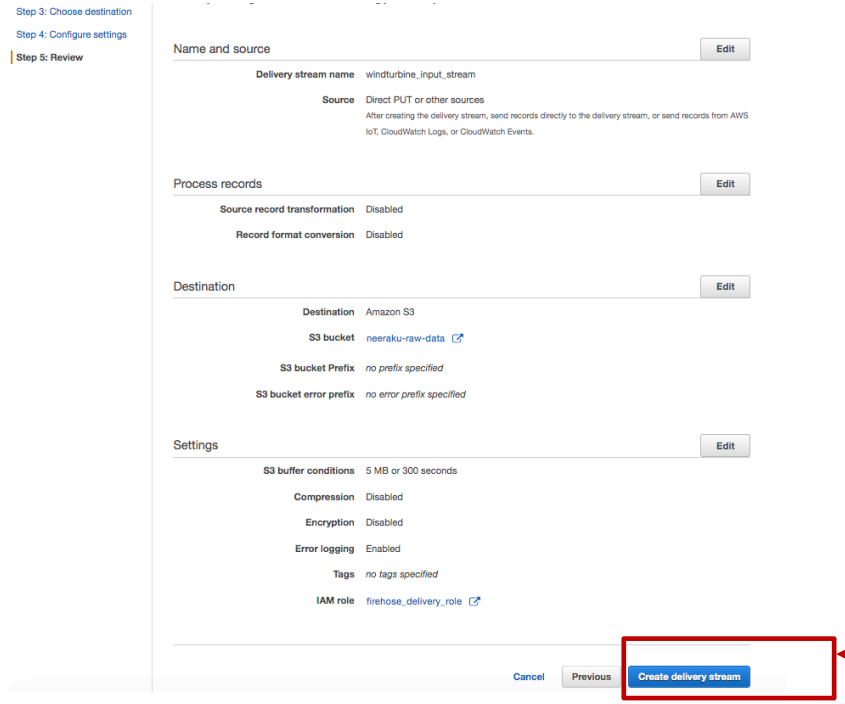
On the next page (Step 4), scroll down to the bottom. If you do not have a existing IAM role for Kinesis Firehose, we will need to create one. This role will be attached to the Kinesis Firehose and enable it to integrate with other other AWS services.

Click Next

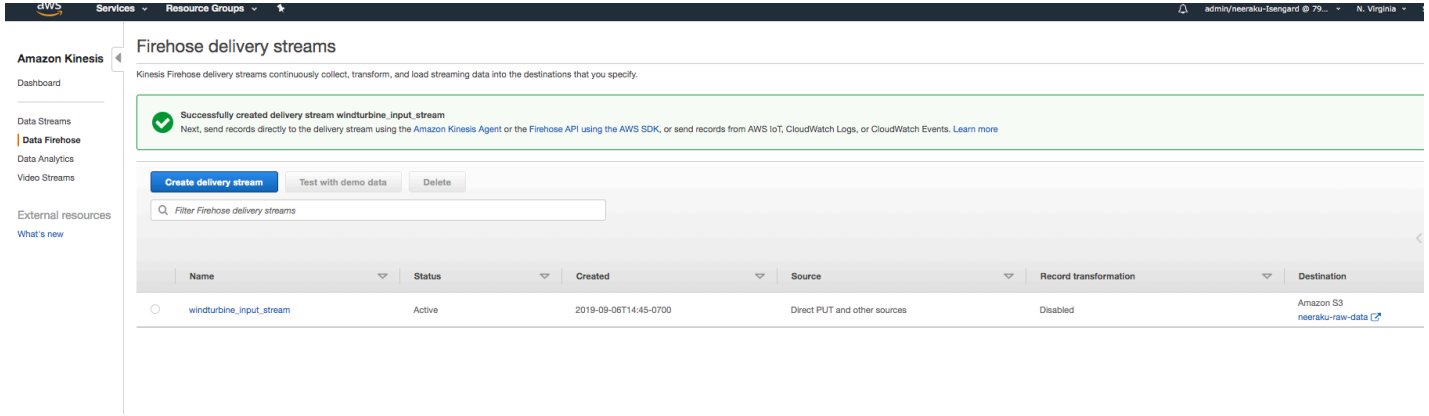


When you select ‘create a new role’, you will see the screenshot on the left. You can change the role name or leave it default. Click ‘Allow’ and continue.

Step5: Review and create Firehose

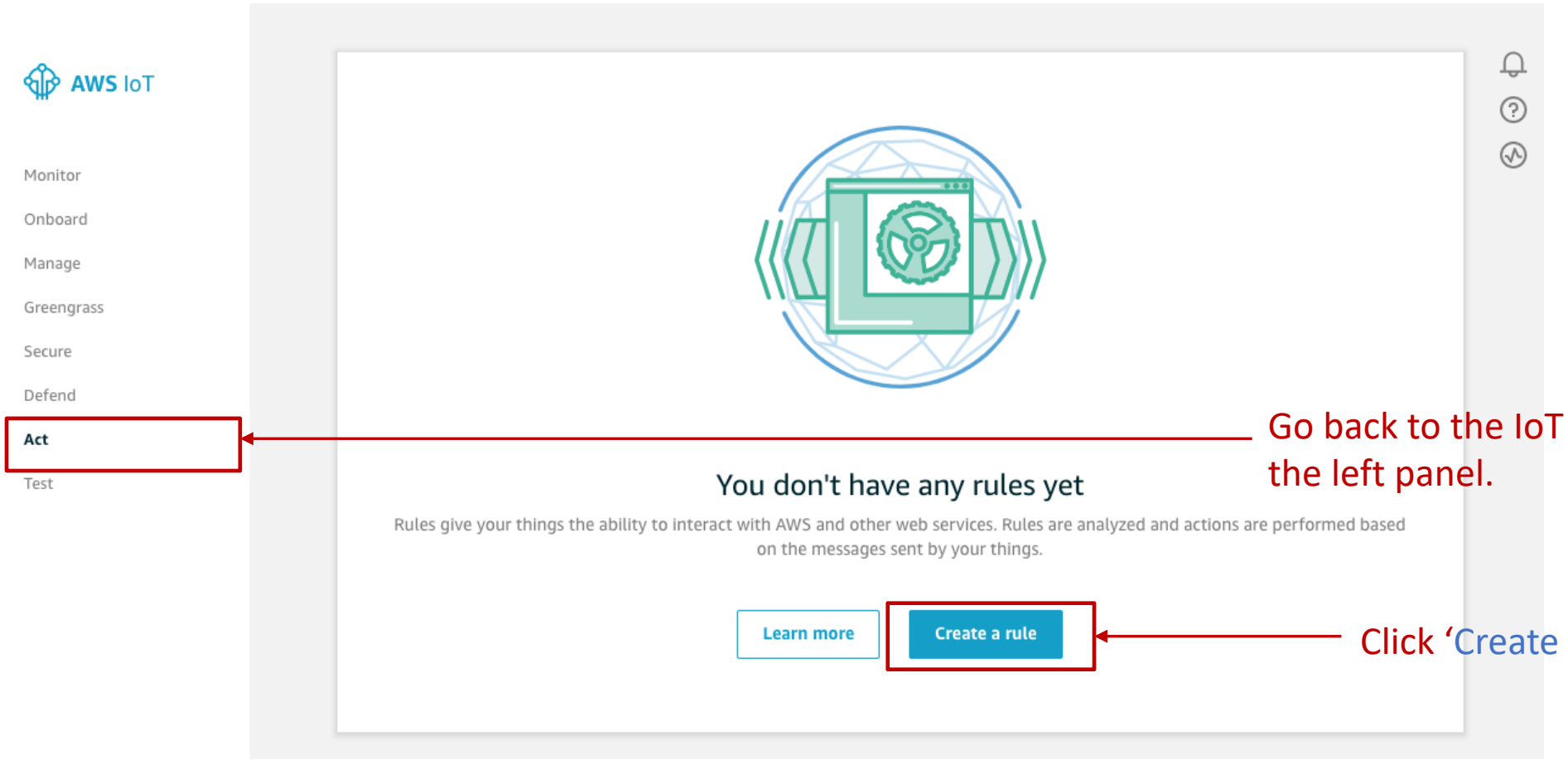


Finally, review the settings (Step 5) and click 'Create delivery stream'



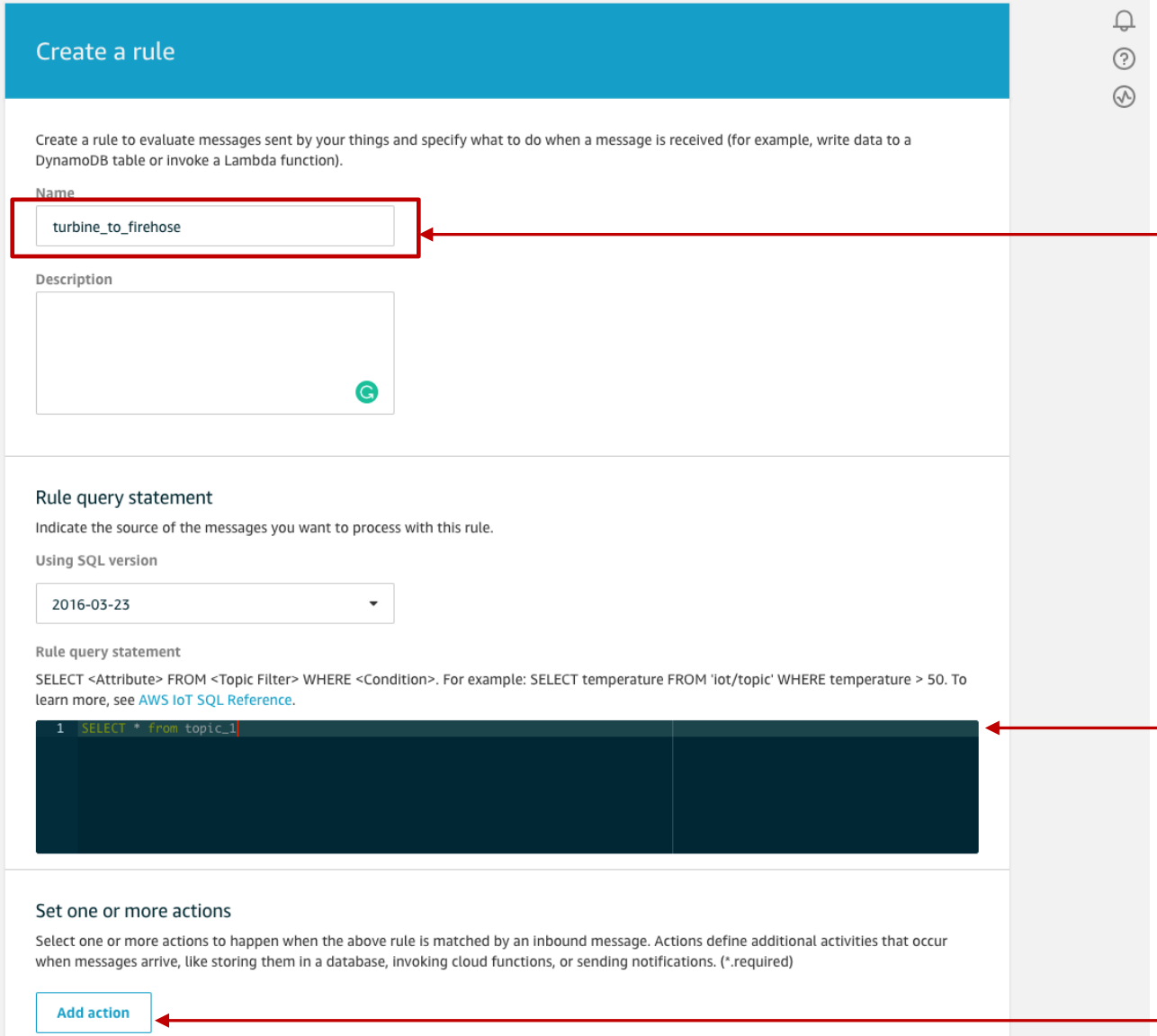
You will now see it on the Firehose Dashboard.

Setup IoT Rule
(Connect IoT Core to Kinesis Firehose)



Go back to the IoT console, and select ‘Act’ from the left panel.

Click ‘Create a rule’

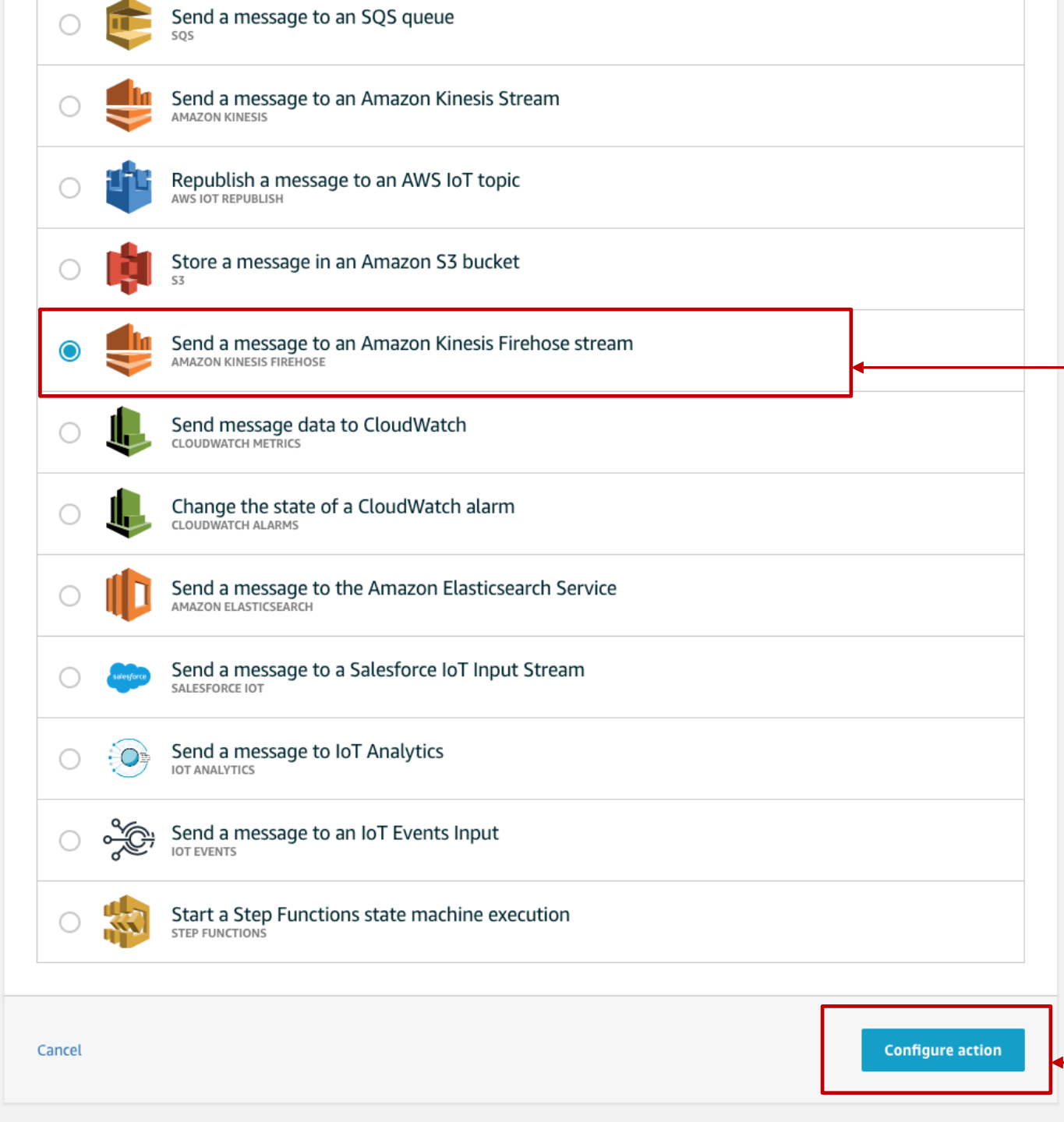


Give this rule a name e.g. turbine_to_firehose

Write SELECT * FROM 'topic_1'
In this box

Important: Ensure the topic name is in single quotes.

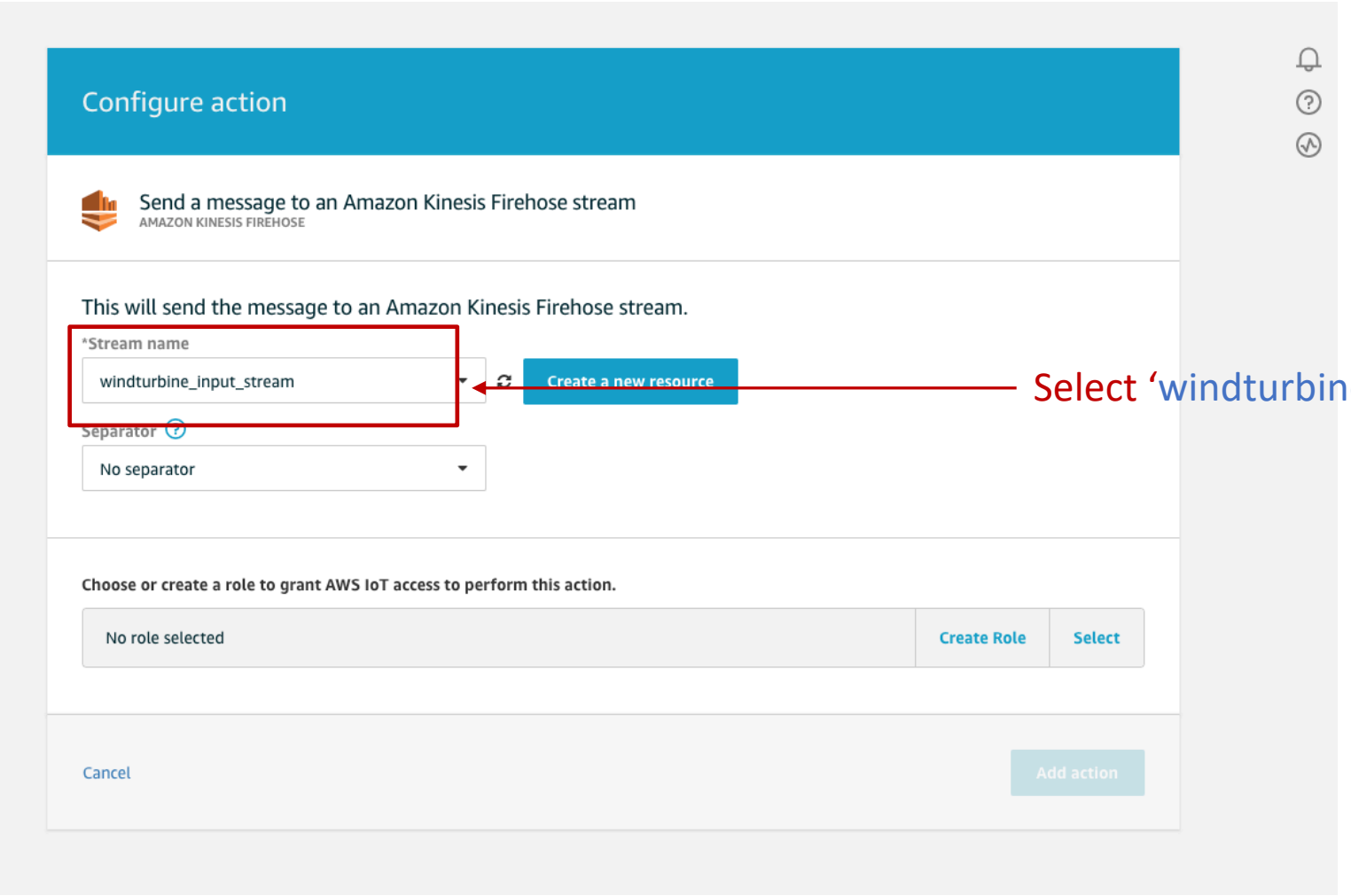
Click Add actions



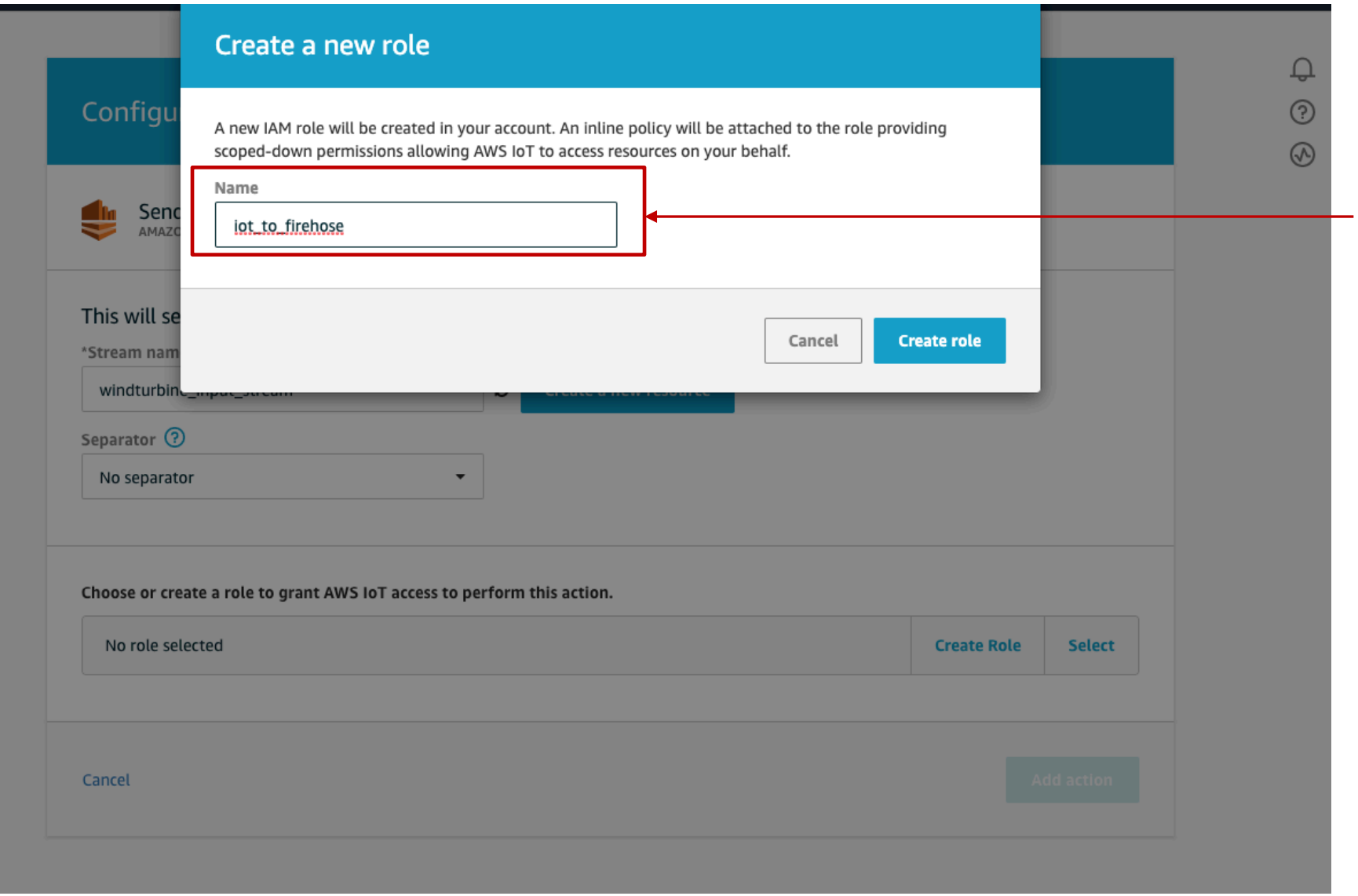
Select Amazon Kinesis Firehose.

*Note: Ensure you are selecting **Kinesis Firehose stream** and NOT **Kinesis Stream***

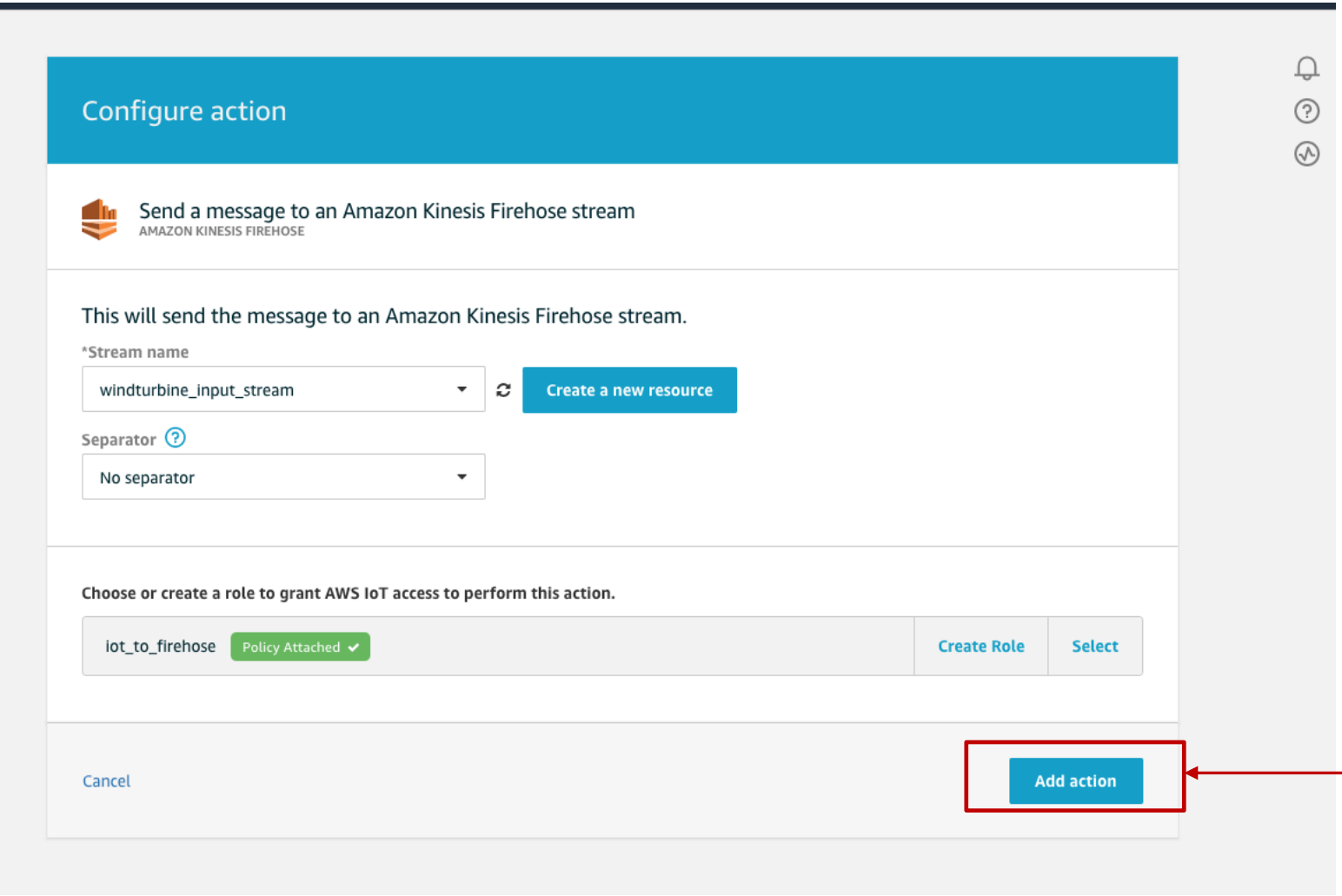
Click 'Configure Action'.



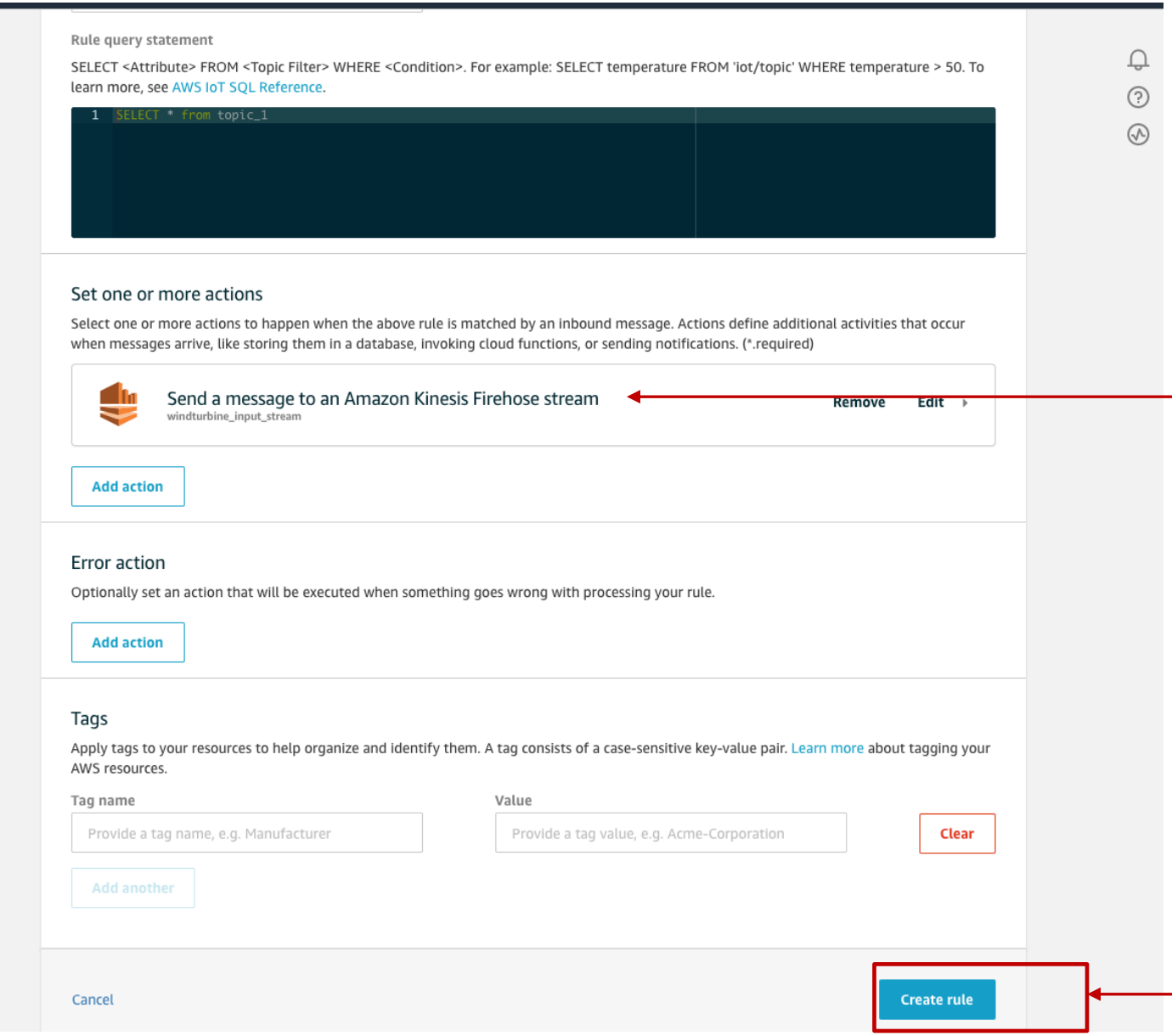
Select ‘windturbine_input_stream’, we created earlier.



Create an IAM role to enable IoT Core to access Kinesis Firehose



Click ‘ Add Action’

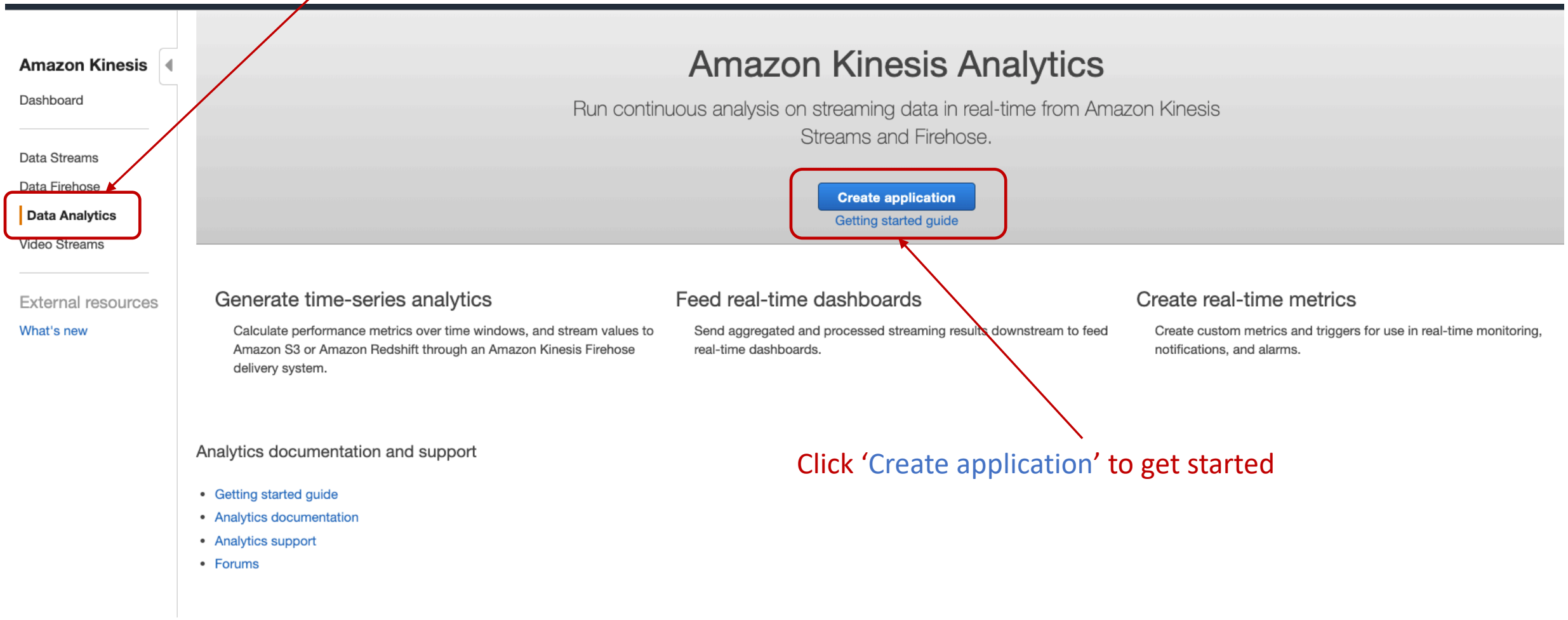


Amazon Kinesis Firehose stream is now setup as target where IoT rules engine will push the incoming data.

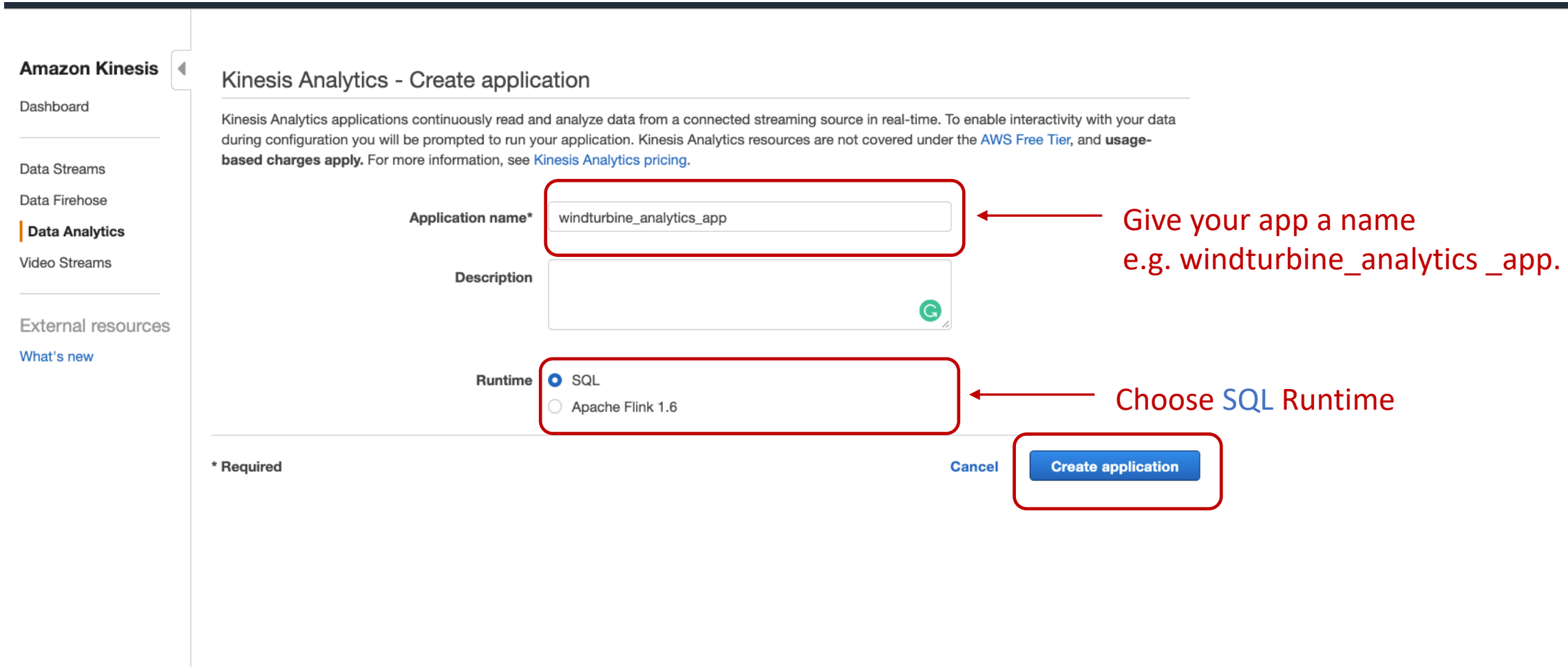
Click ‘ Create rule’ to complete this step

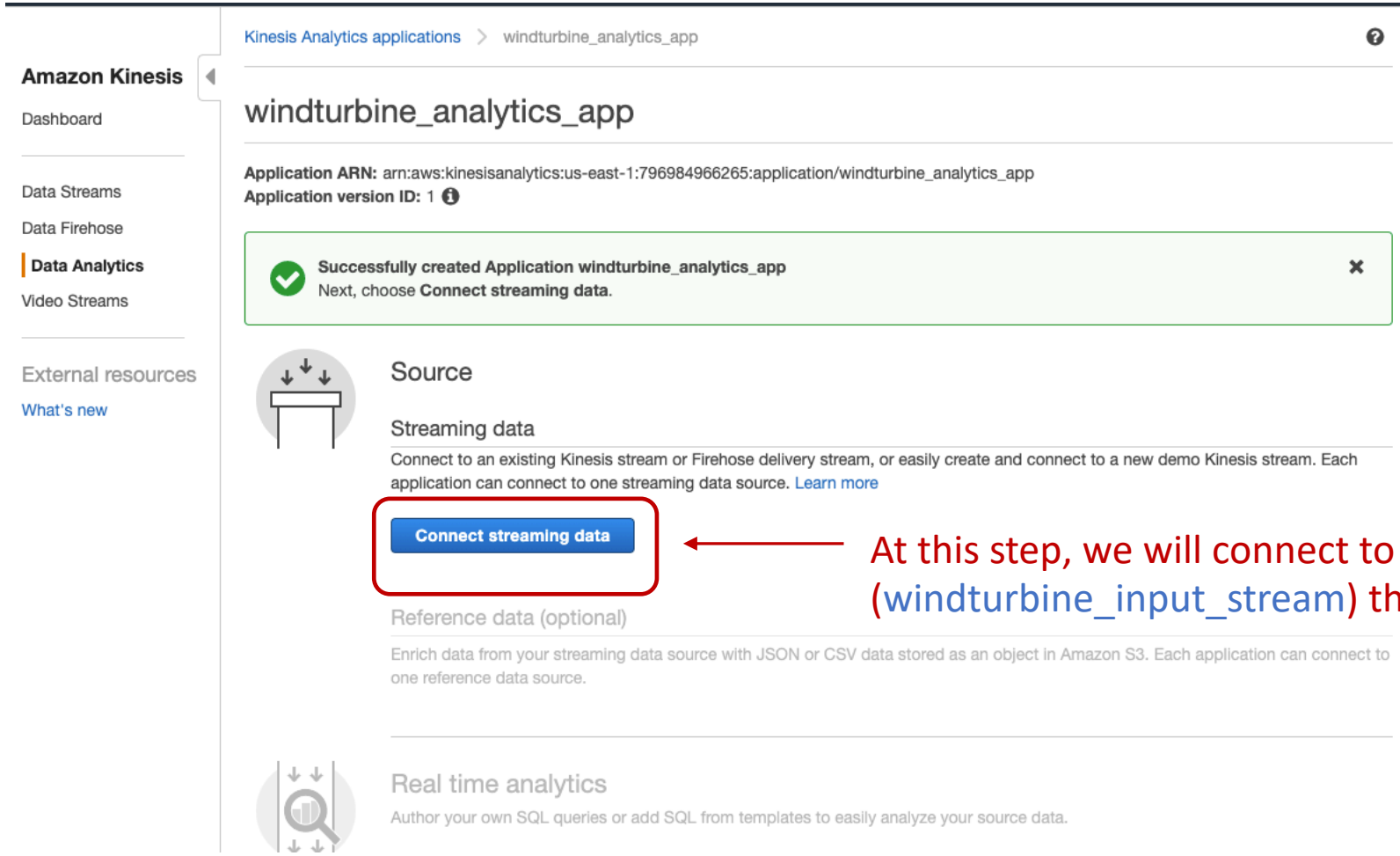
Setup Kinesis Data Analytics Application

On Kinesis dashboard, select ‘Data Analytics’.



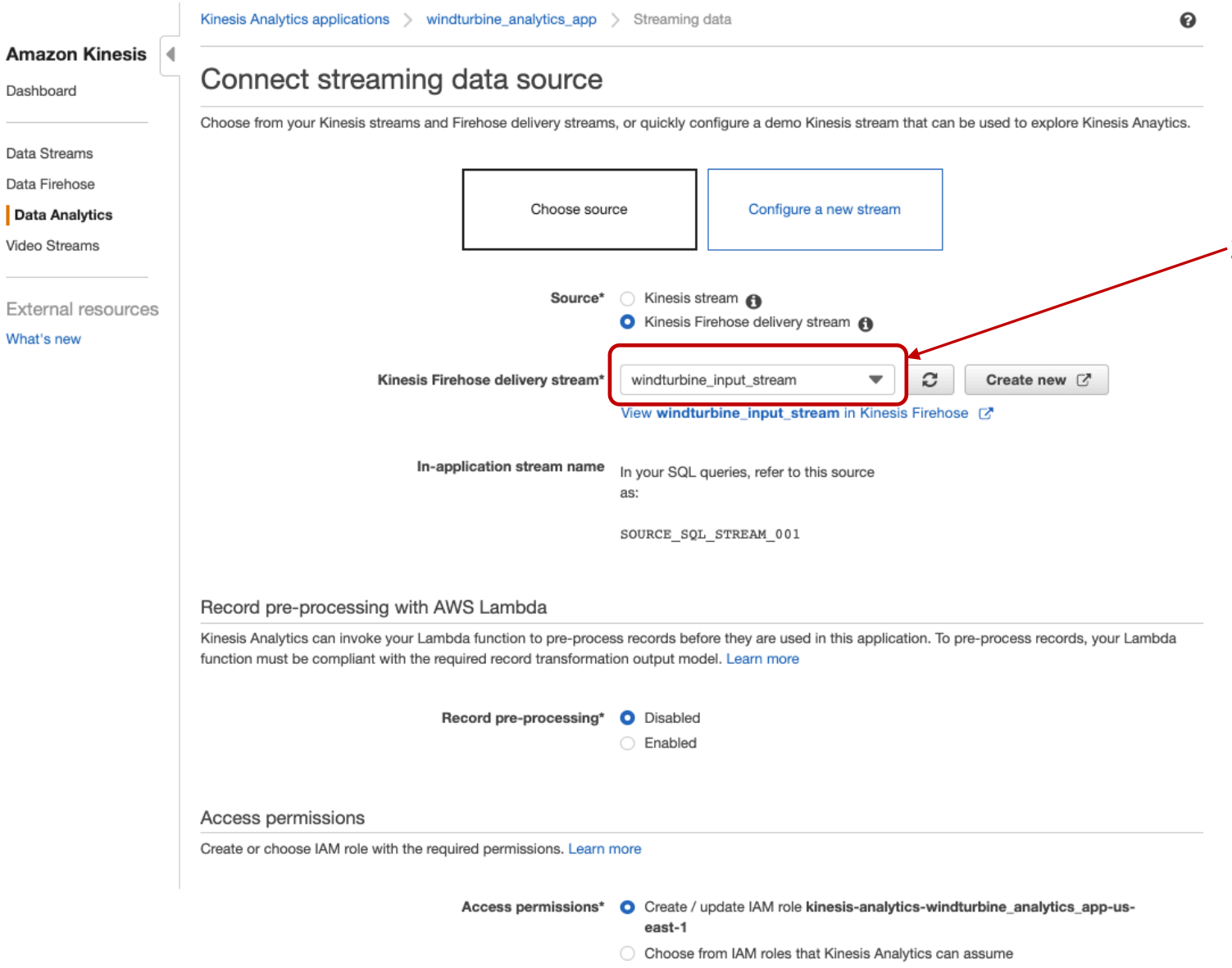
Click ‘Create application’ to get started


Amazon Kinesis
Dashboard
Data Streams
Data Firehose
Data Analytics
Video Streams
External resources
What's new
Kinesis Analytics - Create application
Kinesis Analytics applications continuously read and analyze data from a connected streaming source in real-time. To enable interactivity with your data during configuration you will be prompted to run your application. Kinesis Analytics resources are not covered under the AWS Free Tier, and usage-based charges apply. For more information, see Kinesis Analytics pricing.
Application name*
windturbine_analytics_app
Description
Runtime
SQL
Apache Flink 1.6
* Required
Cancel
Create application
Give your app a name
e.g. windturbine_analytics _app.
Choose SQL Runtime



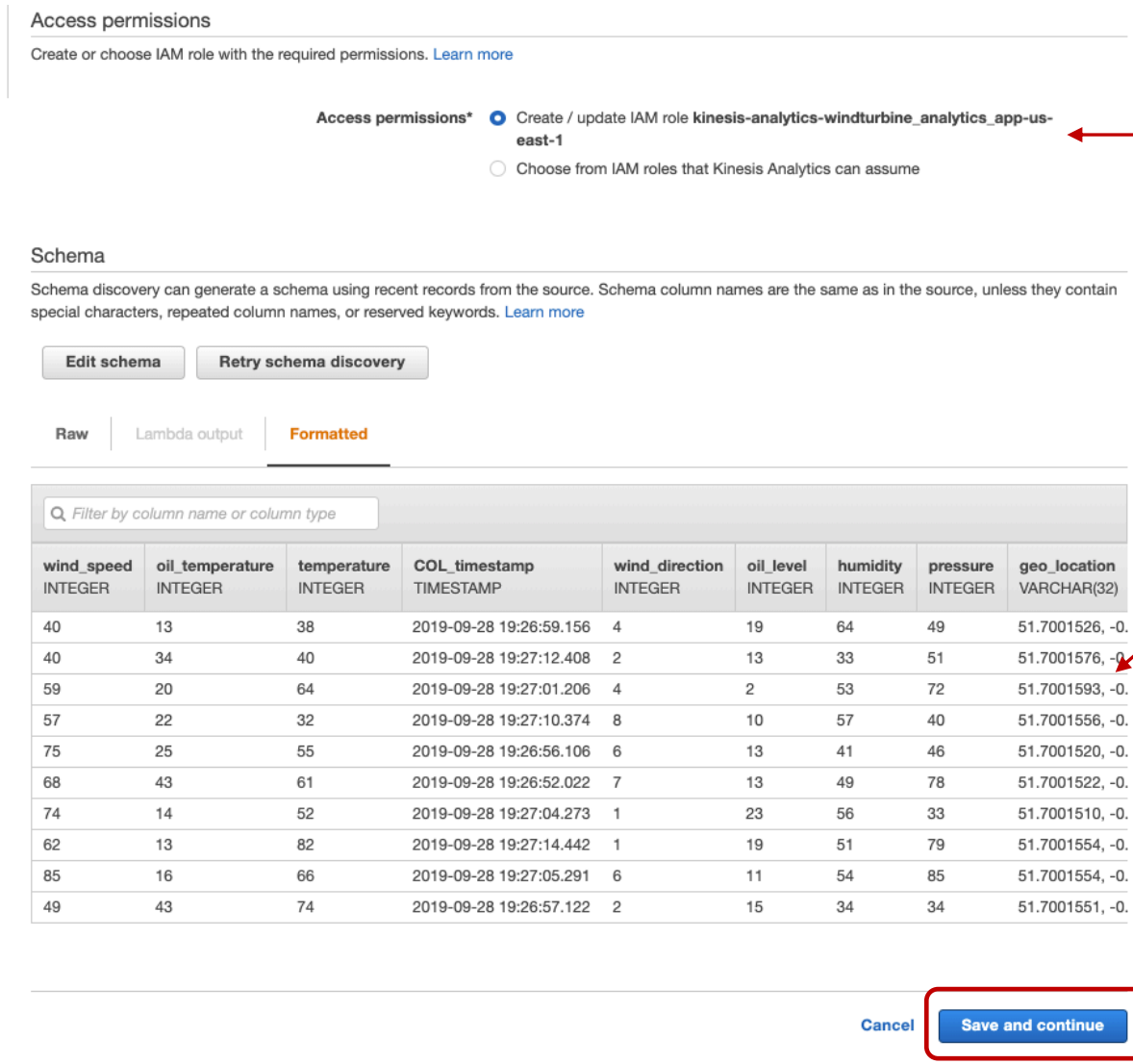
At this step, we will connect to the source, Kinesis Data Firehose (windturbine_input_stream) that we created in the previous step.

Important: Ensure the data simulator is on and it is sending data into Kinesis Firehose. This will enable Kinesis Data Analytics App to discover the schema.



Select 'windturbine_input_stream'

...scroll down



Select to create a new IAM role.

When clicked on ‘Discover Schema’ Kinesis Analytics application will infer the schema from incoming stream, as shows here.

Click ‘Save and continue’ to move to the next step

Data Streams
Data Firehose
Data Analytics
Video Streams

External resources
What's new

Application ARN: arn:aws:kinesisanalytics:us-east-1:796984966265:application/windturbine_analytics_app
Application version ID: 2

Source

Streaming data

Connect to an existing Kinesis stream or Firehose delivery stream, or easily create and connect to a new demo Kinesis stream. Each application can connect to one streaming data source. Learn more

	Source	In-application stream name	ID	Record pre-processing
	Firehose delivery stream windturbine_input_stream	SOURCE_SQL_STREAM_001	2.1	Disabled

Reference data (optional)

Enrich data from your streaming data source with JSON or CSV data stored as an object in Amazon S3. Each application can connect to one reference data source. Learn more

Connect reference data

Real time analytics

Author your own SQL queries or add SQL from templates to easily analyze your source data. Learn more

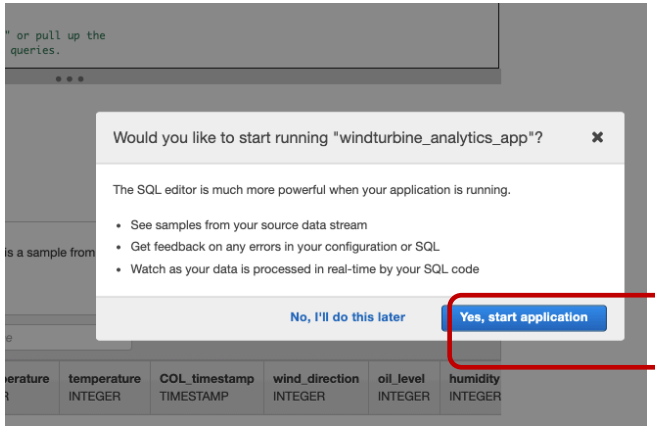
Go to SQL editor

Destination

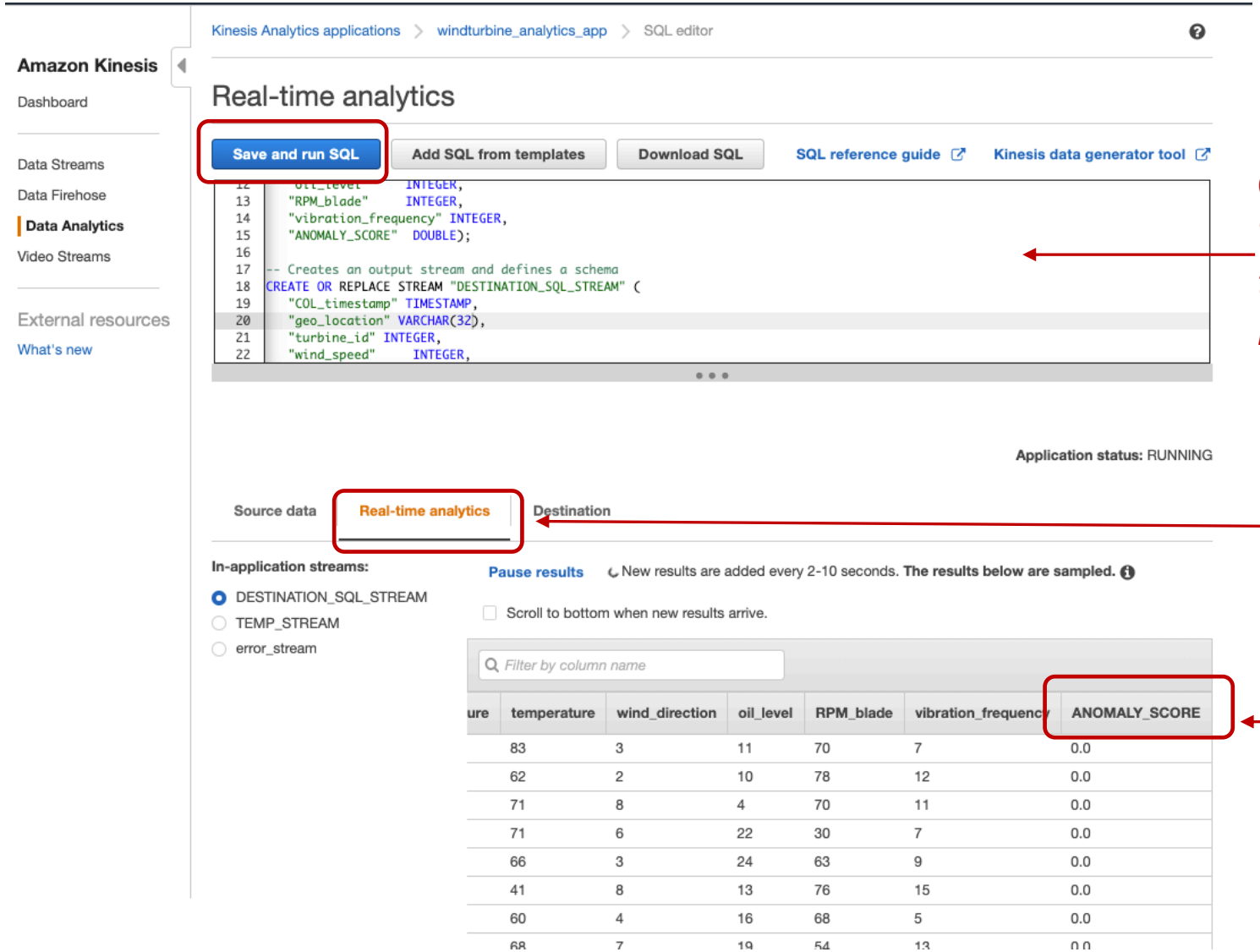
(Optional) Connect an in-application stream to a Kinesis stream, or to a Firehose delivery stream, to continuously deliver SQL results to AWS destinations. The limit is three destinations for each application. Learn more

Exit to Kinesis Analytics applications

Click 'go to SQL editor'



Click to start the application

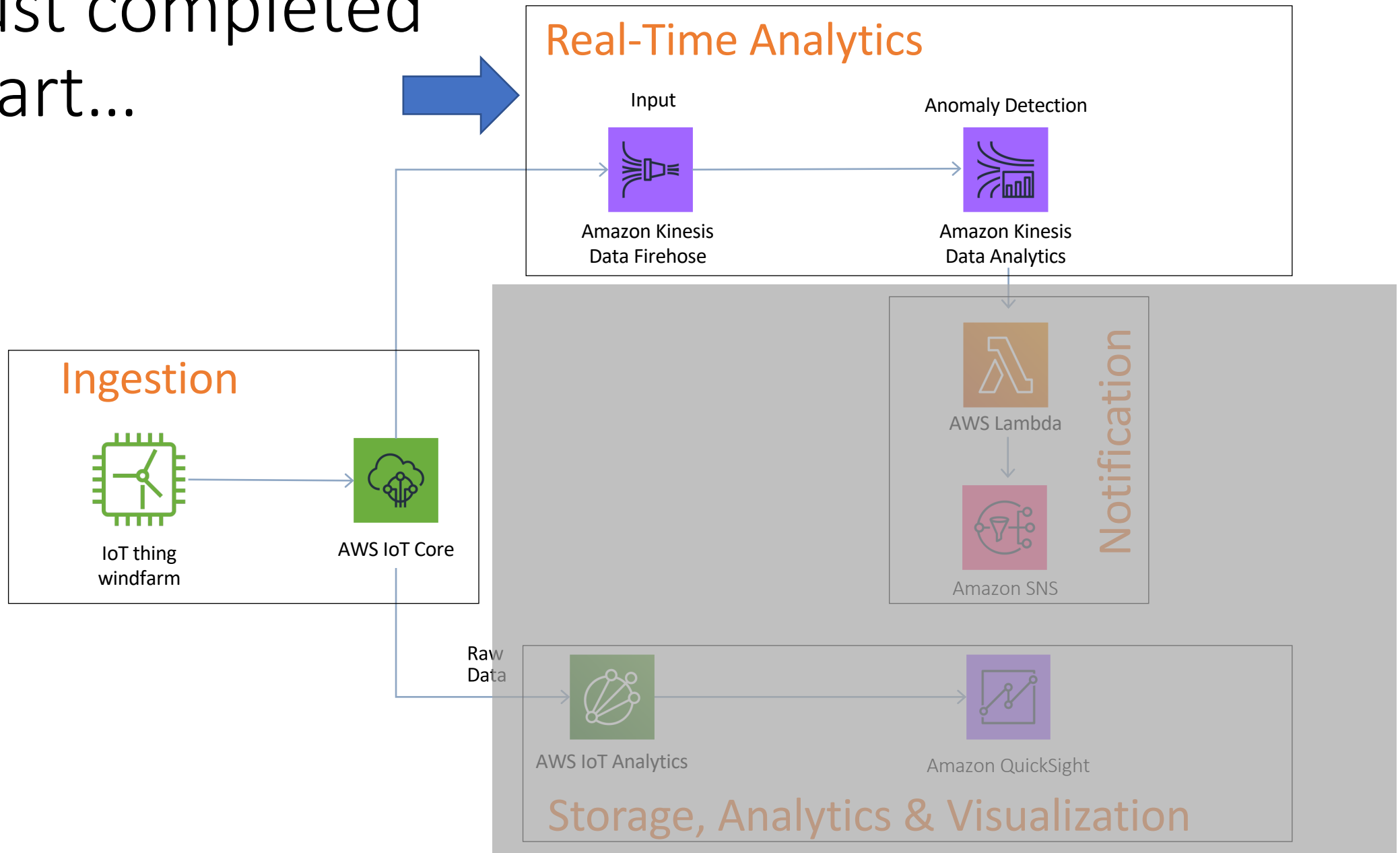


Copy the SQL code from '*kinesis_analytics_app_for_anomaly_detection.sql*', that you downloaded with this guide, and paste here. After that, click '*Save and run SQL*'

Click on '*Real-time analytics*'

Incoming records and ANOMALY_SCORE generated by the application will be shown here.

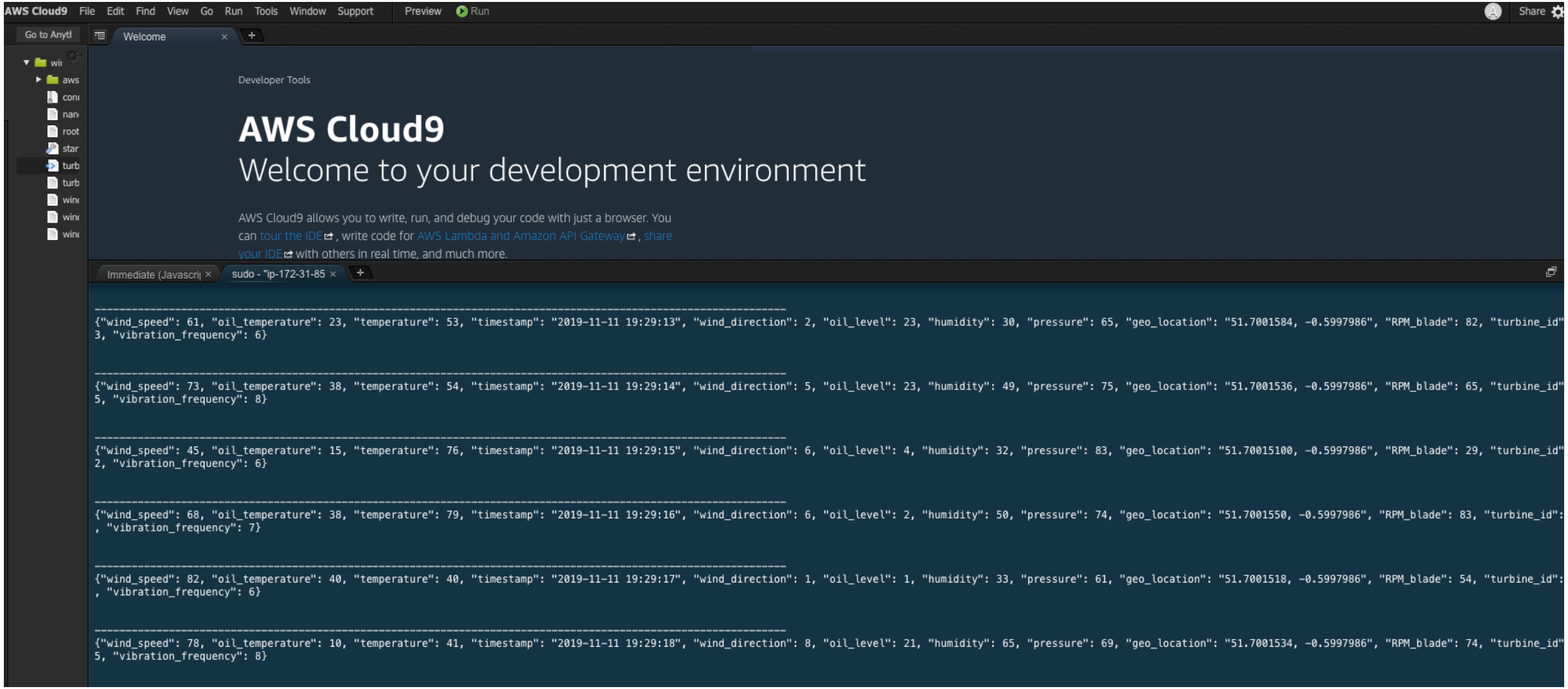
You just completed this part...



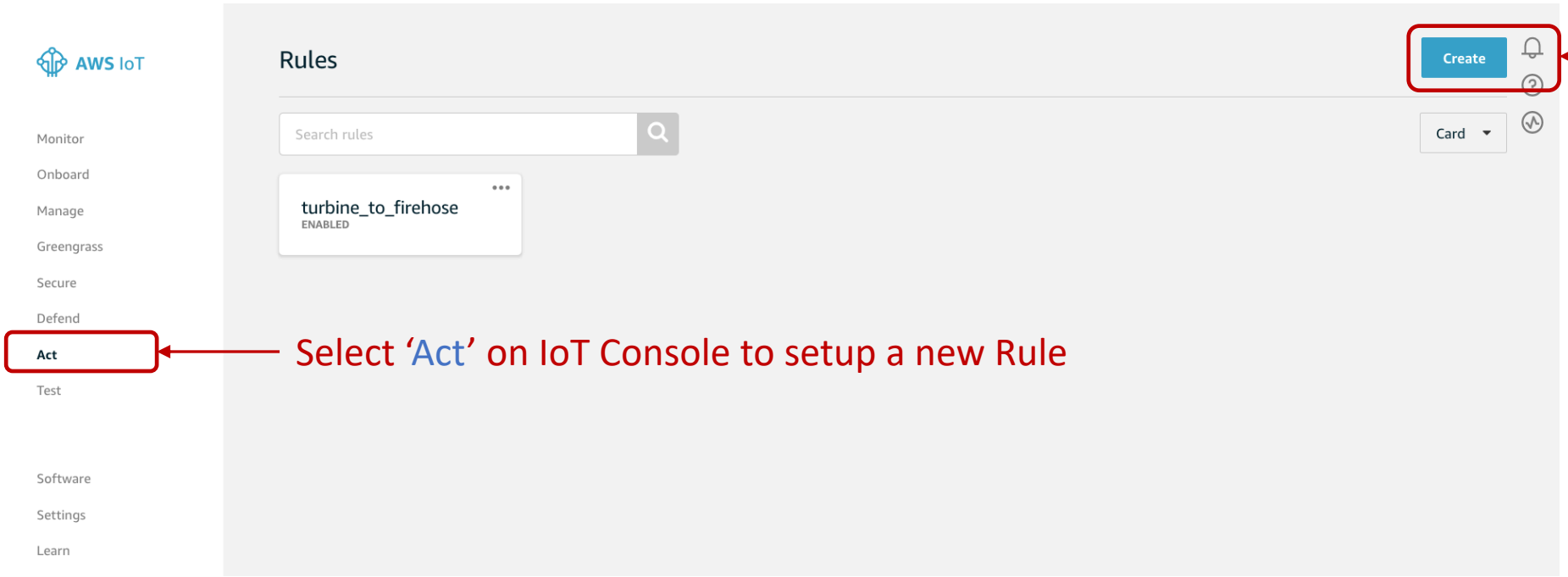
We will go over these labs:

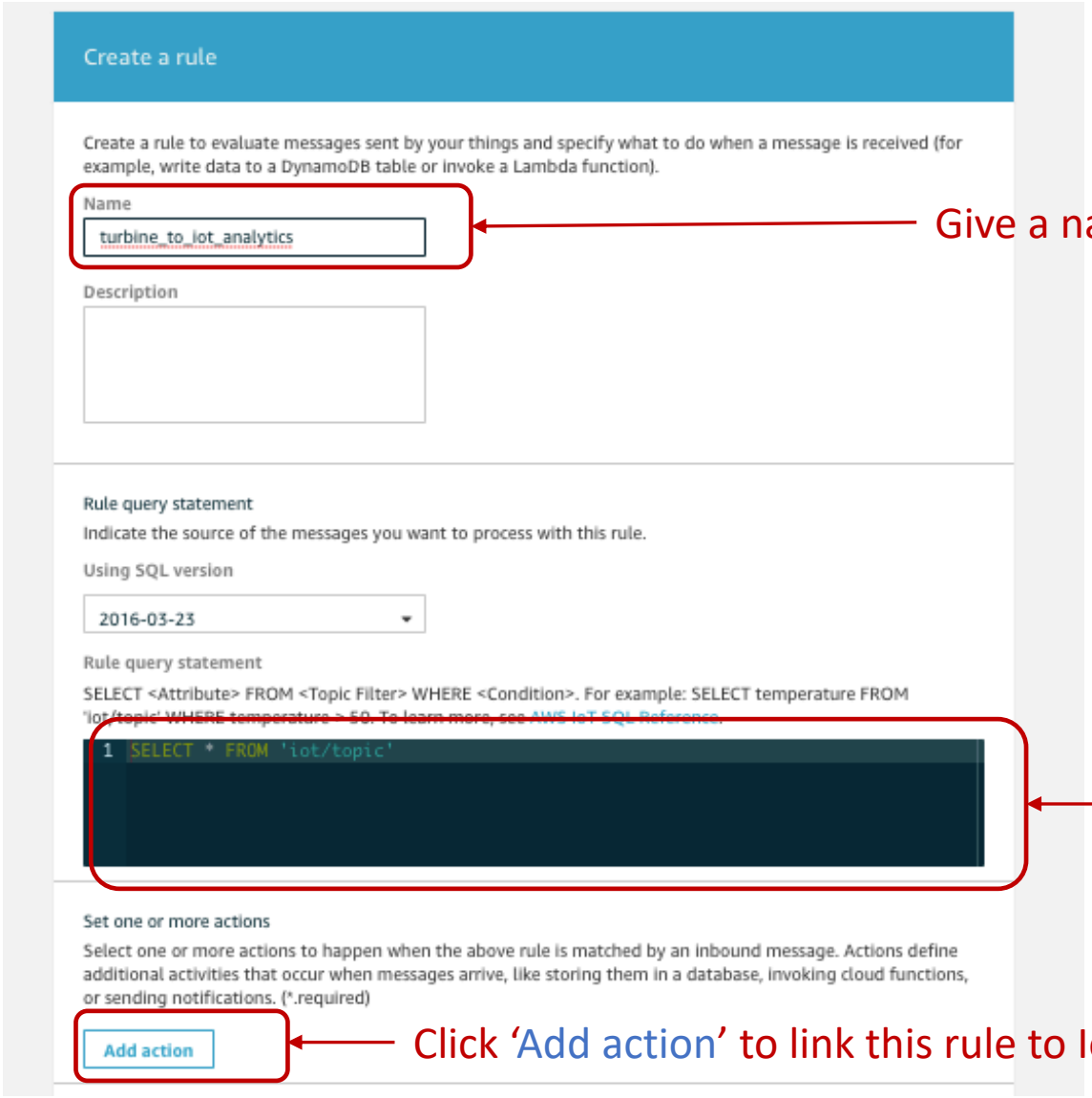
Lab 1 –IoT ‘thing’ & Device Simulation	15 mins
Lab 2 –Kinesis Firehose and Analytics Pipeline	30 mins
Lab 3 –IoT Analytics	30 mins
Lab 4 –Notification (Optional)	30 mins

Before setting up IoT Analytics, ensure that device simulator is running on Cloud9



Setup IoT rule to push data in IoT Analytics

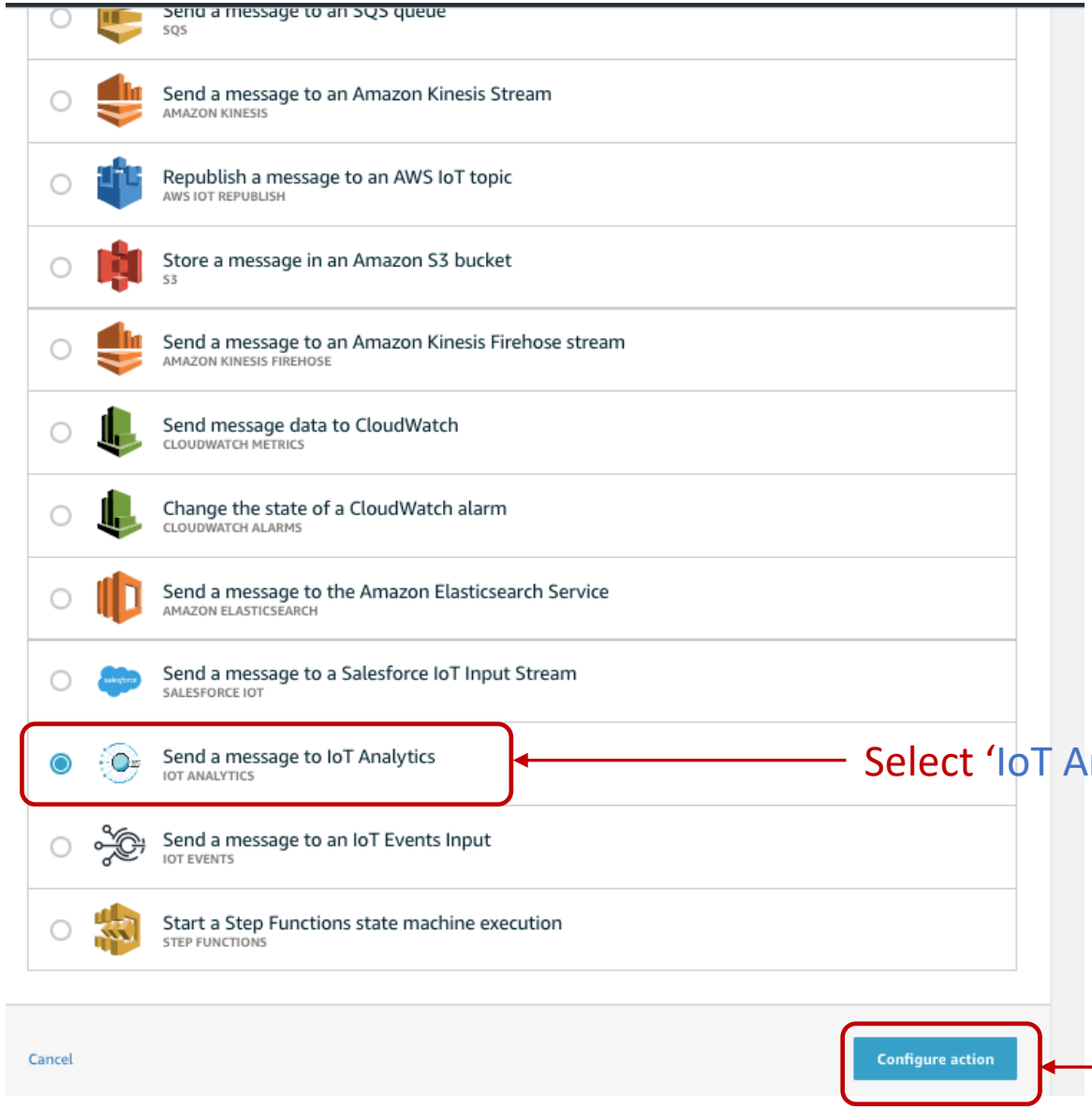

AWS IoT
Monitor
Onboard
Manage
Greengrass
Secure
Defend
Act
Test
Software
Settings
Learn
Rules
Create
Search rules
Card
turbine_to_firehose
ENABLED
Select ‘Act’ on IoT Console to setup a new Rule
Click ‘Create’



Give a name to this rule.

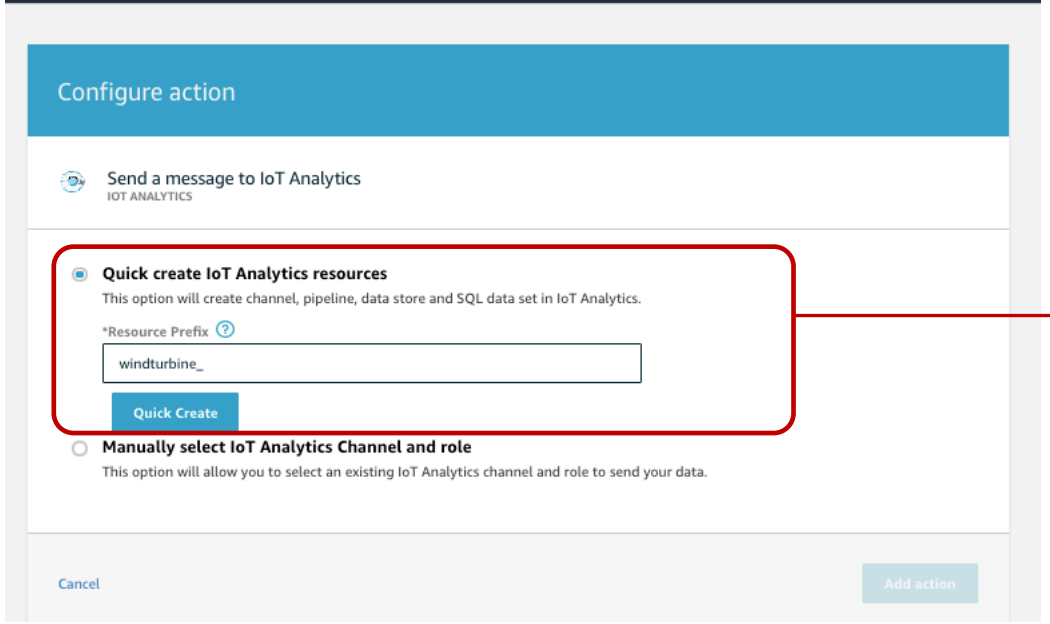
Change the SQL code here to SELECT * FROM 'topic_1'

Click ‘Add action’ to link this rule to IoT Analytics

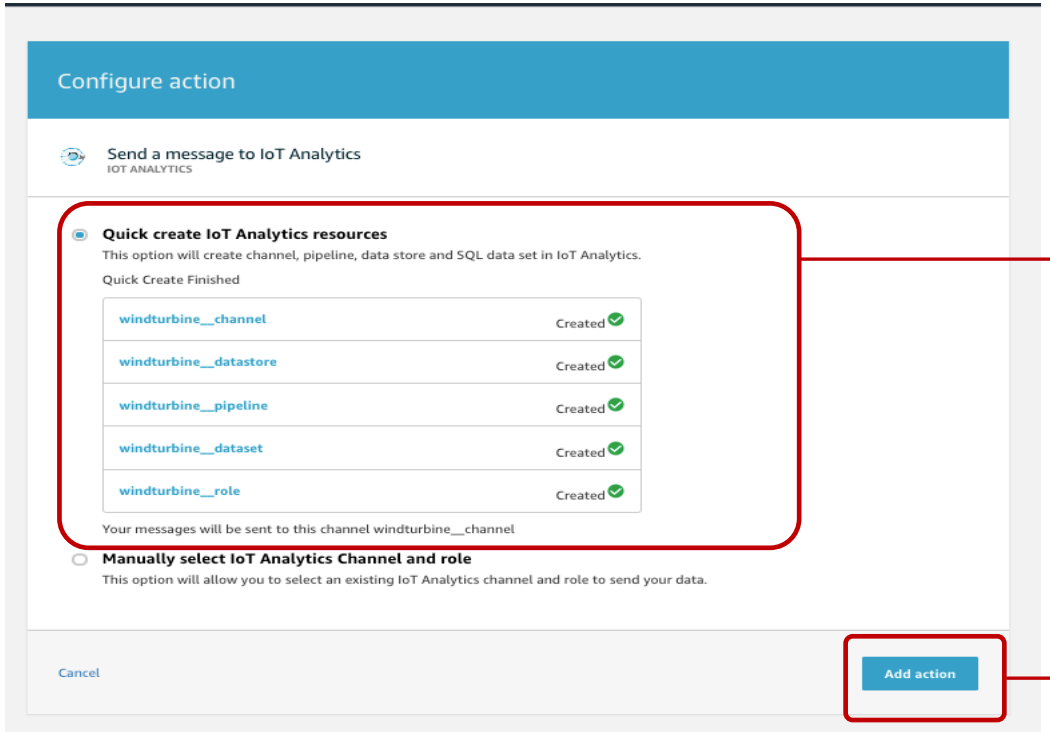


Select 'IoT Analytics' from list of services

Click 'Configure Action'

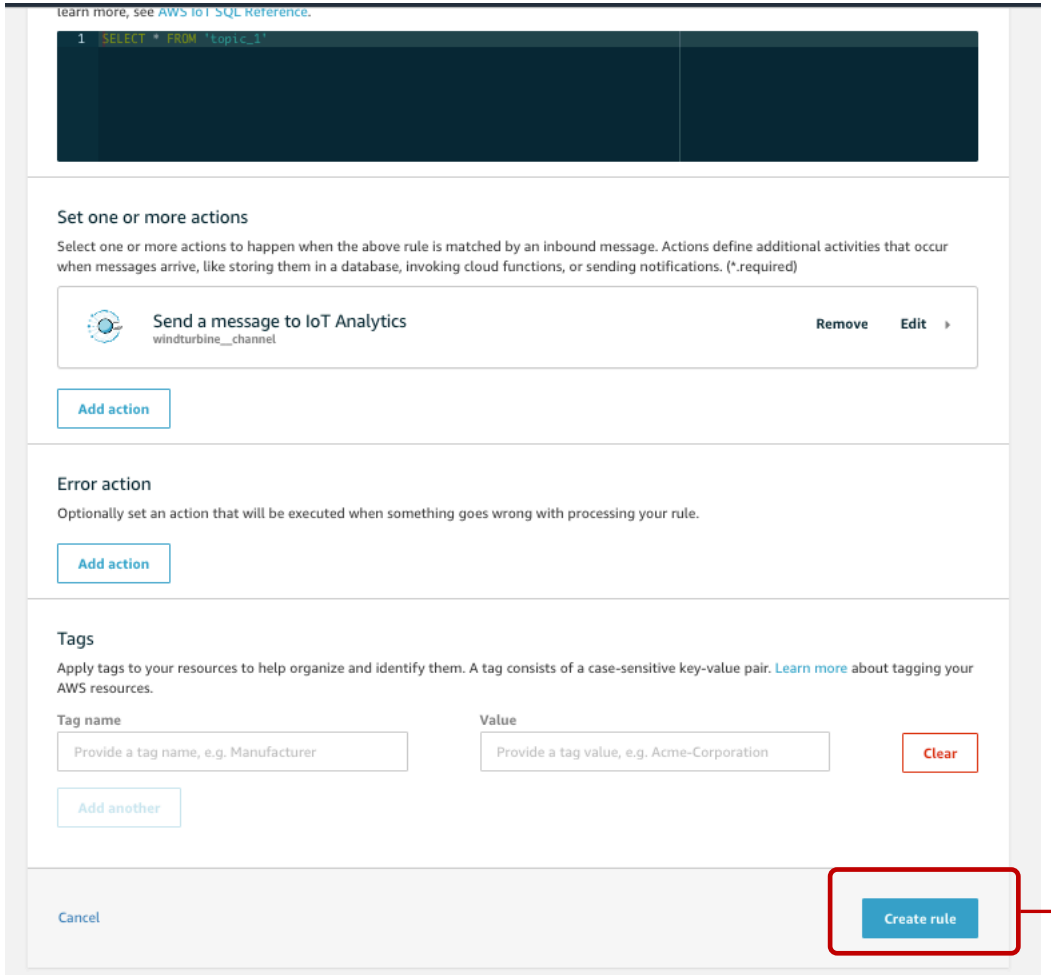


The IoT Analytics resources do not exist at this point.
We will select the ‘Quick Create IoT Analytics option.
Enter 'windturbine' as the resource prefix, then click ‘Quick Create’



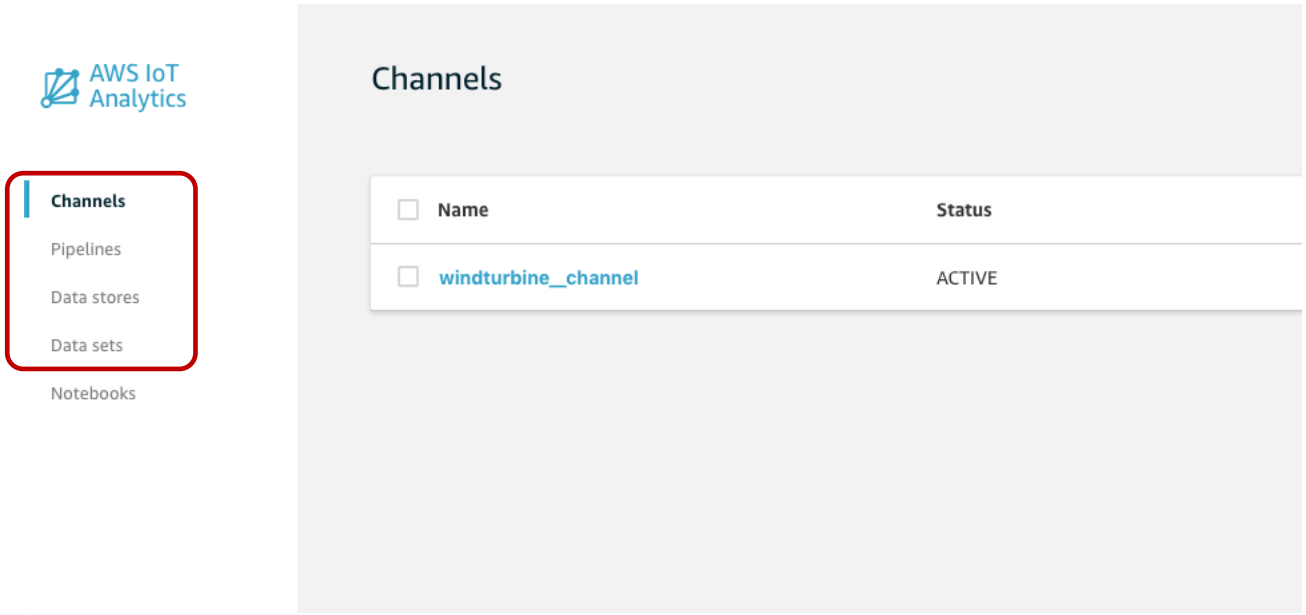
After clicking ‘Quick Create’ you will see the IoT Analytics resources created.

Click ‘Add action’



Click ‘Create Rule’ to complete this process.

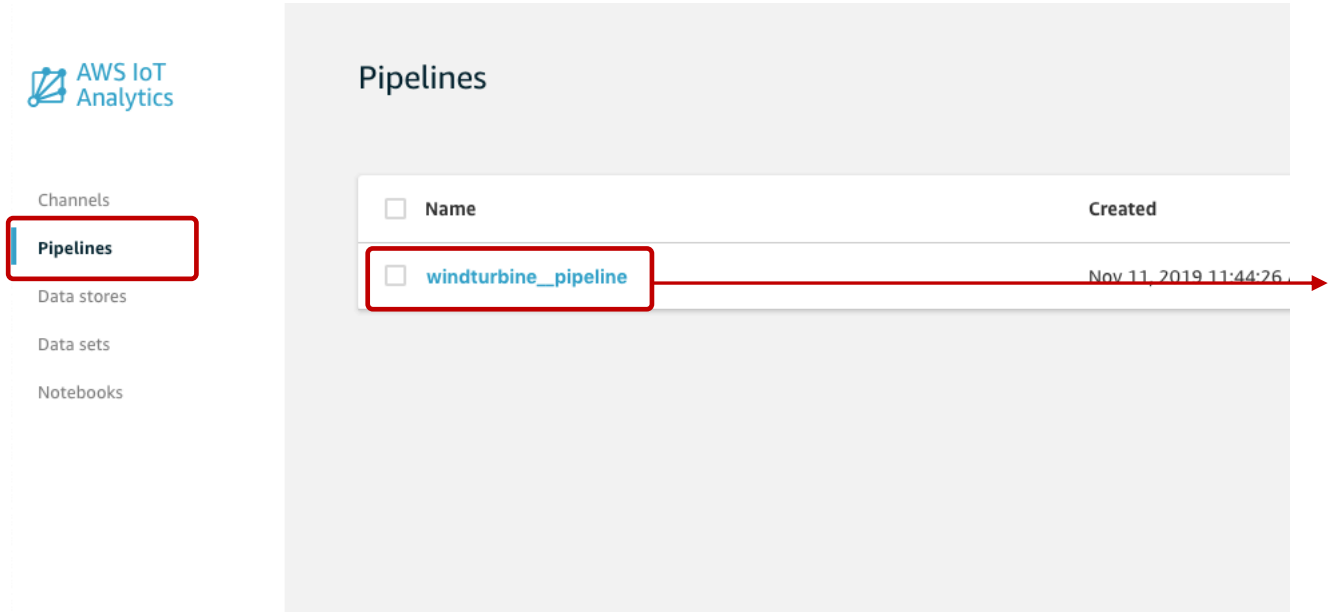
Transforming and querying data in IoT Analytics



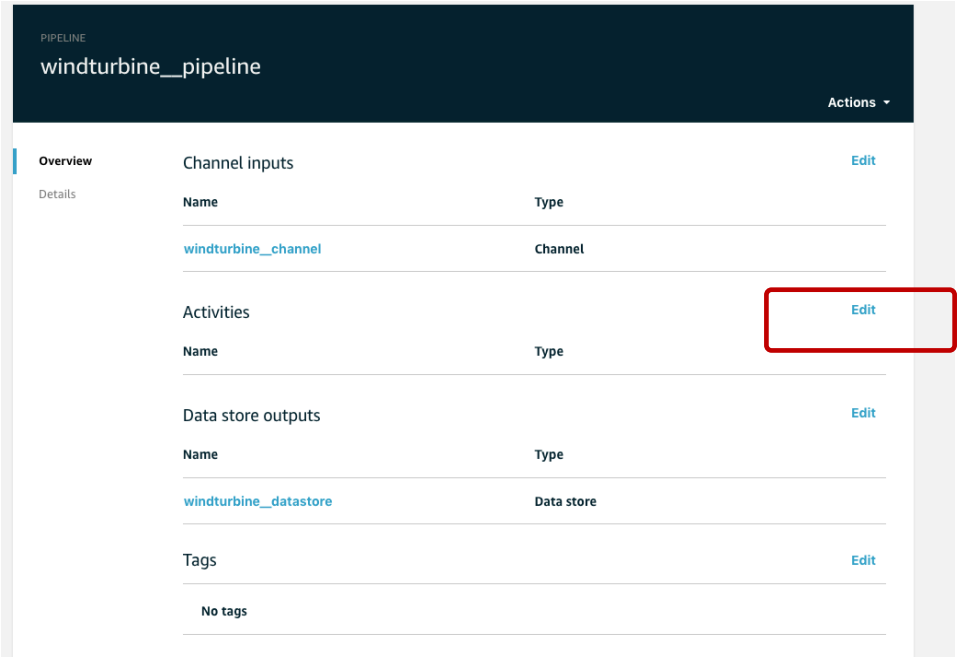
Launch AWS IoT Analytics from AWS Console.

You will see the Channel, Pipeline and Data Store and Data set created from previous step

In this lab we will use pipeline to create a simple transformation – converting temperature from Fahrenheit to Celsius

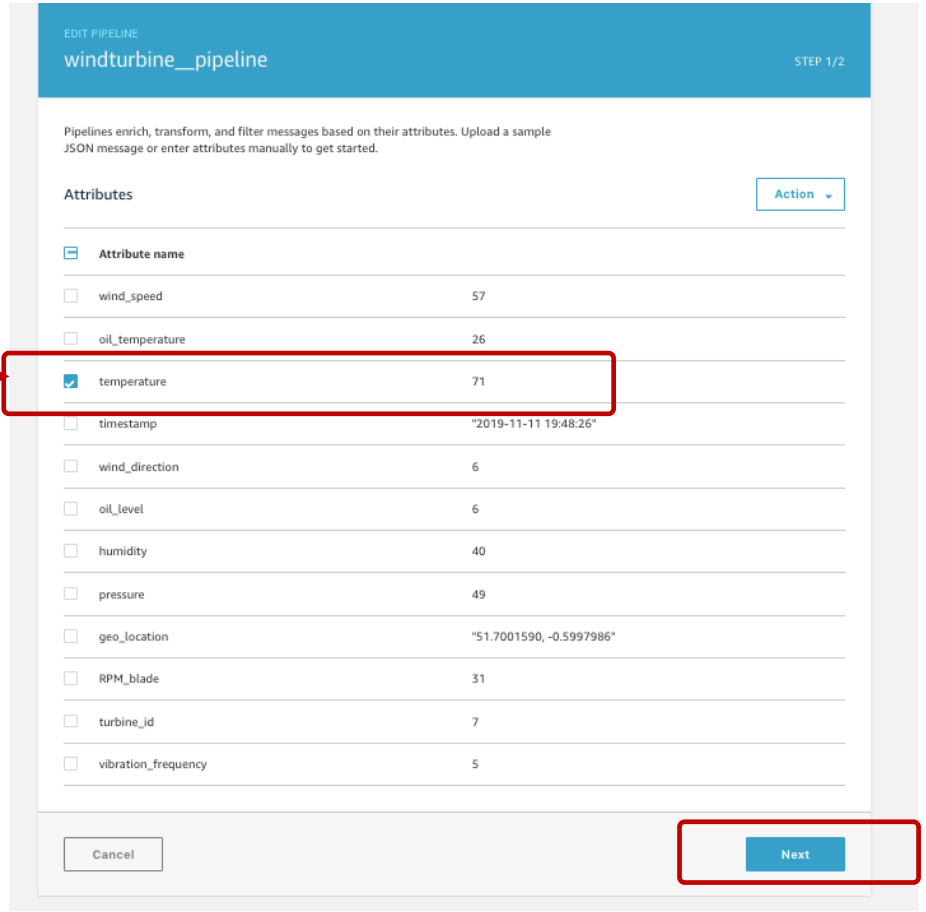


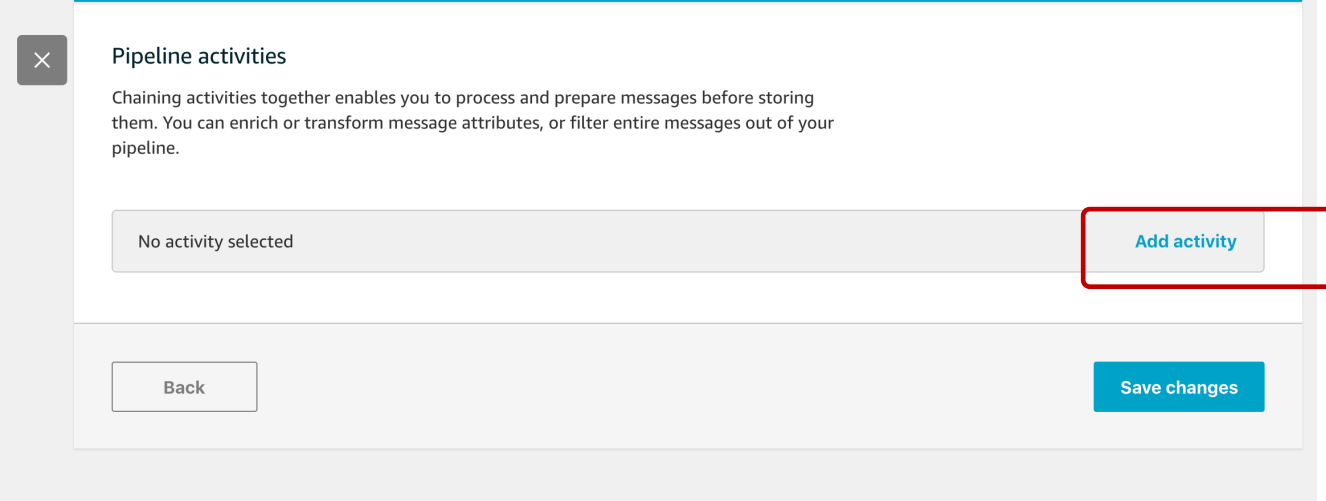
We will start with first selecting the pipeline



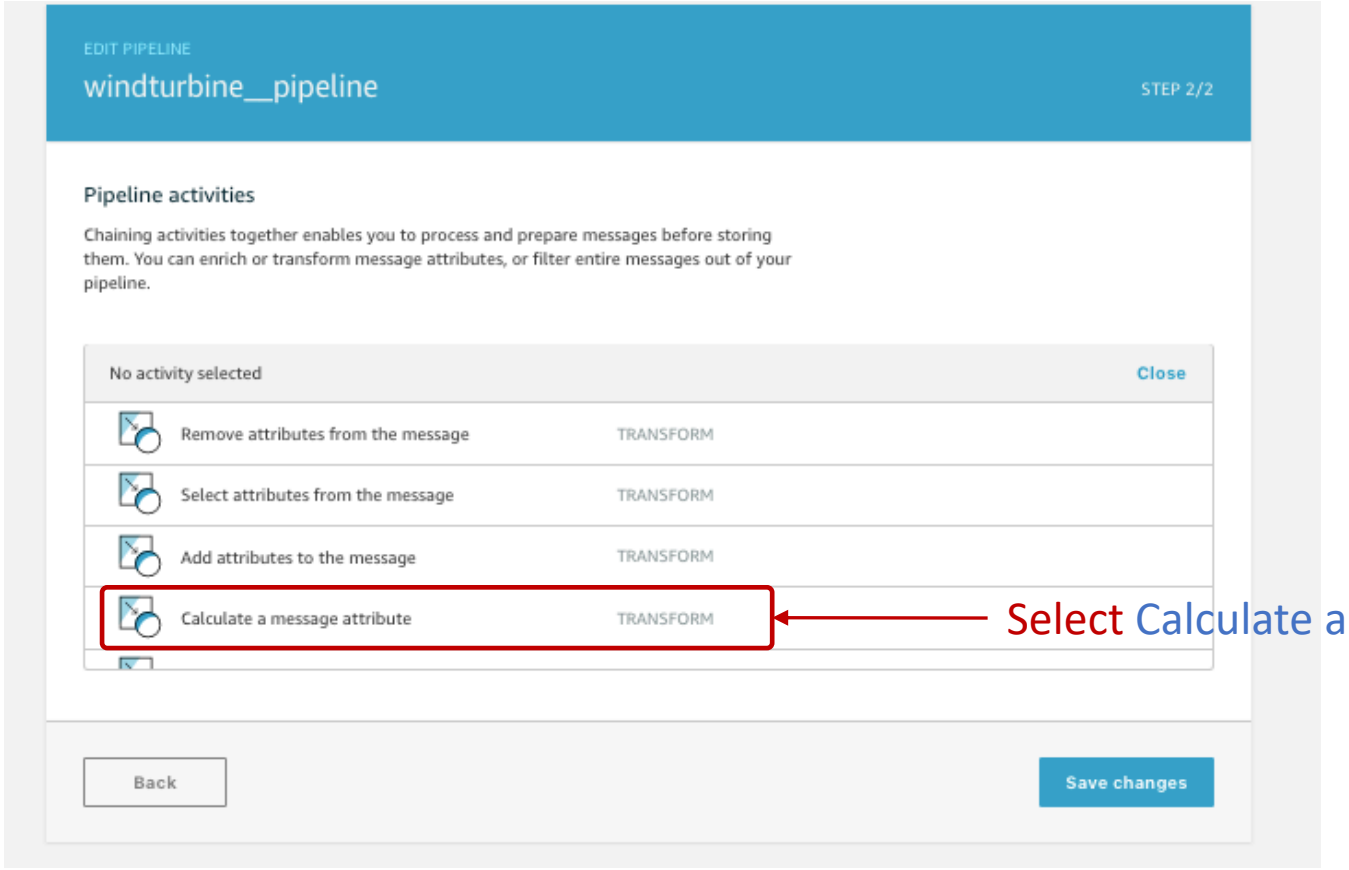
Click on ‘Edit’ Activity, to add a transformation step.

Select ‘temperature’ attribute

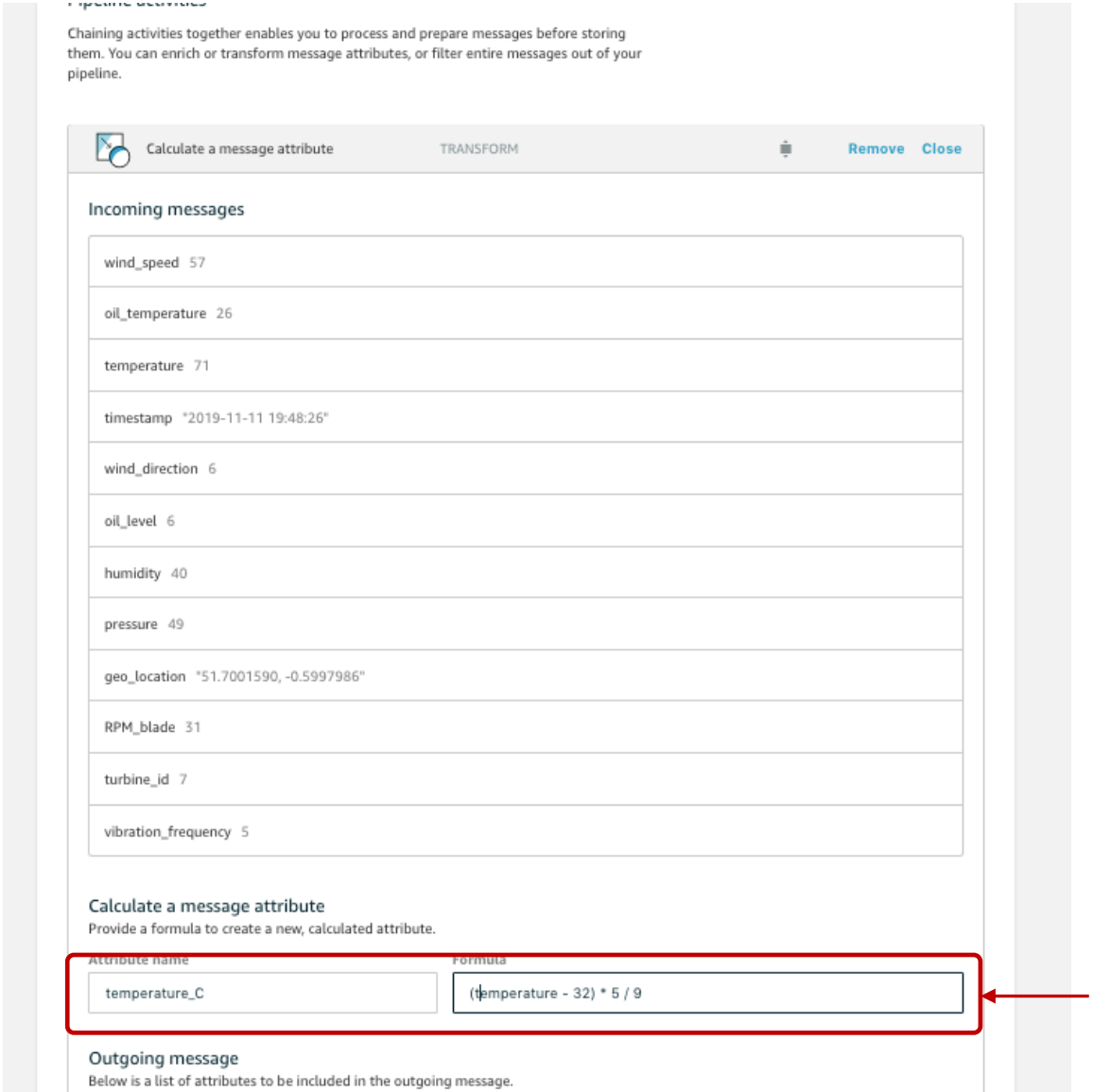




Click Add activity

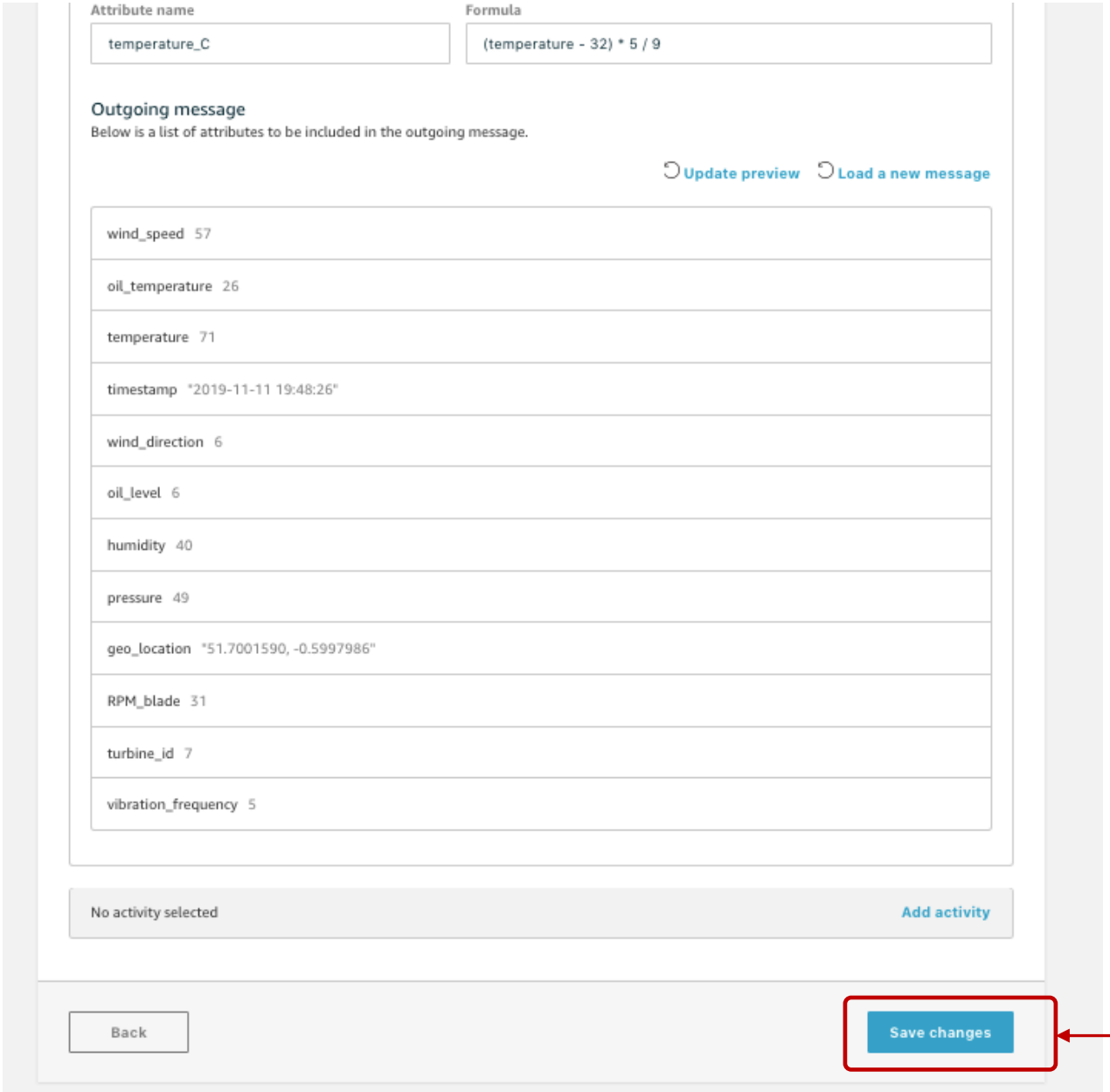


Select Calculate a message attribute

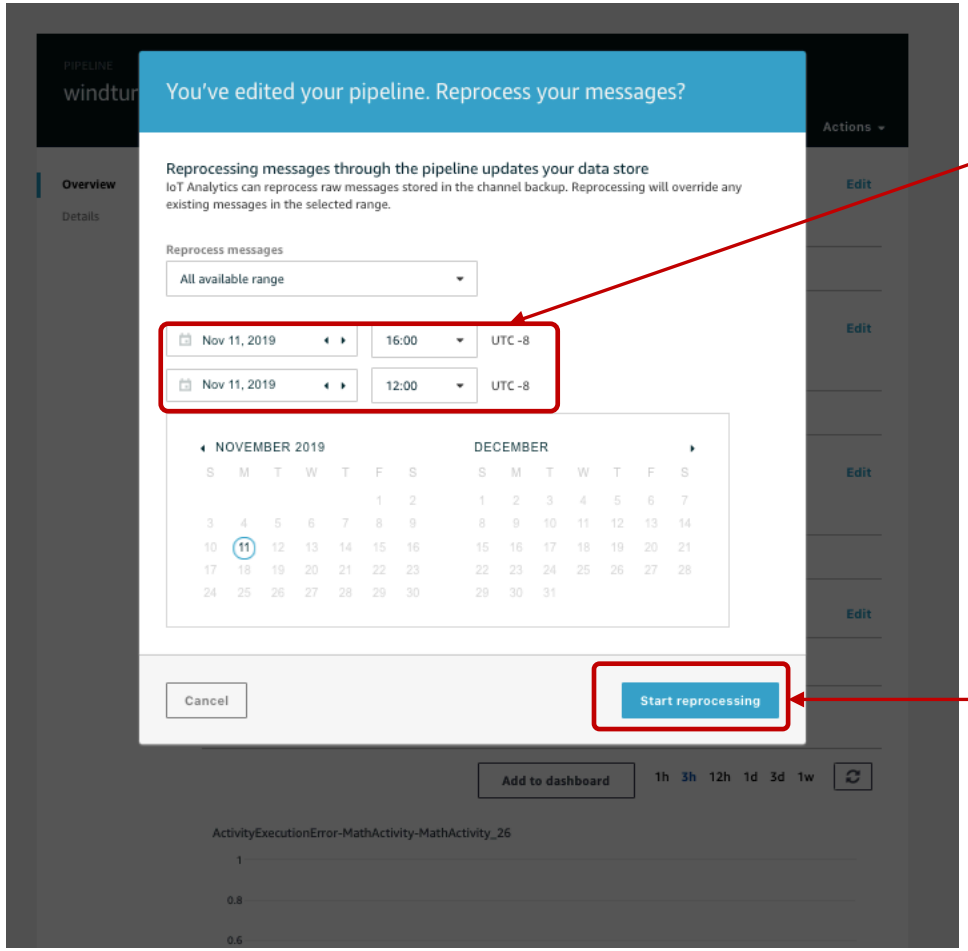


In this step, we convert the temperature to Celsius, and add an additional attribute 'temperature_c'

Select 'Calculate a message attribute'.
The formula here is *(temperature – 32) * 5 / 9*



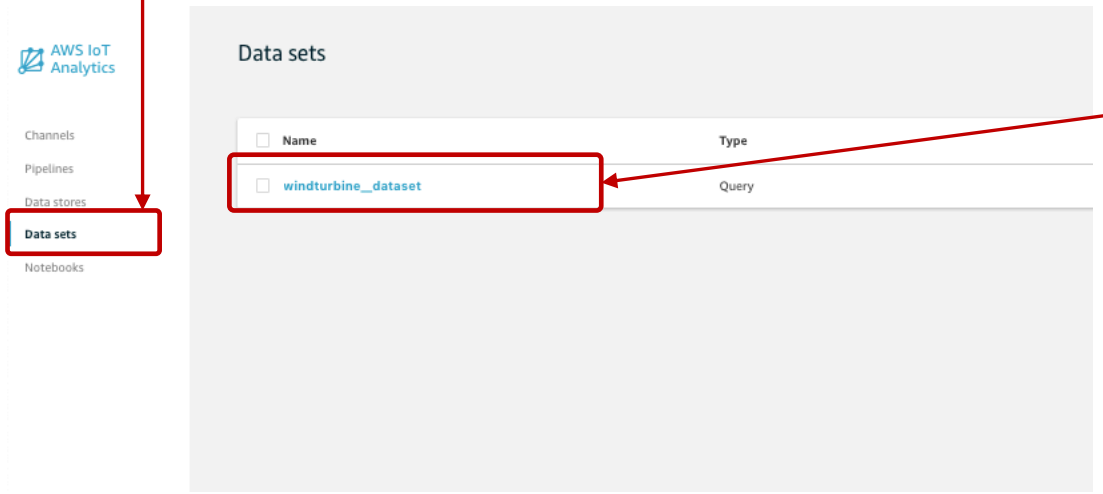
Save changes to apply activity to this pipeline



You will see an option to reprocess the messages. You can choose a timeframe to go back and apply this transformation to previously collected data.

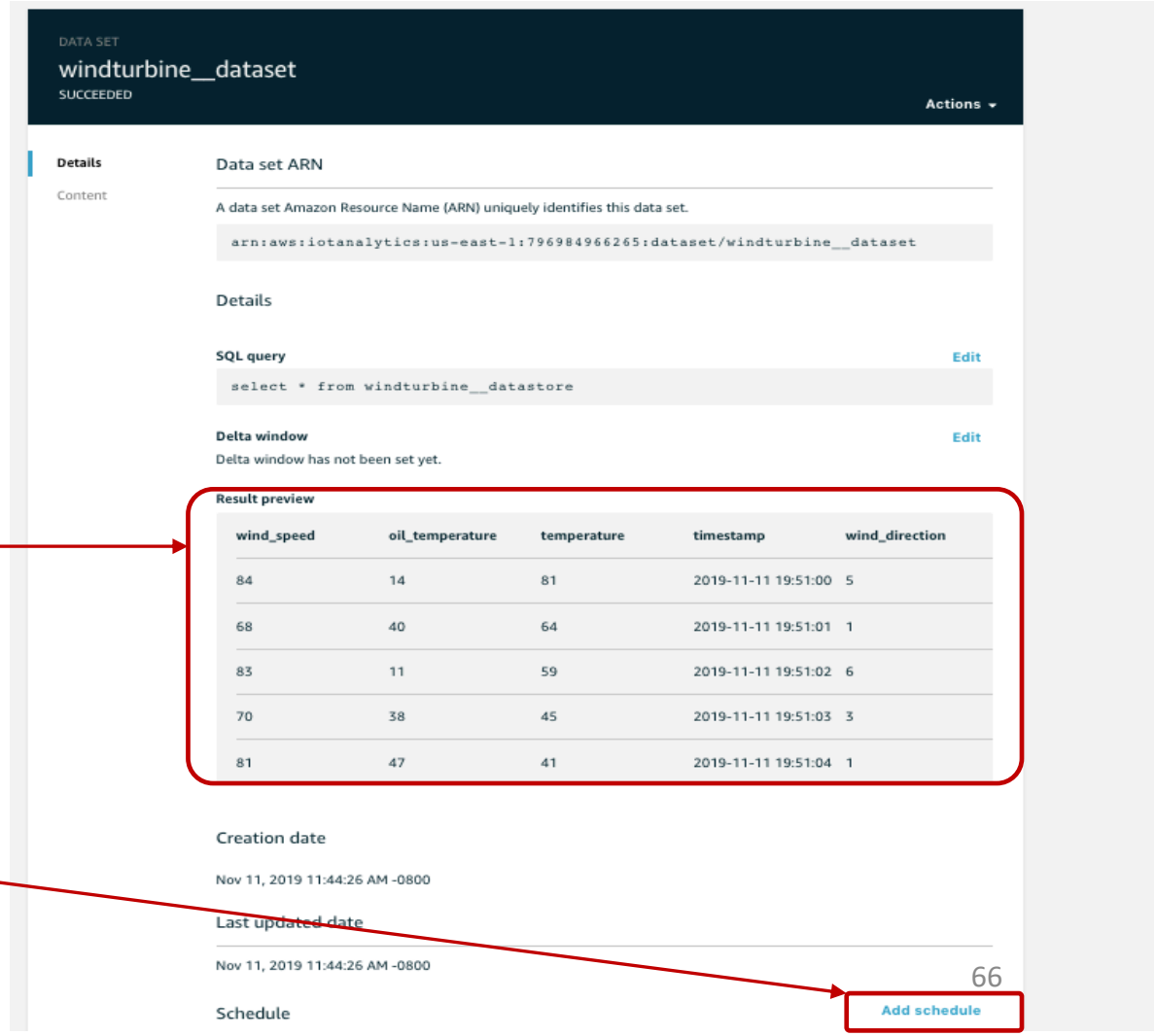
Click 'Start processing'

On the IoT Analytics console, select ‘Data sets’ from the left panel



Click on the Data set ‘Windturbine_dataset’, you will open the screen below.

Here you can preview incoming transformed data



Click ‘Add schedule’ to configure frequency to run this query. You can set frequency between 1 minute and a month to refresh the datasets.

For this workshop, we can set it for 1 minute, so we can visualize this data in later steps.

Visualize IoT data in AWS QuickSight

Select Quicksight from AWS Console
If QuickSight is not setup, you will see the following screen.

Your AWS Account is not signed up for QuickSight. Would you like to sign up now?

AWS Account **796984966265**

Sign up for QuickSight

Click 'Sign up for QuickSight'

To access QuickSight with a different account, log in again.

Create your QuickSight account

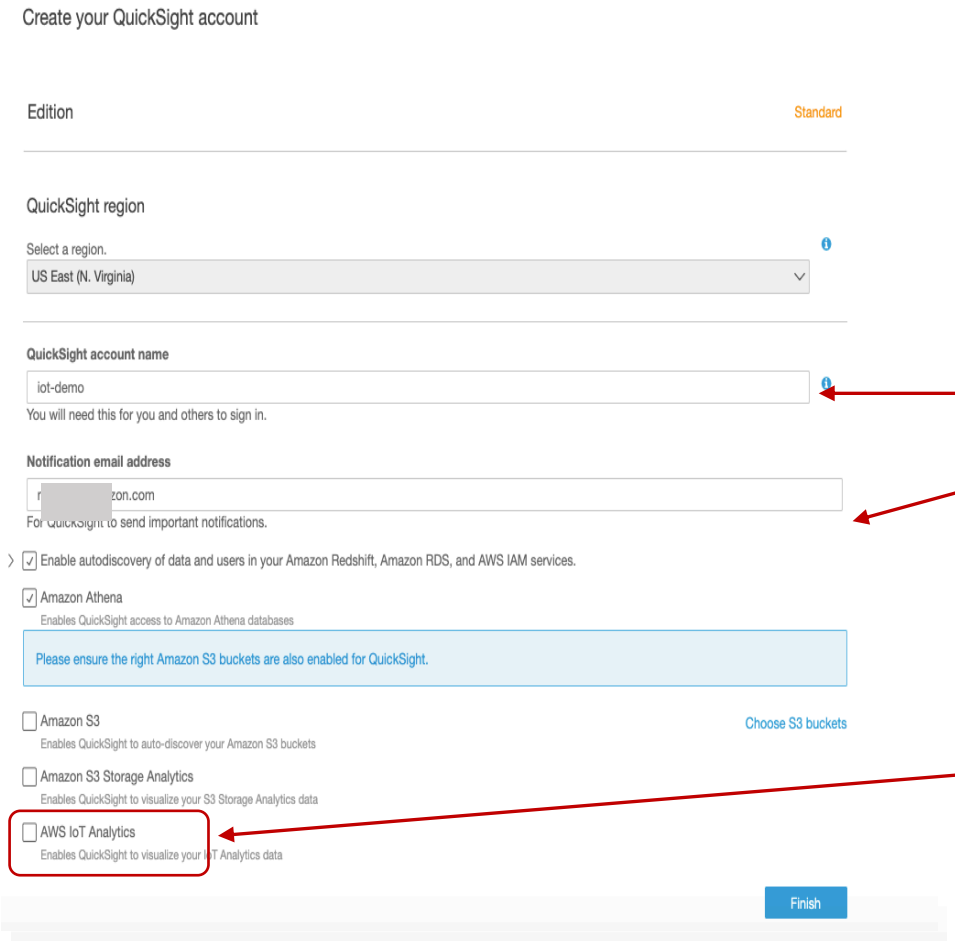
Edition	Standard	Enterprise
First author with 1GB SPICE	FREE	FREE
Team trial for 60 days (4 authors)*	FREE	FREE
Additional author per month (yearly)**	$9	$18
Additional author per month (monthly)**	$12	$24
Additional readers (Pay-per-Session)	N/A	$0.30/session (max $5/reader/month) ****
Additional SPICE per month	$0.25 per GB	$0.38 per GB
Single Sign On with SAML or OpenID Connect	✓	✓
Connect to spreadsheets, databases & business apps	✓	✓
Access data in Private VPCs		✓
Row-level security for dashboards		✓
Hourly refresh of SPICE data		✓
Secure data encryption at rest		✓
Connect to your Active Directory		✓
Use Active Directory Groups ***		✓
Send email reports		✓

Select Standard

* Trial authors are auto-converted to month-to-month subscription upon trial expiry
** Each additional author includes 10GB of SPICE capacity
*** Active Directory groups are available in accounts connected to Active Directory
**** Sessions of 30-minute duration. Total charges for each reader are capped at $5 per month.Conditions apply

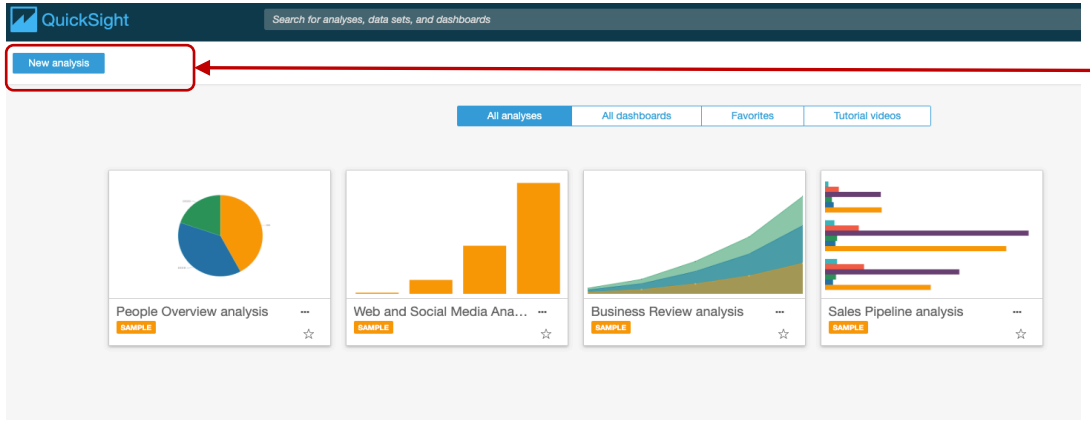
Continue

Click Continue

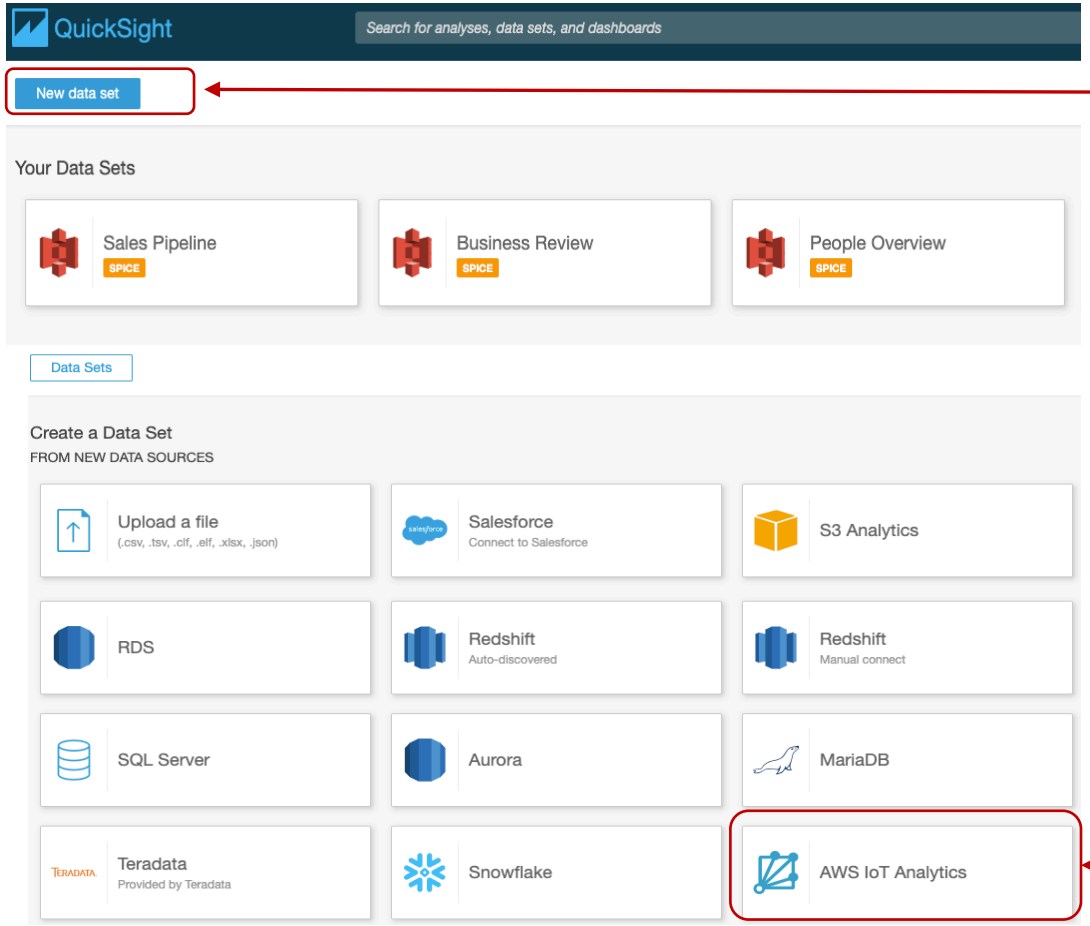


Provide a unique name and email-id

Select AWS IoT Analytics, to give permission to QuickSight, to access Iot Analytics datasets

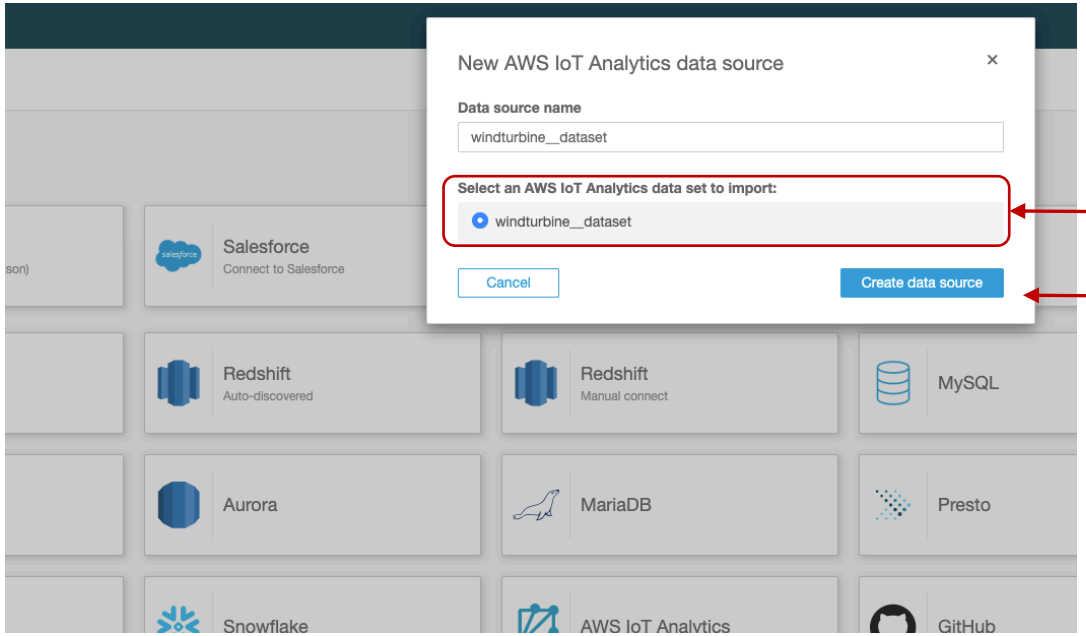


Click ‘New Analysis’



Click ‘New Data Set’

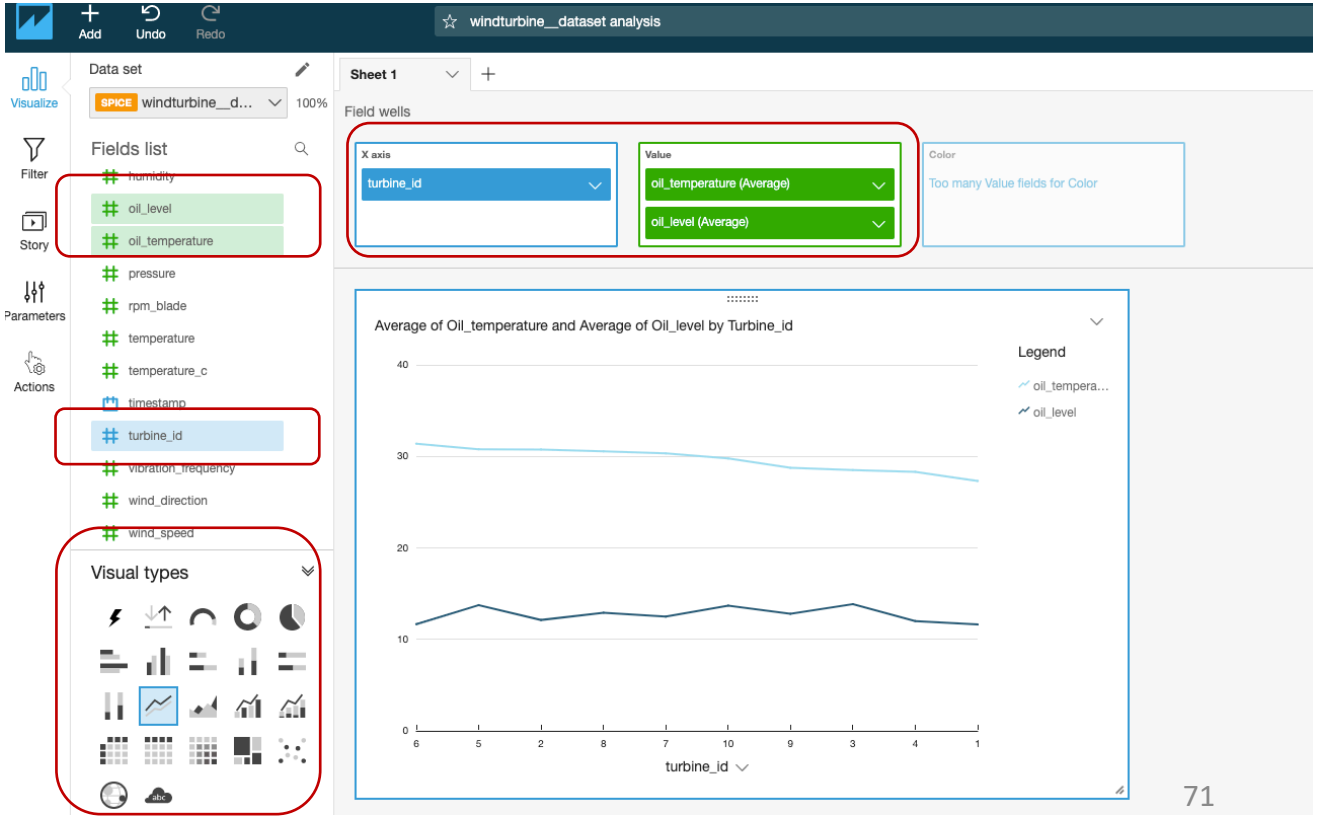
Select ‘AWS IoT Analytics’



Select the Dataset.

Click ‘Create data source’

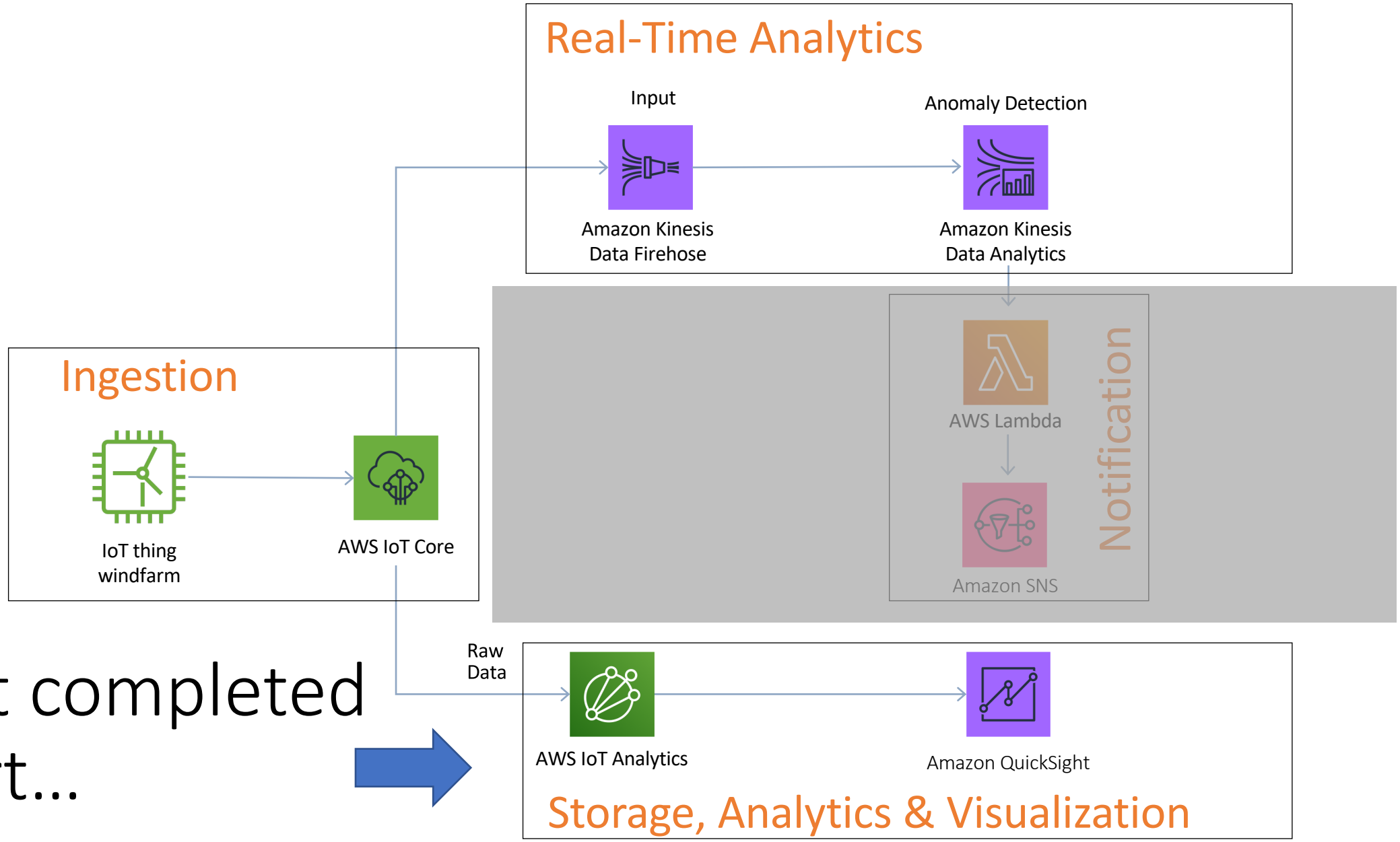
Next, start visualizing the data



In this example, under Visual Types on the bottom left, select Line Graph.

Then choose turbine_id to be the X-axis and temperature and oil_temperature to be the Y-axis.

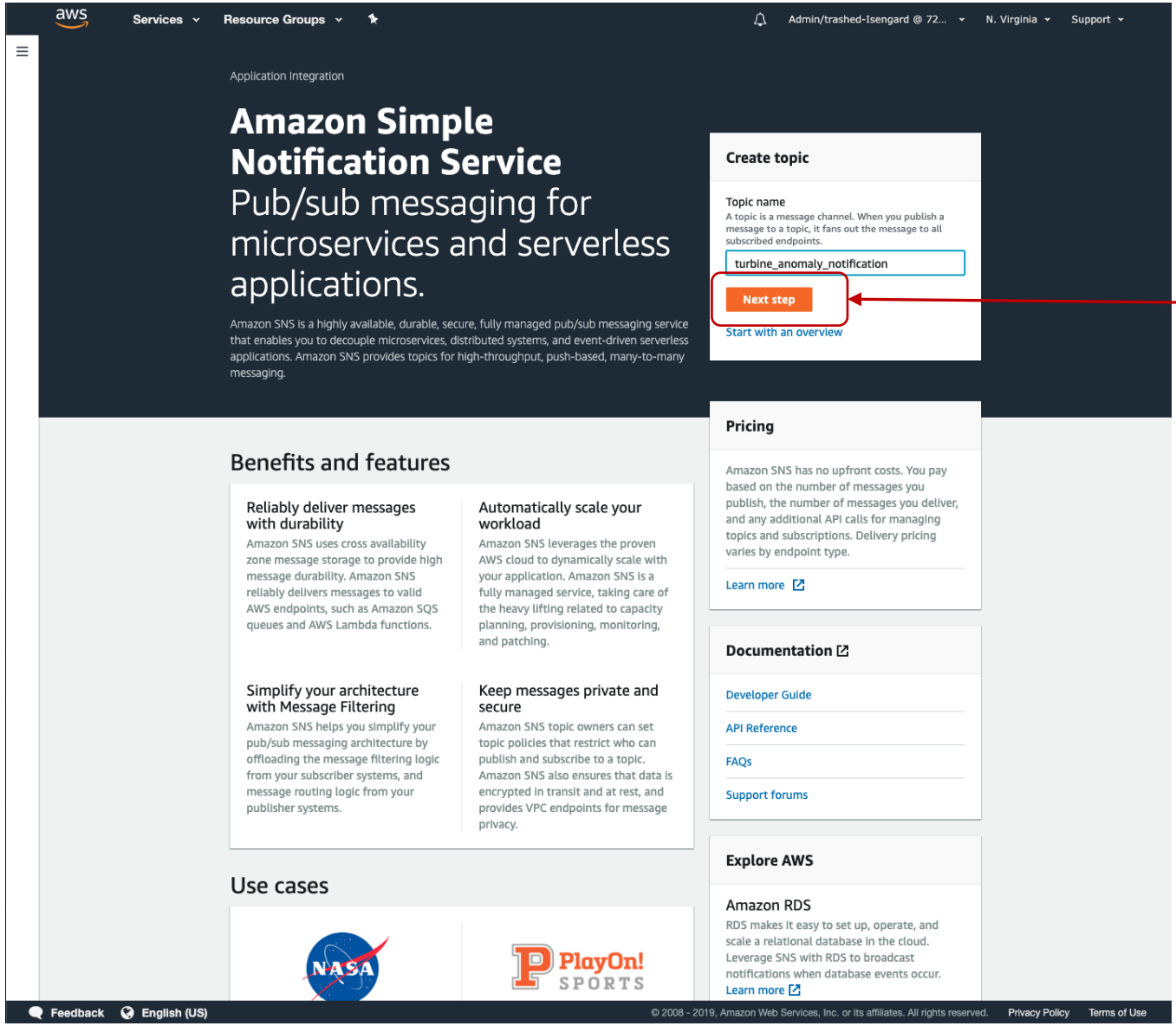
In the drop down of temperature, choose the ‘Aggregate’ option and select ‘Average’. Do the same with oil_temperature



You just completed this part…

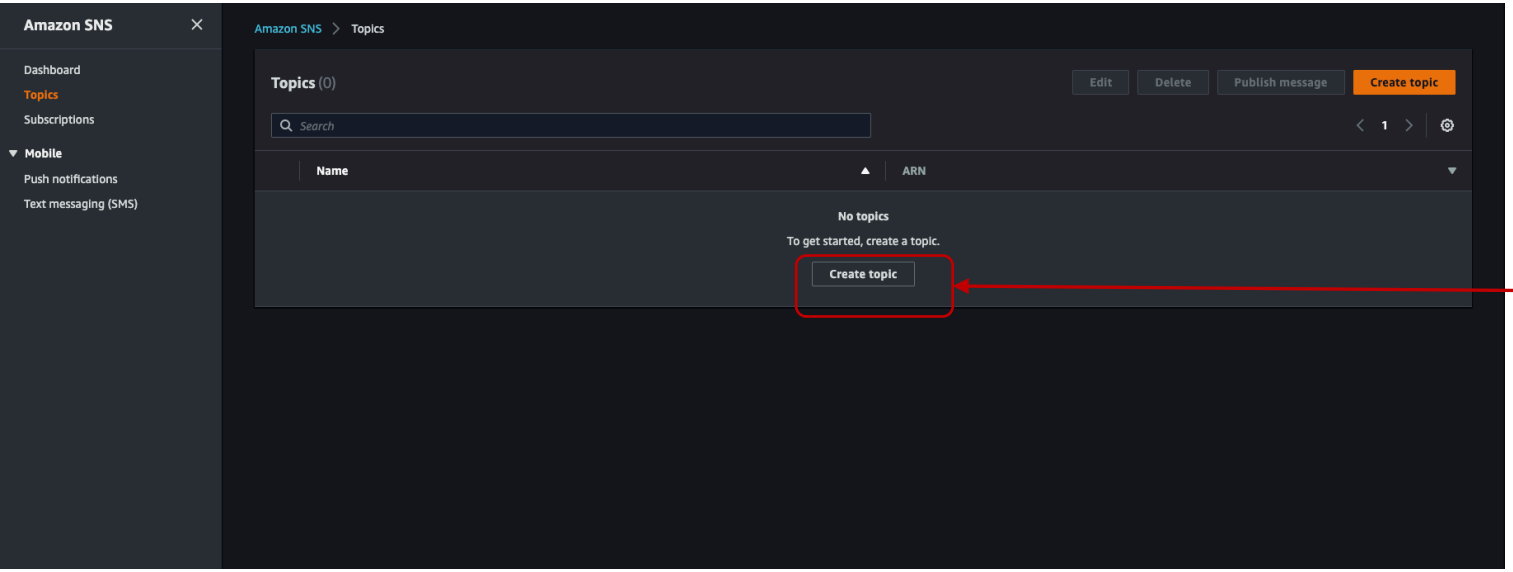
We will go over these labs:

Lab 1 –IoT ‘thing’ & Device Simulation	15 mins
Lab 2 –Kinesis Firehose and Analytics Pipeline	30 mins
Lab 3 –IoT Analytics	30 mins
Lab 4 –Notification (Optional)	30 mins

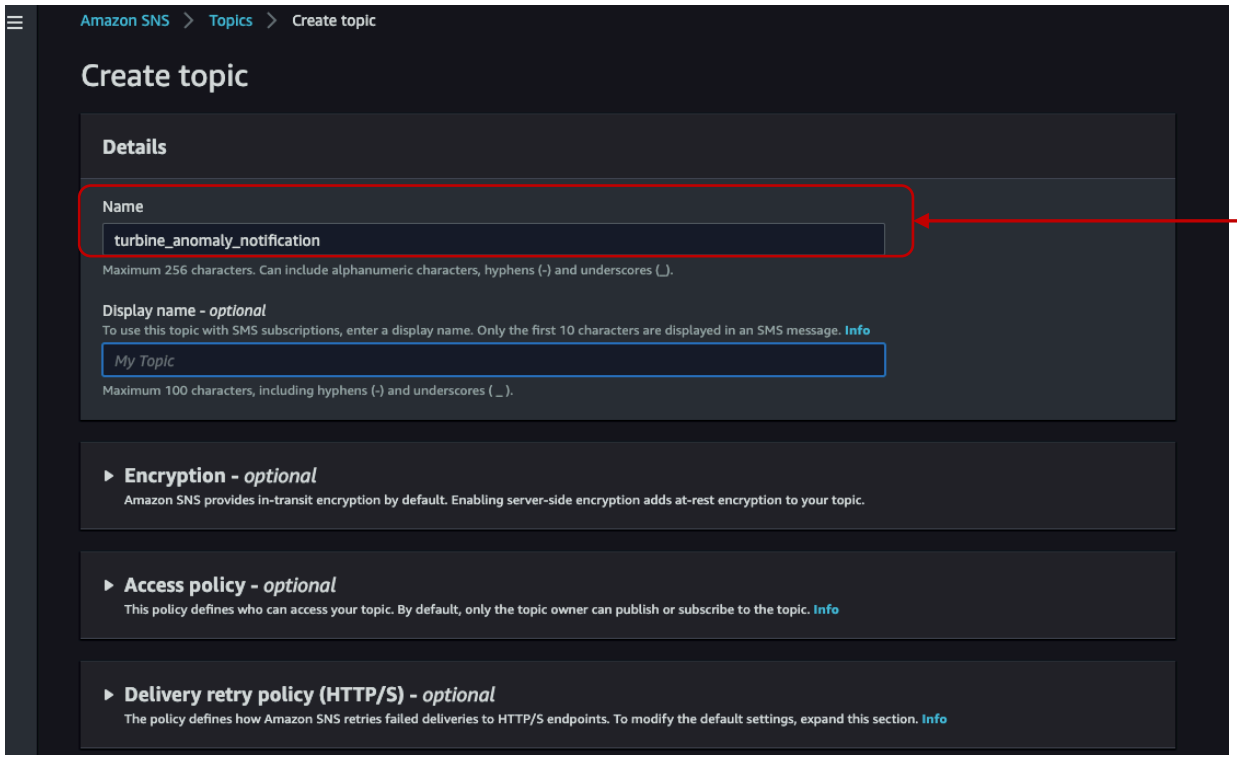


Select SNS from AWS Console.

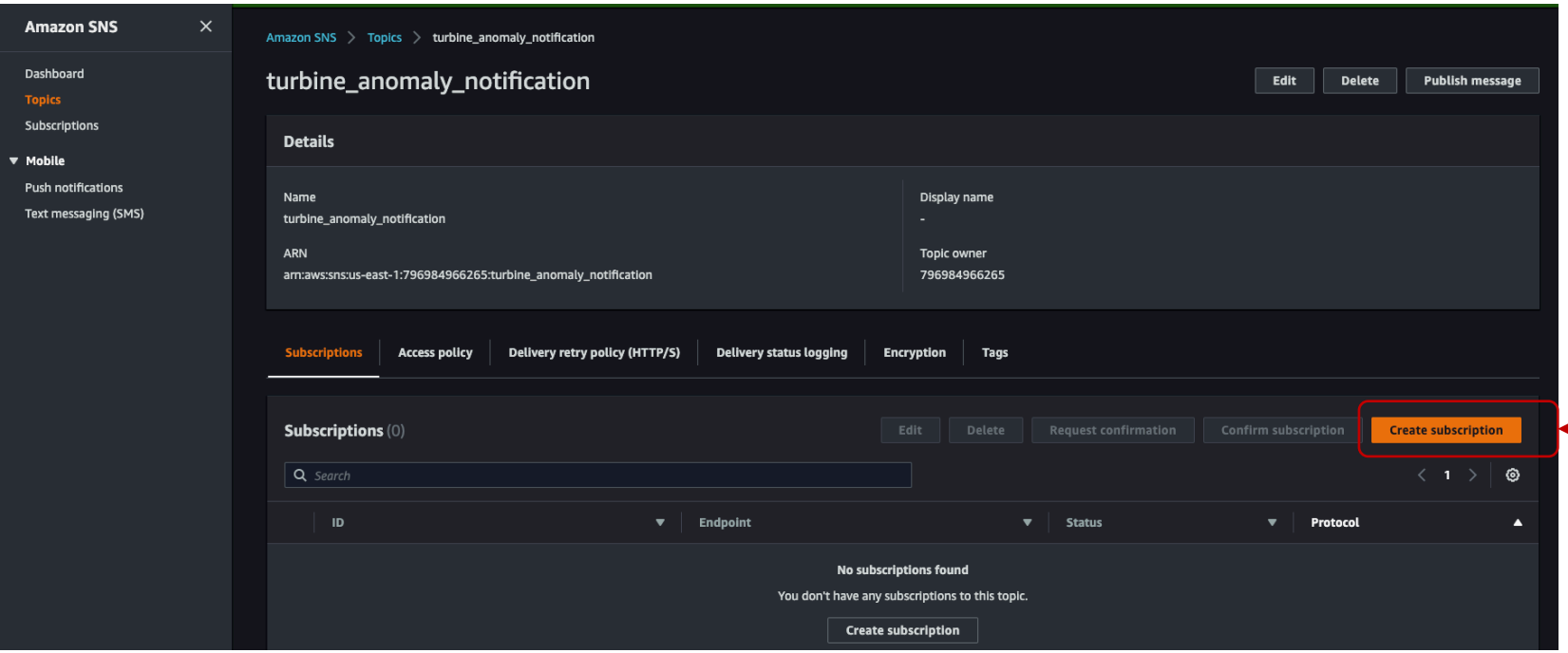
Click ‘Next step’



Click ‘Create topic’



Give a name to this topic. Leave all other settings default, then click ‘Create topic’



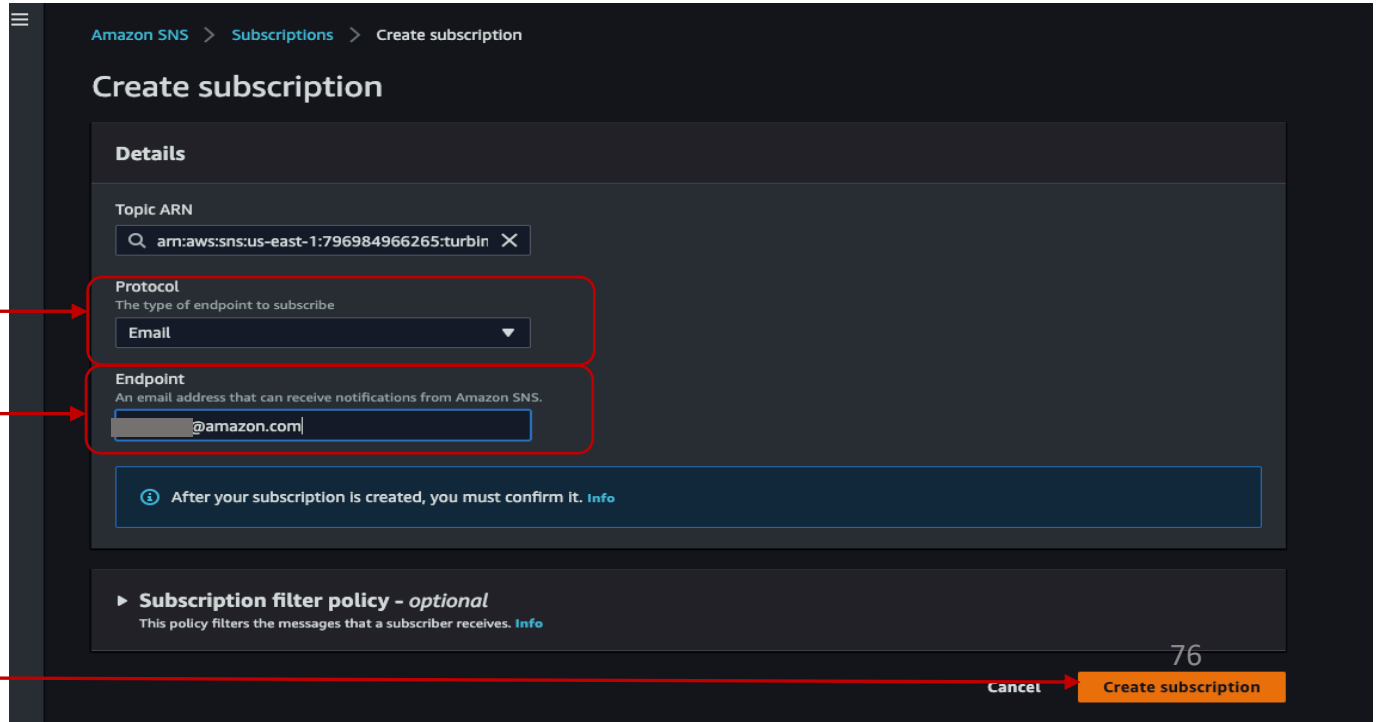
Note: After creating subscription, you will receive an email to confirm the subscription. SNS will not send notification, unless you confirm the subscription.

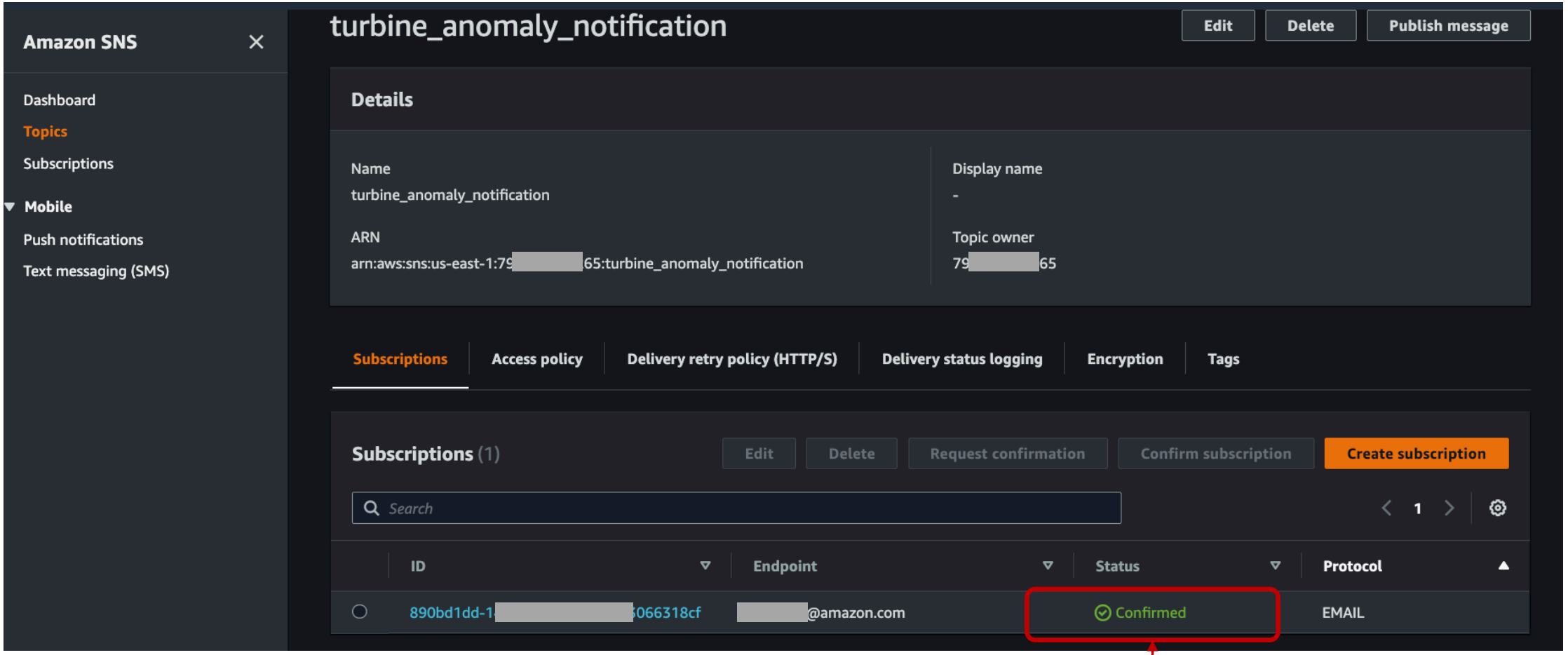
Click ‘Create subscription’

Select ‘Email’ Protocol

Provide your email-id as endpoint

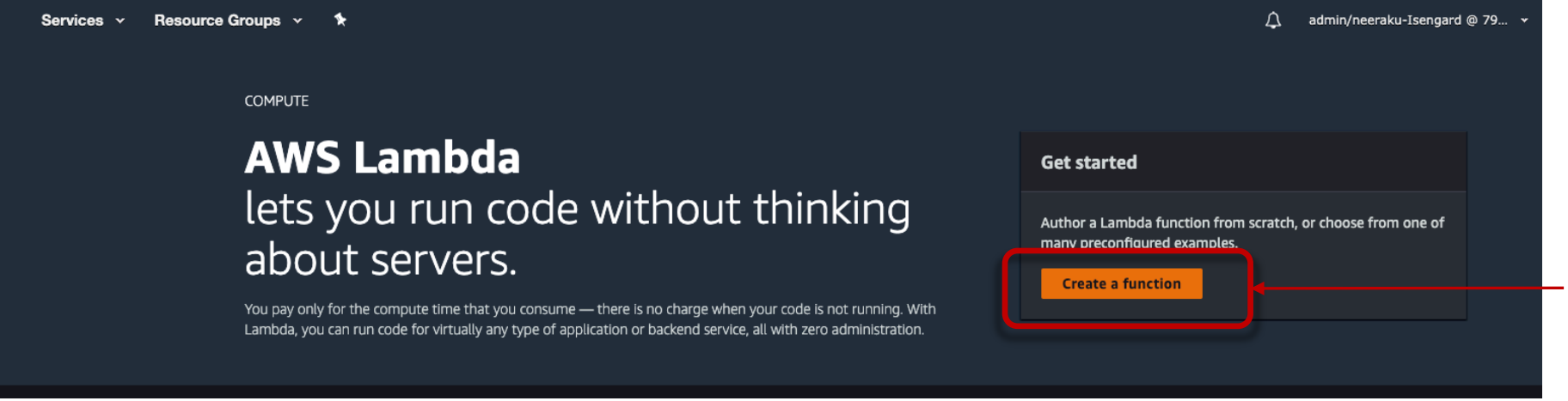
Click ‘create subscription’



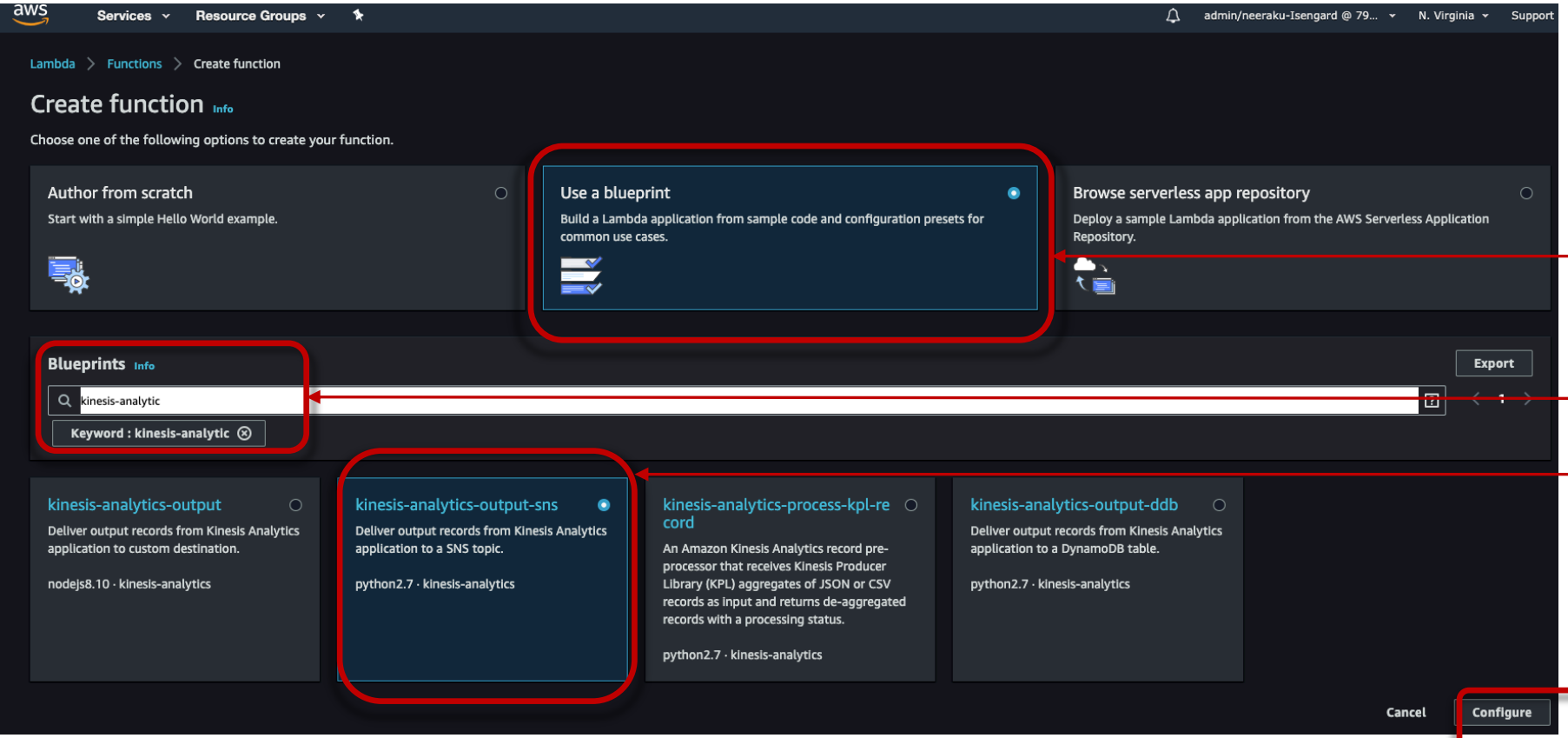


Before proceeding, ensure the status is shown as 'Confirmed'.

Create Lambda Function



Select Lambda from AWS Console. Click 'Create a function'

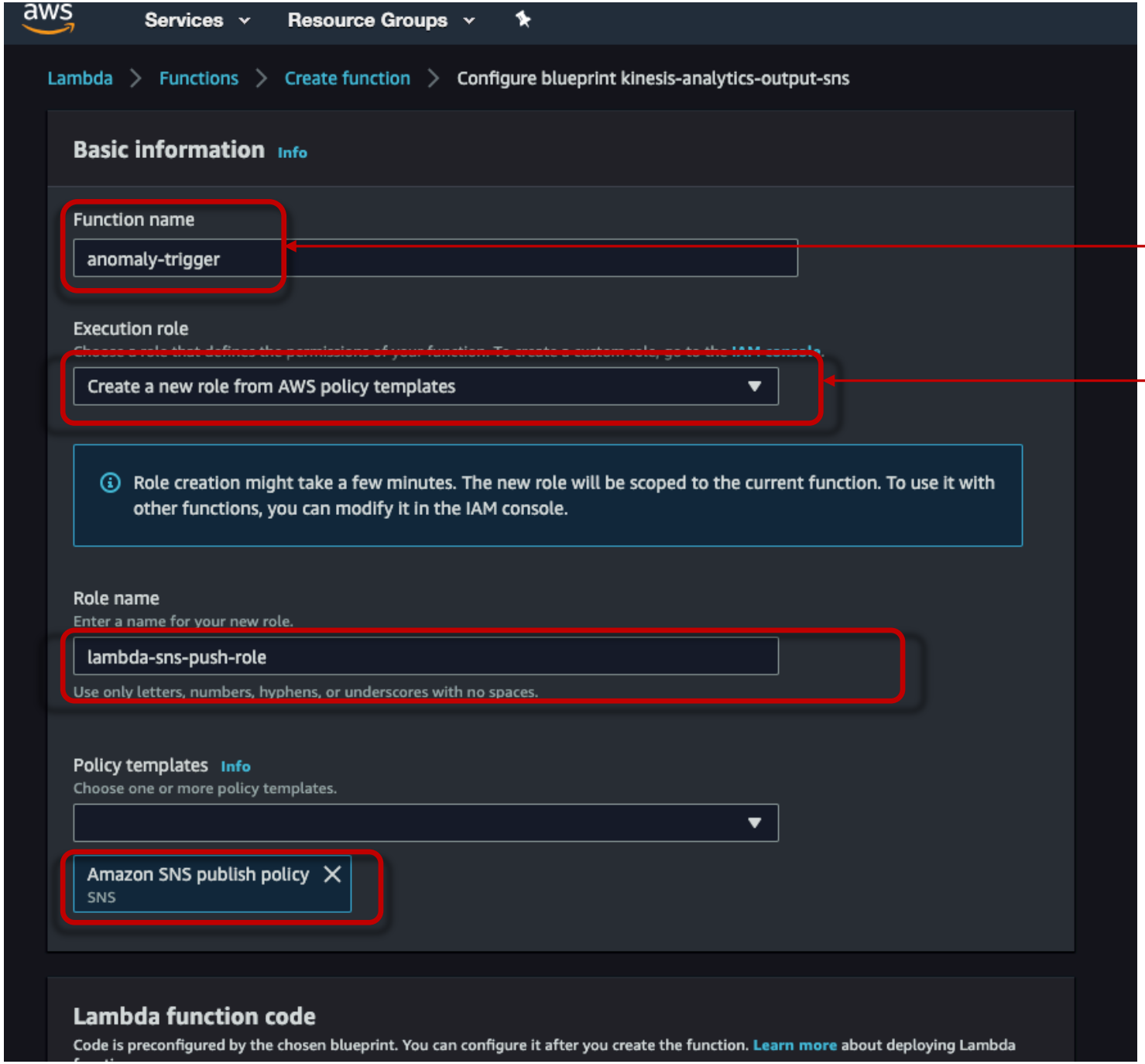


Choose Blueprint

Search for 'kinesis-analytics'

Select 'kinesis-analytics-output-sns' blueprint

Click 'configure'



Give function a name – 'anomaly-trigger'

Create a new IAM role, to enable Lambda to push messages to SNS

Lambda function code

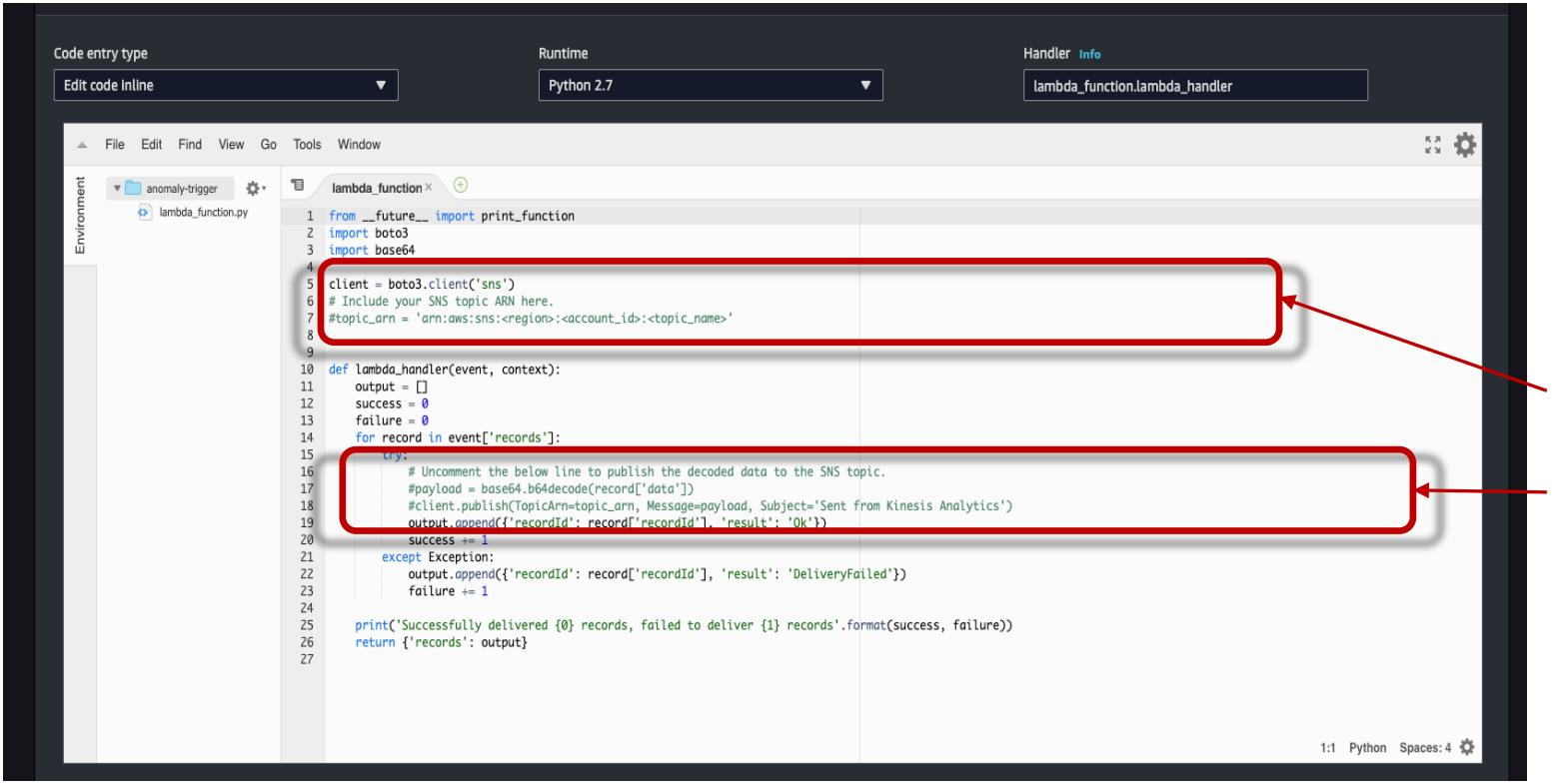
Code is preconfigured by the chosen blueprint. You can configure it after you create the function. Learn more about deploying Lambda functions.

Runtime
Python 2.7

```
import boto3
import base64

client = boto3.client('sns')
# Include your SNS topic ARN here.
#topic_arn = 'arn:aws:sns:<region>:<account_id>:<topic_name>'


def lambda_handler(event, context):
    output = []
    success = 0
    failure = 0
    for record in event['records']:
        try:
            # Uncomment the below line to publish the decoded data to the SNS topic.
            #payload = base64.b64decode(record['data'])
            #client.publish(TopicArn=topic_arn, Message=payload, Subject='Sent from
            output.append({'recordId': record['recordId'], 'result': 'Ok'})
            success += 1
        except Exception:
            output.append({'recordId': record['recordId'], 'result': 'DeliveryFailed
            failure += 1

    print('Successfully delivered {0} records, failed to deliver {1} records'.format
    return {'records': output}
```

Cancel | Create function

Click 'Create function'



On the Lambda console, scroll down to the function code.

The function code is already filled in since we selected an existing blueprint. We need to add a few things before making it functional

Add the SNS ARN that we created earlier, and uncomment this line.

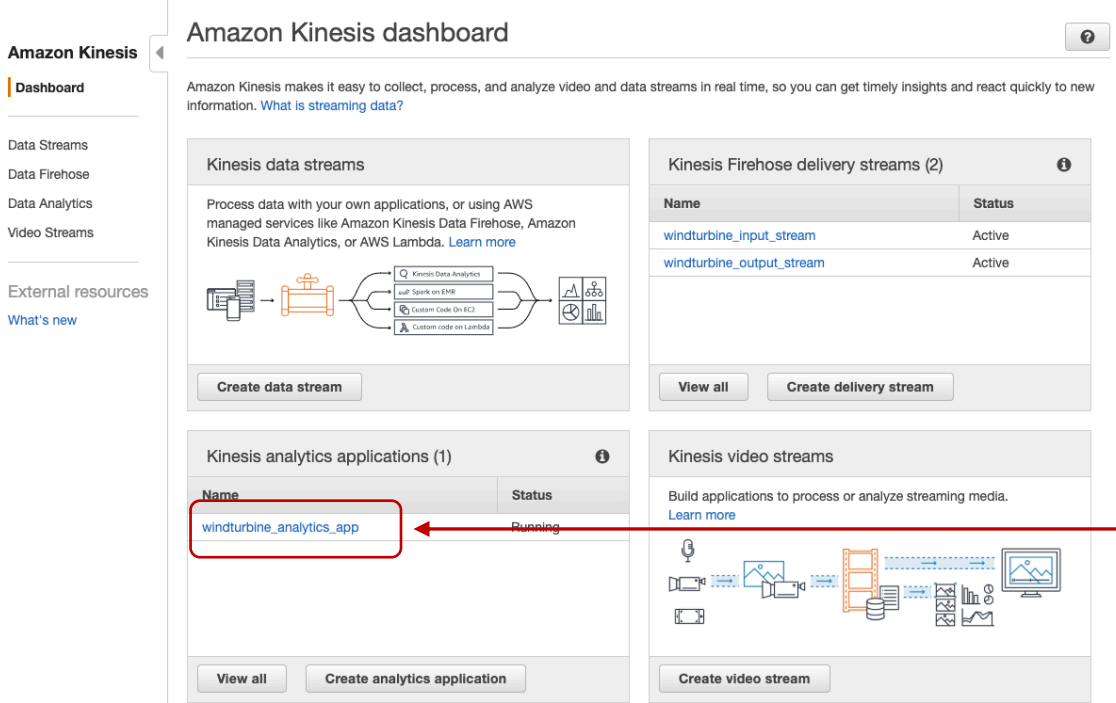
Uncomment the line here, when you are ready to send SNS message.

For reference, the entire lambda code is on the next page, you can copy paste and replace the default code in Lambda console.

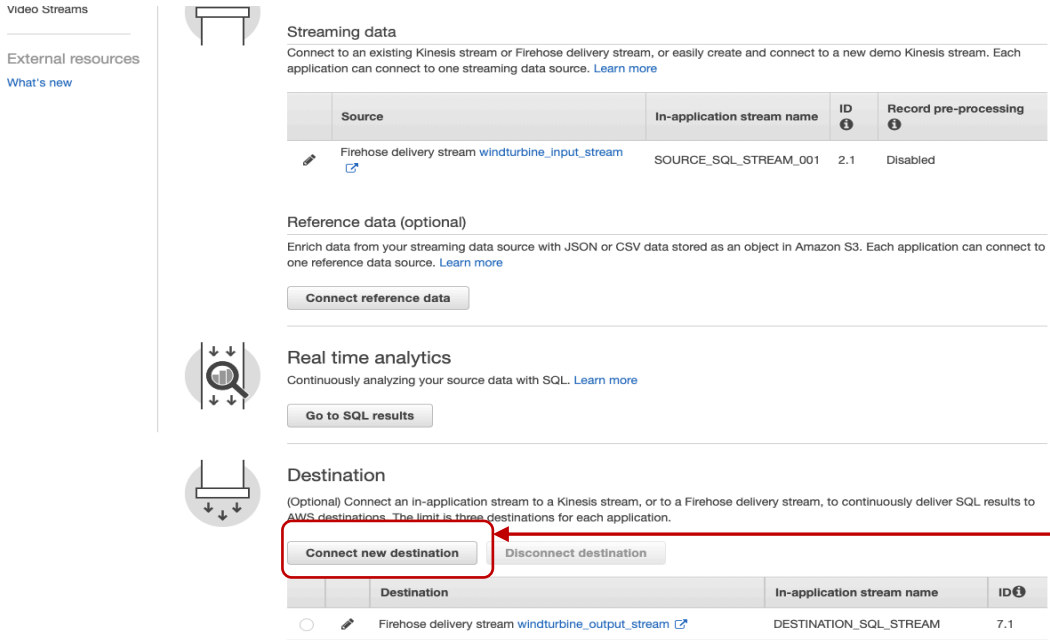
You can copy past the Lambda code from here and replace the existing lambda function code.
Don't forget to add your SNS topic ARN in the code! This code is also available in lambda_code.py you downloaded with the workshop package

```
from __future__ import print_function
import boto3
import base64
from json import loads

client = boto3.client('sns')
# Include your SNS topic ARN here.
topic_arn = 'Add your SNS topic SRN here'


def lambda_handler(event, context):
    output = []
    success = 0
    failure = 0
    for record in event['records']:
        try:

            payload = base64.b64decode(record['data'])
            data_item = loads(payload)
            #print (data_item)

            if (data_item['ANOMALY_SCORE']) > 2:
                client.publish(TopicArn=topic_arn, Message=payload, Subject='Anomaly Detected!')
                print ('Anomaly Detected')
            output.append({'recordId': record['recordId'], 'result': 'Ok'})
            success += 1
        except Exception:
            output.append({'recordId': record['recordId'], 'result': 'DeliveryFailed'})
            failure += 1

    print('Successfully delivered {0} records, failed to deliver {1} records'.format(success, failure))
    return {'records': output}
```

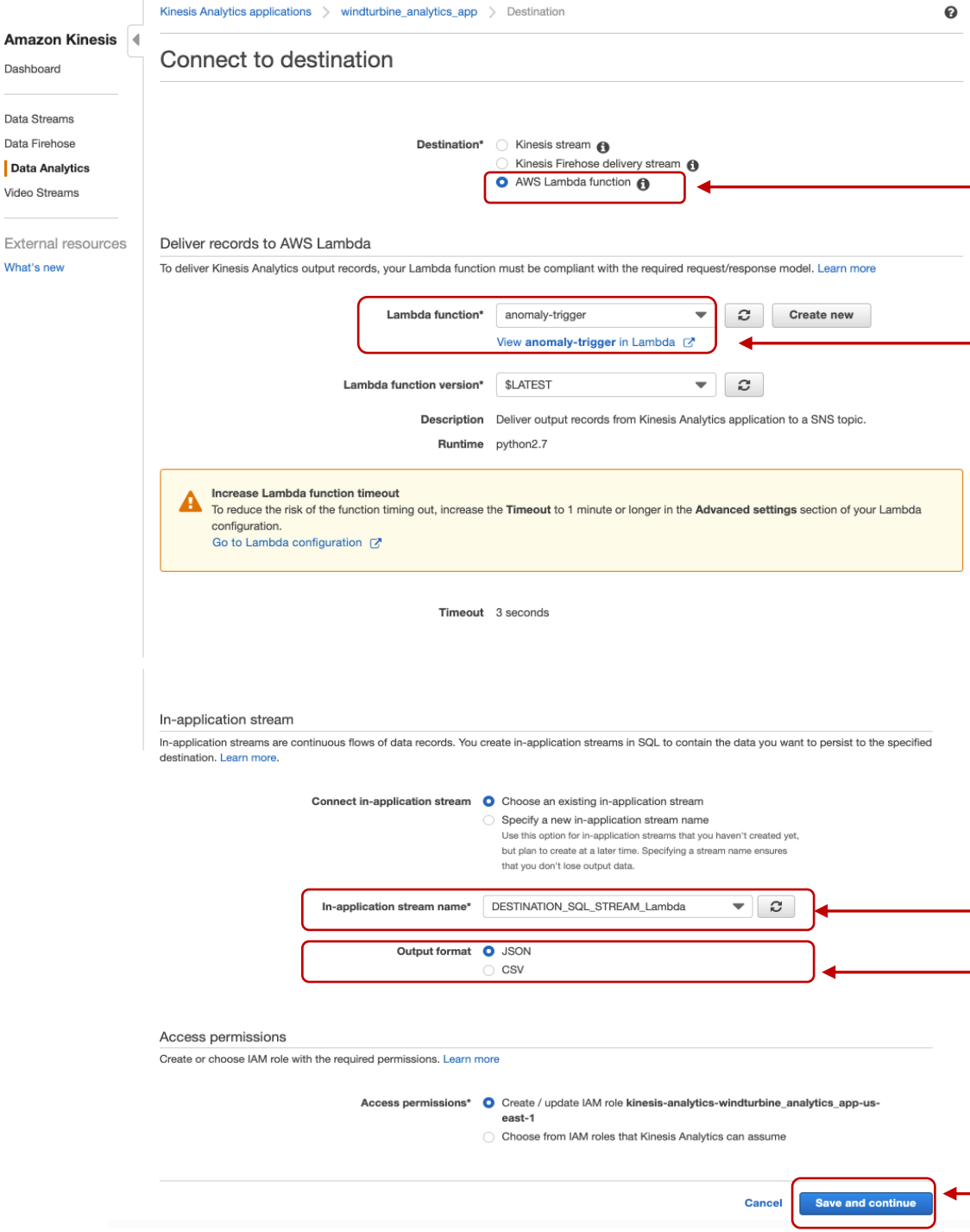
Link Lambda function to Kinesis Analytics App



Go back to the Kinesis dashboard and select Kinesis Analytics App. We will connect a new destination to the app to send data to a Lambda Function. The Lambda Function will trigger SNS notification, when Anomaly is identified.



Click, 'Connect new destination'



Select Lambda as destination

Select the Lambda function we created in previous step

Select In-application Stream 'DESTINATION_SQL_STREAM_Lambda'

Select 'JSON' output format

Click Save and continue

When anomalies are detected, SNS will send a notification to the endpoint (Email) you specified earlier.

{"COL_timestamp":"2019-11-15 22:15:26.000","geo_location":"51.7001597,
-0.5997986","turbine_id":1,"wind_speed":1,"oil_temperature":1,"temperature":1,"wind_direction":1,"oil_level":1,"RPM_blade":1,"vibration_frequency":1,"ANOMALY_SCORE":3.1708571428571433}

--

If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:

https://sns.us-east-1.amazonaws.com/unsubscribe.html?SubscriptionArn=arn:aws:sns:us-east-1:42[redacted]44:turbine_anomaly_detection:3cd0e[redacted]009&Endpoint=[redacted]amazon.com

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at https://aws.amazon.com/support

This is what you built today…