

**IoT Edge Design & Deployment document for Sysgain**

Version. 1.0

Dec, 2017

Table of Contents

**[1. Introduction 3](#_Toc6555)**

**[2. Components required 3](#_Toc11204)**

**[3. Data Flow 3](#_Toc22507)**

**[4. Sequence Diagram 4](#_Toc15874)**

**[5. Setup Steps 4](#_Toc21709)**

**[6. Data Structure 10](#_Toc23337)**

**[7. Next steps 10](#_Toc13835)**

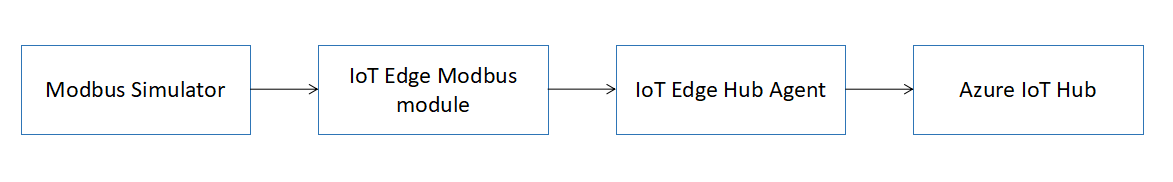
## **Introduction**

This document is created to give the follower an overview of data flow from a modbus device through Azure IoT edge system to Azure IoT Hub. This document also covers the steps to be taken to get the system working on targeted environment and some of the items that need to be taken care of once the data reaches the IoT hub.

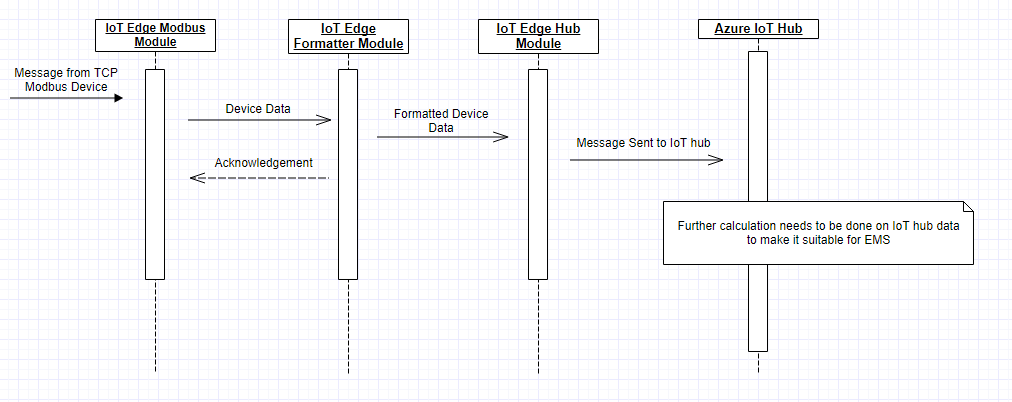
## **Components required**

1. **Modbus Simulator**  
   A PLC simulator could be downloaded from [here](http://www.plcsimulator.org/downloads), which upon installation would simulate the values over modbus-tcp.
2. **IoT Edge V2 environment (properly set up)**  
   To set up the edge environment on a machine (either PC or gateway), steps mentioned in a blog [here](https://docs.microsoft.com/en-in/azure/iot-edge/) could be followed.
3. **IoT Edge Modbus module (V2)**We would be using the available modbus module V2 for Azure IoT Edge to read data from a modbus device(PLC simulator in this case). The source and the steps to get this module on IoT Edge environment are well documented [here](https://github.com/Azure/iot-edge-modbus) and can be followed as mentioned.
4. **IoT Edge Formatter Module**  
   This is a custom module built to restructure the data coming from the modbus module. The docker image for the same could be found [here](agreeyait/formattermodule:1.0).
5. **Azure IoT Hub**It is used to receive the telemetry from IoT Edge runtime environment.

## **Data Flow**

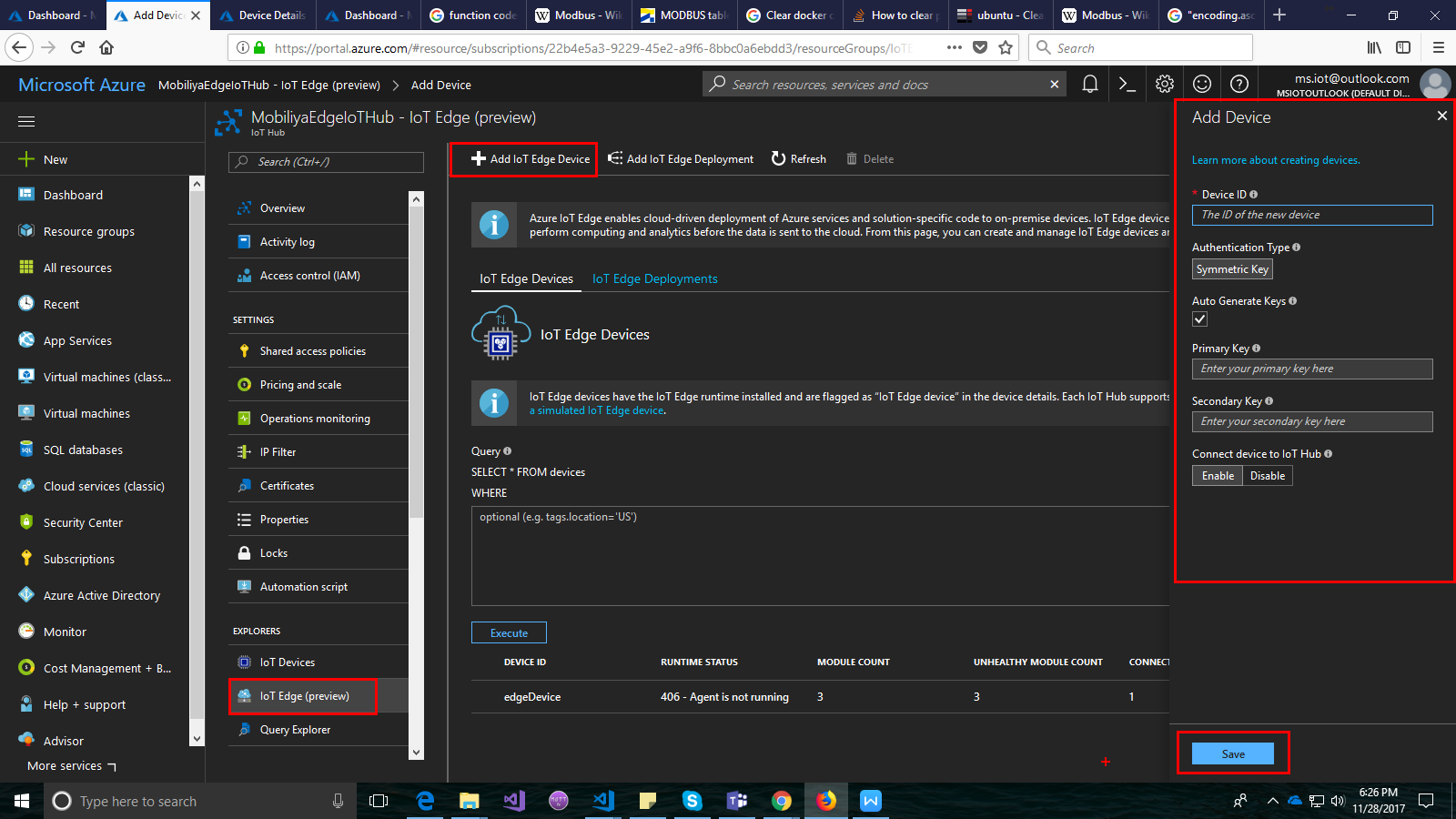


## **Sequence Diagram**



## **Setup Steps**

1. Once the IoT edge environment has been set up on target machine, then start modbus simulator on the same machine and fill values in some of the registers. List of sample values along with the register address could be found in excel sheet on Github repository.
2. Create an edge device in already created Azure IoT hub on Azure portal.



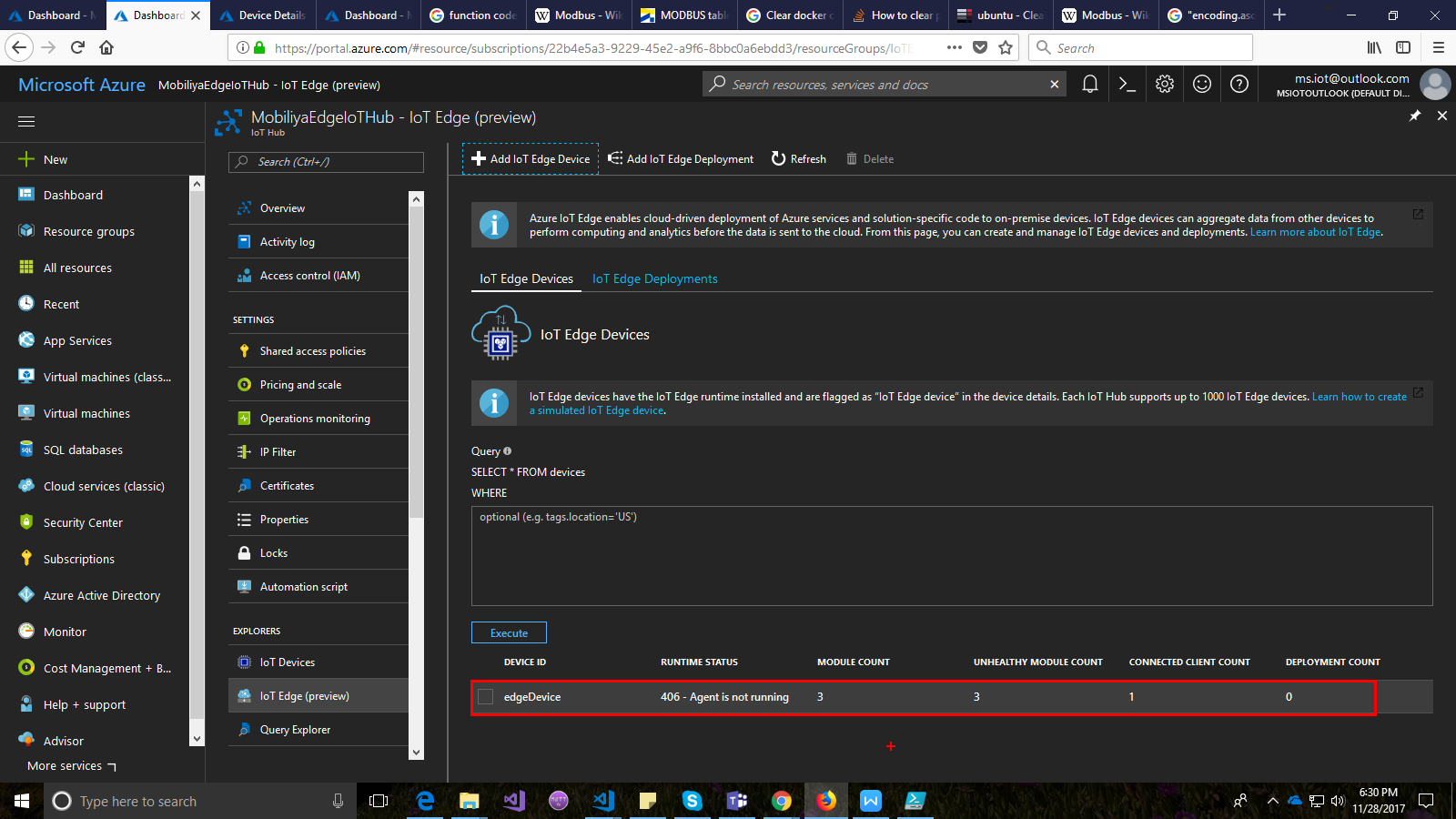
1. Copy the device connection string from the device details screen appearing on Azure portal (scroll down for screen) and set up an IoT edge runtime environment using the following command

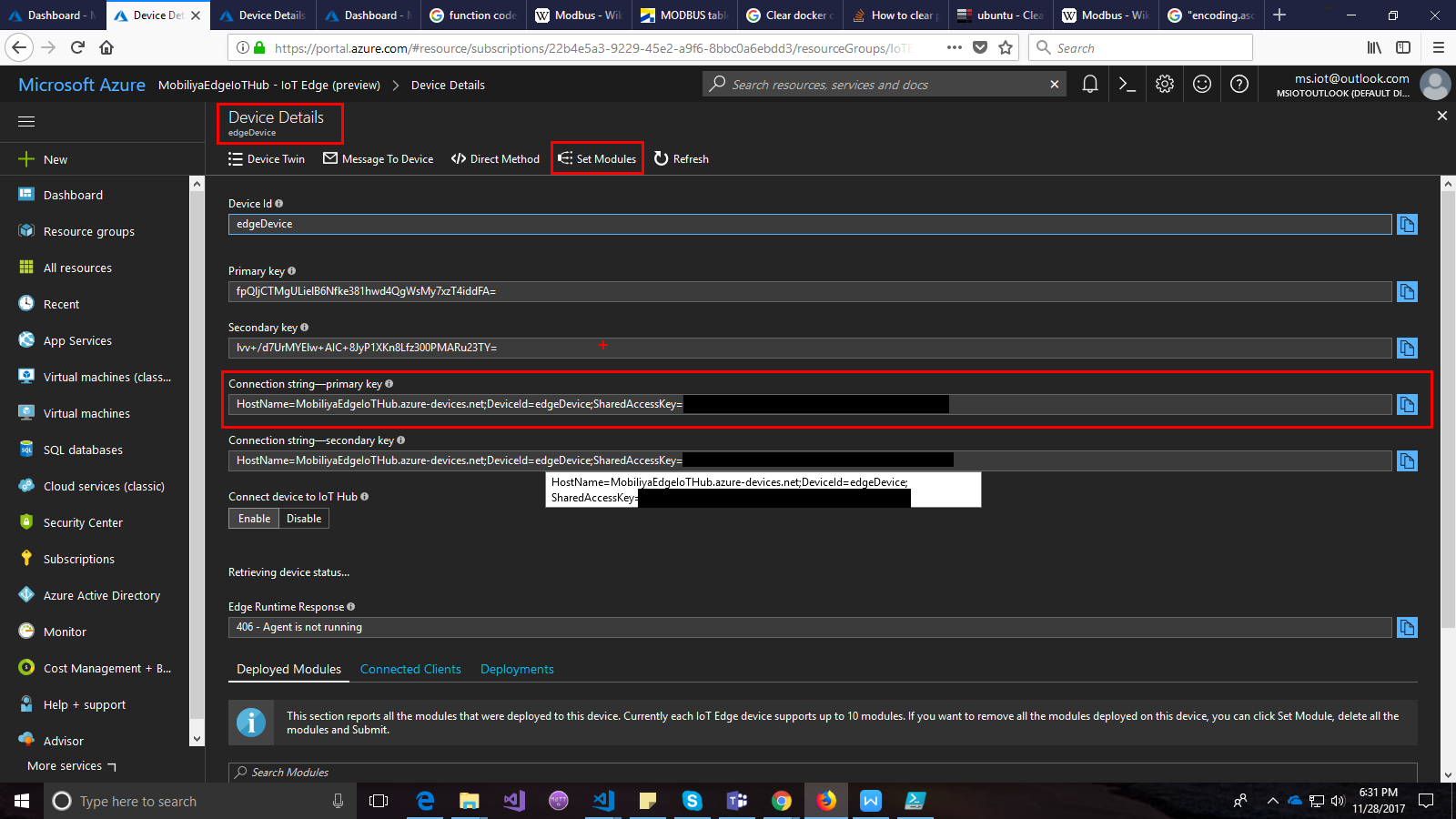
iotedgectl setup --connection-string "<Device Connection String here>" --auto-cert-gen-force-no-passwords

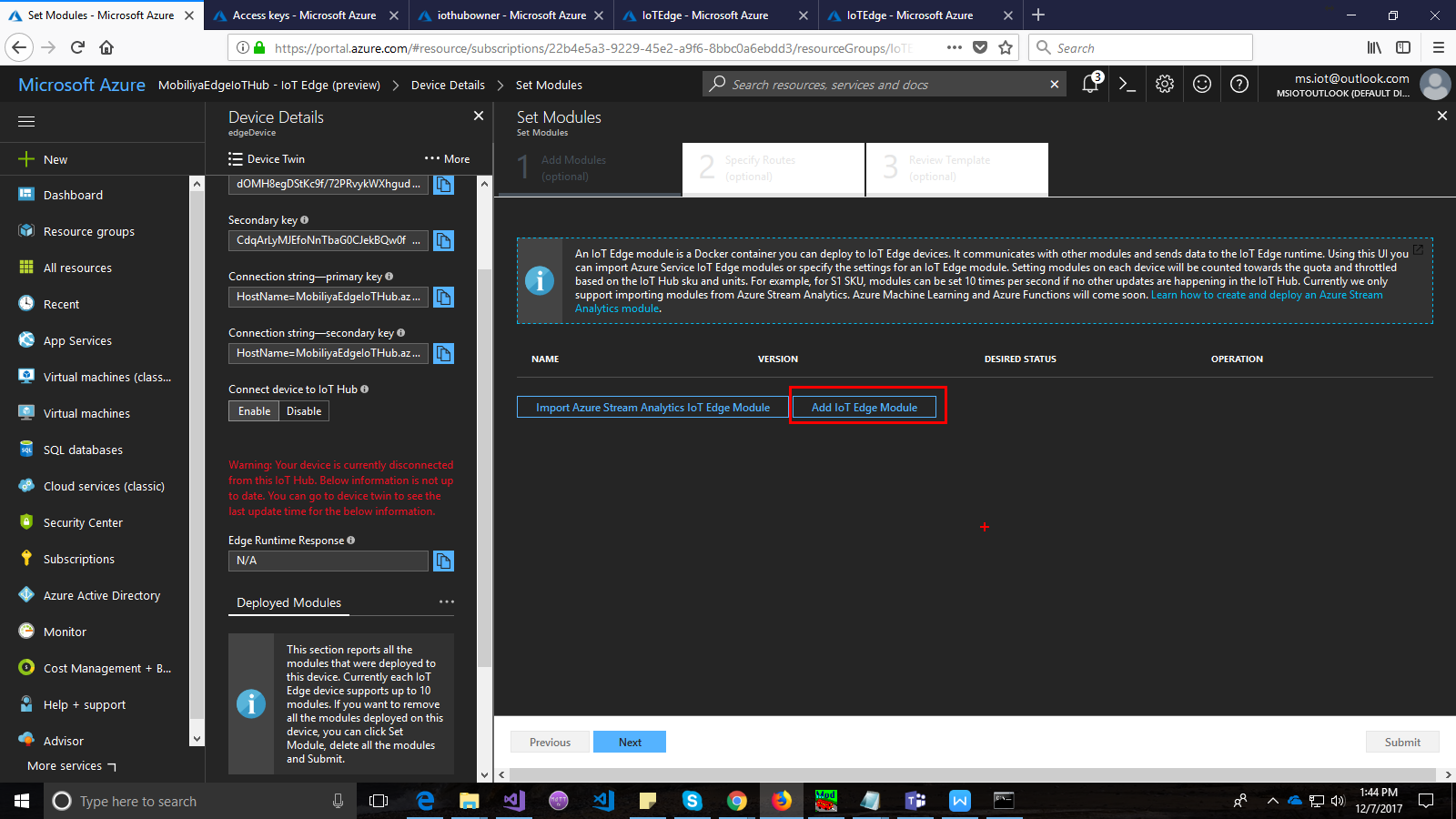
1. Start the IoT edge runtime environment upon successful set up using the command

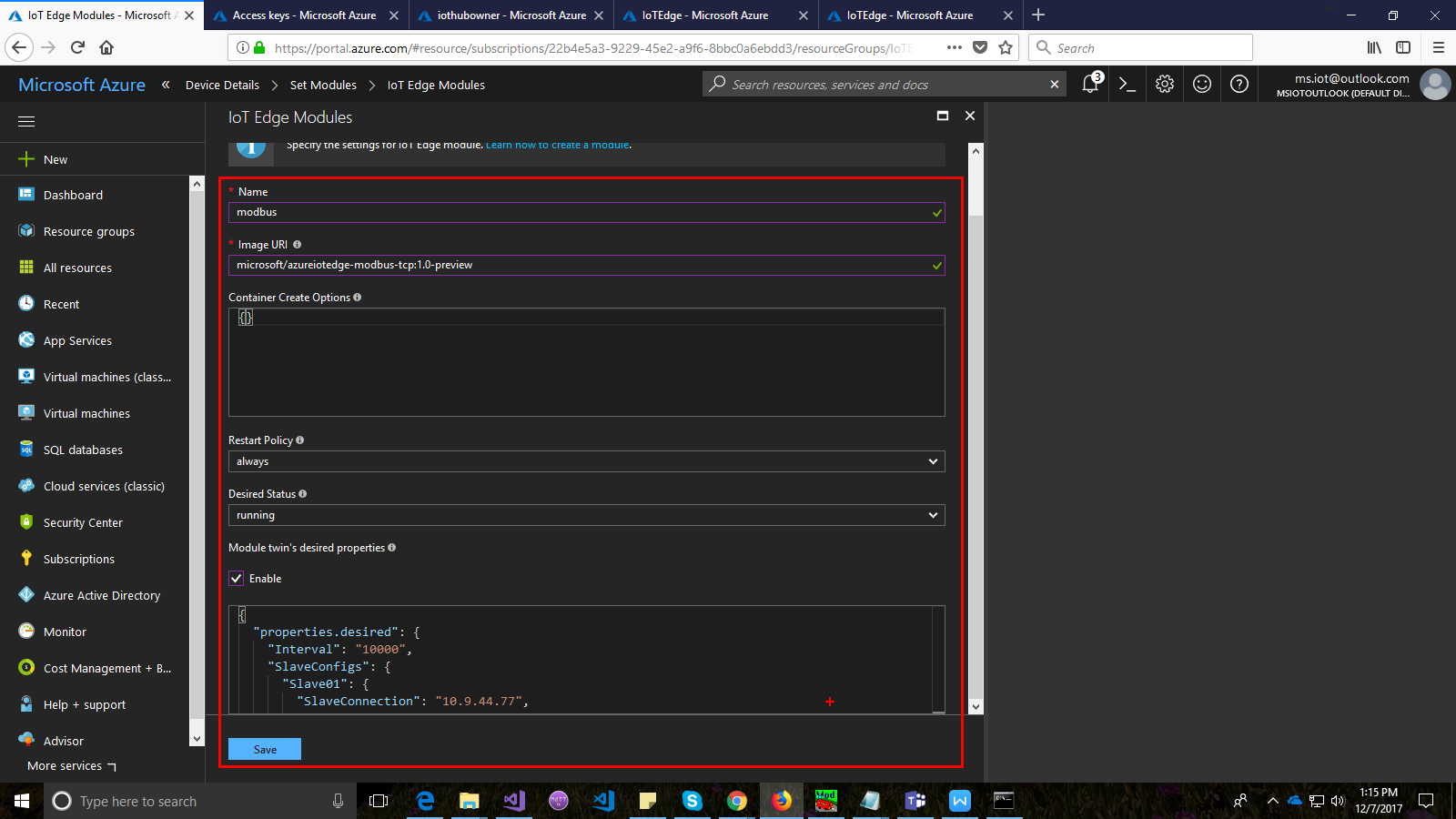
iotedgectl start

1. Create a modbus module for the selected device on Azure portal in device details screen. We need to create two modules, first the modbus module and second the formatter module.

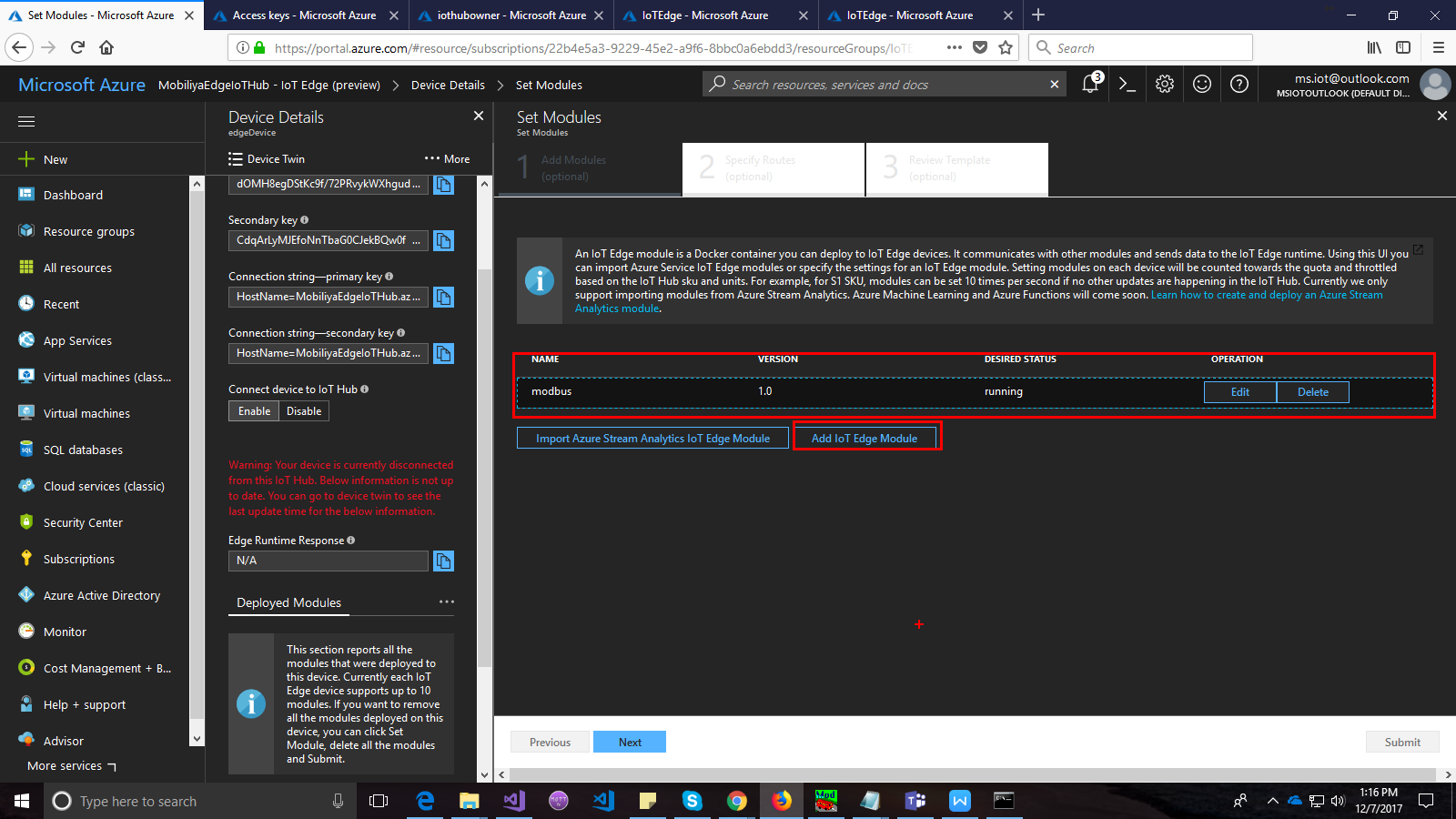


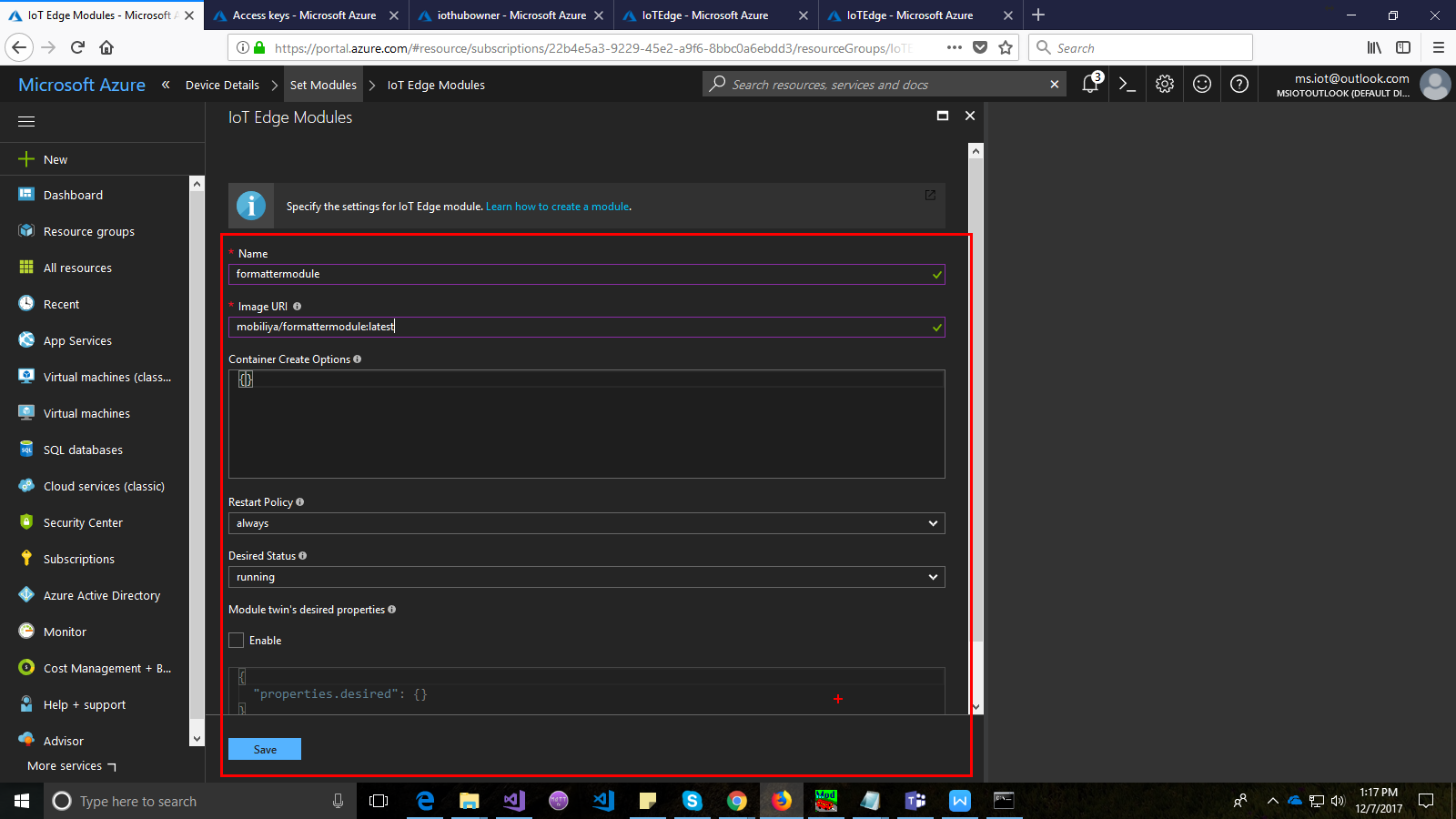




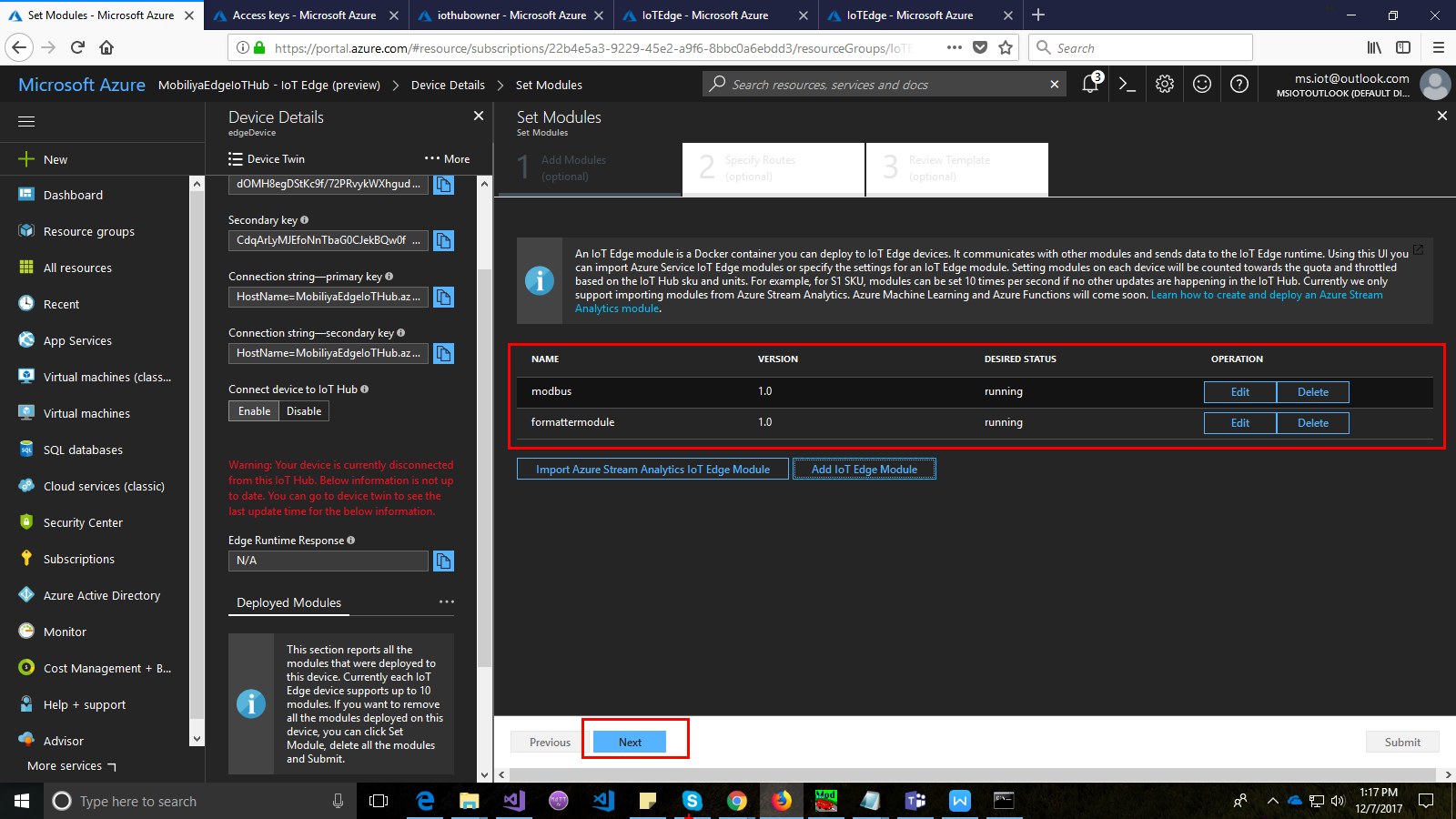


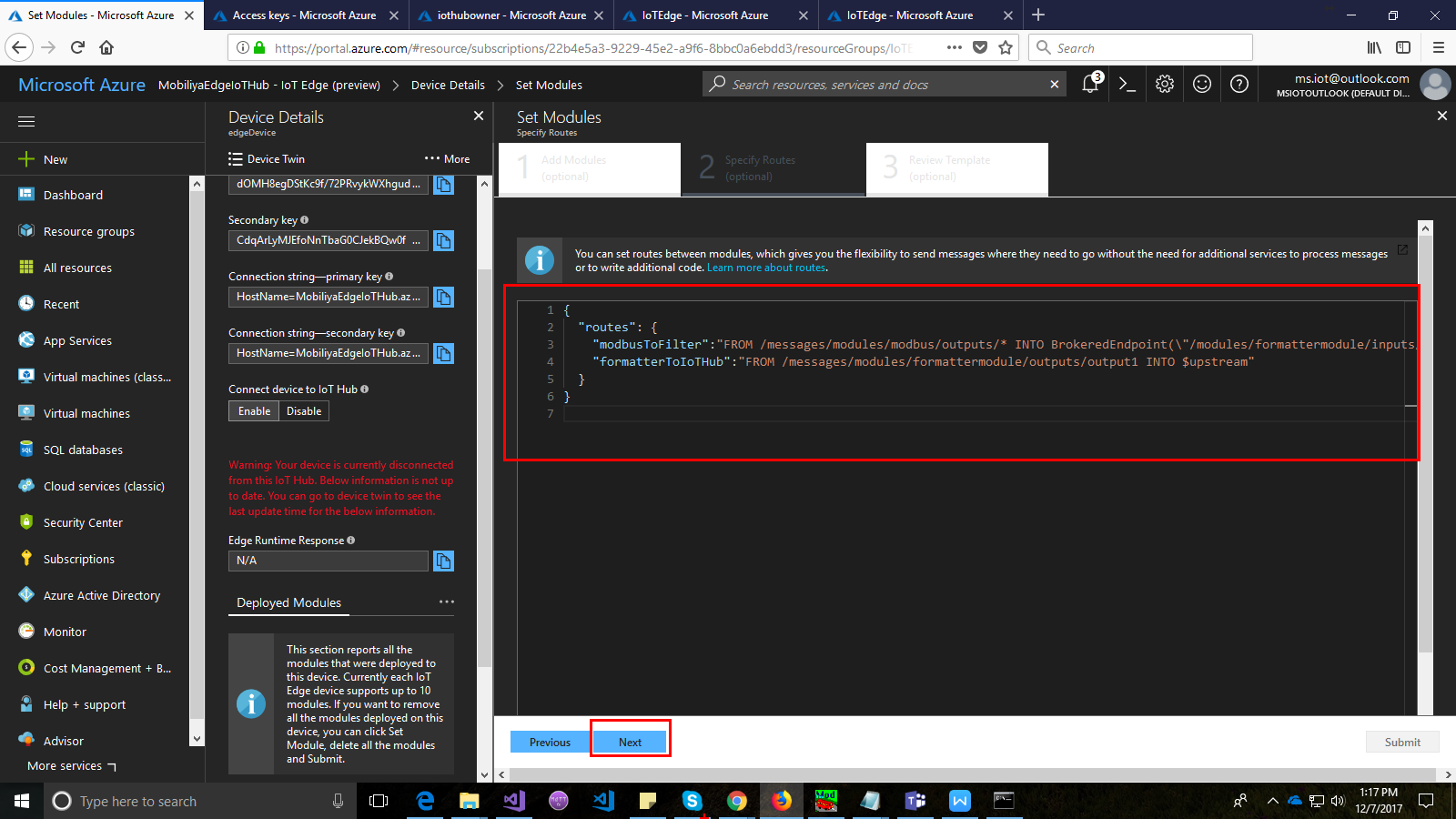
Note: Sample for Module twin configuration in above screen could be found in the Github repository.





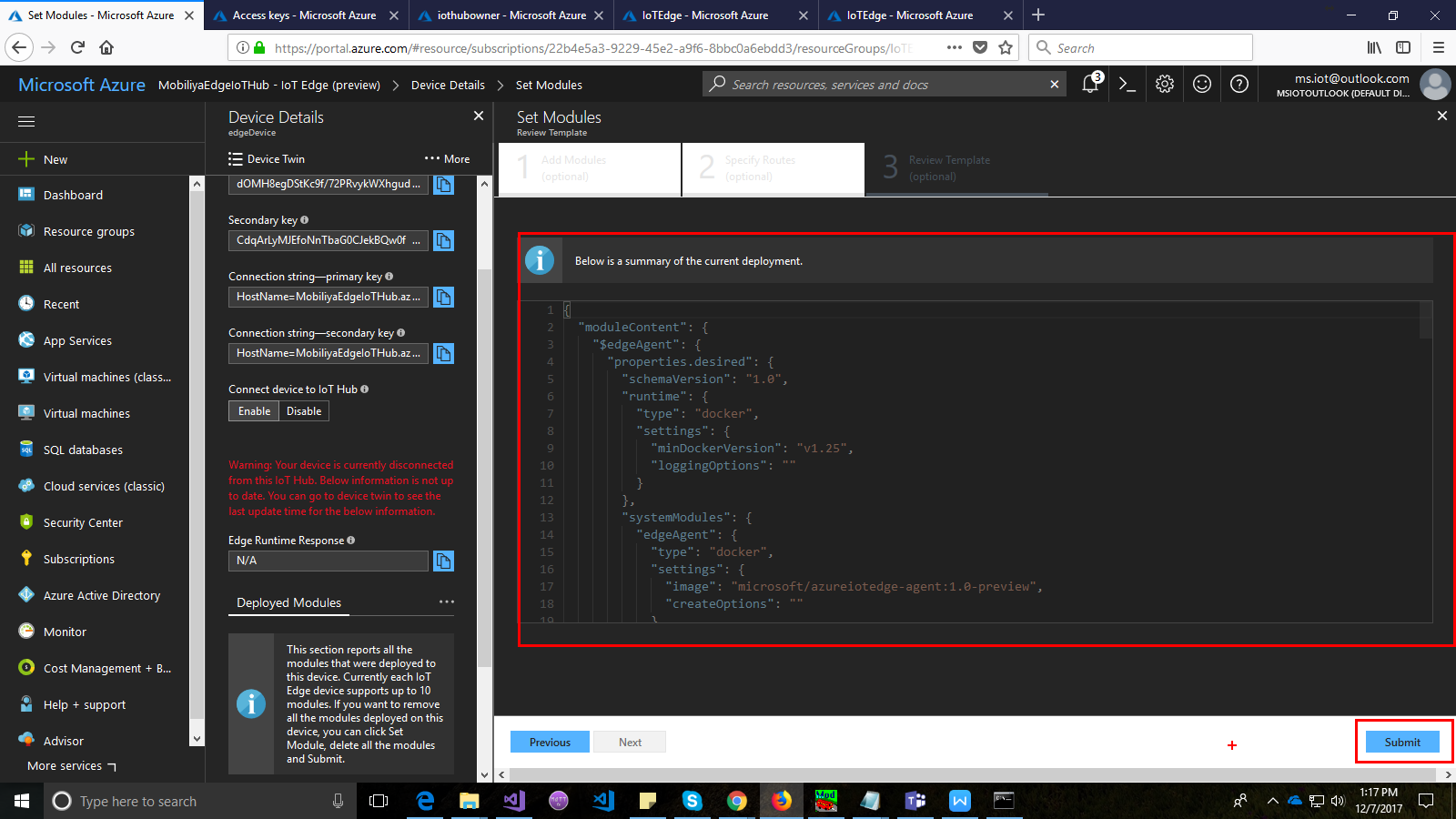
Note: Image URI for formatter module in the above screen is **agreeyait/formattermodule:1.0**





Note: Sample for route mentioned in the above screen could be found in the Github repository.

1. Keep all the other options as mentioned (default) and press **Submit**. This would then push the docker module onto the gateway i.e. IoT edge configured device.



1. You could check the modules running on the IoT edge environment by running command (Note: It might take some time for the modules to be pushed on to the edge device and the modules to show up)

docker ps

1. Check the docker logs to check if the modbus is reading the data and setup is working fine using command

docker logs <Container/module name>

1. Optionally you can add a listener on IoT hub to check the data that is coming in.

## **Data Structure**

The data would be received in the following format on the IoT hub.

{

"DeviceData": [

{

"Device\_Id": "HW1",

"AMPSAvg": "111"

"AMPS-L1": "11",

"AMPS-L2": "12",

"AMPS-L3": "13",

"Volts L1 to Neutral": "14",

"Volts L2 to Neutral": "15",

"Volts L3 to Neutral": "16",

"kW L1": "17",

"kW L2": "18",

"kW L3": "19",

"kW System": "20"

}

]

}

## **Next steps**

1. Once the data in on IoT hub, it would require further processing and calculations to fit it in already created EM solution which could be either done using the web jobs or serverless computing.
2. The formulae for the calculations are as below.

|  |  |
| --- | --- |
| **Parameter** | **Formula** |
| Daily\_electric\_cost | Daily\_KWH\_System \* 0.04367 (Current electricity cost) |
| Daily\_KWH\_System | IF (Hour('\*') = 0 AND Minute('\*') = 0) THEN 0 ELSE TagTot('kW System','t','\*')\*24 |
| Monthly\_electric\_cost | Monthly\_KWH\_System \* 0.04367 (Current electricity cost) |
| Monthly\_KWH\_System | IF (Hour('\*') = 0 AND Minute('\*') = 0 AND Day('\*') = 1) THEN 0 ELSE TagTot('kW System','1','\*')\*24 |
| Rolling\_hourly\_kwh\_system | TagTot('kW System','\*-1h','\*')\*24 |

1. The above formulae are for numeric parameters. In addition to these, other mapping parameters like building etc. also have to be taken care of at the cloud end.