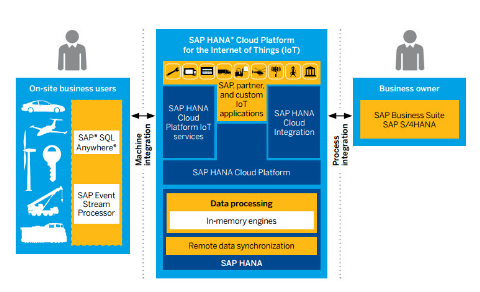
IoT1C01 – Setup IoT Services - HCP

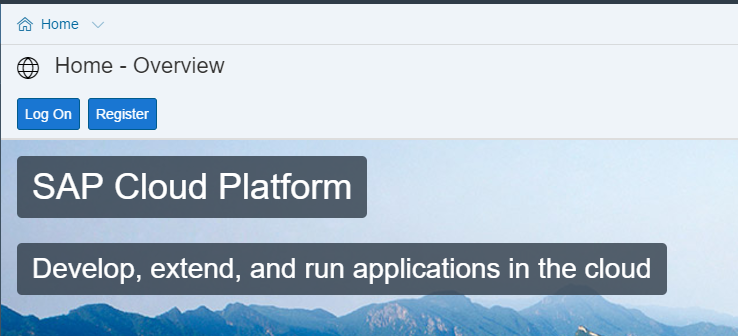
|  |  |
| --- | --- |
| **Product**  HANA Cloud Platform IoT Services  **Level**  Undergraduate/Graduate  Beginner  **Focus**  HANA Cloud Platform  **Author** Ross Hightower | MOTIVATION  In this case you will configure the IoT services on HANA Cloud Platform.  **PREREQUISITES**  None |



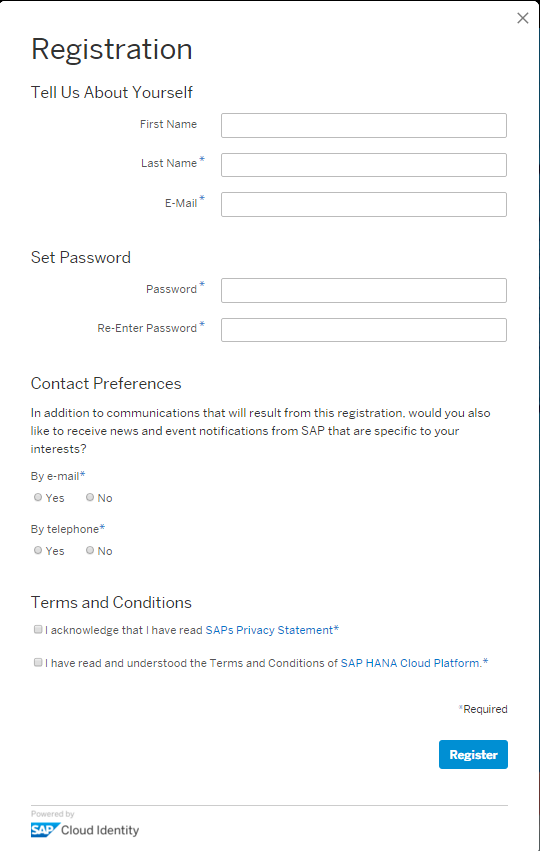
# Registering for HCP

|  |
| --- |
| If you have already registered for any of these services or already have an SAP Community Network (SCN) id, you should use the same email to register for the other services. That allows your accounts to be linked. |

Although other browsers may work, SAP recommends you use Google Chrome. Navigate to <https://hanatrial.ondemand.com>. To create an account click **Register**.

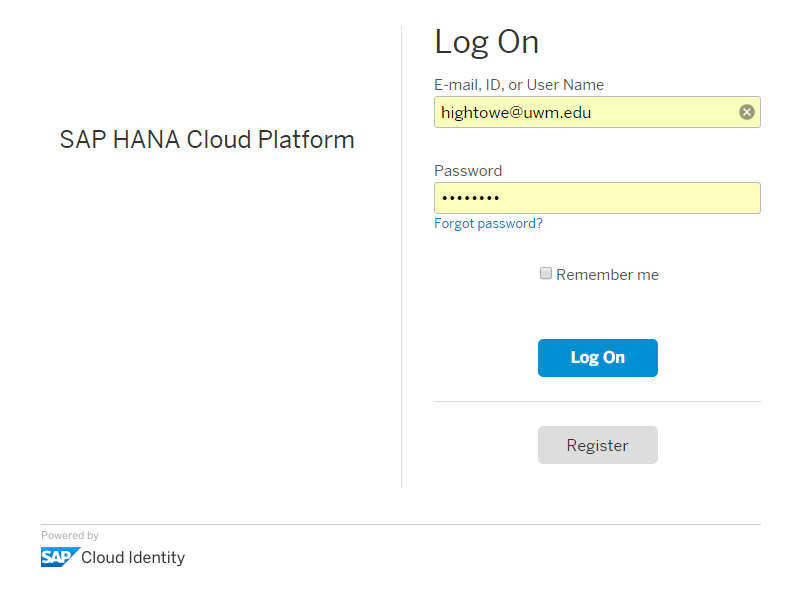


Fill in the Registration form and click **Register**.



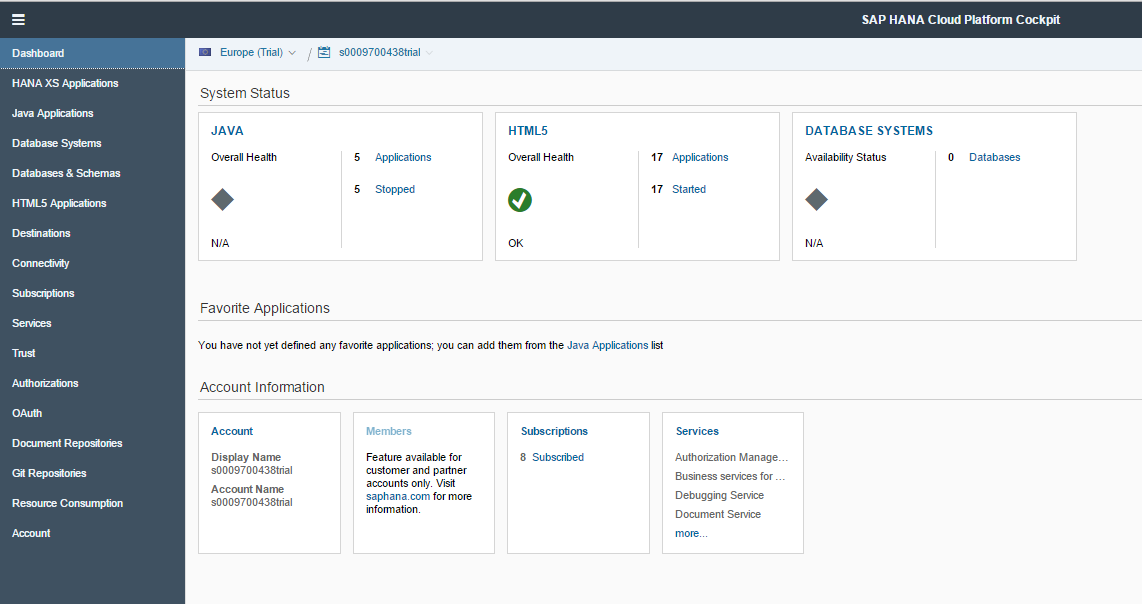
You will receive an email with a link that allows you to activate your account. Once you activate your account click Continue to proceed to the HCP Cockpit which is the administrative tool for HCP. You will also receive another email with links to several useful resources.

You can access the HCP Cockpit using [**https://account.hanatrial.ondemand.com/cockpit**](https://account.hanatrial.ondemand.com/cockpit).

****

When you log on to the cockpit you will often see a dialog that shows News and Announcements which describe updates to the platform. The platform can evolve quickly.

The first screen you will see is the Dashboard which provides a summary of the state of your account.

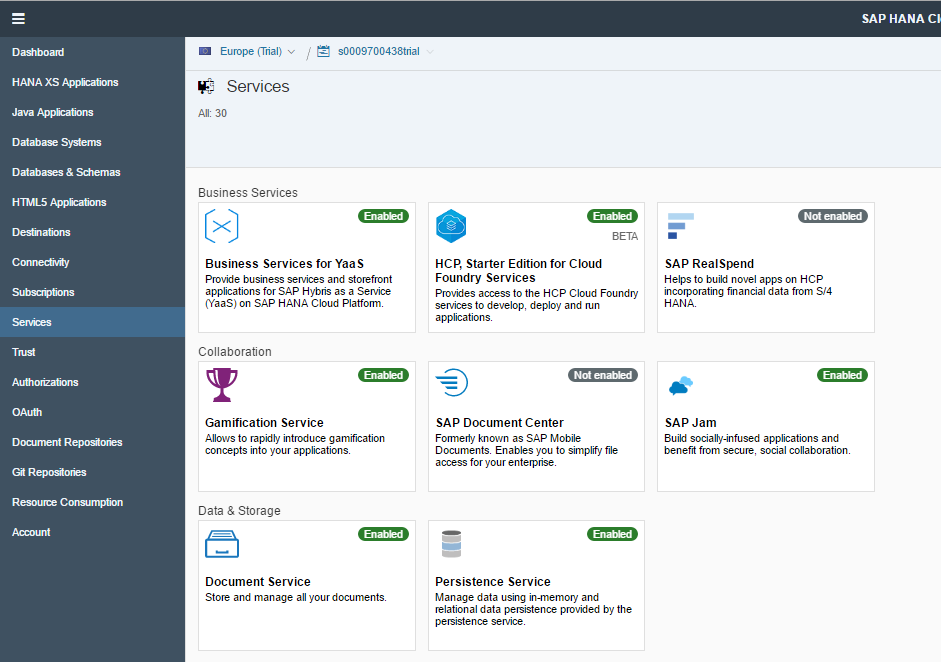


Note the bread crumb navigation near the top of the page. This shows the name of your account (ending in trial), the datacenter where you are hosted (Europe in the image below) and the page you are on (SAP Web IDE configuration on the image below). You can return to the top level by clicking your account id.

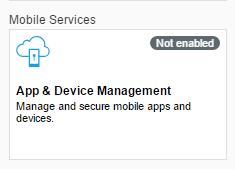


## Setup IoT Services

Navigate to the Services page using the menu on the left side of the screen.



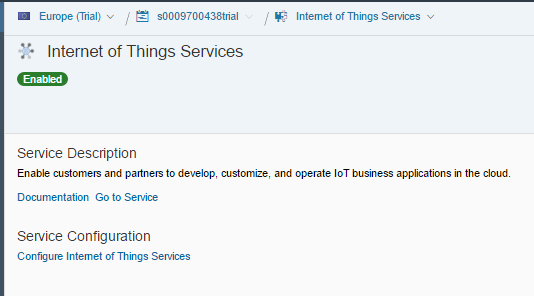
Scroll down until you find the Internet of Things Services tile. Note the tile says Not Enabled.



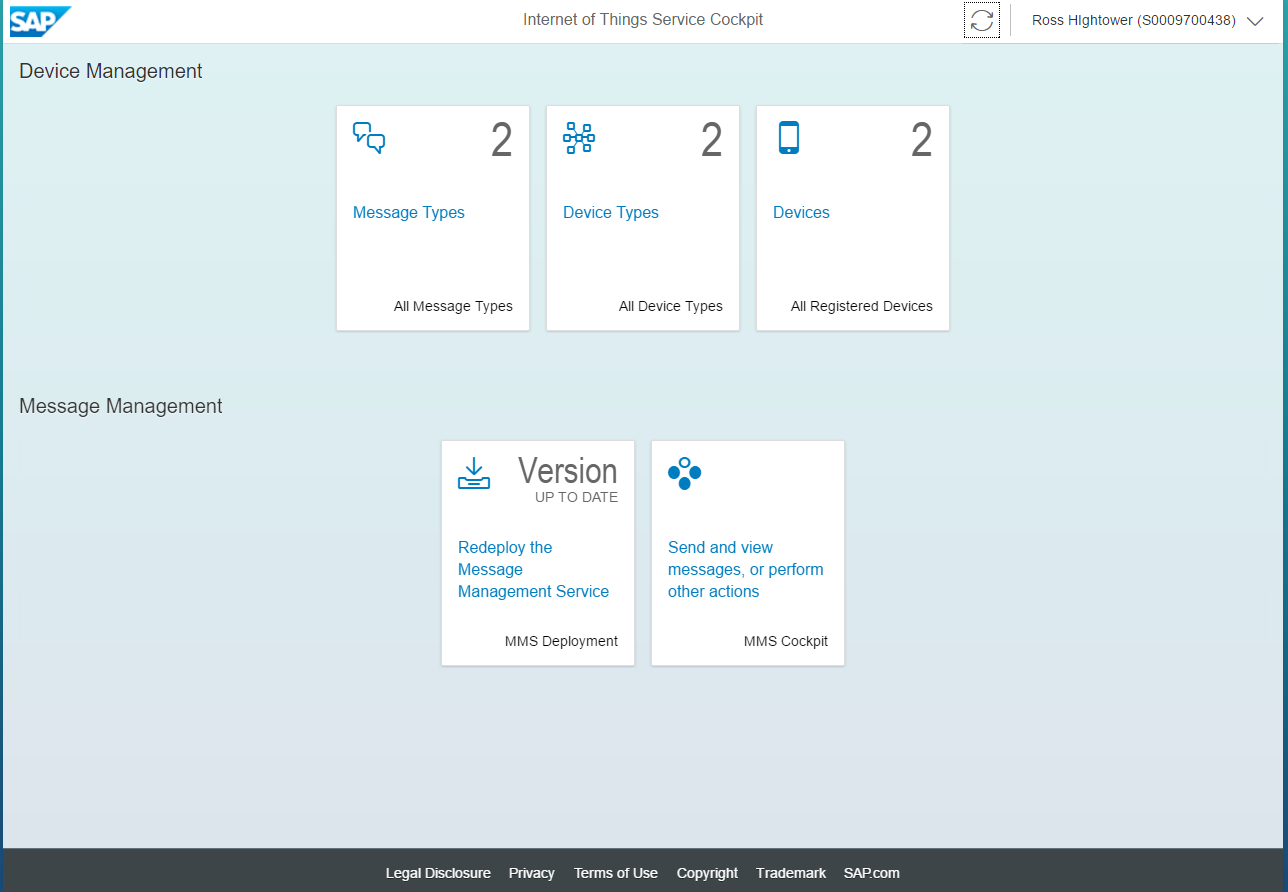
To enable the service, click the tile then click the Enable button.



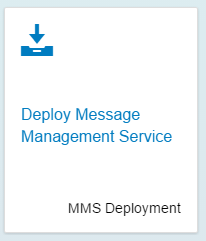
After a few minutes, the service is enabled.



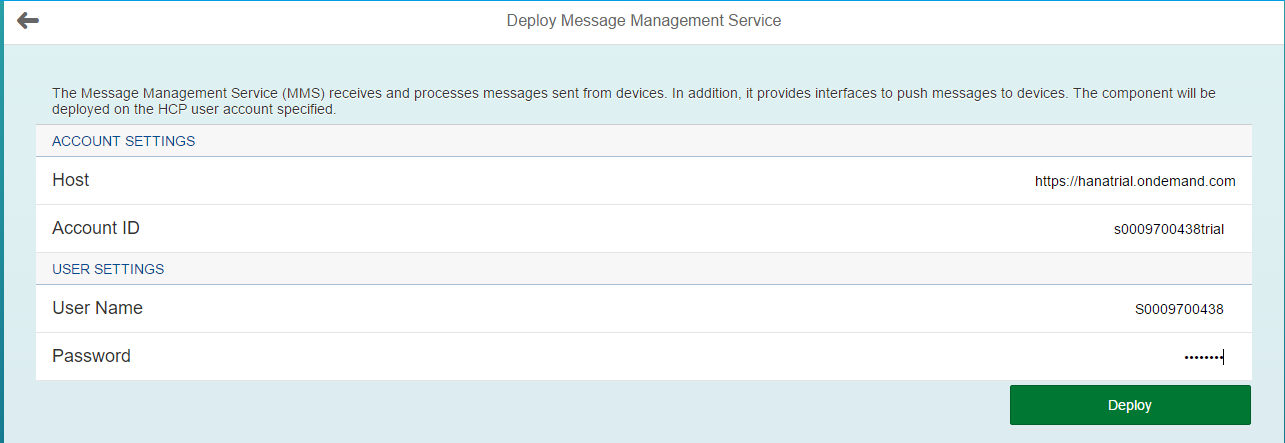
Note the link to the Documentation so you can find it later. Click the **Go to Service** link to open the Internet of Things Services Cockpit. You can bookmark this page so you can return directly to it.



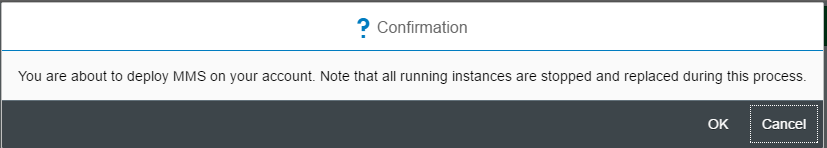
Click the MMS Deployment tile on the lower left to deploy the Message Management Service.



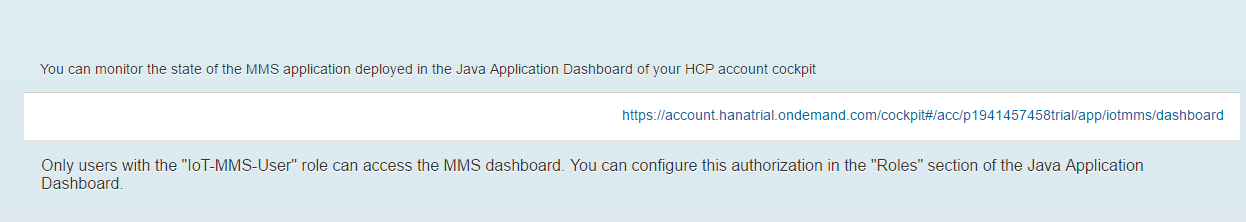
Enter the password for your HCP account and click **Deploy**.



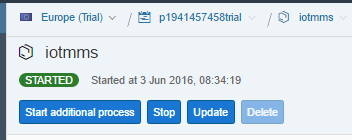
Confirm that you want to deploy the service. The deployment takes several minutes.



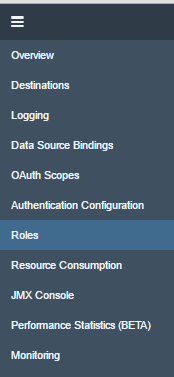
When the service has been deployed you will see an URI to the Java application.



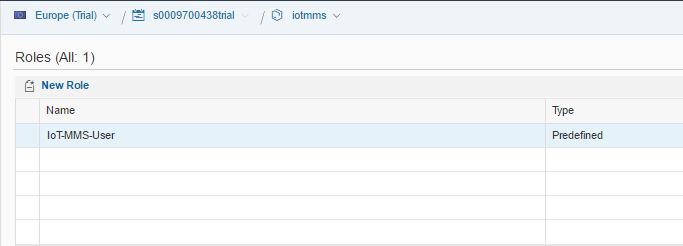
If you click this link, the MMS Dashboard in the HCP Cockpit opens. You can also access this dashboard from the Java Applications page in the HCP Cockpit. The status of the service is shown at the top of the dashboard page and you can control the service from here.



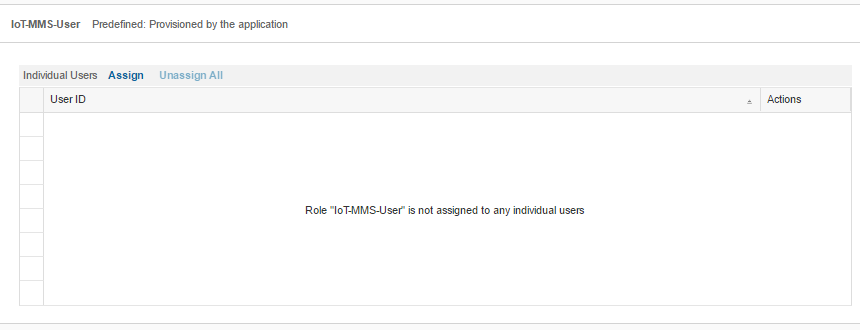
Click Roles in the menu on the left side of the screen.



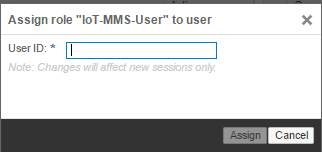
The role that grants access to the MMS service is already created.



To add a role to your id click **Assign** above the table on the lower left.

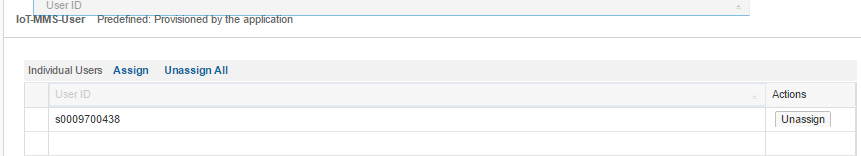


Enter your HCP user id.



|  |
| --- |
| If you don’t remember your id you can find it in the breadcrumbs at the top of the screen. The id is the middle portion without the trial so the id in the image is p1941457458. |

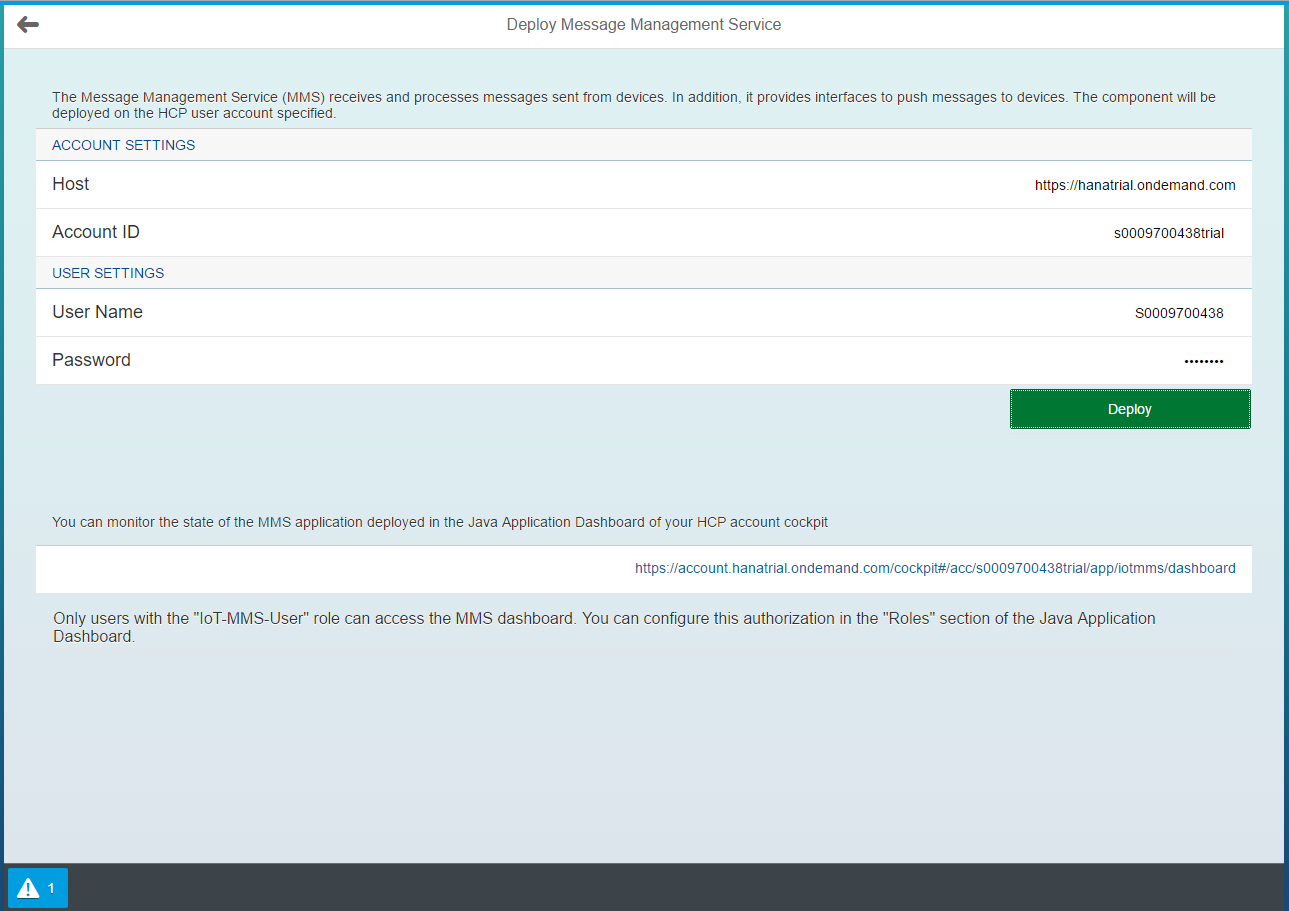
Click **Assign** and the role is assigned to your id.



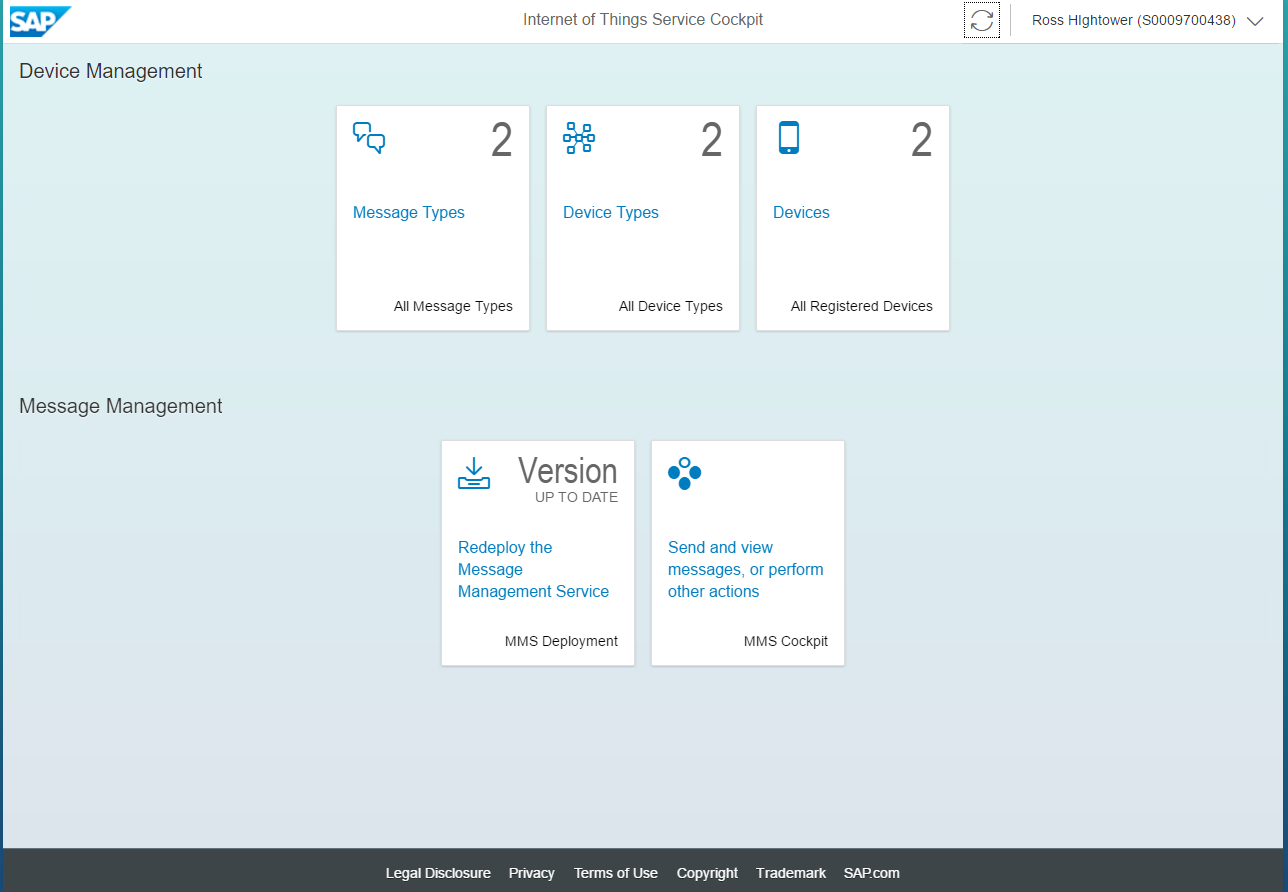
## Test the Service

Now that the service is running and your user ID has authorizations we can test the service with some sample services. The IoT service works by sending messages. Messages can be sent from a device to the server and from the server to the device.

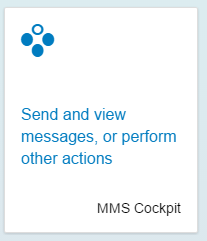
Return to the Deploy Message Management Service screen.



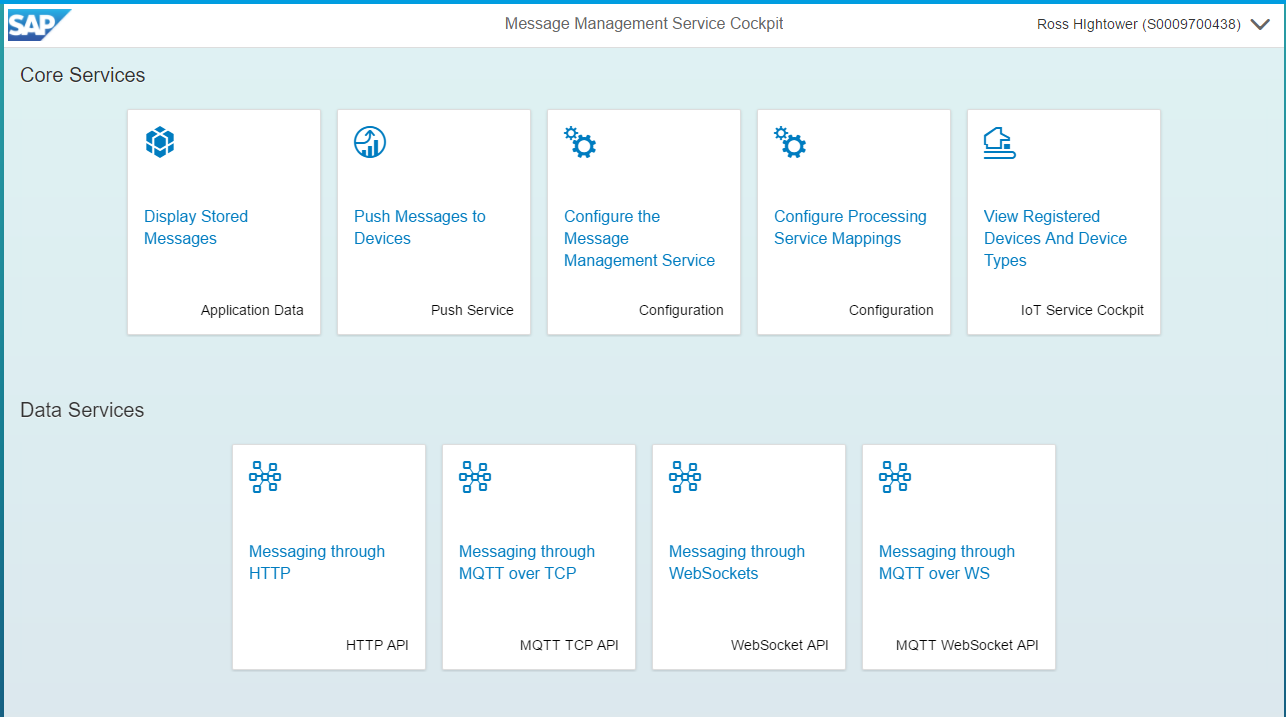
Click the back arrow to return to the IoT Services Cockpit. Note that the Deploy MMS tile has changed.



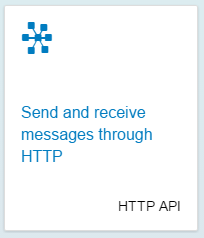
Click the MMS Cockpit tile.



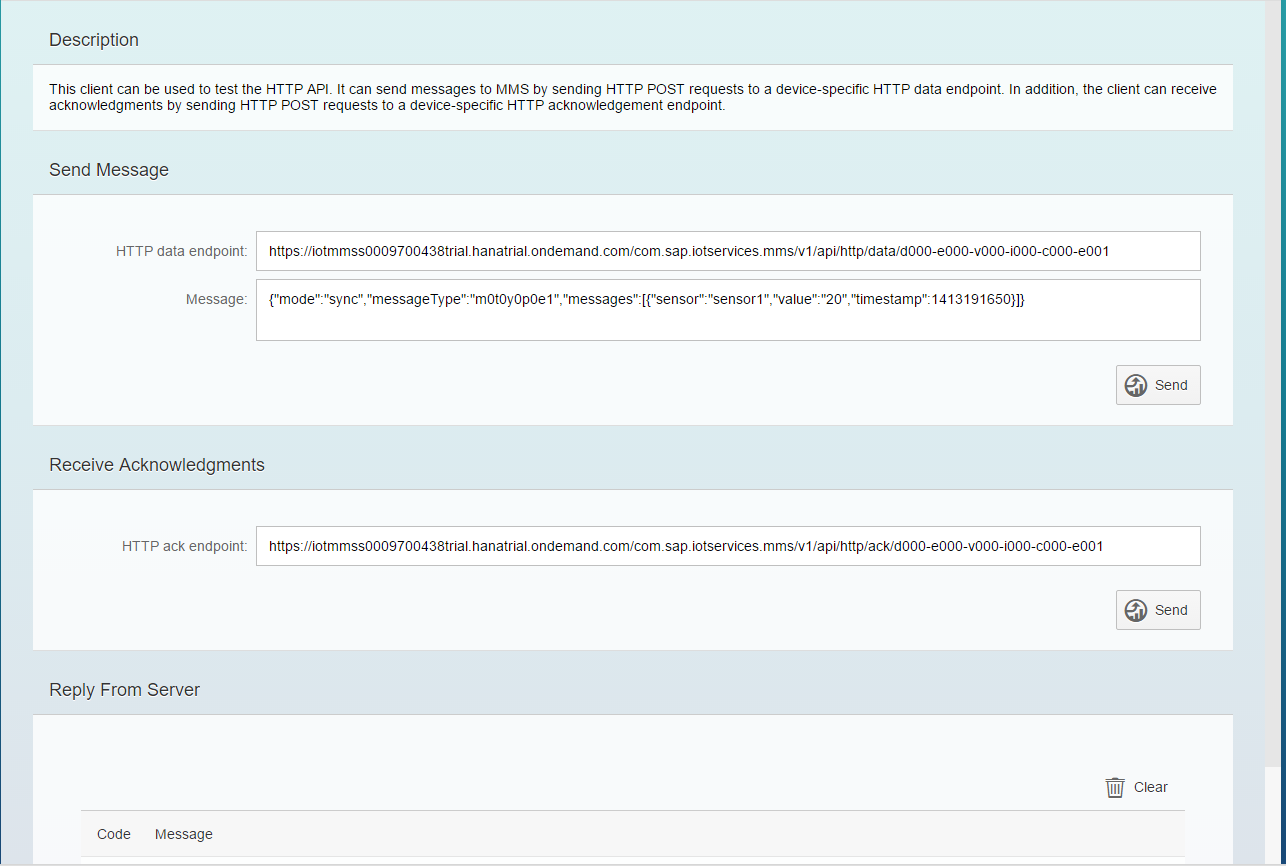
The tiles at the bottom of the screen show the ways to send messages to and from devices.



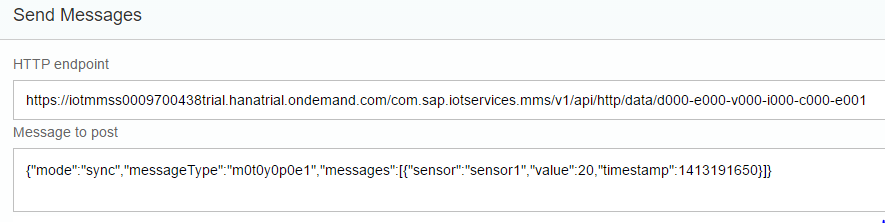
Click the HTTP API tile on the lower left.



This screen has documentation for making RESTful calls to the service and includes a sample service to test the service.



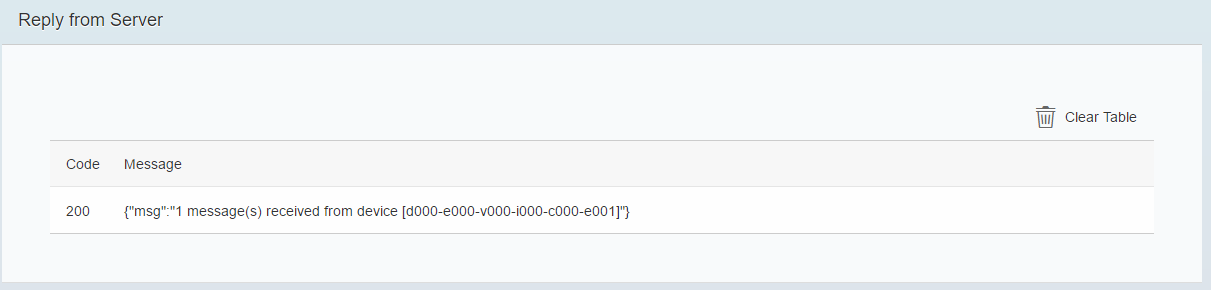
The URI for the Data Endpoint illustrates the format for sending messages to HCP from a device and the Message shows the format for the sample message.



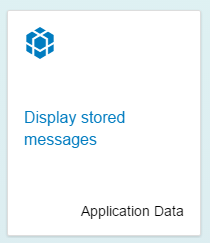
The d000-e000-v000-i000-c000-e001 at the end of the endpoint URI is a dummy device ID. In a real scenario this would be the id of a sensor, for example. We’ll configure our own devices later.

The message is in JSON format and it shows the basic format of a message. The message is being sent synchronously and is of type m0t0y0p0e1. The message type determines the format (e.g. values transmitted) and the database table in which the messages are stored. The sample message contains three fields: sensor, value and timestamp.

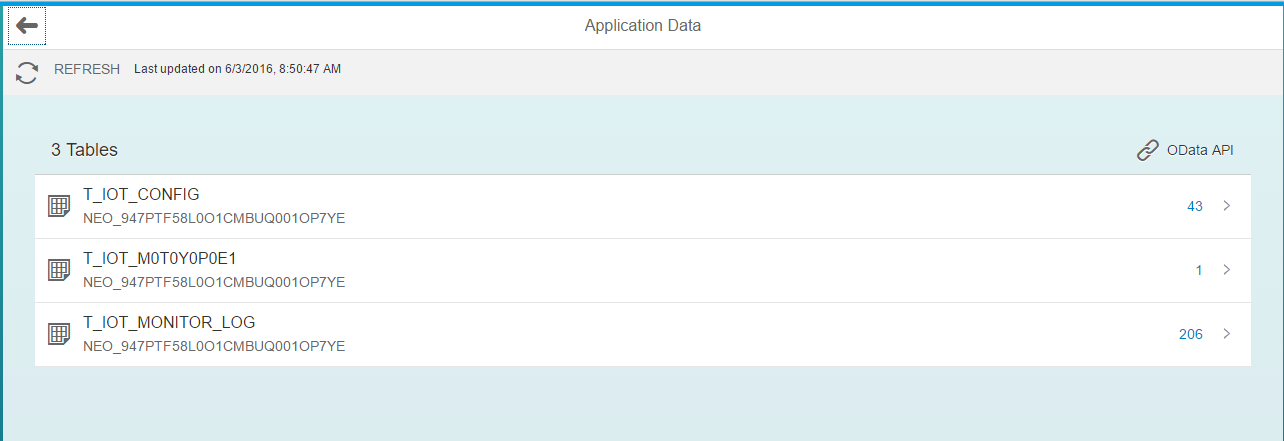
To send a test message, click the **Send** button. In a moment you will see the response from the server in the Reply from server section. The status 200 indicates the message was received and processed successfully.



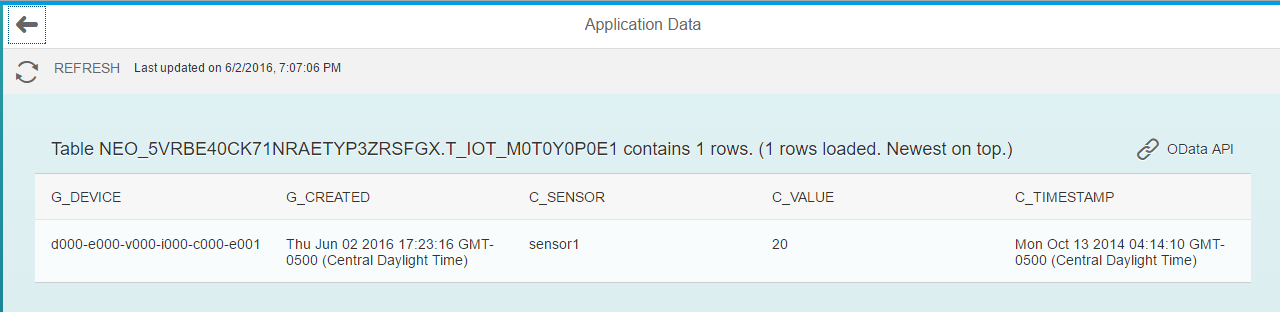
Click the back arrow to return to the MMS Cockpit then click the Application Data tile.



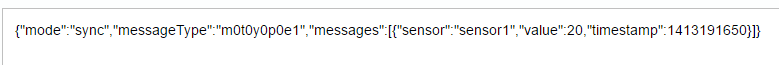
This screen shows the database tables that contain messages. You may have to click Refresh to see all the tables.



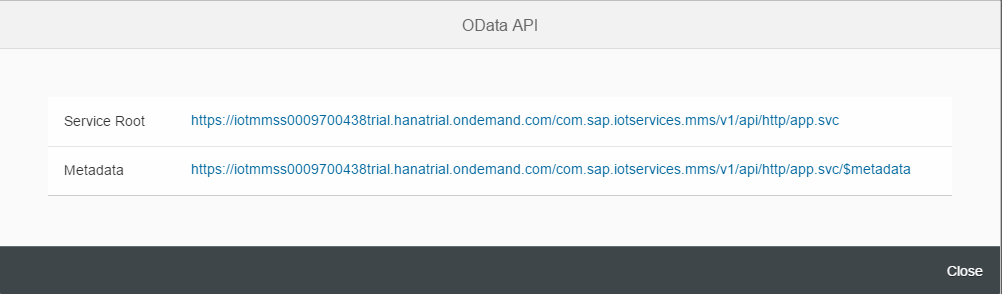
Click the T\_IOT\_M0T0Y0P0E1 table. This table contains the messages of type m0t0y0p0e1 which was the type of the sample message. There is one record that contains the sample message.



Compare the record in the table with the message you sent.



Click  to see an OData service you can use to interact with the service.



The Service Root will provide all the service endpoints available. If you click it the service document will open in a browser window:



The service that contains the actual data is the one with the message name. Copy the href and paste it on the end of the service root url. The URL has the following format:

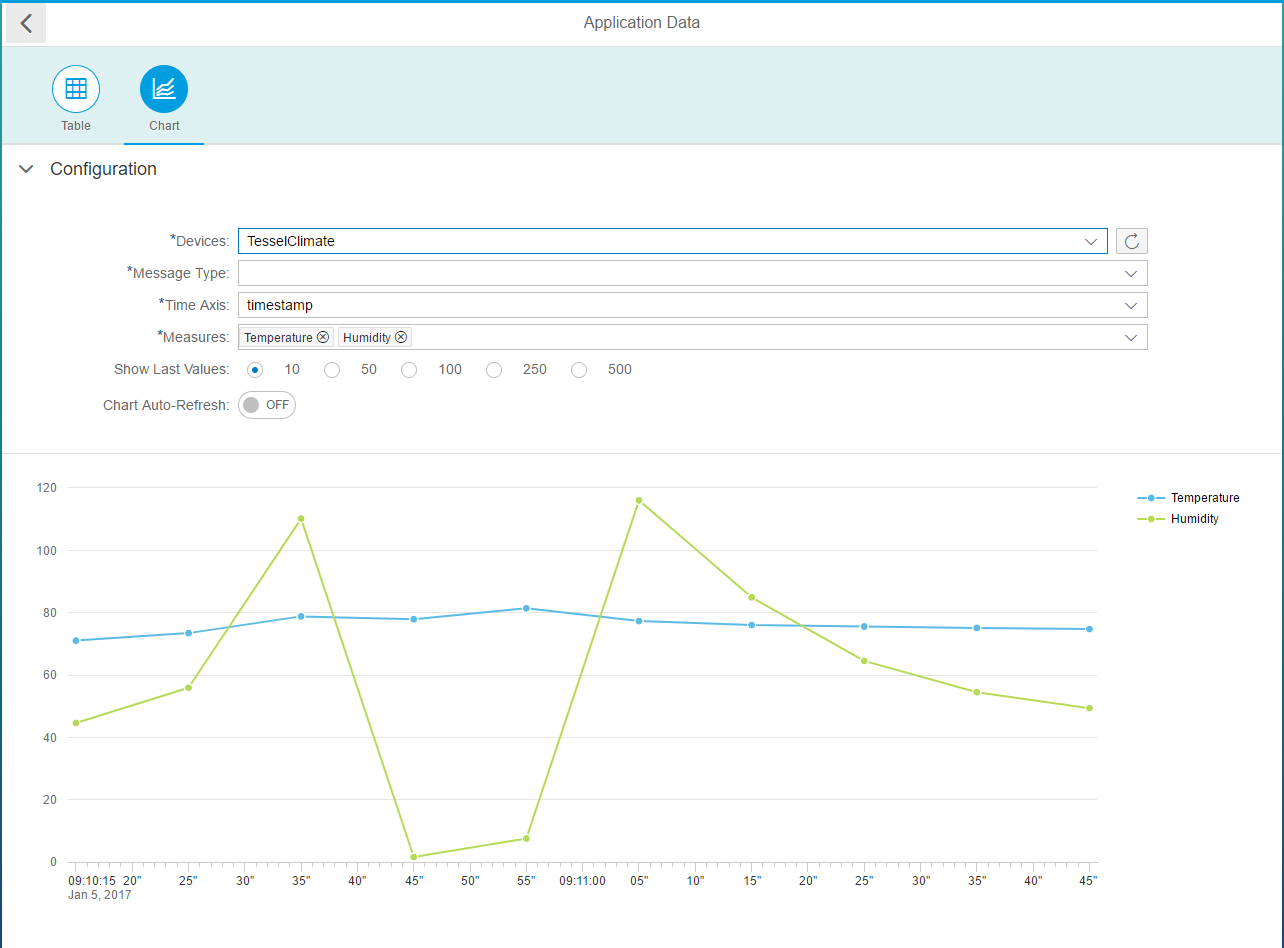
https://iotmms<your id>trial.hanatrial.ondemand.com/com.sap.iotservices.mms/v1/api/http/app.svc/NEO\_CT4FMF3FDW02NYG4UIPXV81BX.T\_IOT\_M0T0Y0P0E1

The host portion is your HCP account and the last part that looks like NEO\_CT4FMF3FDW02NYG4UIPXV81BX.T\_IOT\_M0T0Y0P0E1 is the <database schema>.<table> where the data is stored.

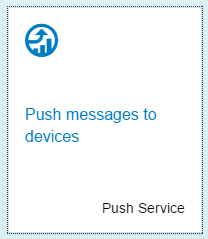
The data, including the test message you just sent, is displayed (Note the image below is formatted nicely because the browser has an extension installed to format XML code).



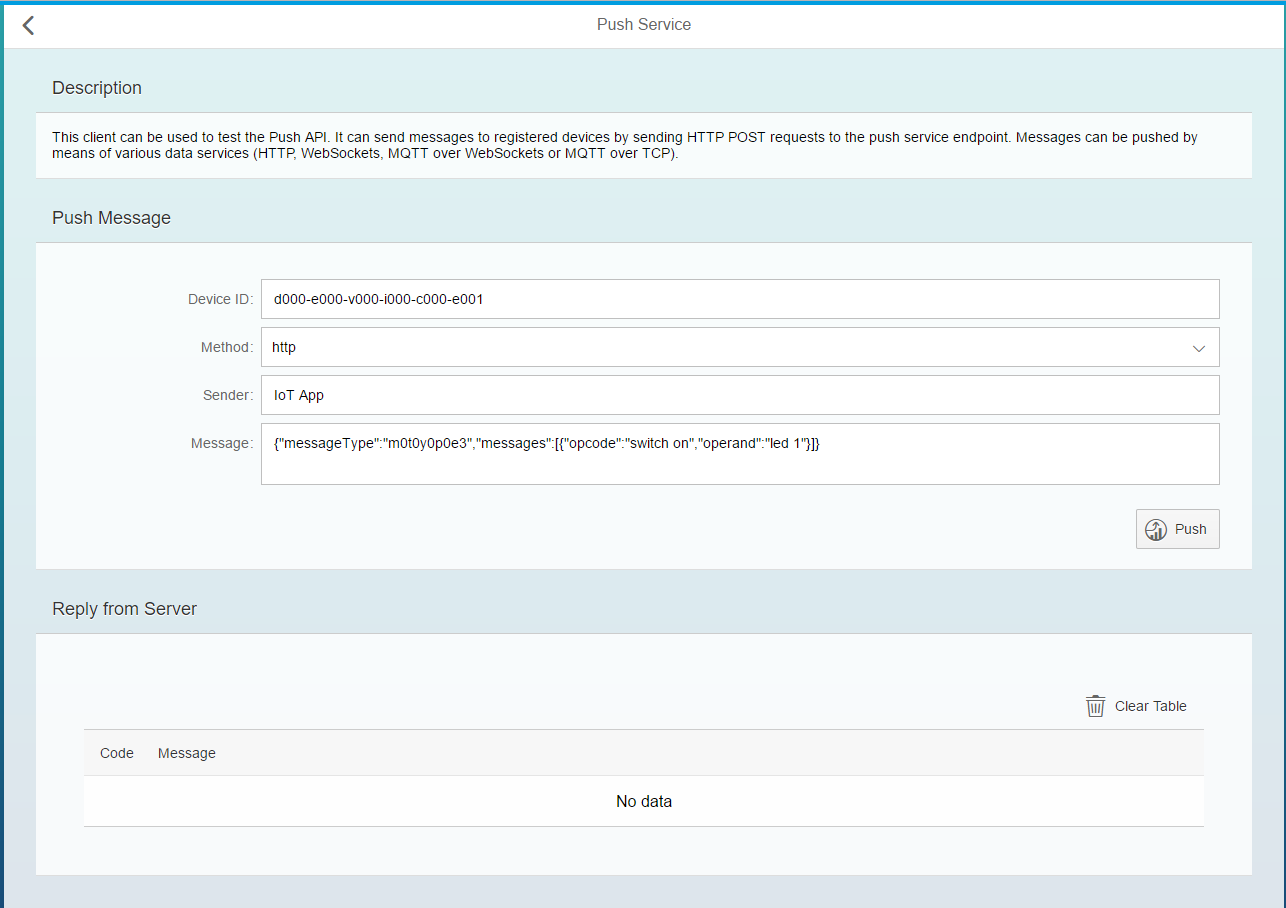
Another thing you can do with the data is to display it in a chart.



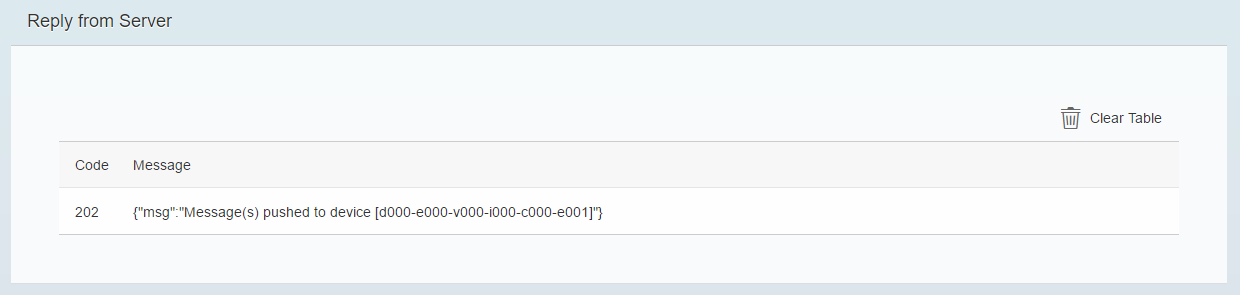
Return to the MMS Cockpit and click the Push Service tile.



This page has information about pushing messages to devices and includes documentation for how to send messages and a sample service.



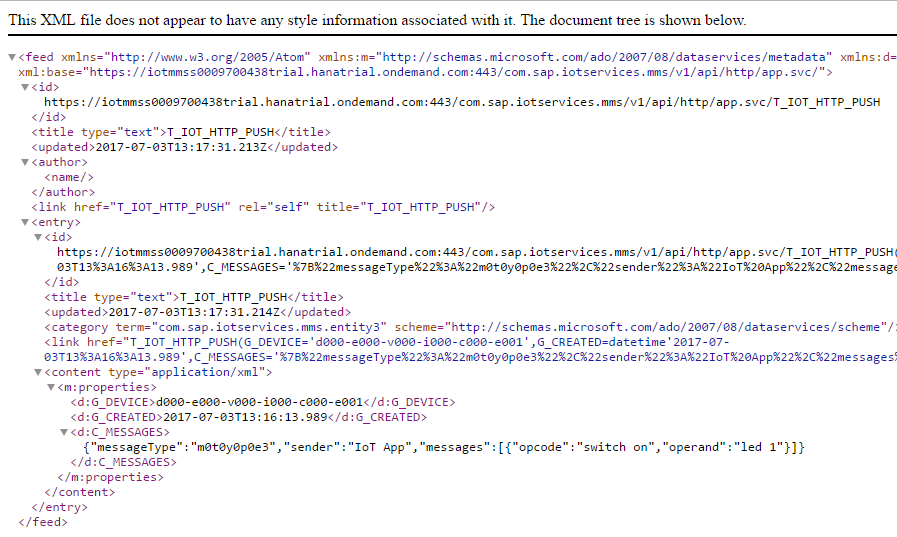
Click Push and you will see a reply from the server. The status 202 indicates your request to send the message was sent successfully.



Return to the service document. Note there is an endpoint for push messages.



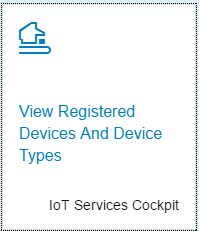
If you invoke this service you will see the push message you just sent.



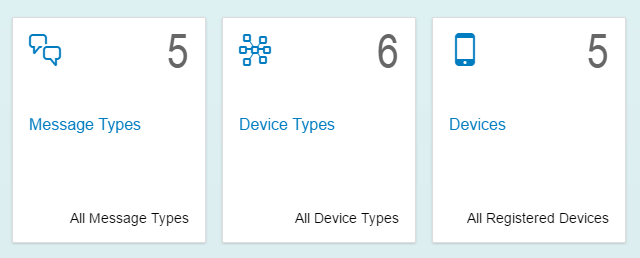
This message is stored here until it is sent to the device.

## Configure a Device and Message Type

Now we will configure a device and message type we can use it to see how the service works. Return to the IoT Service Cockpit.

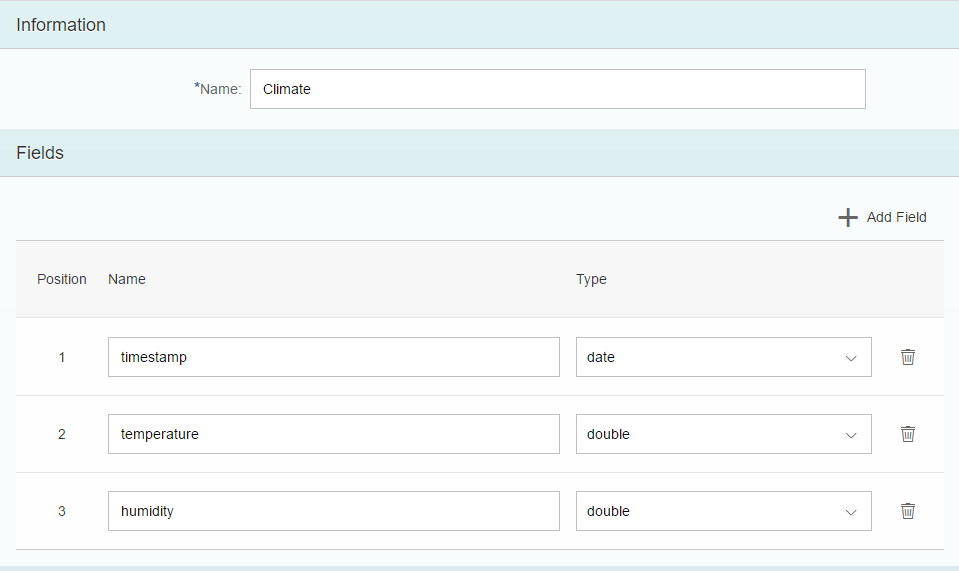


Note the three tiles at the top of the screen. Click the Message Types tile.

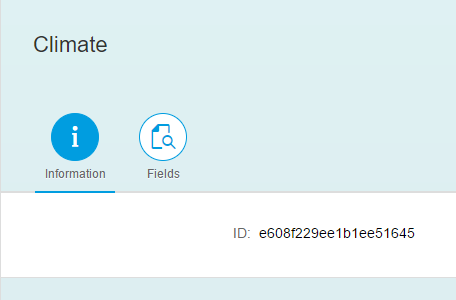


Click the plus sign to create a Message Type. Complete the Information section as shown below. The Name of the message is Climate.

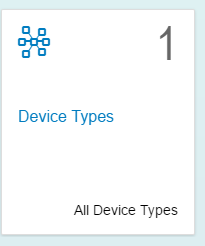
Use the **Add Field** button to add two new fields, temperature and humidity, both of type double.



Click **Create** to create the Message Type.

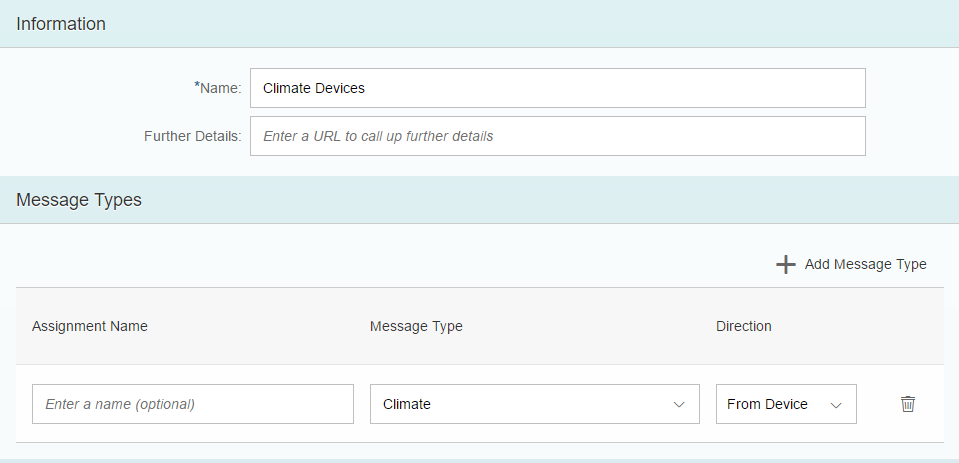


Return to the IoT Service Cockpit and click the Device Types tile.

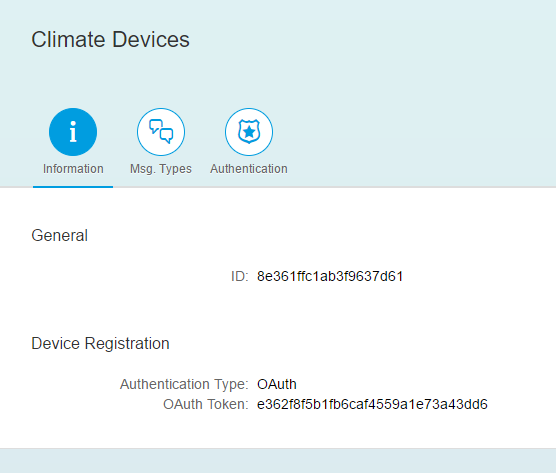


Click the plus sign at the bottom of the screen to create one.

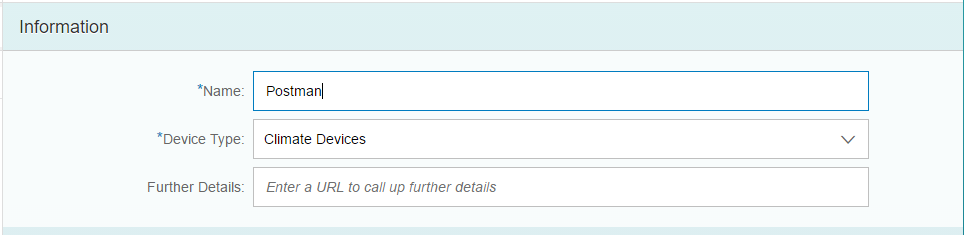
Enter Climate Devices for the Name and add the Climate Message Type. Click **Create**.



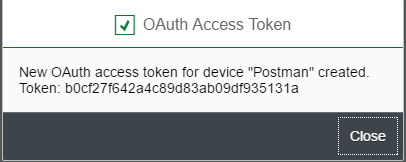
In a moment the Device Type is created. Note there is a Message Types tab which shows the Message Types associated with this type of device and an Authentication tab. We’ll use this later to generate a token so we can authenticate to this device type.



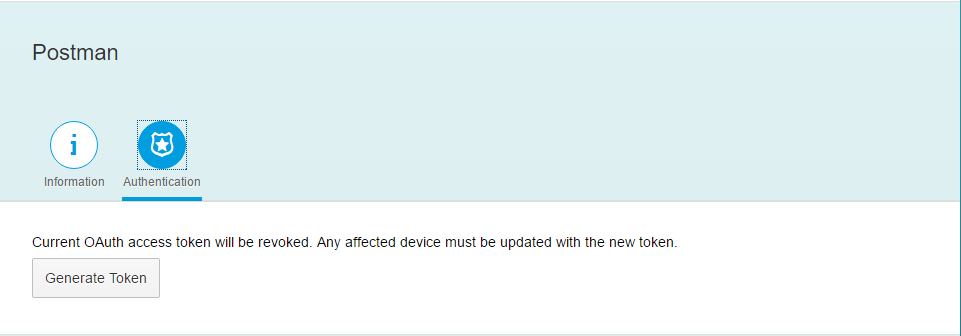
Click the back arrow and click the Devices tile then create a device. We will use a RESTful client called Postman to test this service so enter the name Postman then create the device.



When the device is created a dialog with the device’s OAuth Access Token. OAuth is an authentication technology that is more secure than user ids and passwords. You will need this token so you should copy and paste the token somewhere that is easy to reach.



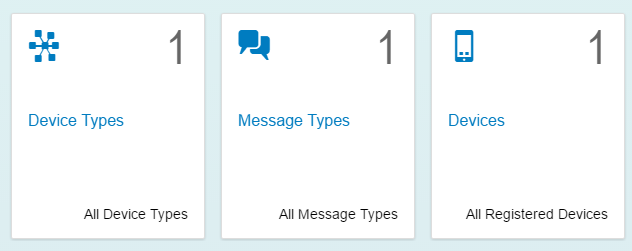
If you lose the token, you can generate a new one on the Authentication tab.



When you click Close on the dialog you will see the device ID. You should save this ID in the same place you saved the token.

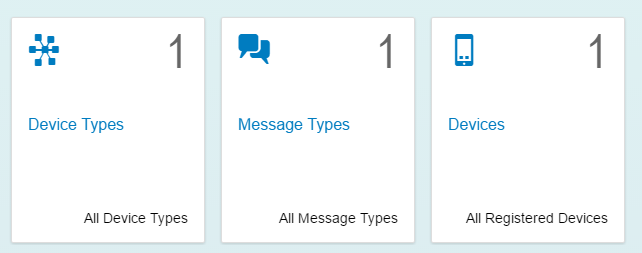


Back in the IoT Services Cockpit you will see that we have configured one Device Type, one Message Type and one Device.



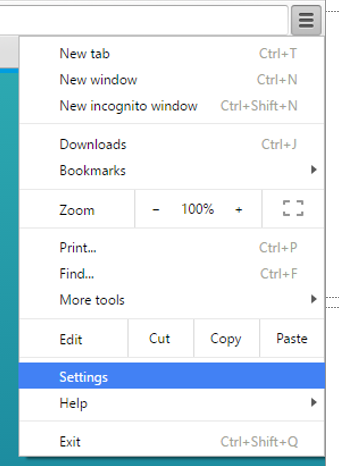
## Testing the IoT Service

In this case we will test the service using the device and message types created in the previous case.



### Install a RESTful Client

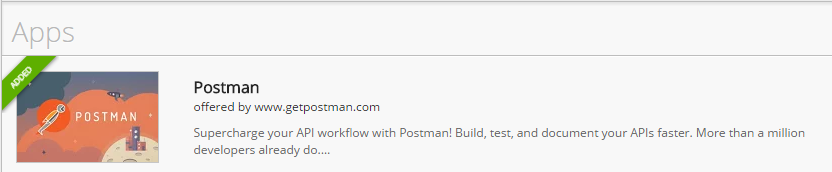
In order to test the service, you need to install a RESTful client. This case uses the Postman client which is a Chrome application. To install it, open Chrome then open the Settings page.



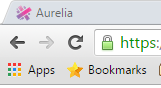
Select Extensions then select at the bottom of the page.

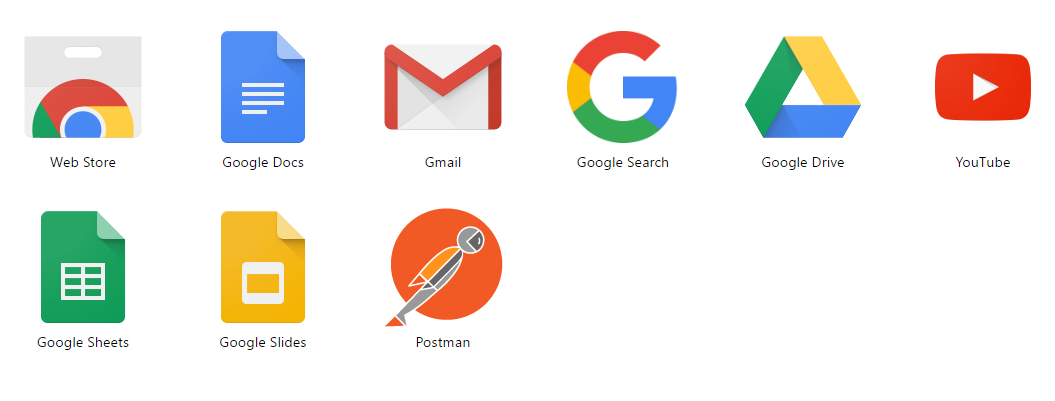


Search for Postman and then click Add to Chrome to install it.



The application is added to the Chrome application group which you can access through this icon.





Open the Postman app.

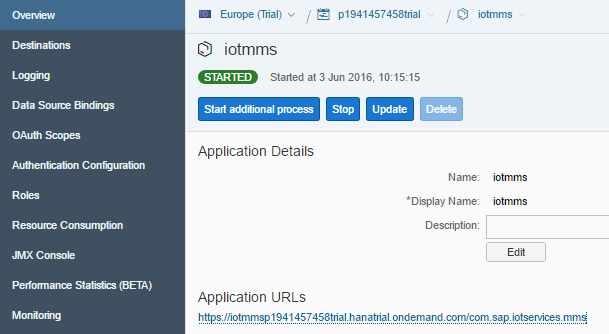
## Send a Message Synchronously

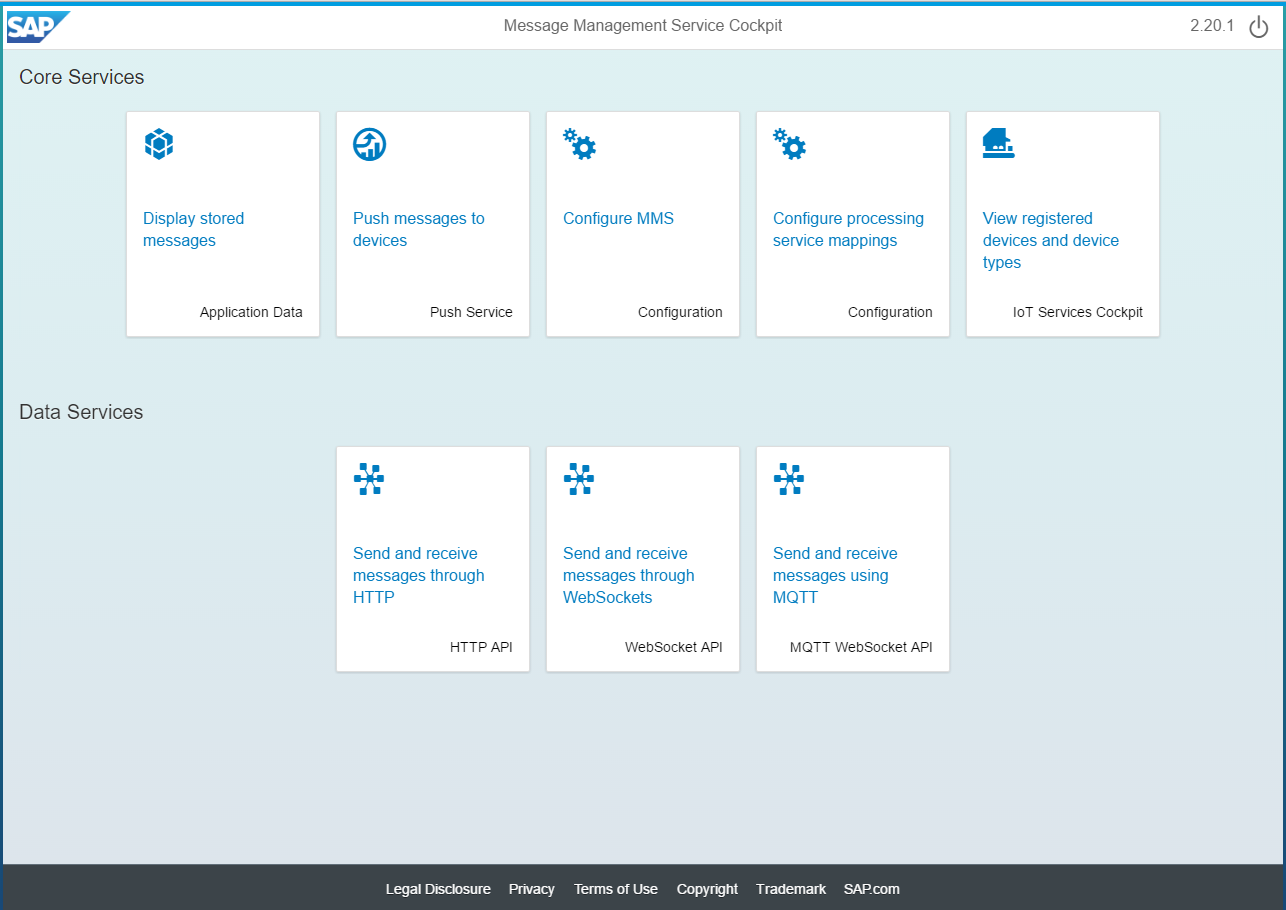
Now you will use Postman to send a message. To send the message you must:

1. Construct the URL using the Device ID.
2. Add an Authentication header with the device’s OAuth token.
3. Construct a message of the format of the Climate Message Type

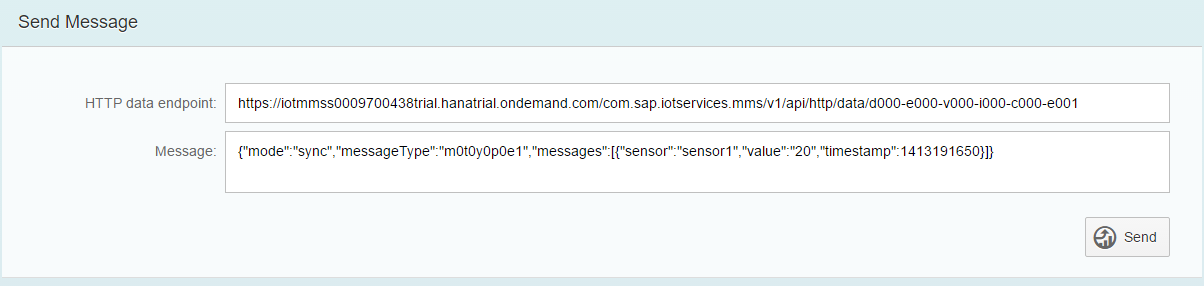
### Construct the URL

Open the MMS Cockpit. You can do this by using a bookmark or by logging into the HCP Cockpit, navigating to the dashboard for the IoT Services dashboard and clicking the Application URLs link.





Click the **HTTP API** tile. The HTTP data endpoint provides the template URI for sending messages from Postman.



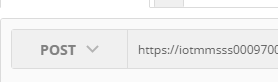
The URI has this format:

https://iotmmss<your id>trial.hanatrial.ondemand.com/com.sap.iotservices.mms/v1/api/http/data/<device ID>

Note that <your id> will be your HCP id. You can find the device ID on the Devices page. Replace d000-e000-v000-i000-c000-e001 with the Postman device’s ID. For example:

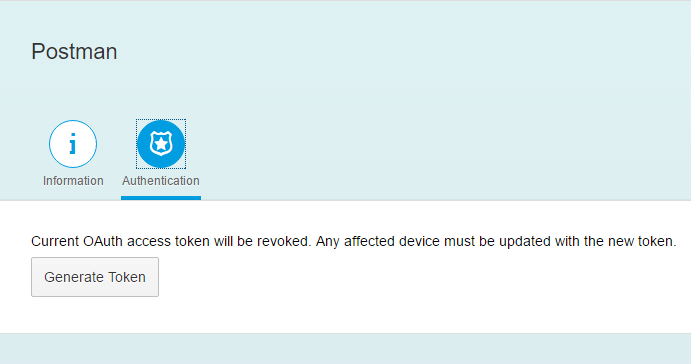
https://iotmmss<your id>trial.hanatrial.ondemand.com/com.sap.iotservices.mms/v1/api/http/data/39b3817f-2863-44bb-8e1c-c82f625b715b.

Paste this URL into Postman and change the method to POST.

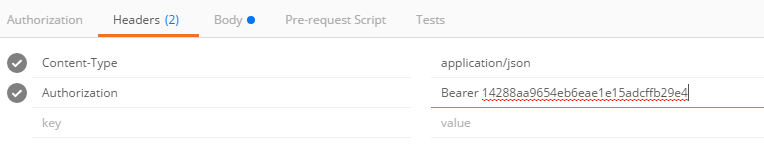


### Add an authentication header

If you saved the authorization token that was created when you created your device, locate it and copy it. If not, open the Authentication tab for the Postman device and click Generate Token.



Now, open the Headers tab in Postman and configure it as shown. You may have to add the Authorization key from the drop-down list. The value for the key is Bearer <your token>.



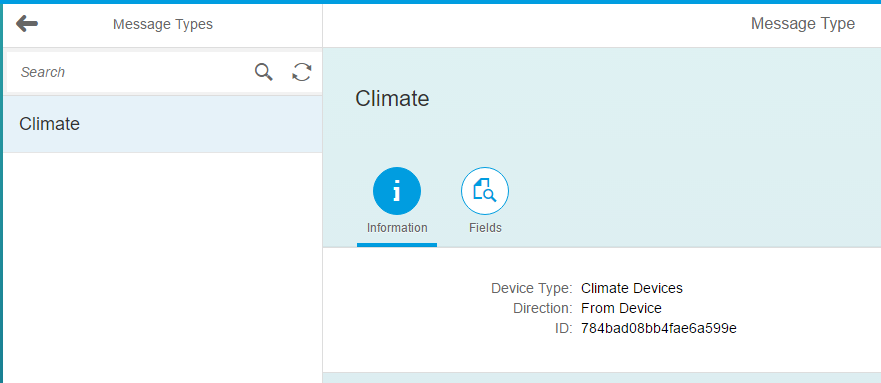
Also add the header Content-Type with a value of application/json.

### Add the message to the body

Finally, you have to add the message. Copy the sample message you send before with the test service.

|  |
| --- |
| {"mode":"sync","messageType":"m0t0y0p0e1","messages":[{"sensor":"sensor1","value":"20","timestamp":1413191650}]} |

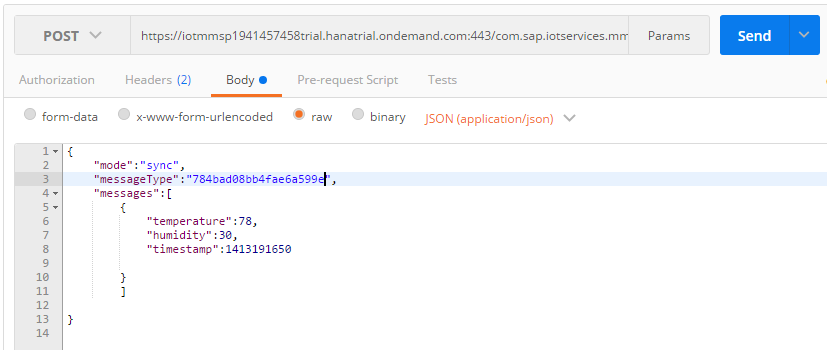
Replace the messageType value with the Climate Message Type ID



Replace the messages array with something like the example below;

|  |
| --- |
| {  "mode":"sync",  "messageType": "791b24bd6a2202ba3187",  "messages":[  {  "temperature":78,  "humidity":30,  "timestamp":1413191650  }  ]  } |

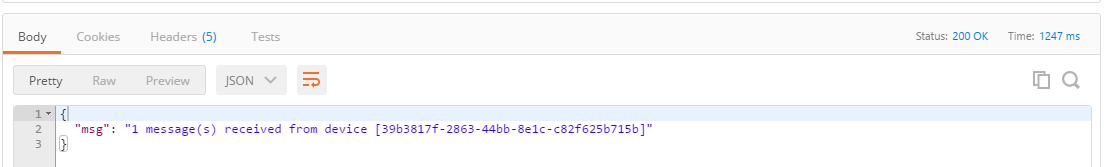
In Postman, select the Body tab and select raw. Paste the message into the body in Postman. Make sure the type is JSON in the dropdown list.



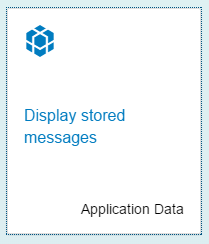
Once everything is entered, click **Send**. This message is sent synchronously which means that an application sending it will wait for the response before continuing. This probably isn’t the mode you want to use in the real world as one message can cause a long delay. The first message will probably return a response like that shown below. This indicates the response time is too long so the server sends back a status of 202 Accepted. This message simply acknowledges that the message was received.

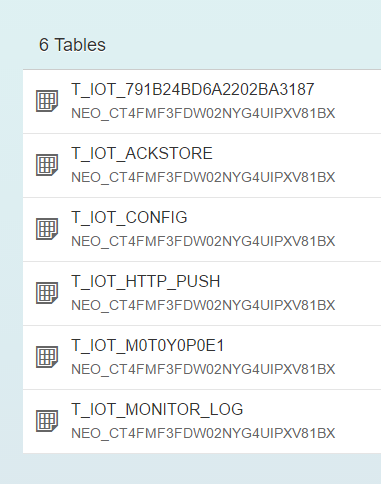


If the server handles the message in sufficient time, it will send back a response with a status of 200 OK. Click Send a few more times and you should get a response like this:

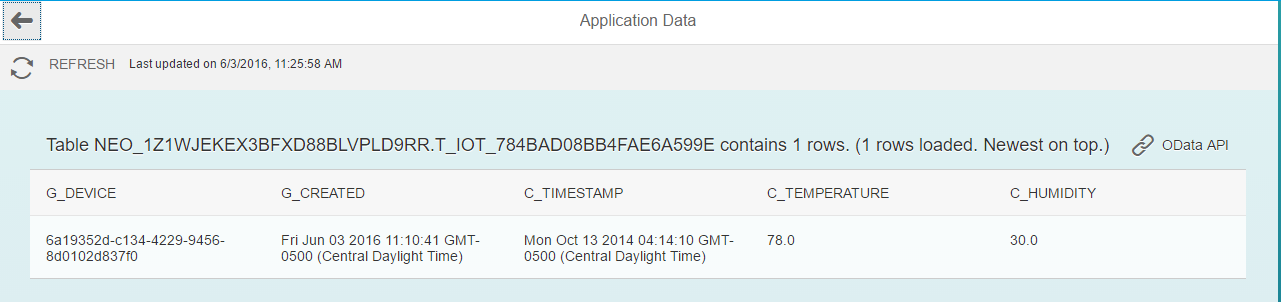


Return to the MMS Cockpit and click the Application Data tile.



You will see a new table for the message type that you sent (if you don’t see it your session may have been timed out. Refresh the browser window). 

Open the table and you will see the message. We’ll access this table in a later case using an application.



Click the OData API button to see the service output.



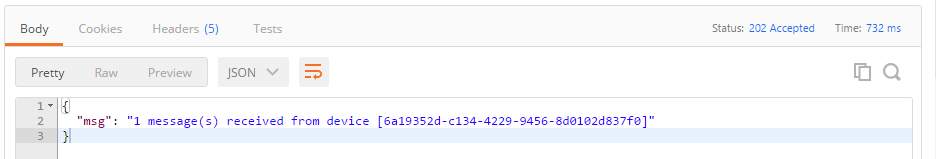
## Send an Asynchronous Message

In most cases you will want to send messages asynchronously. This allows the application sending the message to continue without waiting for a response. You can request an acknowledgement later if you wish.

In Postman modify the message as shown below. You can modify the parameter values if you wish.

|  |
| --- |
| {  "mode":"async",  "messageType":"<message type>",  "messages":[  {  "temperature":78,  "humidity":30,  "timestamp":1413191650    }  ]  } |

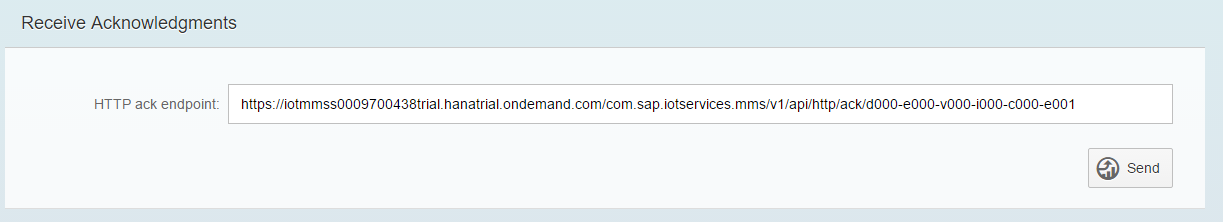
The result is a message with 202 Accepted status.



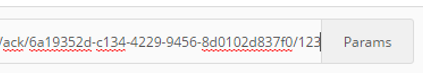
If you wish to request an acknowledgement that the message was received and processed, you can use a message like that shown below.

|  |
| --- |
| {  "mode":"async-ack",  "sequence": "123",  "messageType":"<message type>",  "messages":[  {  "temperature":78,  "humidity":30,  "timestamp":1413191650    }  ]  } |

The sequence value is used to identify this specific message so it would normally be generated each time a message is sent to the server. If you press send you will get the same response as the previous message. However, this time we can request an acknowledgement that the message was processed successfully. To find the URI for the ACK return to the HTTP API page.



Copy this URI and paste it into Postman then replace d000-e000-v000-i000-c000-e001 with your device ID.



Leave the type as POST and make sure you still have the authorization header configured. Click Send and you should receive a response like that shown below.

