**Locomotive Application**

The Locomotive client application consists of the following microservices .

The application has been deployed in student75 org and Training3 space:

1. **locomotive-simulator-service**: This microservice will randomly generate 3 main data set and call the locomotive-dataingestion-service which will in turn push the simulator data into the timeseries database. The data set consists of :
   1. RPM
   2. Torque
   3. Location
      1. Latitude
      2. Longitude

We have introduced two sets of dataset for 2 locomotives namely LOCOMOTIVE\_1 and LOCOMOTIVE\_2.

For each Locomotive, the dataset is being generated by the simulator and pushed to cloud which in turn calls the locomotive\_dataingestion-service to inject data into the TimeSeries Database every 2 seconds. URL to call the service is <http://locomotive-dataingestion-service.run.aws-usw02-pr.ice.predix.io/SaveTimeSeriesData> along with tenantId, clientId, and content json in the body.

URL to start the locomotive-simulator-service from Rest Client:

<http://locomotive-simulator-service.run.aws-usw02-pr.ice.predix.io/simulator/start>

header: Content-Type : application/json

If the Service is started properly it will return a response of “true” in the response body

URL to stop the locomotive-simulator-service:

<http://locomotive-simulator-service.run.aws-usw02-pr.ice.predix.io/simulator/stop>

header: Content-Type : application/json

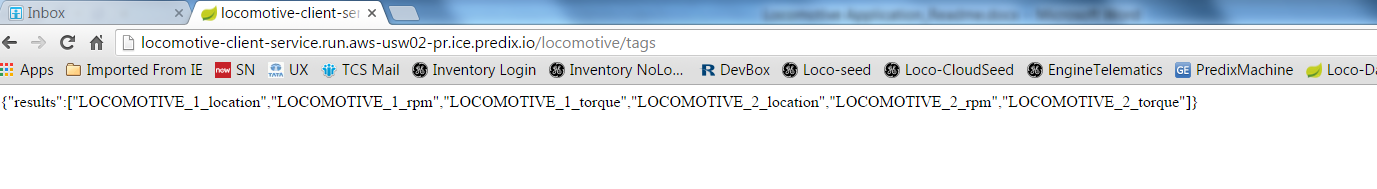
If the Service is stopped properly it will return a response of “false” in the response body

1. **locomotive-dataingestion-service**: This microservice will ingest data from the locomotive-simulator-service and push the data to the timeseries DB.

The service has exposed an endpoint /SaveTimeSeriesData which will accept an authorization token in the header and tenantId, clientId and content json as the request parameter and will inject the content json - rpm, torque and the location data into the TimeSeries Database

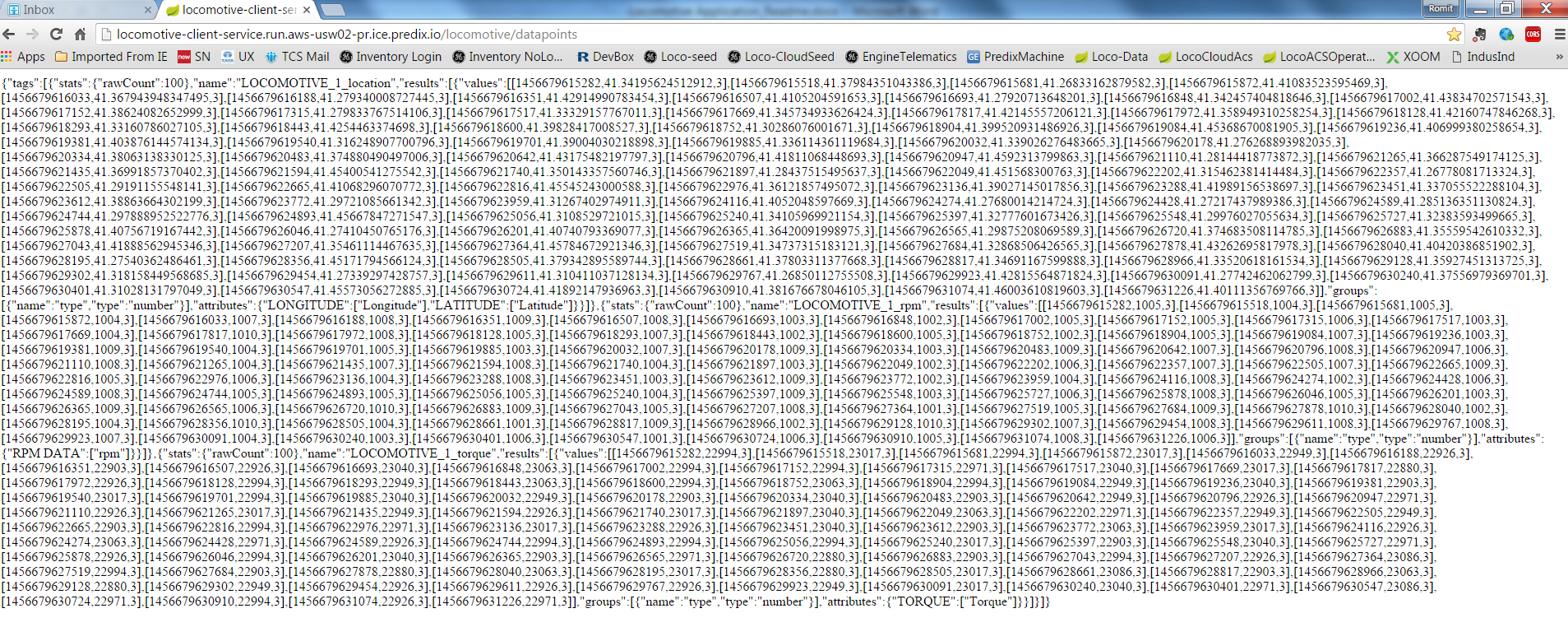
1. **locomotive-client-service**: This microservice exposes the following endpoints
   1. /locomotive/tags : This end point will fetch the tags under which the data is stored in the TimeSeries DB in the UI

<http://locomotive-client-service.run.aws-usw02-pr.ice.predix.io/locomotive/tags>



* 1. /locomotive/datapoints : This end point will fetch the latest 100 datapoints for the LOCOMOTIVE\_1 rpm, torque and location

<http://locomotive-client-service.run.aws-usw02-pr.ice.predix.io/locomotive/datapoints>

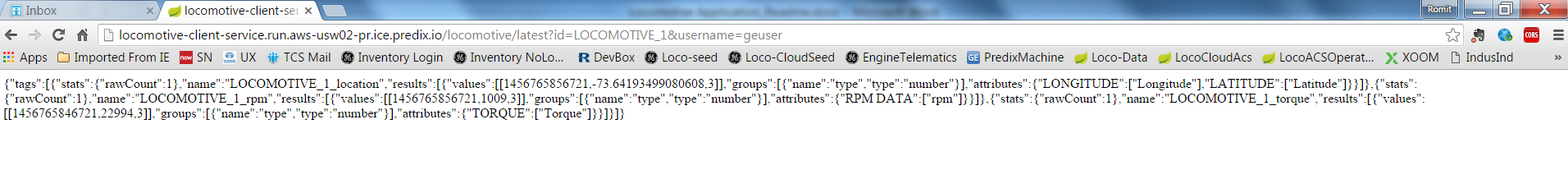


* 1. /locomotive/latest : This endpoint will fetch the latest data from the TS db

<http://locomotive-client-service.run.aws-usw02-pr.ice.predix.io/locomotive/latest?id=LOCOMOTIVE_1&username=geuser>

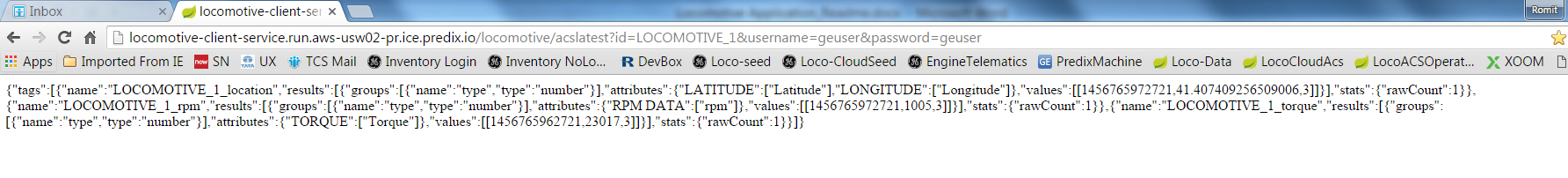
OR

<http://locomotive-client-service.run.aws-usw02-pr.ice.predix.io/locomotive/latest?id=LOCOMOTIVE_2&username=geuser>



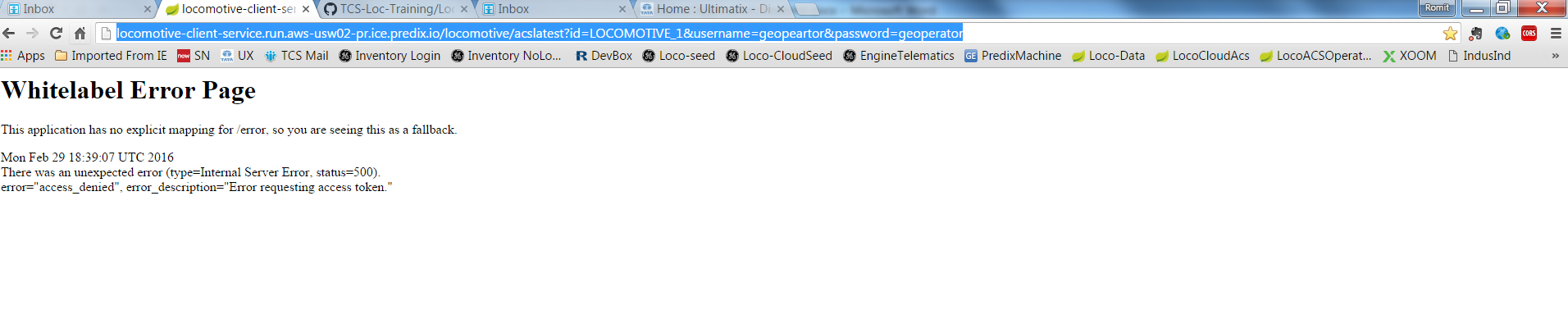
* 1. /locomotive/acslatest : This end point will fetch the latest record from the TS db provided the username has access to the resource after the ACS policy evaluation

<http://locomotive-client-service.run.aws-usw02-pr.ice.predix.io/locomotive/acslatest?id=LOCOMOTIVE_1&username=geuser&password=geuser>

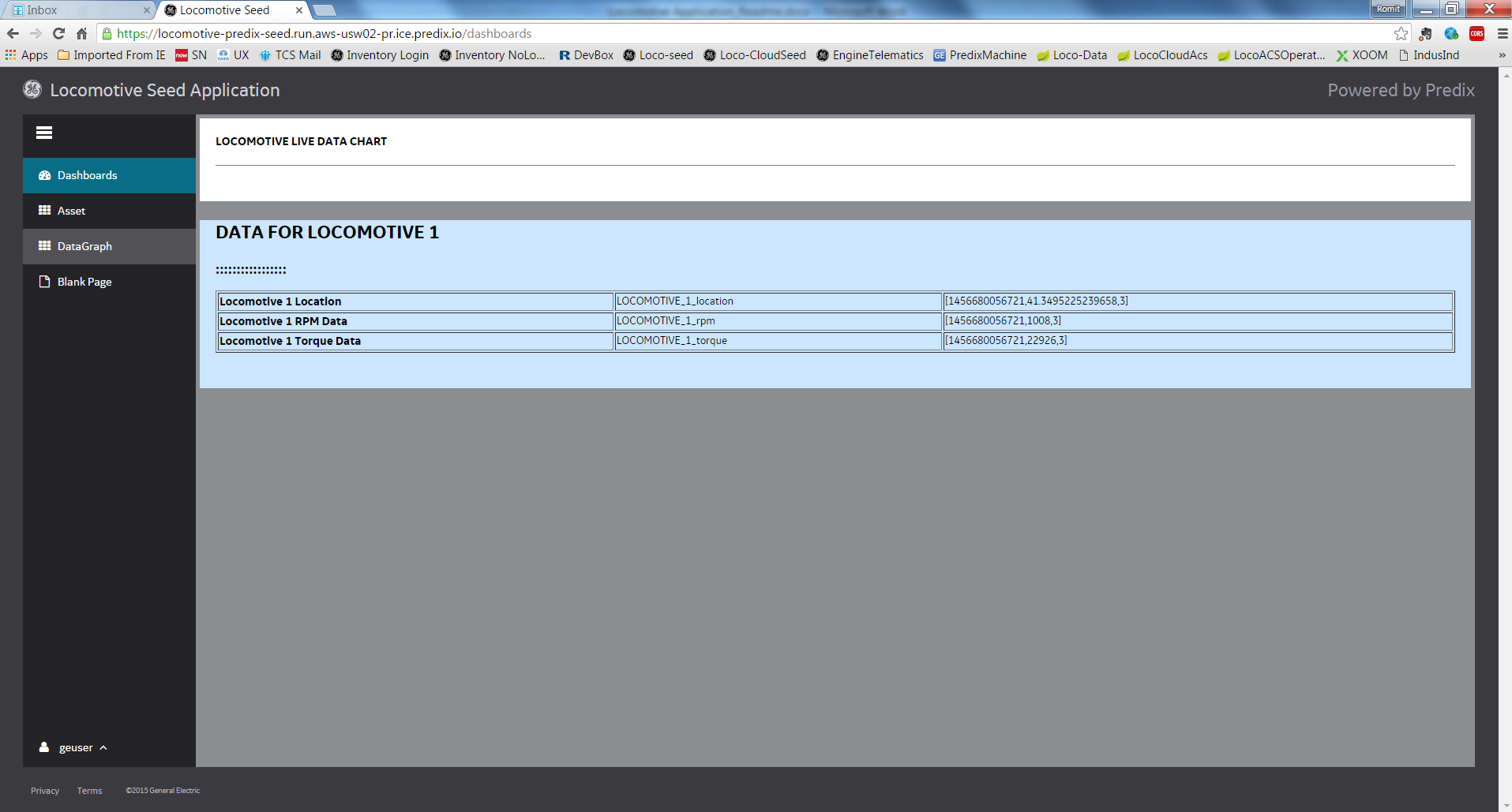


The user e.g. geoperator does not have access to GET latest request of the resource as per ACS policy. If we call the /acslatest with geoperator as user the result will be as shown below

<http://locomotive-client-service.run.aws-usw02-pr.ice.predix.io/locomotive/acslatest?id=LOCOMOTIVE_1&username=geopeartor&password=geoperator>

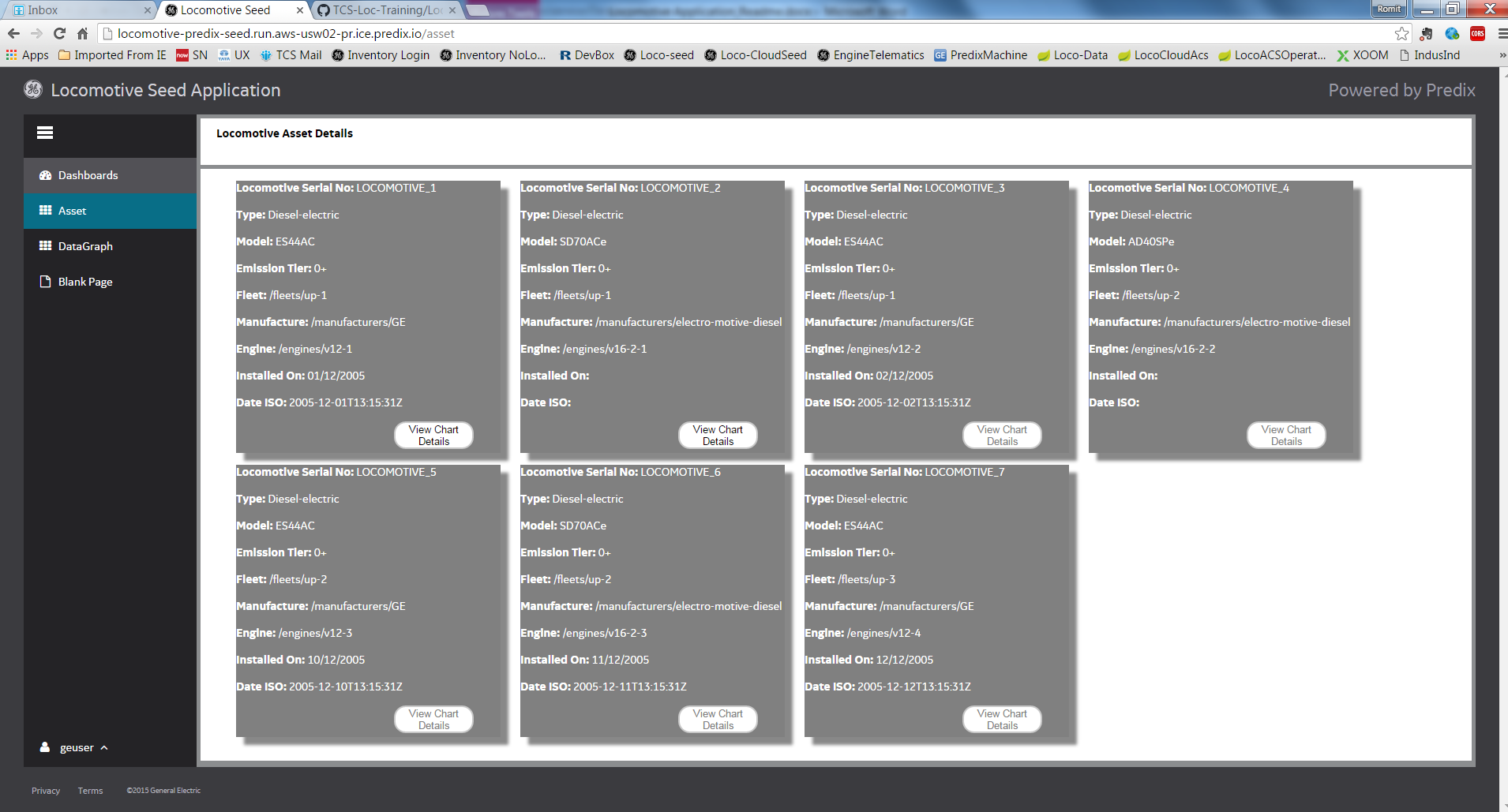


1. **locomotive-predix-seed** : This is the UI microservice which has 3 tabs
   1. **Dashboard** : This tab will show the 3 datapoints in a tabular format for LOCOMOTIVE\_1 which is basically hitting the datapoints URL to get the data



[Time, Data, quality]

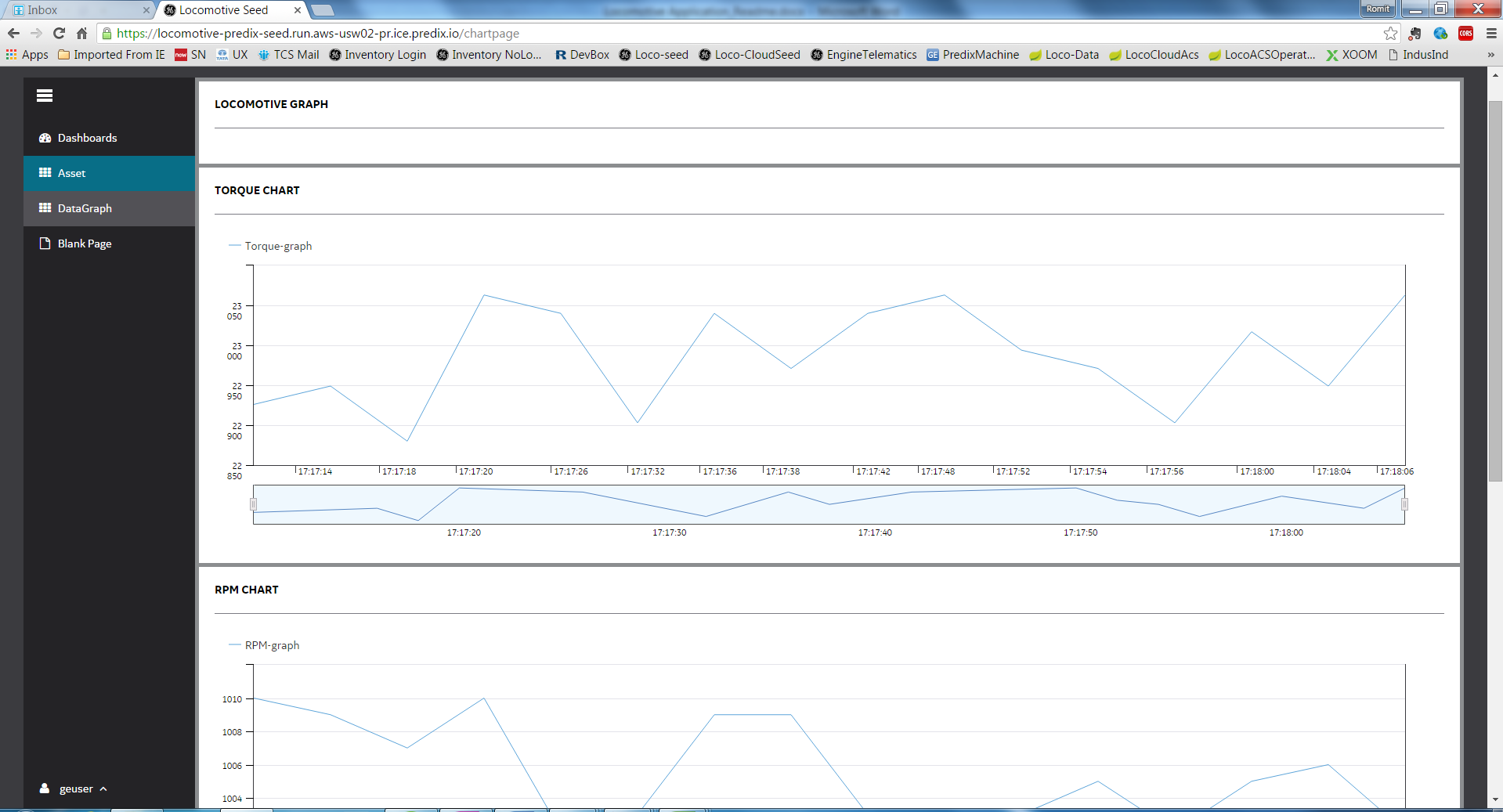
* 1. **Asset :** This tab will show all the assets in a layout format which has been loaded into the asset service (as of now 7 locomotive data has been loaded out of which 2 locomotive has live simulator data)



Click data for Locomotive\_21

Click data for Locomotive\_1\_1

* 1. DataGraph : This tab will show the live datachart for 3 types of data rpm, torque and location depending on the selection of Asset on the Asset tab (by default it will show data for Locomotive\_1)



**ACS:**

1. For ACS , we have created 2 users : geuser/geuser who is the admin and geoperator/geopeartor.

geuser has all the access for GET/POST/PUT/DELETE where as the geoperator does not have the privilege to GET/PUT/POST/DELETE

All the json policy has been put in the ACS\_Setup folder

URL to test the geuser – PERMIT as per policy

[**http://locomotive-client-service.run.aws-usw02-pr.ice.predix.io/locomotive/acslatest?id=LOCOMOTIVE\_1&username=geuser&password=geuser**](http://locomotive-client-service.run.aws-usw02-pr.ice.predix.io/locomotive/acslatest?id=LOCOMOTIVE_1&username=geuser&password=geuser)

URL to test the geoperator –DENY as per policy

[**http://locomotive-client-service.run.aws-usw02-pr.ice.predix.io/locomotive/acslatest?id=LOCOMOTIVE\_1&username=geopeartor&password=geoperator**](http://locomotive-client-service.run.aws-usw02-pr.ice.predix.io/locomotive/acslatest?id=LOCOMOTIVE_1&username=geopeartor&password=geoperator)

**UAA:**

Setting up UAA service is done via all the commands listed in the file ‘UAA Set Up - Training.txt’. All the steps have been explained in the document sequentially.



**P.S.:** For any query about this application please feel free to reach out to Romit Saha ([romit.saha@tcs.com](mailto:romit.saha@tcs.com)) and Preetha Raman ([preetha.raman@tcs.com](mailto:preetha.raman@tcs.com) )