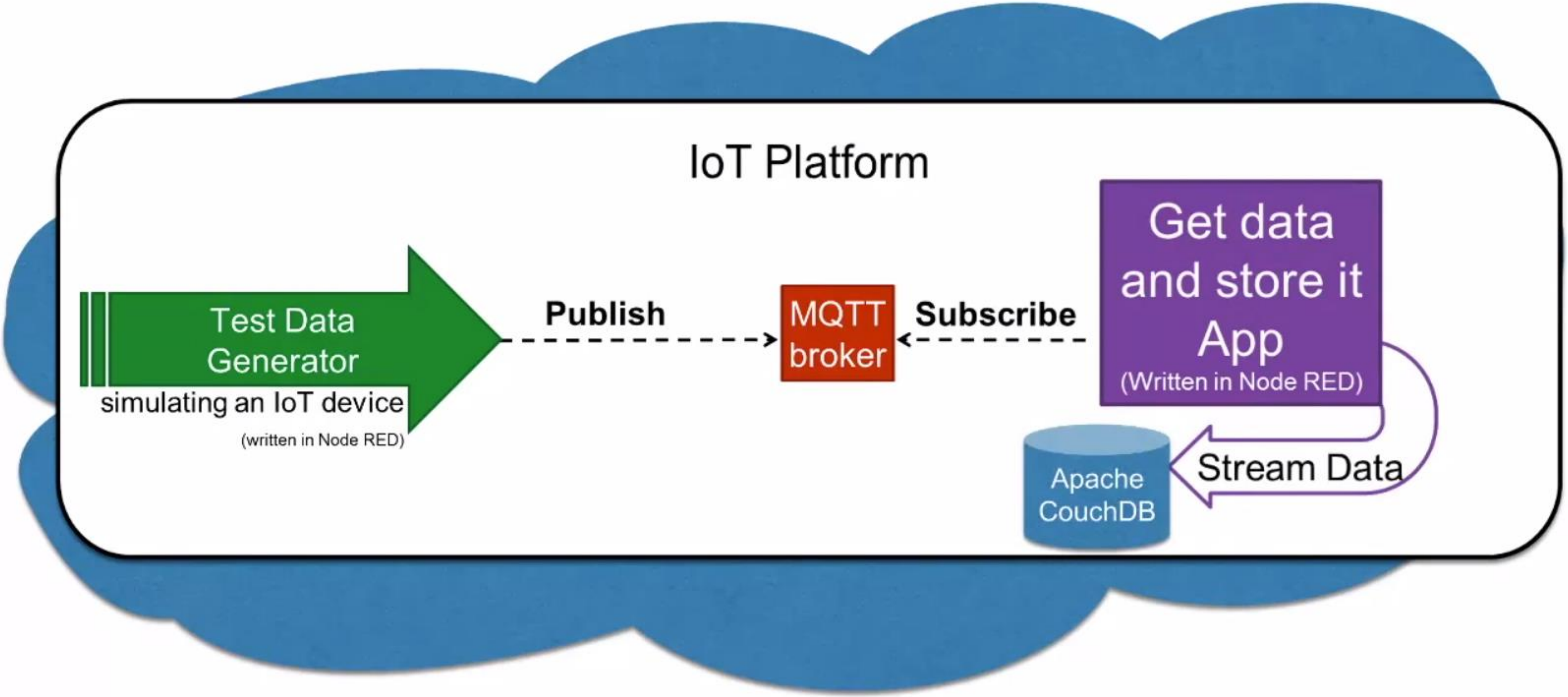
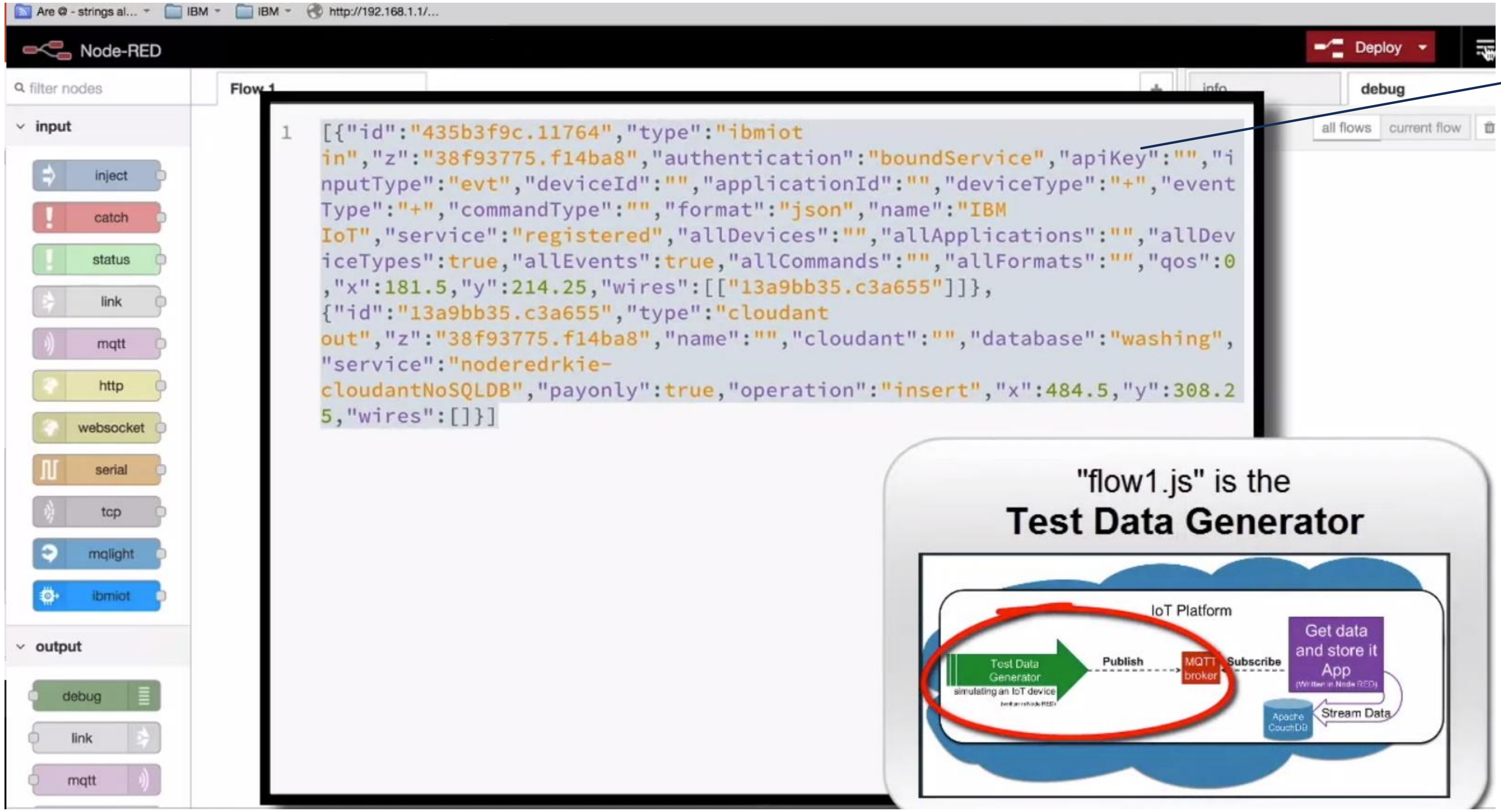
TEST DATA GENERATION

End-to-end scenario



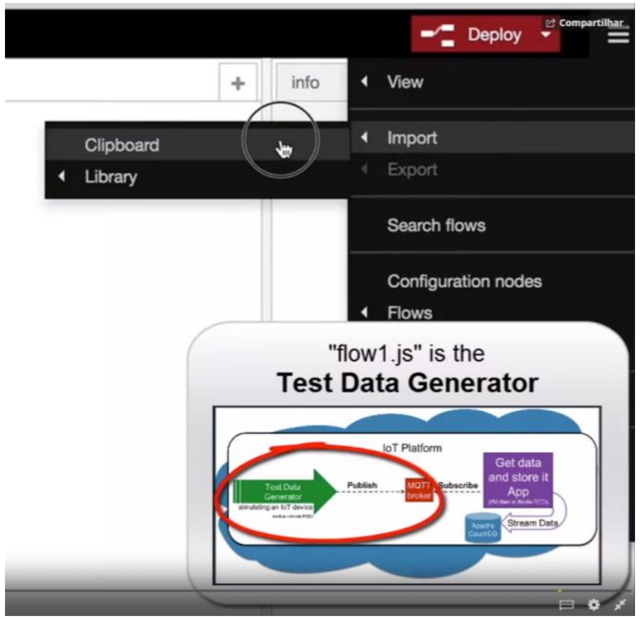
FLOW 1



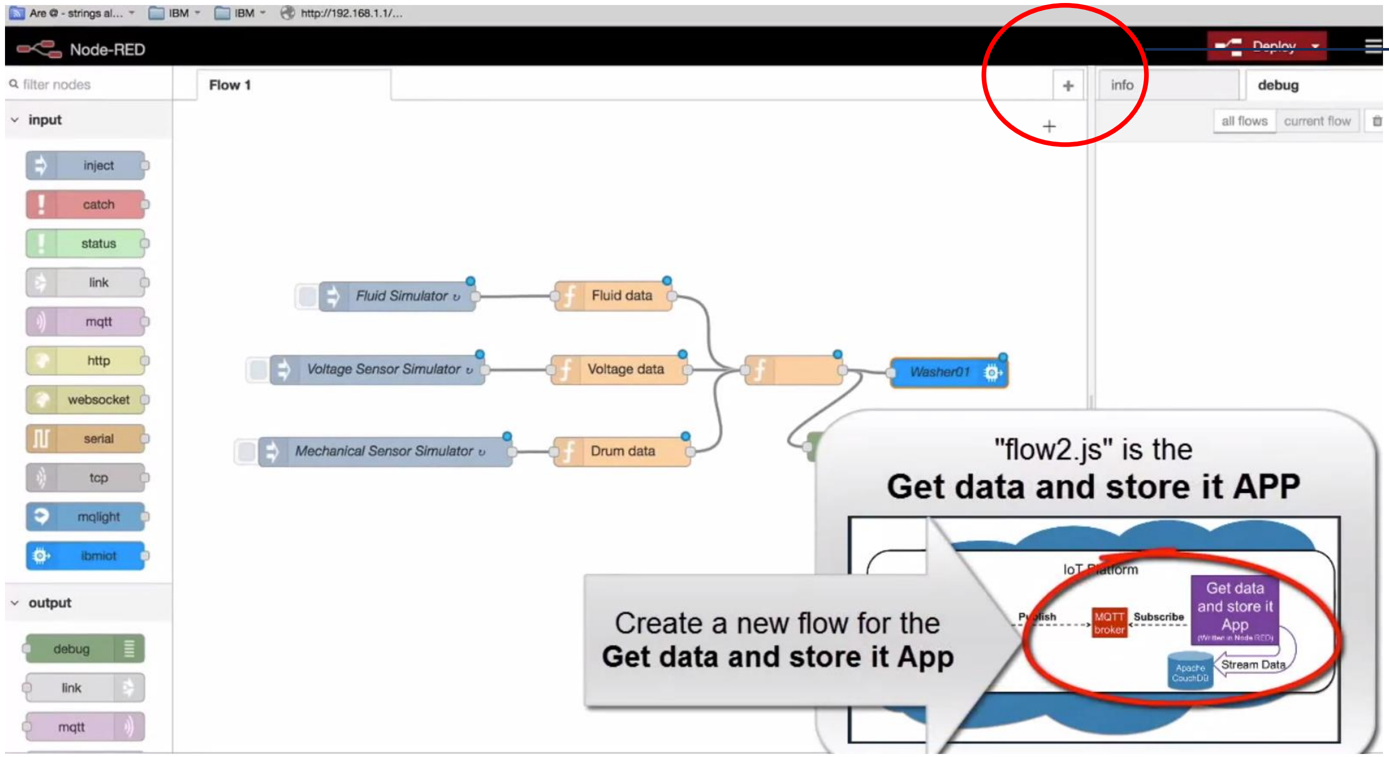
```
[{"id":"435b3f9c.11764","type":"ibmiot
in","z":"38f93775.f14ba8","authentication":"boundService","apiKey":"","i
nputType":"evt","deviceId":"","applicationId":"","deviceType":"+","event
Type":"+","commandType":"","format":"json","name":"IBM
IoT","service":"registered","allDevices":"","allApplications":"","allDev
iceTypes":true,"allEvents":true,"allCommands":"","allFormats":"","qos":0
,"x":181.5,"y":214.25,"wires":[["13a9bb35.c3a655"]]},
{"id":"13a9bb35.c3a655","type":"cloudant
out","z":"38f93775.f14ba8","name":"","cloudant":"","database":"washing",
"service":"noderedrkie-
cloudantNoSQLDB","payonly":true,"operation":"insert","x":484.5,"y":308.2
5,"wires":[]}]
```

Open in a text editor

Past here:

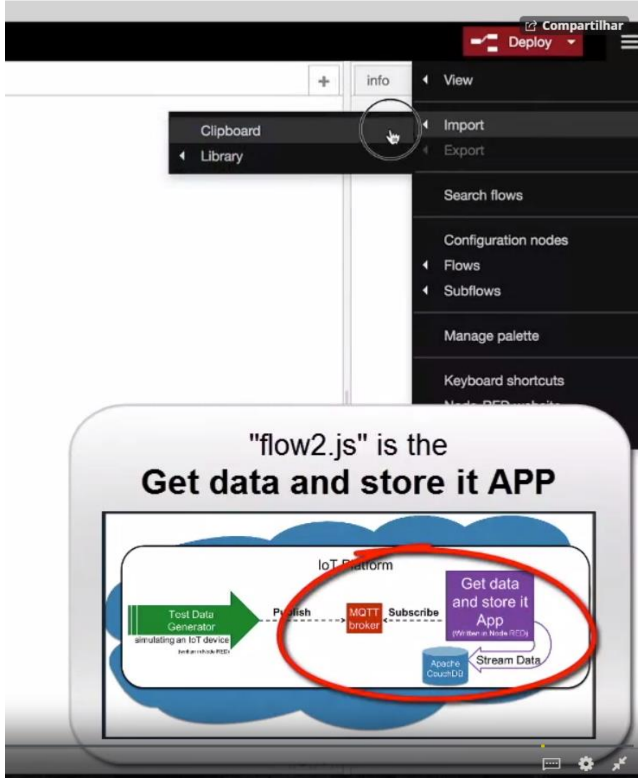


FLOW 2

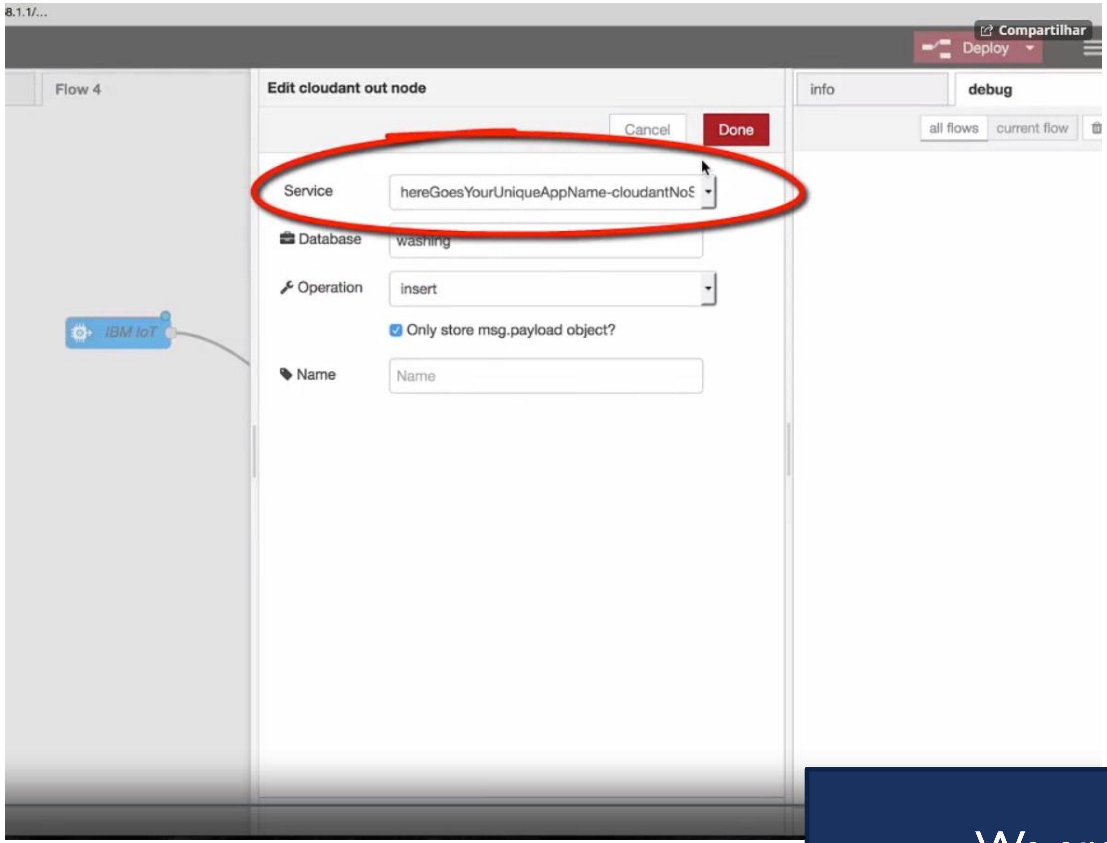


Click here to create a new panel for a new flow

Again:



IBM IOT AND WASHING (FLOW2)



Edit cloudant out node

Cancel Done

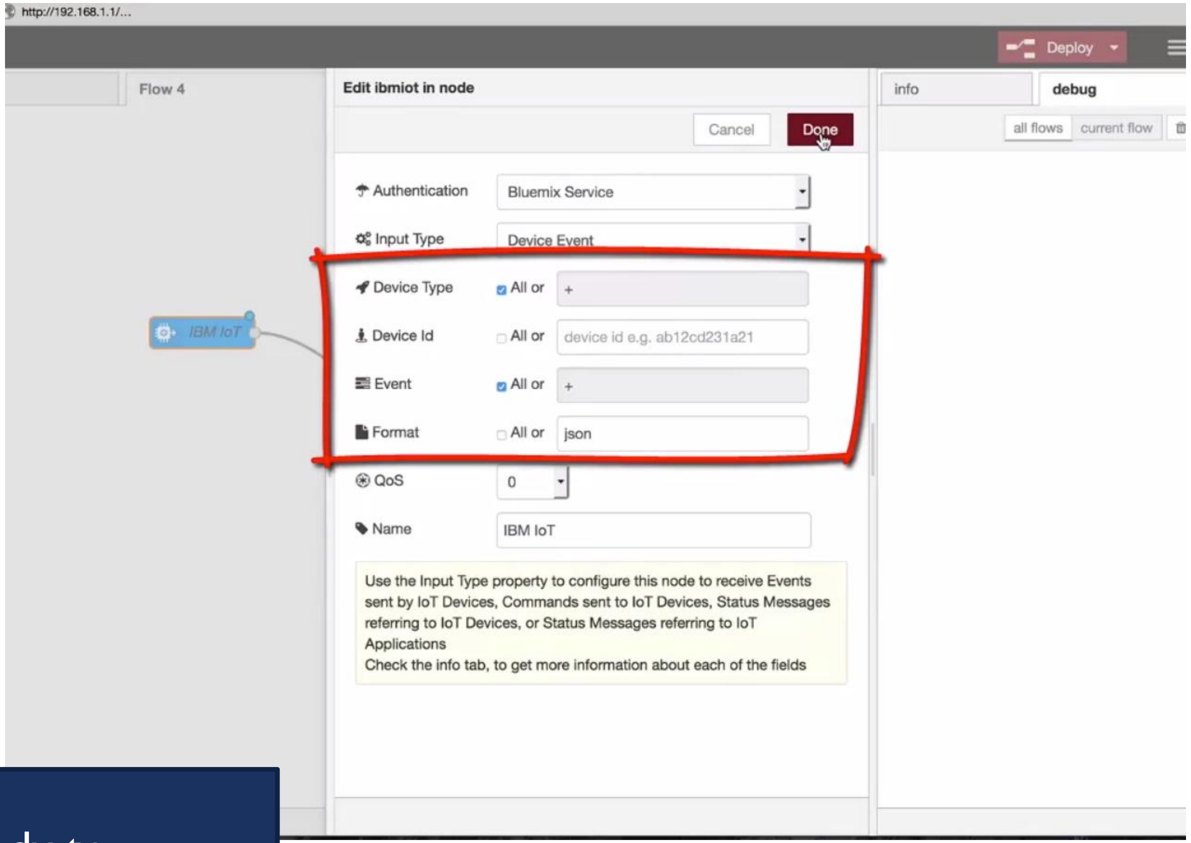
Service: hereGoesYourUniqueAppName-cloudantNoS

Database: washing

Operation: insert

☑ Only store msg.payload object?

Name: Name



Edit ibmiot in node

Cancel Done

Authentication: Bluemix Service

Input Type: Device Event

Device Type: ☑ All or +

Device Id: □ All or device id e.g. ab12cd231a21

Event: ☑ All or +

Format: □ All or json

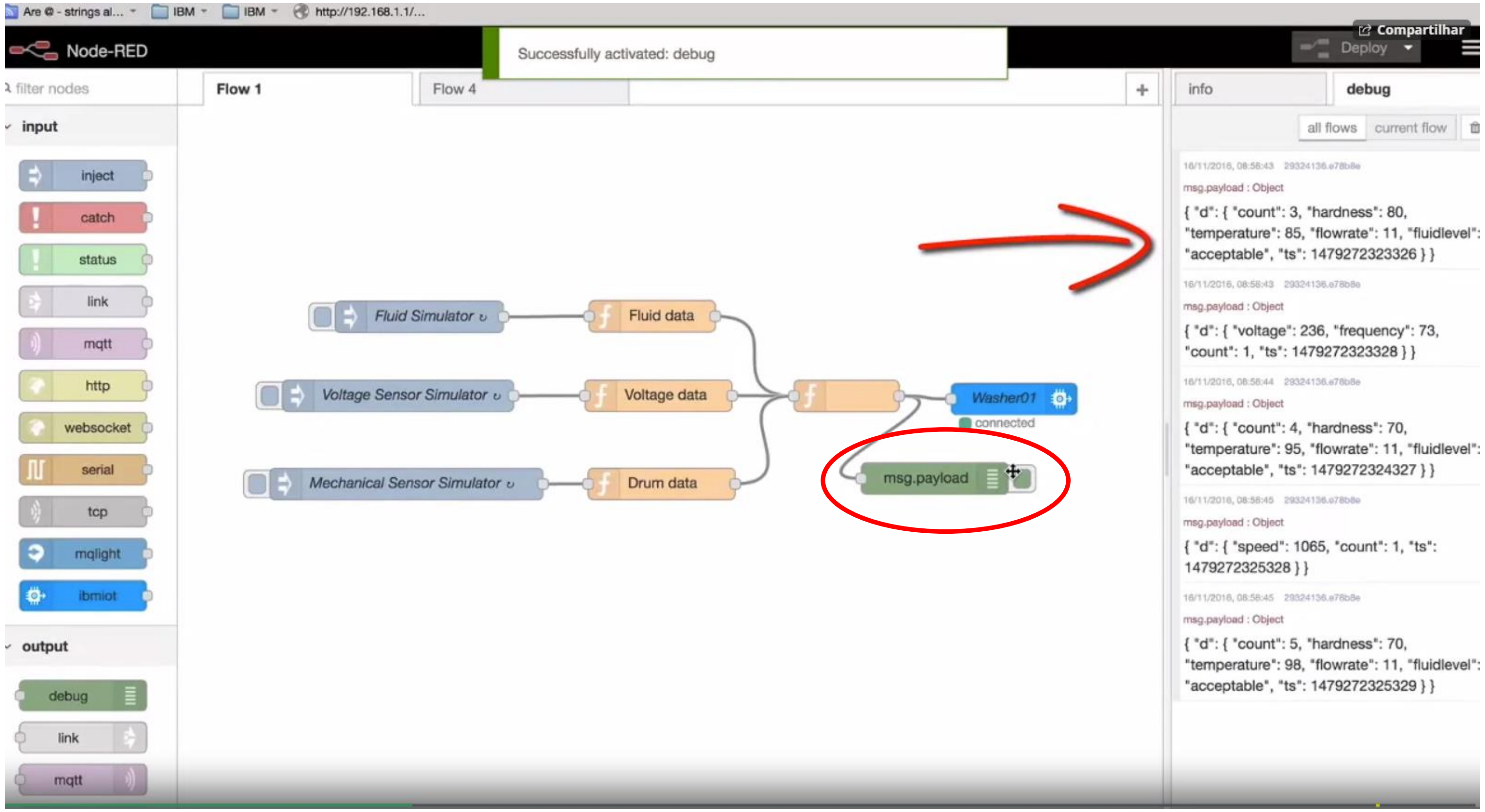
QoS: 0

Name: IBM IoT

Use the Input Type property to configure this node to receive Events sent by IoT Devices, Commands sent to IoT Devices, Status Messages referring to IoT Devices, or Status Messages referring to IoT Applications
Check the info tab, to get more information about each of the fields

We are ready to
Generate test data!

CLICK ON MSG.PAYLOAD



Successfully activated: debug

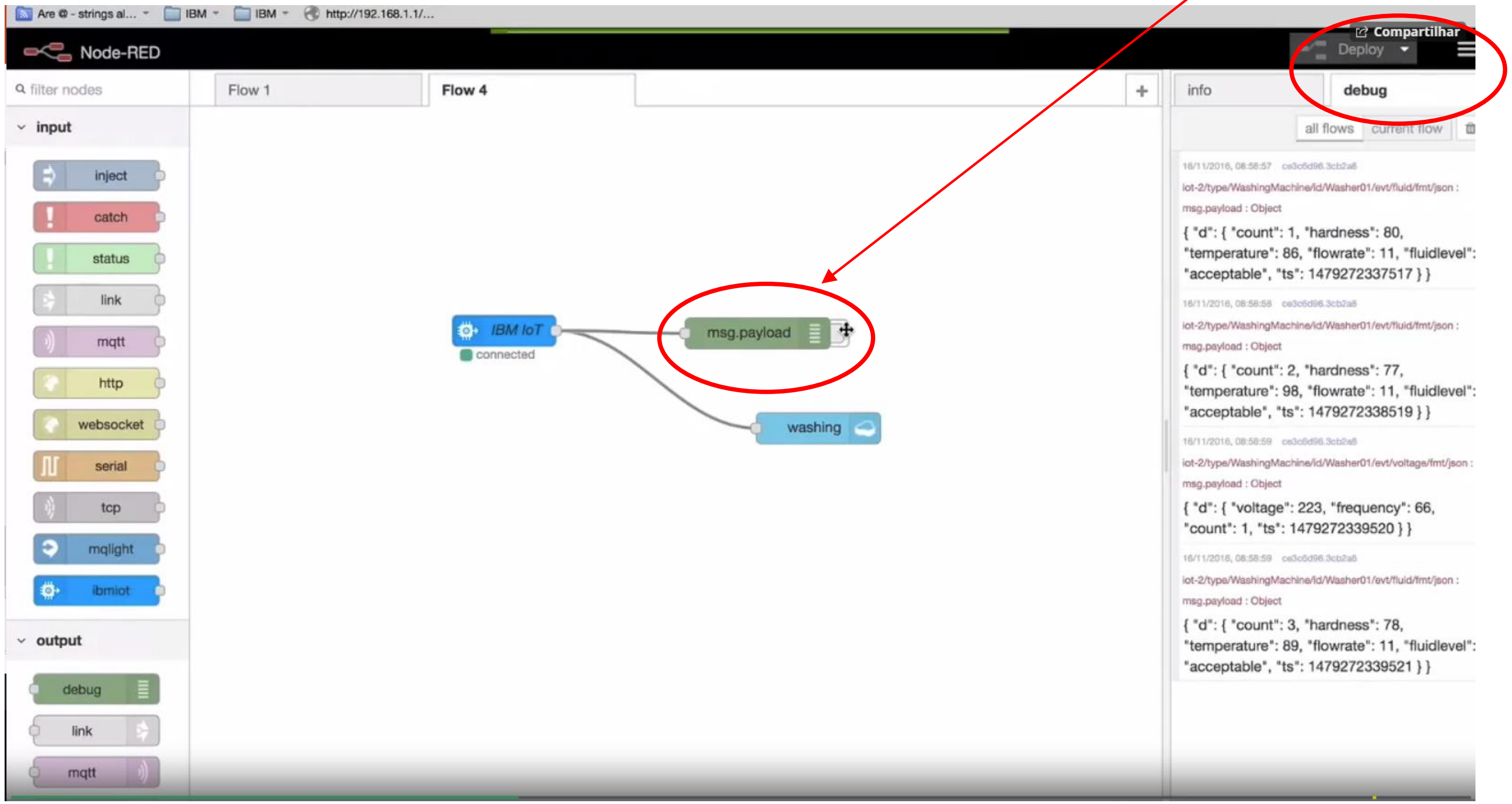
```
16/11/2016, 08:58:43   29324136.e78b8e
msg.payload : Object
{ "d": { "count": 3, "hardness": 80,
"temperature": 85, "flowrate": 11, "fluidlevel":
"acceptable", "ts": 1479272323326 } }

16/11/2016, 08:58:43   29324136.e78b8e
msg.payload : Object
{ "d": { "voltage": 236, "frequency": 73,
"count": 1, "ts": 1479272323328 } }

16/11/2016, 08:58:44   29324136.e78b8e
msg.payload : Object
{ "d": { "count": 4, "hardness": 70,
"temperature": 95, "flowrate": 11, "fluidlevel":
"acceptable", "ts": 1479272324327 } }

16/11/2016, 08:58:45   29324136.e78b8e
msg.payload : Object
{ "d": { "speed": 1065, "count": 1, "ts":
1479272325328 } }

16/11/2016, 08:58:45   29324136.e78b8e
msg.payload : Object
{ "d": { "count": 5, "hardness": 70,
"temperature": 98, "flowrate": 11, "fluidlevel":
"acceptable", "ts": 1479272325329 } }
```

Once we activate the debug node,
we will see test data generated by the test data generator.
We'll have a look at this later.
But basically, it is important to notice that this particular flow normally doesn't run in the cloud, but on an IoT device or gateway reading raw sensor data
directly from the built-in sensors and transmitting those directly to the cloud.
This is hypothetical sensor data coming from a washing machine.

Let's deactivate the debug node and add one on the cloud side.

COPY FOR THE FLOW 1 TO THE FLOW 2 (MSG.PAYLOAD)

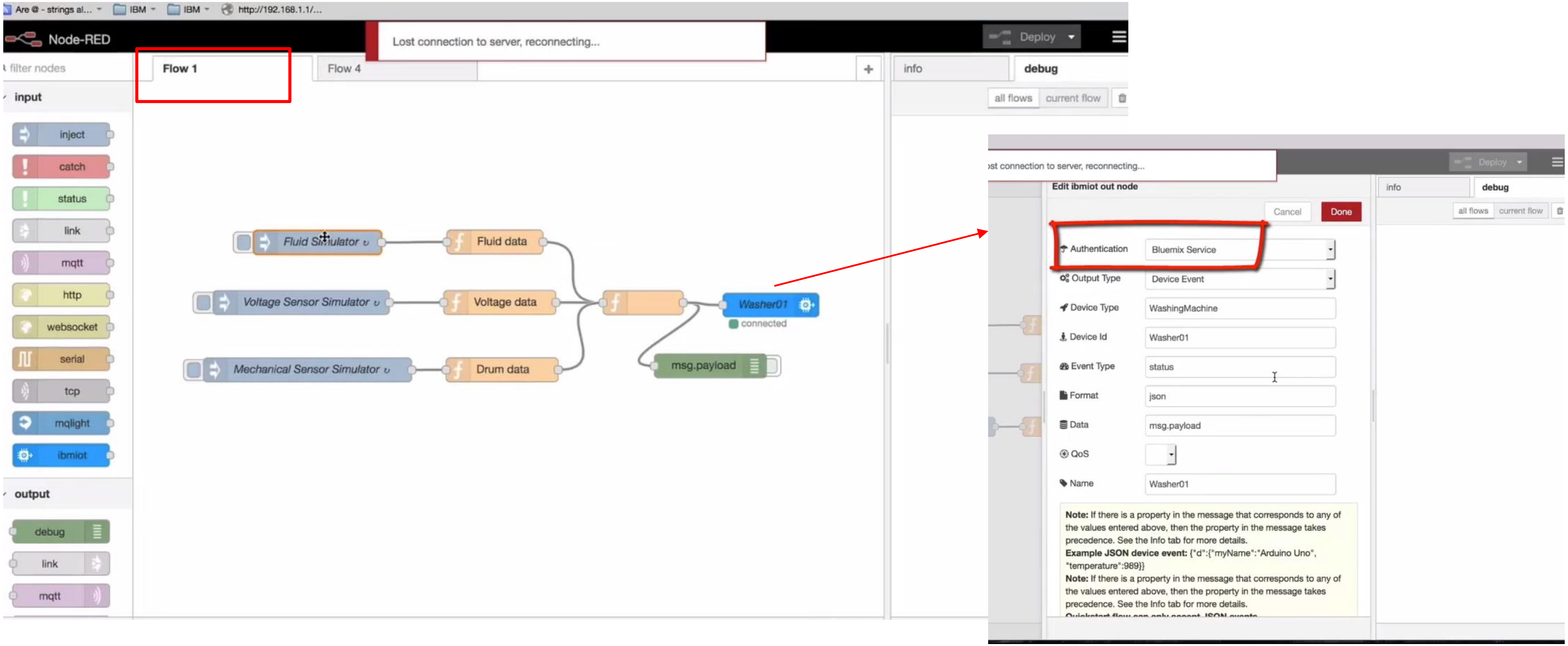
Click here to start:



```
16/11/2016, 08:58:57   ce3c6d96.3cb2a8
iot-2/type/WashingMachine/id/Washer01/evt/fluid/fmt/json :
msg.payload : Object
{ "d": { "count": 1, "hardness": 80,
"temperature": 86, "flowrate": 11, "fluidlevel":
"acceptable", "ts": 1479272337517 } }
16/11/2016, 08:58:58   ce3c6d96.3cb2a8
iot-2/type/WashingMachine/id/Washer01/evt/fluid/fmt/json :
msg.payload : Object
{ "d": { "count": 2, "hardness": 77,
"temperature": 98, "flowrate": 11, "fluidlevel":
"acceptable", "ts": 1479272338519 } }
16/11/2016, 08:58:59   ce3c6d96.3cb2a8
iot-2/type/WashingMachine/id/Washer01/evt/voltage/fmt/json :
msg.payload : Object
{ "d": { "voltage": 223, "frequency": 66,
"count": 1, "ts": 1479272339520 } }
16/11/2016, 08:58:59   ce3c6d96.3cb2a8
iot-2/type/WashingMachine/id/Washer01/evt/fluid/fmt/json :
msg.payload : Object
{ "d": { "count": 3, "hardness": 78,
"temperature": 89, "flowrate": 11, "fluidlevel":
"acceptable", "ts": 1479272339521 } }
```

Now we can see the very same data, but now we are officially on the cloud and not pretending we are on an IoT device anymore.
We are currently streaming those data into Cloudant at the speed of more than one record per second.

SO NOW LET'S HAVE A BRIEF LOOK AT THE TEST DATA GENERATOR (FLOW1):



Lost connection to server, reconnecting...

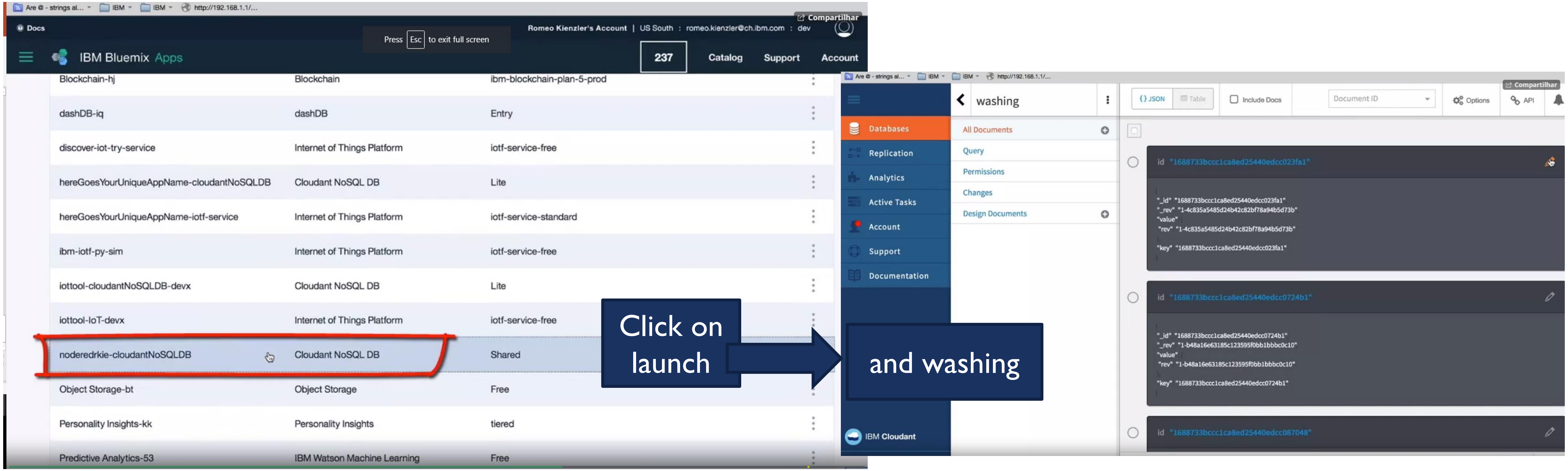
Edit ibmiot out node

Authentication	Bluemix Service
Output Type	Device Event
Device Type	WashingMachine
Device Id	Washer01
Event Type	status
Format	json
Data	msg.payload
QoS	
Name	Washer01

Note: If there is a property in the message that corresponds to any of the values entered above, then the property in the message takes precedence. See the Info tab for more details.
Example JSON device event: {"d":{"myName":"Arduino Uno", "temperature":989}}
Note: If there is a property in the message that corresponds to any of the values entered above, then the property in the message takes precedence. See the Info tab for more details.

LET'S HAVE A LOOK AT THE CLOUDANT USER INTERFACE TO SEE WHETHER THE DATA WE ARE STORING ACTUALLY ARRIVES IN THE DATABASE.

- So we open the Bluemix user interface and
- scroll down to the Cloudant service which is bound to our Node-RED application.
- You can easily make autocorrect one since by default it has the same name as the URL of your application.



Docs
Press Esc to exit full screen
Romeo Kienzler's Account | US South : romeo.kienzler@ch.ibm.com : dev
Compartilhar
IBM Bluemix Apps
237
Catalog
Support
Account

Blockchain-hj	Blockchain	ibm-blockchain-plan-5-prod
dashDB-iq	dashDB	Entry
discover-iot-try-service	Internet of Things Platform	iotf-service-free
hereGoesYourUniqueAppName-cloudantNoSQLDB	Cloudant NoSQL DB	Lite
hereGoesYourUniqueAppName-iotf-service	Internet of Things Platform	iotf-service-standard
ibm-iotf-py-sim	Internet of Things Platform	iotf-service-free
iottool-cloudantNoSQLDB-devx	Cloudant NoSQL DB	Lite
iottool-IoT-devx	Internet of Things Platform	iotf-service-free
noderedrkie-cloudantNoSQLDB	Cloudant NoSQL DB	Shared
Object Storage-bt	Object Storage	Free
Personality Insights-kk	Personality Insights	tiered
Predictive Analytics-53	IBM Watson Machine Learning	Free

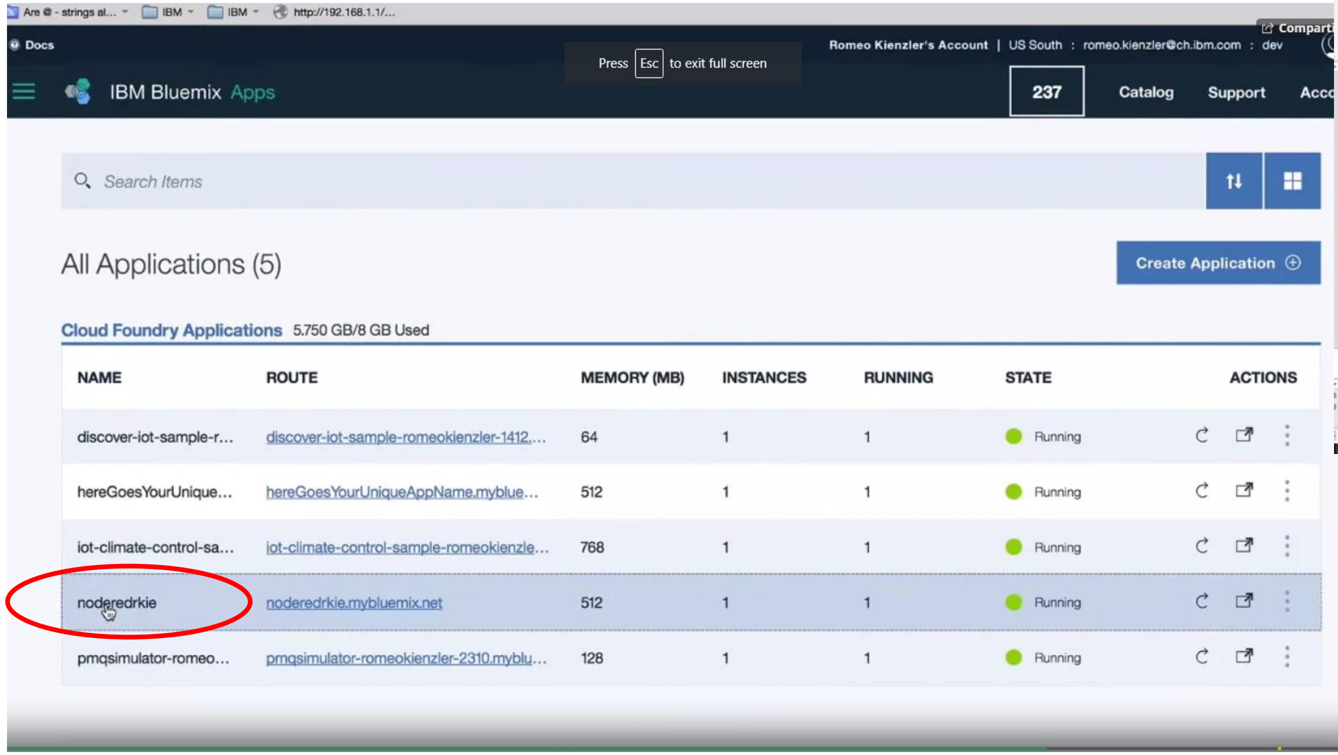
Click on launch

and washing

```
id "1688733bccc1ca8ed25440edcc023fa1"
{
"_id": "1688733bccc1ca8ed25440edcc023fa1",
"_rev": "1-4c835a5485d24b42c82bf78a94b5d73b",
"value": {
 "rev": "1-4c835a5485d24b42c82bf78a94b5d73b"
},
"key": "1688733bccc1ca8ed25440edcc023fa1"
}

id "1688733bccc1ca8ed25440edcc0724b1"
{
"_id": "1688733bccc1ca8ed25440edcc0724b1",
"_rev": "1-b48a16e63185c123595f0bb1bbbc0c10",
"value": {
 "rev": "1-b48a16e63185c123595f0bb1bbbc0c10"
},
"key": "1688733bccc1ca8ed25440edcc0724b1"
}

id "1688733bccc1ca8ed25440edcc087048"
```

ACCESS CLOUDANT DATA ON APACHE SPARK

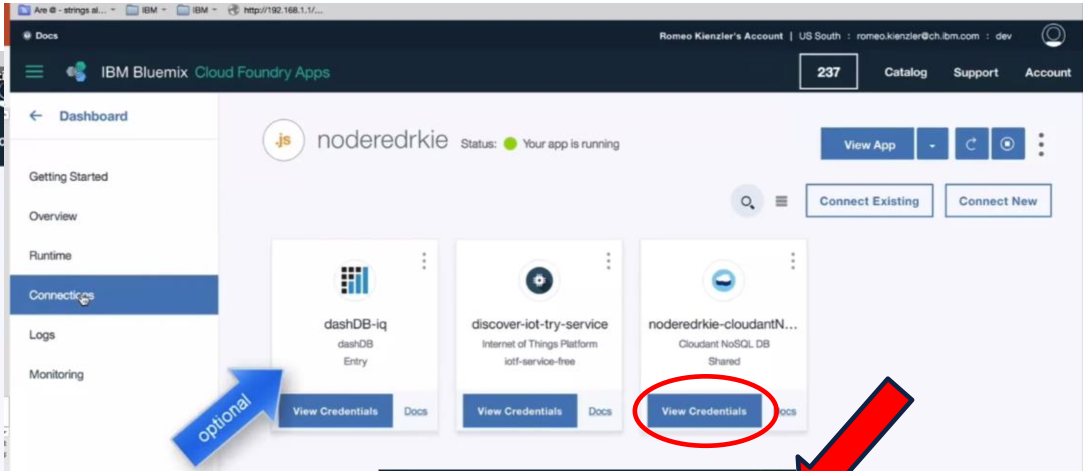
- In order to access Cloudant data on Apache Spark, we need to obtain the Cloudant credentials.
- So again, we enter the IBM Bluemix dashboard, click on our Node-RED application.

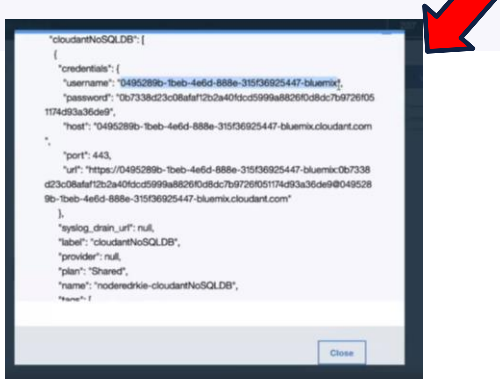


All Applications (5)

Cloud Foundry Applications 5.750 GB/8 GB Used

NAME	ROUTE	MEMORY (MB)	INSTANCES	RUNNING	STATE	ACTIONS
discover-iot-sample-r...	discover-iot-sample-romeokienzler-1412....	64	1	1	● Running	
hereGoesYourUnique...	hereGoesYourUniqueAppName.myblue...	512	1	1	● Running	
iot-climate-control-sa...	iot-climate-control-sample-romeokienzle...	768	1	1	● Running	
noderedrkie	noderedrkie.mybluemix.net	512	1	1	● Running	
pmqsimulator-romeo...	pmqsimulator-romeokienzler-2310.myblu...	128	1	1	● Running	



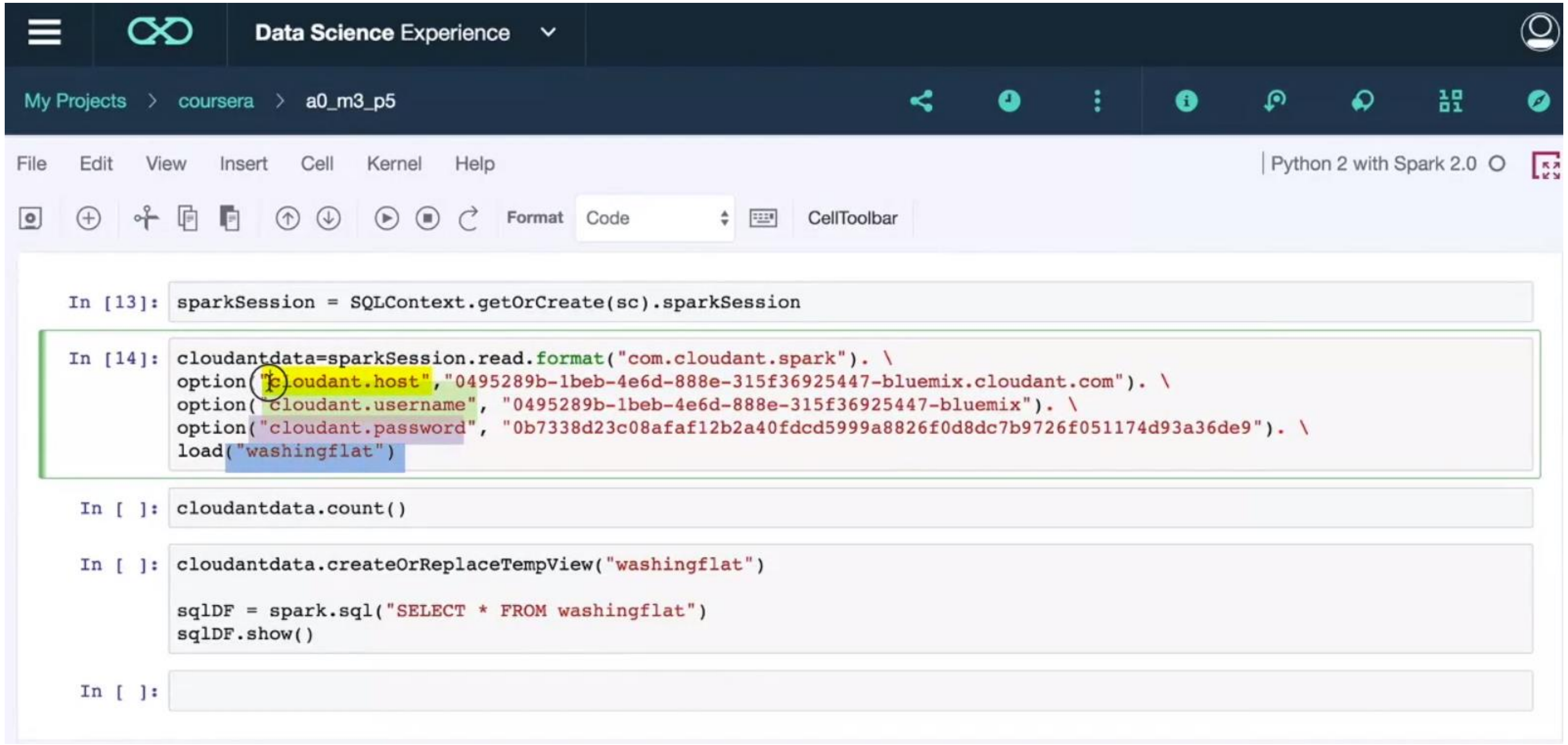


```
 "cloudantNoSQLDB": [
  {
   "credentials": {
    "username": "0495289b-1beb-4e6d-888e-315f36925447-bluemix",
    "password": "0b7338d23c08afaf12b2a40fdcd5999a8826f0d8dc7b9726f05
1174d93a36de9",
    "host": "0495289b-1beb-4e6d-888e-315f36925447-bluemix.cloudant.com
",
    "port": 443,
    "url": "https://0495289b-1beb-4e6d-888e-315f36925447-bluemix:0b7338
d23c08afaf12b2a40fdcd5999a8826f0d8dc7b9726f051174d93a36de9@049528
9b-1beb-4e6d-888e-315f36925447-bluemix.cloudant.com"
   },
   "syslog_drain_url": null,
   "label": "cloudantNoSQLDB",
   "provider": null,
   "plan": "Shared",
   "name": "noderedrkie-cloudantNoSQLDB",
```

SO NOW IF WE HAVE ALL INFORMATION TO ACCESS THE DATABASE FROM APACHE SPARK, SO LET'S HAVE A LOOK HOW THIS WORKS.

- There are basically four parameters you have to specify when accessing Cloudant from Apache Spark using the Cloudant connector:

Flow 1" contains a NodeRED application to simulate IoT sensor data. This flow is running in the cloud. In a real scenario, where would this flow normally run?
- On a IoT Gateway and
- On a IoT Device!


Data Science Experience
My Projects > coursera > a0_m3_p5
File Edit View Insert Cell Kernel Help
Python 2 with Spark 2.0
Format Code
CellToolbar

```
In [13]: sparkSession = SQLContext.getOrCreate(sc).sparkSession

In [14]: cloudantdata=sparkSession.read.format("com.cloudant.spark"). \
         option("cloudant.host","0495289b-1beb-4e6d-888e-315f36925447-bluemix.cloudant.com"). \
         option("cloudant.username", "0495289b-1beb-4e6d-888e-315f36925447-bluemix"). \
         option("cloudant.password", "0b7338d23c08afaf12b2a40fdcd5999a8826f0d8dc7b9726f051174d93a36de9"). \
         load("washingflat")

In [ ]: cloudantdata.count()

In [ ]: cloudantdata.createOrReplaceTempView("washingflat")

        sqlDF = spark.sql("SELECT * FROM washingflat")
        sqlDF.show()

In [ ]:
```

SUMMARY

- IoT data comes from IoT devices
- NodeRED can run on IoT devices
- Easy to simulate test data and run your solution – "the how"
- Straightforward to stream data to Cloudant
- ApacheSpark cloudant connector supports dataframe from Apache Spark