IOT over MQTT
pull zip file from github then go to the folder IoT-over-MQTT
CMD To view docker-compose file-
$ nano docker-compose.yml

docker-compose file contains definition of all the services that we want to run

Services:

1) Orion context broker
2) IoT agent
3) Database MongoDB
4) MQTT- publish-subscribe network protocol for transporting messages between devices.

FLOW:

Firstly, MQTT broker receives data from sensors(devices) and acts as a central communication point, passing MQTT topics between the IoT Agent and IoT devices as necessary.

Then, IoT agent receives data from MQTT Broker on registered topics and forwards to Orion Context Broker.

Orion Context Broker stores context information
-Query context information.
-Update context information, e.g. send updates of temperature
-Get notified when changes on context information take place (e.g. the temperature has changed)
-Register context provider applications, e.g. the device for the temperature sensor.

Finally the data goes to MongoDB database for storage.

CMD To run docker-compose file in detach mode,

$ docker-compose -p fiware up -d
$ docker ps

Checking Health of IoT agent and Context broker services

POSTMAN:
In Collections, go to IoT agent health check,
URL is localhost since it is running on our local system,
port number check from 'docker ps' for IoT agent and click SEND.

If STATUS is 200 OK means successful.
Similarly repeat steps for Context broker health check.

Connecting IoT Devices

The IoT Agent is middleware between devices and the context broker.
So,to create context data entities with unique IDs,
provision a service
unknown device records some data,
then the IoT Agent add this to the context using the supplied <device-id>.

Since Orion context broker is responsible for context, i.e information
So,
STEP 1) Registering a service
for success, Check STATUS- 201 Created

[Provisioning a Service Group for MQTT
For MQTT communication, provisioning supplies the authentication key so the IoT Agent will know which topic it must subscribe to.]

STEP 2) Provision IoT device
entity_name, entity_type and attributes[object_id, name, type]
entity_name is primary key
for success, Check STATUS- 201 Created

STEP 3) Publish message to a particular topic "t|30"
docker run -it --rm --name mqtt-subscriber --network fiware_default efrecon/mqtt-client sub -h mosquitto -m "t|30" -t "/4jggokgpepnvsb2uv4s40d59ov/TempHumd001/attrs"

where
4jggokgpepnvsb2uv4s40d59ov is api key of service registered above
TempHumd001 is device_id of device above
attrs are the attributes

STEP 4) Display published data in Context broker
go to POSTMAN, use entity_name as primary key to fetch published data
see value as "30".

STEP 5) Display registered services
Open a new terminal, and create a new running mqtt-subscriber Docker container as follows:
docker run -it --rm --name mqtt-subscriber --network fiware_default efrecon/mqtt-client sub -h mosquitto -t "/#"

DATABASE

$ docker exec -it db-mongo mongo
To show database
> show dbs;

address
admin
config
iotagentui
local
orion
orion-openiot

>use iotagentui
>show collections //here collections refers to tables

>db.groups.find();

>db.groups.find().pretty();

>use orion
>show collections