Prof. Vittorio Scarano

SERVERLESS COMPUTING FOR IOT

MQTT Temporary traffic light

Candidate: Cioffi Daniele

PROJECT

The project consists of simulating a temporary traffic light that allows pedestrian passage. The temporary traffic light is used before the installation of a permanent traffic light, to verify its frequency of use.

His "state" is always green. This state changes to red when a pedestrian makes a "remote call" through a remote button, because he wants to cross the road.

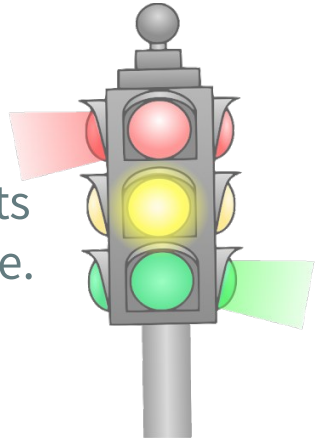
https://github.com/daniele321b/serverless

PROJECT - 2

Steps:

- the pedestrian will press the button;
- after a few seconds the traffic light will turn red;
- the pedestrian will be able to cross.

To verify how necessary it is to install a traffic light and then its actual use, all calls are saved in a database with date and time.

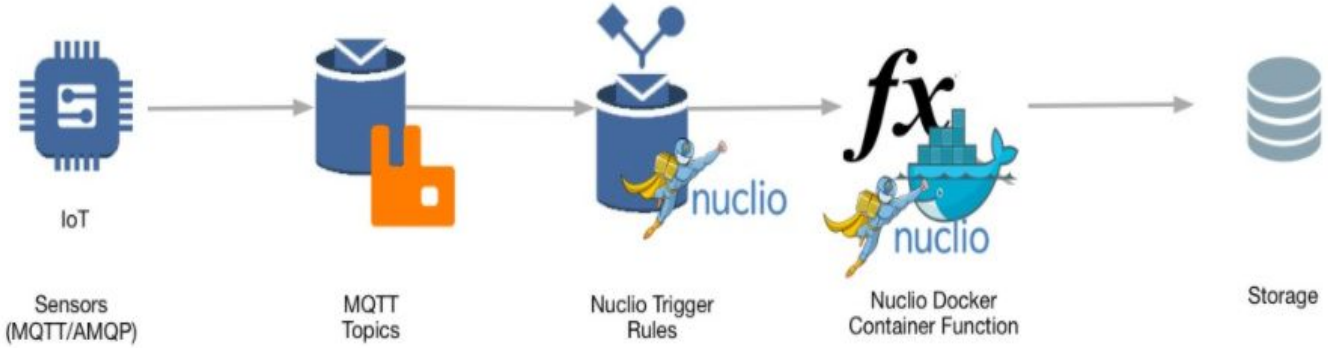


01

ARCHITECTURE

Which components are used for implementation

TOPIC: iot/sensors/trafficLights



02

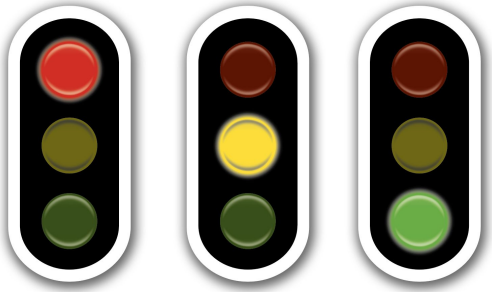
IMPLEMENTATION

Components

https://github.com/daniele321b/serverless

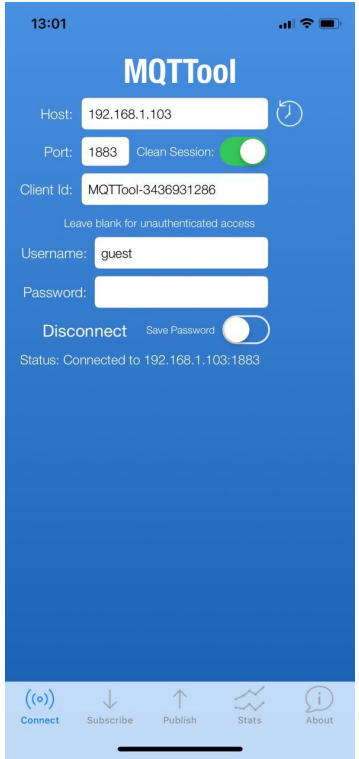
Implementation

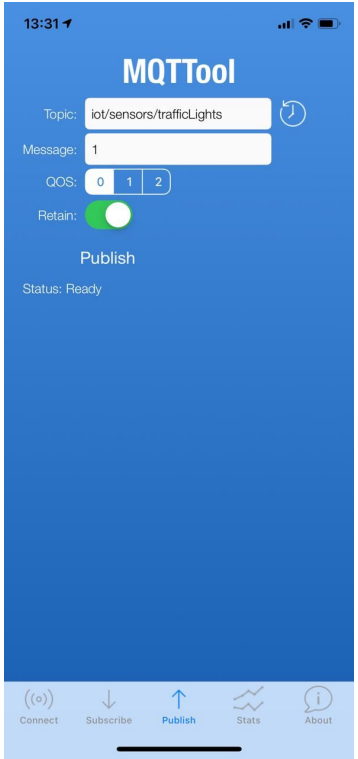
Send Call: send a new “call value” on the MQTT to the queue iot/sensors.



Consume Call: is triggered by a new MQTT message on the queue iot/sensors.

MQTTool

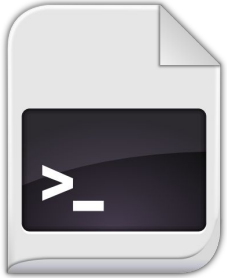




Client for iOS: In addition to the Nuclio test, I also used a mobile client (app) to simulate the call..

Implementation -2

Logger: is waiting for a new messages on the queue and it makes the data persistent by saving them in a database.



ShowData: retrieve data from database

send.js

```
var mqtt = require('mqtt'), url = require('url');

var mqtt_url = url.parse(process.env.CLOUDAMQP_MQTT_URL || 'mqtt://guest:guest@192.168.1.103:1883');
var auth = (mqtt_url.auth || ':').split(':');
var url = "mqtt://" + mqtt_url.host;

var options = {
  port: mqtt_url.port,
  clientId: 'mqttjs_' + Math.random().toString(16).substr(2, 8),
  username: auth[0],
  password: auth[1],
};

exports.handler = function(context, event) {
    var client = mqtt.connect(url, options);

    client.on('connect', function() {
        client.publish('iot/sensors/trafficLights', "1", function() {
                    client.end();
                    context.callback('MQTT Message Sent');
                } );

        });

};
```

https://github.com/daniele321b/serverless

consumer.js

```
var amqp = require('amqplib');
        var FUNCTION_NAME = "consumer";
        var STATE = 1;
        var count = 2; // {0 red, 1 yellow, 2 green}
         function send_feedback(msg){

            var q = 'iot/logs'; //path
            amqp.connect('amqp://guest:guest@192.168.1.103:5672').then(function(conn) { //connection
                return conn.createChannel().then(function(ch) { //connection channel
                    var ok = ch.assertQueue(q, {durable: false});
                    return ok.then(function(_qok) {
                    ch.sendToQueue(q, Buffer.from(msg));
                    //console.log(" [x] Sent '%s'", msg);
                    return ch.close();
                    });
                }).finally(function() {
                        conn.close();
                    });
            }).catch(console.warn);
         }

         //change var state
        function decrementCount(){
         return count--;
        }

       function resetCount(){
         return count = 2;
       }

       function changeState(){
         return STATE=1;
       }
```

consumer.js

```
37  //convert from binary to string
38  function bin2string(array){
39    var result = "";
40    for(var i = 0; i < array.length; ++i){
41      result+= (String.fromCharCode(array[i]));
42    }
43    return result;
44  }
45
46  exports.handler = function(context, event) {
47      var _event = JSON.parse(JSON.stringify(event)); /
48      var _data = bin2string(_event.body.data);
49      context.callback("feedback "+_data);
50      //context.callback(STATE + " - TL");
51
52      //send_feedback("CALL MADE " + _data);
53      //send_feedback("CALL MADE!");
```

https://github.com/daniele321b/serverless

consumer.js

```
55              if(STATE==1){
56                send_feedback("CALL MADE!");
57                STATE--;
58              setTimeout(() => {
59                decrementCount(), send_feedback("TRAFFICLIGHT STATE CHANGED --- NOW: YELLOW ");
60                }, 5000);
61
62                setTimeout(() => {
63                  decrementCount(), send_feedback("TRAFFICLIGHT STATE CHANGED --- NOW: RED ");
64                }, 15000);
65
66                setTimeout(() => {
67                  resetCount(), send_feedback("TRAFFICLIGHT STATE RESET --- NOW: GREEN ");
68                }, 30000);
69
70                setTimeout(() => {
71                  changeState()
72                }, 30010);
73              }
74          };
```

newPersistent.js

```
var amqp = require('amqplib');
var db = require('./persistent');

var ACK = 3;

amqp.connect('amqp://guest:guest@192.168.1.103:5672').then(function(conn) {
  process.once('SIGINT', function() { conn.close(); });
  return conn.createChannel().then(function(ch) {

    var ok = ch.assertQueue('iot/logs', {durable: false});

    ok = ok.then(function(_qok) {
      return ch.consume('iot/logs', function(msg) {
        if(ACK == 3){
              db.databaseInsert();
              ACK--;
        } else if (ACK == 0){
          ACK=3;
        }else {
          ACK--;
        }
        console.log(" [x] Received '%s'", msg.content.toString());
      }, {noAck: true});
    });

    return ok.then(function(_consumeOk) {
      console.log(' [*] TRAFFICLIGHT STATE --- NOW: GREEN ');
    });
  });
}).catch(console.warn);
```

https://github.com/daniele321b/serverless

showData.js

```
var mysql = require('mysql');

//connection
var HOST = "localhost";
var USER = "root";
var PASS = "password";
var MYDB = "trafficlights";

var showData = "SELECT * FROM calls";


var con = mysql.createConnection({
    host: HOST,
    user: USER,
    password: PASS,
    database: MYDB
  });


    con.connect(function(err) {
        if (err) throw err;
        //Select all customers and return the result object:
        con.query(showData, function (err, result, fields) {
          if (err) throw err;
          console.log(result);
        });

      });
```

https://github.com/daniele321b/serverless

03

SIMULATION

Video playback of a simulation

Video

04

FUTURE DEVELOPMENTS

Possible modifications and implementations

FUTURE DEVELOPMENTS

01

WEB/MOBILE APP

User interface implementation

02

REAL "THINGS"

implementation of a real button and a real traffic light

THANKS

https://github.com/daniele321b/serverless