## Shane Ryan – x15745561

# IoT Principles – Assignment 1

## Python Script:

This project begins with writing a python script. In order to accomplish the objectives of this assignment we first need to import the necessary libraries at the beginning of the script, namely:

Dweepy – This library allows us to output our sensor readings to dweet.io

Grovepi – This library allows us to make us of the grove board and it’s sensors for gathering data

Configparser – This allows us to parse the data in an external config file (config.ini for me) and pass it to the script.

Sqlite3 – This library allows us to establish an SQL database to store the data gathered by the sensors.

I wrote a get() method for each of the 6 sensors (e.g getSound()) and call these methods later in a larger getandSaveReadings() method. This method accomplishes several things, it calls the other get methods to return data from the sensors and stores it in a key value pair dictionary. It also passes the data into a tuple formatted correctly and performs an SQL INSERT statement to deposit data in our external database ‘readings’ table. The method then returns the dictionary, a while loop at the bottom of the method calls the getandSaveReadings() method every 5 seconds and calls the post() method which submits them to dweet.io. I also wrote a separate python script that I run once to create the SQL database and accompanying ‘readings’ table. If everything is successful the readings will be updating every 5 seconds on dweet.io.

## HighCharts

In order to accomplish the highcharts requirement of this assignment I first downloaded a simple bootstrap template and formatted and the relevant containers to hold my charts. I then used the code from the highcharts example on moodle and placed it in a createChart.js file. I took the configurable pieces from the example and placed variables there instead, these would then be parameters passed to the function from my other method called showCharts. The showCharts method accepts an array called ‘data’ as it’s only parameter, it then calls the dweetio.get function to retrieve the data from dweet.io. This method then passes the dweet data into the createChart function with custom parameters in place to give each chart different axis labels, tooltips, chart types etc. Finally this method calls my LocalData function and passes each chart (stored as vars) through as parameters. The localData functions accomplishes two things, it first calls a function to check the .length of the array to make sure the records are below 100, using a while loop if the amount of records gets too high the project makes use of the array.splice(0,1) function to remove 1 record from the first index(0). After this function is called for each data type, an addRecords function is then called to add the most recent data to the array, the array is then passed to a window.localStorage.setItem() function to store the records in localStorage (this JSON is stringified at this point).

## Work Beyond Scope

I use three sensors to gather 6 different types of data: sound, light, pentiometer, voltage, degrees rotation. I also have an LED bulb that is activated by sounds over a certain threshold and another LED bulb which can be dimmed or made brighter by the rotation sensor.

For the Javascript portion I am displaying all 6 data types from Dweet on one responsive bootstrap page. I have carefully tailored tooltips and relevant axis, labelling and make use of spline, line, bar, pie and scatter chart variants. My scripts are written following OOP protocol and are spread across several .js files keeping things clean and functional.