**FINAL PROJECT**

Comp 216 - Networking for Software Developers

This is the final project of COMP216.

It is a group effort that will replace your second test and will contribute 25% towards your final grade. This builds on the previous week’s lab works, Assignments, which may require some tweaking to assemble into a complete IoT solution.

# Before you start.

This is a large undertaking considering the time constraints and the number of components required. One person would not do able to and should not be doing all the work because of the volume and the scope. Because the time is not enough sufficient for one person to adequately complete all the prescribe tasks.

I suggest short (not more than 15 minutes) meeting frequent (at least 5 time per week), where designs are finalized, tasks are farmed out and reported back on. You can also examine the milestones and see if more effort is need to bring any lagging task up to the expected completeness. Unresponsive, uncooperative team members are to be identified and brought to my knowledge.

Make use of your favourite collaboration platform and relegate e-mails to formal correspondence.

# Technical Constraints

You will only use the libraries that were covered in classes. You will use Python version 3.6.5 or later. The only external frameworks/libraries allowable are requests, flask and mosquito. You may use the libraries in a standard python distribution with restraints.

# Overview.

We will implement an end-to-end IoT solution that will satisfy your client needs (this is based on your previous labs). We will use the MQTT protocol for this implementation. There will be publisher clients that sends data to a broker as well as subscriber clients that receives data from the broker. The broker sends data to the appropriate subscriber.

You will design and build both types of clients according to specifications. The diagram shows an overly simplified architecture of the intended system.

Broker

Subscriber

Publisher

Publisher

Publisher

Subscriber

Each component is described more fully below along with the weight contribution towards the final grade.

# Detailed description

## Broker

You will install the Eclipse mosquito broker and ensure that it is working as expected.

## Publisher

The publisher will generate one data value at a time to send to the broker at regular intervals indefinitely. The data value must be random with a pattern (I know that this is a contradiction). Think of the value of a particular stock on the stock exchange or the outdoor temperature around your home.

This can be implemented either a CLI or GUI interface.

Corrupt data?

Mutate value

Get a single value from generator

Sleep

Send data?

Package value as dict

1. Connect to broker
2. Send data
3. Disconnect from broker

No

Yes (about 99% of the time)

No

Yes (about 0.5% of the time)

You will configure and run multiple publisher clients to simulate multiple devices. You will have to pass some customization property to the publisher such as location etc.

### Publisher – Value generation

This must be implemented in a class in a separate file. (Just import the filename without the py extension in the file where you want to use the logic). The values generated must mimic the data that might be generated in your real-world situation.

The design should be such that it is easy to use (multiple variation started by just specifying different arguments when starting) and flexible enough to be able to give the data value in the required pattern.

### Publisher – Packaging the above values

The above value must be tagged with at least a time stamp and packaged as a json object before transmission. You decide what other features you need to encapsulate in your package,

### Publisher – Sending data to broker

* You will send the above packaged data to the broker under an agreed topic.
* Your transmission **will fail** with a frequency of about 1 in very 100 transmissions. This must not be deterministic!

### Publisher – Extras

These extras are for bonus points and will only be considered if all the normal specifications are satisfied adequately.

* To simulate a real-world scenario, occasionally skip blocks of transmissions (or sets of transmission). This condition must not throw the subscriber into confusion.
* Transmit “wild data” something that is completely off the chart. Again your subscriber should be able to handle this.
* Anything that will add value to your project.

## Subscriber

## The subscriber accept data from the broker and process it. It will decode the data and decide how to process it.

This is best implemented as a GUI application.

You will configure and run multiple subscriber clients to simulate multiple devices. Just as the case with the publisher, you will have to pass some customization property to the subscriber such as location etc.

### Subscriber – Receiving data from the broker

* You will listen to messages from the broker under an agreed topic.
* You will decode the message and decide how to handle the data

Subscriber – Handling the data/absence of data

This section is important, because this is where you imbue your personality on this project. You have to decide what is out of range data. You also have to be able to detect missing transmission.

* Handle sensible data. Display data both in text (print to the console) and visual formats on a tkinter application.
* Detecting and handling out of range (erroneous) data.
* Detecting and handling missing data.

**Quality of Code**

These 5 points are to lose. You automatically start with full points and as the instructor notices code aberration these 5 points will evaporate. Code aberration will include design flaws and implementation blunders.

## Project presentation

Every project team must present the project in the class. If an item is not clearly presented, there might be a possibility that you might not get a point for that item and marks will be deducted. You need not demonstrate your contribution to project, quality of code, design, and implementation.

**Communication**

Keep constant communication with the professor. Any issues related to the team work are to be brought to the teacher’s knowledge through mail. Remember the peer evaluation form (weightage of marks allotted) and the due date of submission (20% penalty if late)

## Due date:

## At the end of week 12

#### Submission

1. You will upload all your files individually to the course dropbox. The video link should be pasted in your submission as a .mp4 file. The names of the files must be prefixed with group\_«your\_group\_number» e.g. group\_1.
2. Your publisher code files must be called group\_«your\_group\_number»\_publisher.py e.g. group\_1\_publisher.py.
3. Your subscriber code files must be called group\_«your\_group\_number»\_subscriber.py e.g. group\_1\_subscriber.py.
4. All your other code files must be prefixed by you group number e.g. group\_1\_data\_generator.py
5. Must be uploaded to course drop box named as final project.