# Chapter 8: Class Project

## Objective

## Time: 4 ½ Hours

## Fundamentals

## Exercise(s)

Your project is to build an IoT weather station. It will:

1. Measure local temperature and humidity. This information can be read from the analog co-processor shield kit using I2C (see I2C exercises in the peripherals chapter).
2. You will connect to a provided MQTT broker:

amk6m51qrxr2u.iot.us-east-1.amazonaws.com

1. Your *thing* name will be “ww101\_<nn>” where <nn> will be a number assigned to you. For example ww101\_01.
2. The credential and private key for your *thing* can be found in the class material folder.
   1. Hint: After updating the key files, you should run a “Clean” on the project. Otherwise, the project will not see the new keys.
3. Update the state of the “thing”. The parameters are named “temperature” (float), “humidity” (float), “weatherAlert” (true or false) and “IPAddress” (ipv4 4dot syntax).
4. Implement a serial terminal to allow the following commands (see UART exercises in the peripherals chapter):

t – read + publish temperature

h – read + publish humidity

A – publish weather alert on

a – publish weather alert off

S – turn on subscriptions

s – turn off subscriptions

P – turn on printing of updates

p – turn off printing of update

x – print the current known state of data

l – print the list of known things

c – clear the screen

? – print out a help screen

u – turn off auto updating

U – turn on auto updating

For subscriptions, the easiest thing to do is to have a static list of known things from a few classmates (i.e. initials\_ww101).

It would be cool if you:

1. Used the linked\_list library to maintain a local database
2. Used threads
3. Used the console library functions to build the interface
4. Used VT100 escape codes to make a pretty screen (<http://ascii-table.com/ansi-escape-sequences-vt-100.php)>
5. Used the DCT to write the configuration
6. Created an HTTP server to display all of the information