# Team 1: IOT Sensor

Your task is to put together an environmental sensor, collecting data and transmitting it to a centralised internet end point.

You’ll be using a Raspberry Pi and a BMP180 sensor to collect temperature and pressure.

You are free to build the project anyway you wish, but you must send the data to the internet API endpoint in the following format using a HTTP GET request:

[http://](http://52.50.60.213/api.php?function=addReading&type=%3Csensor)52.16.112.87/api.php?function=addReading&type=<sensortype>&value=<sensor value>

The sensor type can be anything, but “temperature” and “pressure” are suggested…

An XML response will return success if your request is successful.

Here are some suggested steps you’ll need to go through to get your project up and running:

* Learn how to power up and connect your Pi to the network and obtain it’s IP address (you’ll need to be on the VPN to connect to the Pi)
* You’ll probably want to SSH into the Pi (the default user password is pi / raspberry)
* Understand the I2C interface at a high level
* Obtain the pin layouts to connect the BMP180 to your Pi
* Learn how to push and pull code from Git
* Choose an appropriate library to read the data from the BMP180 (hint, most are in python)
* Write a simple program to poll the raspberry Pi sensor
* Find an appropriate module in your chosen language to make a HTTP request to the endpoint (hint, python has plenty and the Pi sensor modules are mostly written in python…)
* Connect up your additional sensor and see if you can get a reading from it
* Work out how to run a scheduled task to collect data at periodic times OR make your program loop and startup at boot

## Useful URLs

<https://learn.adafruit.com/using-the-bmp085-with-raspberry-pi/overview>

<http://www.learnpython.org>

<https://docs.python.org/2/howto/urllib2.html>

<http://rogerdudler.github.io/git-guide/>

# Team 2: Web App

Your task is to create a web app to read data from a database based on user input and display the information back.

It is suggested that you use a docker container to build your web app so it can be tested on your laptop, and then uploaded to Docker hub for deployment on the internet.

A VM running on vCloud Air has been provided where you could run a container.

Probably the most popular language for web apps is PHP, although you could choose others.

A database has been created for you (MySQL) with some data in to query. The database schema is very simple:

DB: temperature.clmv89mfjqjm.eu-west-1.rds.amazonaws.com:3306

Host: hackathon

User: hackathon123

Pass: hackathon

Table: weather

datetime – a timestamp of the point the data was entered

type – a text field describing what the type of data is – “temperature” and “pressure” are populated

value – a “float” data type with a numerical value

Please don’t delete this table as the other teams will be using it ☺

You may wish to download an interface to connect to your database – for example adminer.php (google).

Some suggested steps:

* Learn how to run a basic container on your laptop. Learn about how docker handles port mapping and how you can “mount” local files into the container (-v)
* Find an appropriate container to base your project on
* Learn how to push and pull code from git
* Write a basic php based application to connect to the database and display the last 10 values (read up on how to create SQL queries)
* Work out how to push your container to DockerHub (read up on DockerFiles) and run it on the central webserver
* Extend the app to accept user input on how many to read, time ranges and what type to read
* Read about HTML5 and see if you can create a “nicer” looking interface
* See if you can add a graph to the page (hint ChartJS)

## Useful URLs

<https://docs.docker.com/docker-for-mac/>

<http://www.w3schools.com/Php/>

<http://www.chartjs.org>

<https://hub.docker.com/r/webdevops/php-nginx/>

<http://rogerdudler.github.io/git-guide/>

# Team 3: Mobile App

Your task is to build a simple iOS application which will display the latest readings from an internet based API.

Your app should make a call to the following URL with a GET request and parse the XML that is returned:

[http:// 52.16.112.87/api.php?function=getReading&type=<sensortype>&value=<num](http://52.50.60.213/api.php?function=getReading&type=%3Csensortype%3E&value=%3Cnum) of readings>

where sensortype can either be temperature or pressure and num of readings is the last x readings you’d like to gather.

The page will return an XML document which you will need to parse.

Suggested Steps:

* Follow a “Swift” tutorial online to learn how to build a basic application with buttons, labels and drop down boxes
* Learn how to push your code into git
* Determine how to make a HTTP call using Swift
* Work out how to parse XML appropriately
* Display readings on the screen in a label
* Work out how to display a graph of the data

<https://developer.apple.com/library/ios/referencelibrary/GettingStarted/DevelopiOSAppsSwift/>

<http://ashishkakkad.com/2014/10/xml-parsing-in-swift-language-ios-9-nsxmlparser/>

<http://rogerdudler.github.io/git-guide/>

You may need this for your Info.plist file to prevent security error messages when making HTTP requests

<key>NSAppTransportSecurity</key>

<dict>

<!--Include to allow all connections (DANGER)-->

<key>NSAllowsArbitraryLoads</key>

<true/>

</dict>