Vac Work 2019-2020

Progress of development report

|  |  |
| --- | --- |
| Written By | Michael Katsoulis |
| Student Number | KTSMIC005 |
| Date: | 11 Nov ‘19 - 7 Feb ‘20 |

Report on the path taken and work completed in the development of a system to retrieve, store and display the data collected by the sensors in Antarctica.

horizontal line

# 

[**Intro**](#_ajwgm1idk85w) **3**

[Goal for vac work:](#_8usv70yg3fs9) 3

[Requirements:](#_jyt71ccvm0sz) 3

[Reflection on progress and opportunities for improvement](#_nljdj4dlj5fl) 3

[**Planing**](#_mvq02cngxld5) **4**

[Retrieving data](#_mrplussdbyjr) 5

[Getting data from the satellites](#_oeyjpmyrr570) 5

[Storage of data](#_la5jp5tnimjw) 5

[Choice of database software](#_uifrrk9t3ng) 5

[Retrieving data](#_fychd04fp8p2) 6

[Getting data from the satellites](#_ytegmbmxz0g3) 6

[Storage of data](#_3q8os17op22z) 6

[Choice of database software](#_m8iimuc11ozk) 6

[Display of data](#_u36l7uq5sw1z) 6

[**Setup of the server and Login Details:**](#_dcaapyld21ic) **8**

[**Docker**](#_y6pbz33mj50e) **9**

[What is Docker](#_wrtnnsmsvp1l) 9

[Why am I using Docker](#_es1zaxlfzufw) 9

[**Starting the server: -- old**](#_ut8p7xglq2ot) **9**

[Accessing the docker Influx Database:](#_ebza01cbe9wi) 10

[Running an influx database with Docker](#_g2jxa7ygdobr) 10

[**Node-Red**](#_lmzbq4hbt14t) **12**

[**Accessing the DNS proxy server**](#_bim7i9whe4xg) **12**

[**Node-red Flows Backup**](#_duc05yaczvmp) **15**

[**Errors Log**](#_r2lokp1wkv0s) **20**

# 

## 

# Intro

## Goal for vac work:

To make a server which would store all data that was received from the buoys in antarctica and turn this raw data into something that is accessible and usable.

## Requirements:

|  |  |
| --- | --- |
| Data received as a raw binary string must be decoded. | Completed as best as possible with current knowledge about the data  Completed by: Amaan Vally |
| This decoded data must then be brought into a database which can support access by various data visualisation apps. | .csv → influxdb  By: Michael Katsoulis |
| This database must also be easy to access and use. Management of the data also needs to be simple. |  |
| The system must be portable. By this it means that the system must be able to be installed on another machine and must work with minimal reconfiguration. | Attempted but not completed |
| The development process needs to be documented well enough that someone new can pick it up and continue. |  |

## Reflection on progress and opportunities for improvement

This project was particularly challenging as everything that was done had to be learnt from scratch. In the Mechatronics stream there was not much information and learning about the topics that were used during this vac work.

As a result, there are plenty of chances to improve the way things were done.

Todo:

* The linking of email to .csv needs to be done.   
  Unfortunately Michael’s Laptop broke and he lost his data. Amaan Vally has this code though as it was his task
* Once the .csv files can be made, the node-red program needs to know where it is located.
* This should put the data straight into InfluxDB but:
  + The database it’s going to be injected into needs to be chosen in the node-red node
  + It should put the right number of sensor values in on it’s own but the “measurement name” in the node red node needs to be changed to say which buoy the information is from.
  + To have multiple buoys, the node red program can be expanded to route the information to different nodes which should be set up for each buoy.

Chronograf provides a basic visualisation of the data however Grafana is useful if you would like to have data where there is heatmaps and various other more complicated visualisations over the standard graphs in chronograf. There are plenty of resources online, I just wasted time trying to get Docker to work well.

Issues with docker:

Docker should have provided a very efficient and elegant way to package up the software for use somewhere else but the issue was with communication between containers (services) running with docker.   
I could not manage to get this right but if you do, all the containerised software versions of node-red, influxdb, etc. all function the same way in their containers and so you can just copy across the code.

Debugging the system:

It’s hard to debug the system when we have no idea of what the data that will be received will look like.   
I did the best I could to analyse crashes and make sure that everything should be reliable but the wheels may come off when we start actually using it.   
Please contact me (Michael) if it happens and I’m happy to help debug.

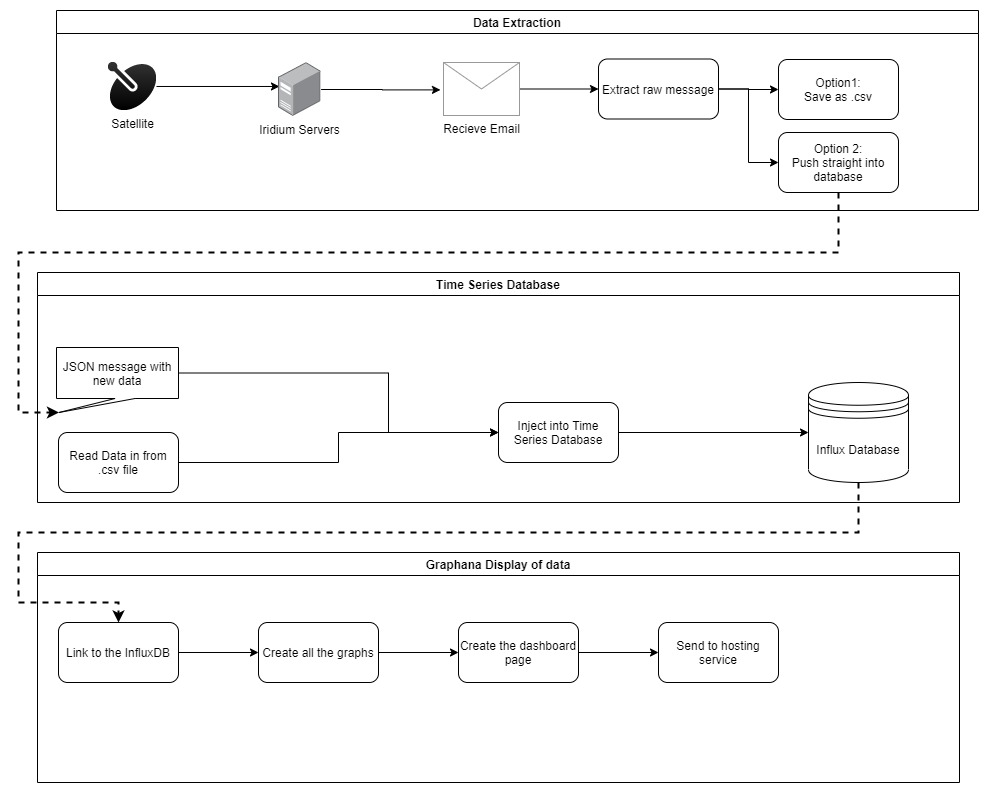
To test the system I built a dashboard to generate data. It can be accessed from <http://localhost:1880/ui/>

The data created will be sent out of the nodes as messages which need to be fed into the influx database.

This dashboard needs to be changed to produce the data we expect to receive once we know how the system is going to work.

# Planing

Flow of the system:



## Retrieving data

### Getting data from the satellites

A python script needs to be written to retrieve the data sent in emails from the satellite link’s servers. This will access the email, find the data and download the binary files.

This data is then decoded and sent to the database. The best way to do this is either to use a .csv file as a temporary backup and messenger or to use ports on the local machine.

## Storage of data

The data that will be gathered needs to be stored in a database. Since the data is time dependent, a time-series database makes the most sense.  
 Some advantages of time series databases are that they allow faster queries with any system where time based calculations are needed (such as average over time).   
They also allow for rules that decide how long to keep data for and how it must be compressed. We may be interested in what our sensors read every hour for this month but after a year we can compress this data and store the max, min, mean and median for the day.

### Choice of database software

Influx was chosen as the database management suite. It is open source and consistently ranked amongst the top options. It has also gained a large amount of popularity over the past couple years and therefore there is plenty of community support available.

There is a suite of different add ons which work together with influx to give visual data management,

Influx also offers enterprise level solutions if it is decided that this project needs to be scaled up.

## Retrieving data

### Getting data from the satellites

A python script needs to be written to retrieve the data sent in emails from the satellite link’s servers. This will access the email, find the data and download the binary files.

This data is then decoded and sent to the database. The best way to do this is either to use a .csv file as a temporary backup and messenger or to use ports on the local machine.

## Storage of data

The data that will be gathered needs to be stored in a database. Since the data is time dependent, a time-series database makes the most sense.

Some advantages of time series databases are that they allow faster queries with any system where time based calculations are needed (such as average over time). They allow for rules that decide how long to keep data for and how it must be compressed. We may be interested in what our sensors read every hour for this month but after a year we can compress this data and store the max, min, mean and median for the day.

### Choice of database software

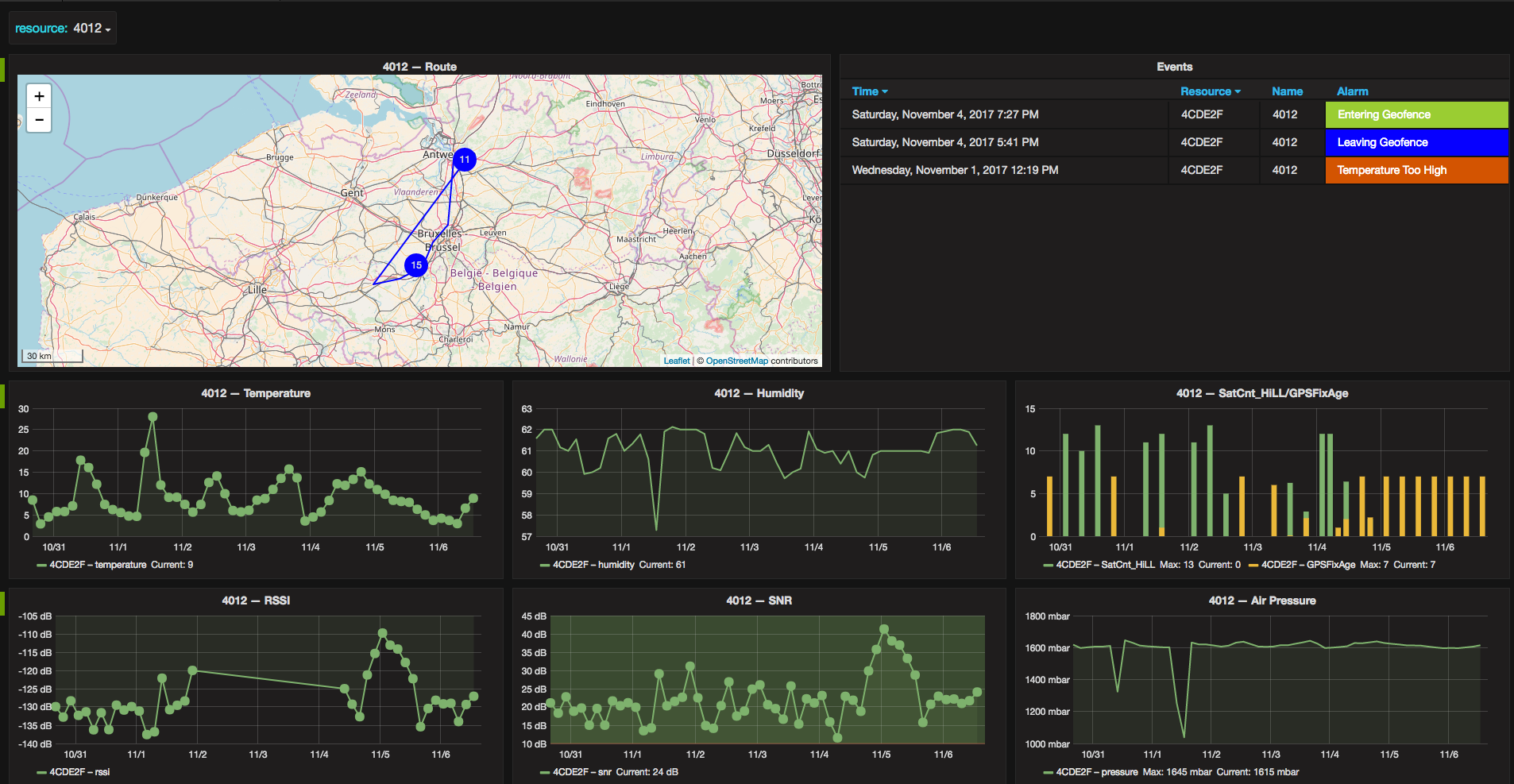
Influx was chosen as the database management suite. It is open source and consistently ranked amongst the top options. It has also gained a large amount of popularity over the past couple years and therefore there is plenty of community support available.

They also offer enterprise level solutions if it is decided that this project needs to be scaled up.

## Display of data

Grafana was chosen to produce the graphs required for the data. It is easy to directly integrate with Influx databases and there is a large pool of support and tutorials from it’s community of users.

An example of what a grafana dashboard looks like is shown below:



# Setup of the server and Login Details:

|  |  |  |
| --- | --- | --- |
| **Service** | **Username** | **Password** |
| Logging into the server | user | admin |
| \*gmail | [floatyboi2k19@gmail.com](mailto:floatyboi2k19@gmail.com) | iamafloatyboi |
| Docker.com | Username: sharcwavebuoy  Email : [floatyboi2k19@gmail.com](mailto:floatyboi2k19@gmail.com) | iamafloatyboi |
| Grafana online login | SharcWaveBuoy | iamafloatyboi2k19 |
| Grafana database  -- Locally on server | admin | admin |
|  |  |  |

# Docker

* The attempt to get everything portable and easy to set up

## What is Docker

Docker is a solution for app development to try and give each deployment the same environment variables.

Think of it as a big shipping container

You build everything, get it working with all the environmental variables on the development side. Once you are happy it works, you package everything in a (docker) container and this can then be sent off to another user.

When we are ready to use the project on another machine, we simply import the container.

We can then run everything inside our little box and it should work on any other machine.

No more fiddling with environmental variables or checking if the right software was installed or if we’re on the same version of a piece of software as when it was all written.

Just run everything from within our shipping container.

This should decrease the effect and risk of one of the services in our container crashing. It can then be debugged and run again and the whole system should continue to work.

Docker can also be setup so that if something crashes it will automatically be restarted

For a good intro to docker see: <https://docs.docker.com/>

## Why am I using Docker

I am trying to have everything running from Docker based images so that we can package off all the work we build so that it can be installed onto any machine running ubuntu and it should run out of the box.

This should allow us to package our software and reuse it on a different machine with fewer problems.

# Starting the server: -- old

I have written a shell script located in the desktop folder called automation scripts.

If you would like to run the system using the docker based influx database (I could not get nodered to communicate with it) then run the ./SHARC\_wave\_bouy\_startup.sh

Otherwise use ./start\_v2.sh

If there are any errors, I have described the common errors I faced and their solutions in the error log.

Update 06 Feb 2020:  
Everything is now configured as a service on the machine.

It will all start and run automatically.

If there are any errors or crashes, you will need to check the system services logs.

## Accessing the docker Influx Database:

In terminal type:

~$ docker exec -it influxdb /bin/bash

And once it launches you into the bash enviroment type:

influx

This will put you into the influx command line

If you would like a graphical interface with the database, go into terminal and type:

~$ chronograf

If it is already running, you will recieve an error saying that port 8888 is already in use.

Now head over to your web browser and in the search bar go to:

<http://localhost:8888/>

This should put you straight into the suite that the Influx DB team designed to view and manage data in the database.

## Running an influx database with Docker

# Note:

I could not get influxdb to work well with node red

Use:

docker run -d \

--name="influxdb" \

--restart on-failure \

-p 8086:8086 \

-v influxdb\_data:/var/lib/influxdb \ ←This is the volume where the data is stored

influxdb -config /etc/influxdb/influxdb.conf

To make a new database:

sudo docker exec -it influxdb /bin/bash

Influx

CREATE DATABASE data\_log1

exit

# Node-Red

To access the configuration of the node red service file:

sudo nano /etc/systemd/system/node-red.service

# Accessing the DNS proxy server

1. Run the docker container using “docker start mydns-proxy-server”
2. Once you have the mydns-proxy-server container running in the docker, use the browser to access “<http://localhost:5380/static/>”

* This is the GUI to add and manage assigned names for IP addresses on the machine.

# Node-red Flows Backup

Paste this into the import from clipboard option in nodered

[{"id":"4f359664.ac5fa8","type":"tab","label":"Data Generating Dashboard","disabled":false,"info":""},{"id":"75f0c5bb.be131c","type":"tab","label":"Load .csv into database","disabled":false,"info":""},{"id":"8d2e2eb6.647758","type":"tab","label":"Reading emails","disabled":false,"info":""},{"id":"45cf183d.c5dd48","type":"influxdb","z":"4f359664.ac5fa8","hostname":"localhost","port":"8086","protocol":"http","database":"sensors\_test","name":"","usetls":false,"tls":""},{"id":"6013b12d.9ee22","type":"influxdb","z":"4f359664.ac5fa8","hostname":"myinfluxdb","port":"8086","protocol":"http","database":"sensors\_test","name":"","usetls":false,"tls":""},{"id":"3b5d2a26.3b75ce","type":"influxdb","z":"4f359664.ac5fa8","hostname":"127.0.0.1","port":"8086","protocol":"http","database":"sensors\_test","name":"","usetls":false,"tls":""},{"id":"83dac0cc.fca8e","type":"influxdb","z":"","hostname":"127.0.0.1","port":"8086","protocol":"http","database":"sensor\_test","name":"","usetls":false,"tls":""},{"id":"d407489a.b191d","type":"influxdb","z":"","hostname":"127.0.0.1","port":"8086","protocol":"http","database":"bitmex\_data","name":"","usetls":false,"tls":""},{"id":"f00fcaf6.761ae","type":"influxdb","z":"4f359664.ac5fa8","hostname":"127.0.0.1","port":"8086","database":"aTimeSeries","name":"aTimeSeries"},{"id":"a732981.1becbe8","type":"ui\_tab","z":"","name":"Home","icon":"dashboard","disabled":false,"hidden":false},{"id":"61e3fac3.47455c","type":"ui\_group","z":"","name":"Waves","tab":"be70b47e.65f4c8","disp":true,"width":"6","collapse":false},{"id":"67c89e77.52c9d8","type":"darksky-credentials","z":"","key\_identifier":"MichaelKatsoulis' Darksky Key"},{"id":"b07f5597.19ce88","type":"darksky-credentials","z":"","key\_identifier":"Testing"},{"id":"229ff8c2.3a7f58","type":"ui\_group","z":"","name":"weather","tab":"a732981.1becbe8","order":2,"disp":true,"width":"12","collapse":true},{"id":"541654f5.ff5e6c","type":"ui\_group","z":"","name":"Wetter Aktuell","tab":"","order":1,"disp":true,"width":"6","collapse":false},{"id":"be70b47e.65f4c8","type":"ui\_tab","z":"","name":"Sensor Readings","icon":"dashboard","disabled":false,"hidden":false},{"id":"e7d37254.807f38","type":"ui\_base","theme":{"name":"theme-dark","lightTheme":{"default":"#0094CE","baseColor":"#0094CE","baseFont":"-apple-system,BlinkMacSystemFont,Segoe UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif","edited":true,"reset":false},"darkTheme":{"default":"#097479","baseColor":"#010028","baseFont":"-apple-system,BlinkMacSystemFont,Segoe UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif","edited":true,"reset":false},"customTheme":{"name":"Untitled Theme 1","default":"#4B7930","baseColor":"#4B7930","baseFont":"-apple-system,BlinkMacSystemFont,Segoe UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif","reset":false},"themeState":{"base-color":{"default":"#097479","value":"#097479","edited":true},"page-titlebar-backgroundColor":{"value":"#097479","edited":false},"page-backgroundColor":{"value":"#111111","edited":false},"page-sidebar-backgroundColor":{"value":"#000000","edited":false},"group-textColor":{"value":"#0eb8c0","edited":false},"group-borderColor":{"value":"#555555","edited":false},"group-backgroundColor":{"value":"#333333","edited":false},"widget-textColor":{"value":"#eeeeee","edited":false},"widget-backgroundColor":{"value":"#097479","edited":false},"widget-borderColor":{"value":"#333333","edited":false},"base-font":{"value":"-apple-system,BlinkMacSystemFont,Segoe UI,Roboto,Oxygen-Sans,Ubuntu,Cantarell,Helvetica Neue,sans-serif"}},"angularTheme":{"primary":"indigo","accents":"blue","warn":"red","background":"grey"}},"site":{"name":"Node-RED Dashboard","hideToolbar":"false","allowSwipe":"false","lockMenu":"false","allowTempTheme":"true","dateFormat":"DD/MM/YYYY","sizes":{"sx":48,"sy":48,"gx":6,"gy":6,"cx":10,"cy":6,"px":5,"py":0}}},{"id":"d8e70ed1.12d8c","type":"ui\_group","z":"","name":"Airtemp","tab":"be70b47e.65f4c8","disp":true,"width":"6","collapse":false},{"id":"2beb61af.17db4e","type":"ui\_group","z":"","name":"Water temp","tab":"be70b47e.65f4c8","disp":true,"width":"6","collapse":false},{"id":"bf40deef.7a32c","type":"ui\_group","z":"","name":"Windspeed","tab":"be70b47e.65f4c8","disp":true,"width":"6","collapse":false},{"id":"2fdb233b.2c277c","type":"ui\_spacer","name":"spacer","group":"bf40deef.7a32c","order":3,"width":1,"height":1},{"id":"b9ce92aa.1863a8","type":"serial-port","z":"","serialport":"/dev/ttyUSB0/","serialbaud":"57600","databits":"8","parity":"none","stopbits":"1","waitfor":"","dtr":"none","rts":"none","cts":"none","dsr":"none","newline":"\\n","bin":"false","out":"char","addchar":"","responsetimeout":"10000"},{"id":"4058cc6f.815b14","type":"ui\_spacer","name":"spacer","group":"61e3fac3.47455c","order":3,"width":1,"height":1},{"id":"bf15a0bc.ac9af8","type":"inject","z":"75f0c5bb.be131c","name":"Fetch file","topic":"","payload":"","payloadType":"date","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":100,"y":380,"wires":[["55a6dca2.8a0114"]]},{"id":"1a394972.4f412f","type":"csv","z":"75f0c5bb.be131c","name":"","sep":",","hdrin":true,"hdrout":"","multi":"one","ret":"\\n","temp":"","skip":"0","strings":true,"x":510,"y":440,"wires":[["221c2f4a.c9d158","85c942f5.a5dd6"]]},{"id":"b81ffb58.1370a","type":"debug","z":"75f0c5bb.be131c","name":"","active":false,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","targetType":"msg","x":1510,"y":560,"wires":[]},{"id":"55a6dca2.8a0114","type":"file in","z":"75f0c5bb.be131c","name":"","filename":"/home/user/Documents/SHARC Wave Bouy/Data/Raw data/bitmex\_xbtusd\_1m\_2016-12-31\_2018-06-17.csv","format":"lines","chunk":false,"sendError":false,"encoding":"none","x":590,"y":340,"wires":[["1a394972.4f412f"]]},{"id":"219fecf0.e5fb34","type":"debug","z":"75f0c5bb.be131c","name":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"false","x":1510,"y":680,"wires":[]},{"id":"4d00399b.cb8578","type":"function","z":"75f0c5bb.be131c","name":"Remove id field","func":"var n\_json = msg.payload;\ndelete msg.payload.id;\nreturn msg;\n","outputs":1,"noerr":0,"x":1260,"y":680,"wires":[["219fecf0.e5fb34"]]},{"id":"a432f8c2.cc7978","type":"inject","z":"75f0c5bb.be131c","name":"Inject reset","topic":"","payload":"","payloadType":"date","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":980,"y":120,"wires":[["9a2e5c81.0c747"]]},{"id":"6ee629e.bd4ba58","type":"debug","z":"75f0c5bb.be131c","name":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"false","x":1480,"y":140,"wires":[]},{"id":"efebcae7.f4df5","type":"inject","z":"75f0c5bb.be131c","name":"Increment position","topic":"","payload":"","payloadType":"date","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":1010,"y":180,"wires":[["b76cf306.625358"]]},{"id":"b76cf306.625358","type":"function","z":"75f0c5bb.be131c","name":"increment counter","func":"var c = global.get(\"counter\");\nglobal.set(\"counter\", c + 1);\nmsg.payload = c +1;\nreturn msg;","outputs":1,"noerr":0,"x":1260,"y":180,"wires":[["6ee629e.bd4ba58"]]},{"id":"515e663d.eb1d28","type":"function","z":"75f0c5bb.be131c","name":"Check if it's been added","func":"var c = global.get(\"counter\");\nif ( msg.payload[\"id\"] > c){\n global.set(\"counter\", c + 1);\n return msg;\n}\n","outputs":1,"noerr":0,"x":1250,"y":560,"wires":[["4d00399b.cb8578","b81ffb58.1370a"]]},{"id":"a34f4f4f.be19f","type":"watch","z":"75f0c5bb.be131c","name":"","files":".\\University of Cape Town\\floatyBoi Vac work 2019-2020 - Documents\\Data work\\Ship Example Data\\Export Data.csv","recursive":"","x":410,"y":220,"wires":[["55a6dca2.8a0114"]]},{"id":"9a2e5c81.0c747","type":"function","z":"75f0c5bb.be131c","name":"Reset counter","func":"global.set(\"counter\", 0);\nmsg.payload = global.get(\"counter\");\nreturn msg;\n","outputs":1,"noerr":0,"x":1260,"y":120,"wires":[["6ee629e.bd4ba58"]]},{"id":"c1a5a0f8.26bbc8","type":"debug","z":"75f0c5bb.be131c","name":"","active":false,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","targetType":"msg","x":1390,"y":460,"wires":[]},{"id":"51145a49.22c93c","type":"switch","z":"8d2e2eb6.647758","name":"Routing based on sender","property":"from","propertyType":"msg","rules":[{"t":"eq","v":"TODO","vt":"str"},{"t":"eq","v":"michaelkatsoulis","vt":"str"},{"t":"else"}],"checkall":"true","repair":false,"outputs":3,"x":330,"y":240,"wires":[["23639a75.d8d52e"],[],[]],"inputLabels":["Email-in"],"outputLabels":["Satellite data feed","Testing @michaelkatsoulis","Otherwise"]},{"id":"23639a75.d8d52e","type":"switch","z":"8d2e2eb6.647758","name":"Has attatchments","property":"attachments","propertyType":"msg","rules":[{"t":"nnull"},{"t":"null"}],"checkall":"true","repair":false,"outputs":2,"x":610,"y":320,"wires":[["ee84826d.97e168","8f62d6b7.4dc85"],["d788cb72.56348"]],"outputLabels":["Has attatchments","No attatchments"]},{"id":"ee84826d.97e168","type":"debug","z":"8d2e2eb6.647758","name":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"true","targetType":"full","x":930,"y":200,"wires":[]},{"id":"8ac29570.377918","type":"comment","z":"8d2e2eb6.647758","name":"What this does","info":"This will output what the attatchements are to the debug terminal\n\nThis data will need to either be proccessed elsewhere or it will need to be proccessed here. \n\nThis flow can be used to run the python script to fetch and decode the data using the exec node\n","x":620,"y":100,"wires":[]},{"id":"5765f535.bf594c","type":"exec","z":"8d2e2eb6.647758","command":"","addpay":true,"append":"","useSpawn":"false","timer":"","oldrc":false,"name":"Extract data into .csv","x":1140,"y":440,"wires":[["855e6e8f.0f462"],["cd70b272.44e2e8"],[]]},{"id":"855e6e8f.0f462","type":"link out","z":"8d2e2eb6.647758","name":"Link to .csv reader","links":[],"x":1350,"y":380,"wires":[],"l":true},{"id":"cd70b272.44e2e8","type":"debug","z":"8d2e2eb6.647758","name":"Report error","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","targetType":"msg","x":1390,"y":440,"wires":[]},{"id":"815c2dfb.c28fd8","type":"inject","z":"8d2e2eb6.647758","name":"","topic":"","payload":"","payloadType":"date","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":120,"y":60,"wires":[["8dd45ebd.e393b8"]]},{"id":"7cc9192a.9db4e","type":"debug","z":"8d2e2eb6.647758","name":"","active":false,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","targetType":"msg","x":350,"y":120,"wires":[]},{"id":"f825aaa3.1ea738","type":"debug","z":"8d2e2eb6.647758","name":"","active":false,"tosidebar":true,"console":false,"tostatus":false,"complete":"attachments ","targetType":"msg","x":370,"y":160,"wires":[]},{"id":"d788cb72.56348","type":"debug","z":"8d2e2eb6.647758","name":"","active":false,"tosidebar":true,"console":false,"tostatus":false,"complete":"true","targetType":"full","x":830,"y":420,"wires":[]},{"id":"8f62d6b7.4dc85","type":"exec","z":"8d2e2eb6.647758","command":"python \"%USERPROFILE%\\University of Cape Town\\floatyBoi Vac work 2019-2020 - Documents\\Code\\Python Code\\Attachment Downloader\\AttachmentDownloader.py\"","addpay":true,"append":"","useSpawn":"false","timer":"","oldrc":false,"name":"Download Attatchments","x":970,"y":260,"wires":[["60ad2606.75123"],["60ad2606.75123"],["60ad2606.75123"]]},{"id":"60ad2606.75123","type":"debug","z":"8d2e2eb6.647758","name":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"true","targetType":"full","x":1230,"y":260,"wires":[]},{"id":"35f866fc.0c6512","type":"inject","z":"8d2e2eb6.647758","name":"","topic":"","payload":"","payloadType":"date","repeat":"","crontab":"","once":false,"onceDelay":0.1,"x":720,"y":180,"wires":[["8f62d6b7.4dc85"]]},{"id":"8dd45ebd.e393b8","type":"e-mail in","z":"8d2e2eb6.647758","name":"","protocol":"IMAP","server":"imap.gmail.com","useSSL":true,"port":"993","box":"INBOX","disposition":"None","criteria":"UNSEEN","repeat":"300","fetch":"trigger","inputs":1,"x":115.99993896484375,"y":127,"wires":[["51145a49.22c93c","7cc9192a.9db4e","f825aaa3.1ea738"]]},{"id":"44703d1f.de5adc","type":"sort","z":"75f0c5bb.be131c","name":"","order":"ascending","as\_num":false,"target":"","targetType":"seq","msgKey":"","msgKeyType":"elem","seqKey":"payload","seqKeyType":"msg","x":1110,"y":460,"wires":[["515e663d.eb1d28","c1a5a0f8.26bbc8"]]},{"id":"4546c318.cd118c","type":"debug","z":"75f0c5bb.be131c","name":"","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"true","targetType":"full","x":850,"y":560,"wires":[]},{"id":"221c2f4a.c9d158","type":"trigger","z":"75f0c5bb.be131c","op1":"","op2":"0","op1type":"pay","op2type":"str","duration":"0","extend":false,"units":"ms","reset":"","bytopic":"all","name":"","x":660,"y":560,"wires":[["4546c318.cd118c"]]},{"id":"529ee235.8cd9fc","type":"influxdb out","z":"75f0c5bb.be131c","influxdb":"d407489a.b191d","name":"","measurement":"btc\_test","precision":"n","retentionPolicy":"","x":850,"y":680,"wires":[]},{"id":"85c942f5.a5dd6","type":"function","z":"75f0c5bb.be131c","name":"Processing for influx","func":"\n// format of insert: |measurement,tag\_set field\_set timestamp|\nvar newPoint = {};\n\nnewPoint.measurement = \"btc-usd\"; //this is the mesurement for influx\n\n// storing the time for the timestamp\nvar mes\_time = (msg.payload.time)\*1000000000;// converting to ns time\ndelete msg.payload.time;\n\nnewPoint.fields = {};\n\nnewPoint.fields= msg.payload;\nnewPoint.fields.time = mes\_time;\n\n// msg.payload = [\n// [{\n// close:msg.payload.close,\n// high:msg.payload.high,\n// open:msg.payload.open,\n \n// time: mes\_time\n// },\n// {\n// tag1:\"sensor1\",\n// tag2:\"device2\"\n// },\n// {\n// measurement:newPoint.measurement\n// }\n// ]];\n\nmsg.payload = [\n [ //fields\n newPoint.fields,\n { //tags\n currencytag:\"btc\" \n },\n \n \n ]\n \n ];\n\n\nreturn msg;","outputs":1,"noerr":0,"x":560,"y":640,"wires":[[]]},{"id":"fc5f8f83.63e4b8","type":"serial in","z":"4f359664.ac5fa8","name":"Serial Input","serial":"b9ce92aa.1863a8","x":170,"y":240,"wires":[["644febba.99d394"]]},{"id":"ce2b5dbd.463b5","type":"comment","z":"4f359664.ac5fa8","name":"Serial input from out STM/Arduino","info":"","x":240,"y":160,"wires":[]},{"id":"644febba.99d394","type":"function","z":"4f359664.ac5fa8","name":"Serial to weather values","func":"\n\n//convert from read string into:\n\n// msg.payload:\n// .sensors:\n// .airTemp\n// .waterTemp\n// .airSpeed\n// .waveHeight\n// .waveDirection\n \n\n\n\nreturn msg;","outputs":1,"noerr":0,"x":390,"y":240,"wires":[["59627782.4297e8"]]},{"id":"b8fe61a0.67e168","type":"ui\_chart","z":"4f359664.ac5fa8","name":"Windspeed chart","group":"bf40deef.7a32c","order":4,"width":0,"height":0,"label":"Windspeed","chartType":"line","legend":"false","xformat":"HH:mm:ss","interpolate":"linear","nodata":"","dot":true,"ymin":"","ymax":"","removeOlder":1,"removeOlderPoints":"","removeOlderUnit":"60","cutout":0,"useOneColor":false,"colors":["#1f77b4","#aec7e8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"useOldStyle":false,"outputs":1,"x":1170,"y":260,"wires":[[]]},{"id":"a6b5161f.243c5","type":"ui\_chart","z":"4f359664.ac5fa8","name":"","group":"61e3fac3.47455c","order":4,"width":0,"height":0,"label":"Wave height","chartType":"line","legend":"false","xformat":"HH:mm:ss","interpolate":"linear","nodata":"","dot":false,"ymin":"","ymax":"","removeOlder":1,"removeOlderPoints":"","removeOlderUnit":"60","cutout":0,"useOneColor":false,"colors":["#1f77b4","#aec7e8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"useOldStyle":false,"outputs":1,"x":1150,"y":380,"wires":[[]]},{"id":"b8909bef.717e48","type":"ui\_chart","z":"4f359664.ac5fa8","name":"Water temp chart","group":"2beb61af.17db4e","order":1,"width":0,"height":0,"label":"Water Temp","chartType":"line","legend":"false","xformat":"HH:mm:ss","interpolate":"linear","nodata":"","dot":true,"ymin":"","ymax":"","removeOlder":1,"removeOlderPoints":"","removeOlderUnit":"60","cutout":0,"useOneColor":false,"colors":["#1f77b4","#aec7e8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"useOldStyle":false,"outputs":1,"x":1170,"y":200,"wires":[[]]},{"id":"e15ee97.89dcb98","type":"ui\_chart","z":"4f359664.ac5fa8","name":"Airtemp Chart","group":"d8e70ed1.12d8c","order":1,"width":0,"height":0,"label":"Air temp","chartType":"line","legend":"false","xformat":"HH:mm:ss","interpolate":"linear","nodata":"","dot":true,"ymin":"","ymax":"","removeOlder":1,"removeOlderPoints":"","removeOlderUnit":"60","cutout":0,"useOneColor":false,"colors":["#1f77b4","#aec7e8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"useOldStyle":false,"outputs":1,"x":1160,"y":140,"wires":[[]]},{"id":"8ee2ac8f.9e7128","type":"ui\_gauge","z":"4f359664.ac5fa8","name":"Wind Direction","group":"bf40deef.7a32c","order":1,"width":0,"height":0,"gtype":"compass","title":"Wind Direction","label":"Degrees","format":"{{}}","min":0,"max":"359","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":1160,"y":320,"wires":[]},{"id":"7ffd2edc.cf129","type":"ui\_gauge","z":"4f359664.ac5fa8","name":"Wave Direction","group":"61e3fac3.47455c","order":1,"width":0,"height":0,"gtype":"compass","title":"gauge","label":"units","format":"{{}}","min":0,"max":"359","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":1160,"y":440,"wires":[]},{"id":"59627782.4297e8","type":"switch","z":"4f359664.ac5fa8","name":"","property":"payload","propertyType":"msg","rules":[{"t":"eq","v":"msg.payload.sensor.airtemp","vt":"str"},{"t":"eq","v":"msg.payload.sensor.watertemp","vt":"str"},{"t":"eq","v":"msg.payload.sensor.windspeed","vt":"str"},{"t":"eq","v":"msg.payload.sensor.winddirection","vt":"str"},{"t":"eq","v":"msg.payload.sensor.waveheight","vt":"str"},{"t":"eq","v":"wavedirecton","vt":"str"}],"checkall":"true","repair":false,"outputs":6,"x":570,"y":240,"wires":[["84bf8b35.237718"],["e0961b27.0f6e5"],["4e6d8272.469e54"],["af974e20.dfda48"],["8a6d950a.4d7bd8"],["6b007306.49255c"]]},{"id":"84bf8b35.237718","type":"ui\_slider","z":"4f359664.ac5fa8","name":"","label":"Airtemp slider","tooltip":"","group":"d8e70ed1.12d8c","order":2,"width":"5","height":"1","passthru":true,"outs":"all","topic":".sensor.airtemp","min":"-30","max":"60","step":1,"x":860,"y":140,"wires":[["e15ee97.89dcb98"]]},{"id":"e0961b27.0f6e5","type":"ui\_slider","z":"4f359664.ac5fa8","name":"","label":"Water temp slider","tooltip":"","group":"2beb61af.17db4e","order":2,"width":0,"height":0,"passthru":true,"outs":"all","topic":"","min":"-20","max":"40","step":1,"x":870,"y":200,"wires":[["b8909bef.717e48"]]},{"id":"4e6d8272.469e54","type":"ui\_slider","z":"4f359664.ac5fa8","name":"","label":"Wind speed slider","tooltip":"","group":"bf40deef.7a32c","order":5,"width":0,"height":0,"passthru":true,"outs":"all","topic":"","min":0,"max":"100","step":1,"x":870,"y":260,"wires":[["b8fe61a0.67e168"]]},{"id":"af974e20.dfda48","type":"ui\_slider","z":"4f359664.ac5fa8","name":"","label":"Wind direction slider","tooltip":"","group":"bf40deef.7a32c","order":2,"width":0,"height":0,"passthru":true,"outs":"all","topic":"","min":0,"max":"359","step":1,"x":880,"y":320,"wires":[["8ee2ac8f.9e7128"]]},{"id":"8a6d950a.4d7bd8","type":"ui\_slider","z":"4f359664.ac5fa8","name":"","label":"Wave height slider","tooltip":"","group":"61e3fac3.47455c","order":5,"width":0,"height":0,"passthru":true,"outs":"all","topic":"","min":0,"max":10,"step":1,"x":870,"y":380,"wires":[["a6b5161f.243c5"]]},{"id":"6b007306.49255c","type":"ui\_slider","z":"4f359664.ac5fa8","name":"","label":"Wave Direction slider","tooltip":"","group":"61e3fac3.47455c","order":2,"width":0,"height":0,"passthru":true,"outs":"all","topic":"","min":0,"max":"359","step":1,"x":880,"y":440,"wires":[["7ffd2edc.cf129"]]},{"id":"b8878430.dc725","type":"comment","z":"4f359664.ac5fa8","name":"TODO: Finish sorting of inputs","info":"","x":380,"y":280,"wires":[]},{"id":"c3a558b6.bda848","type":"comment","z":"4f359664.ac5fa8","name":"Route to dials","info":"","x":610,"y":160,"wires":[]},{"id":"58d8c854.9f938","type":"comment","z":"4f359664.ac5fa8","name":"Need to write to .csv","info":"","x":430,"y":600,"wires":[]},{"id":"7a0b8f9f.e4a1e","type":"comment","z":"8d2e2eb6.647758","name":"README for info","info":"This flow executes scripts stored locally using the terminal\n\n\nThis was designed for a windows machine so \*\*will not \*\* work at the moment. The scripts are will need to be brought in from either the google drive or from the microsoft share drive.","x":100,"y":20,"wires":[]}]

# Errors Log

Node-red:

Cannot connect to influxdb (docker influxdb)

* Struggled to get this to work
* I tried to troubleshoot the networking between containers.
* I tried using a dns proxy server so that nodered could access containers by their names instead of IP addresses

Influxdb (not the docker version)

* run: open server: listen: listen tcp 127.0.0.1:8088: bind: address already in use
  + Work out what is running on port 8088 and kill it. Then restart
  + Normally is another instance of influxdb.
  + Sometimes if influxdb was run as a service on the machine it will not shut down but will be running in the background immediately after startup