National College of Ireland

BSc in Computing

2017/2018

Darragh McKernan

x14516637

darraghmckernan@gmail.com

Atmospheric reading device.

Technical Report

Table of Contents

Web App Link: https://atmospheric-reading-device.shinyapps.io/Downloads/

Executive Summary

The objective of this project is to measure and store atmospheric conditions, there are many options for ways the data to be used for example people working in facility management could use the data to make sure their facilities are all safe and up to code, the data may also be used by companies to analyse the working conditions of their spaces/offices. The will be a combination of an Arduino board independent sensor and Raspberry pi. The sensor will detect conditions and pass the data to the Arduino board for processing and will be passed again to be stored on the Raspberry pi. This data will be able to be accessed via the Raspberry pi and be displayed differently depending on what type of user the system is dealing with, if a general user with little technical knowledge of the data is using the system they will have access to mostly graphically displayed data, while users with advanced technical knowledge in the system and the data will have access to the raw data. The Device produced will have great commercial potential for industrial and possibly even home use. This will be because of the expense of the parts in general are cheap and in future evolutions it could evolve greatly and expand to cover a much wider area and gather more valuable data that would be of great use.

1 Introduction

1.1 Background

the background of this project includes many forms of people using Arduino boards and sensors to collect/monitor the atmosphere of an area. There are many examples of people using the same sort of hardware to monitor the atmosphere in certain areas such as example 1. In this example the aim is to monitor the CO level of an area, this project differs from these as it takes in the data and will save it. The saved data will be able to be recalled and analysed for many purposes. Such purposes would include using the raw data in case studies where atmospheric readings are applicable and for the use of companies and people running long term monitoring on spaces, this gives the project in question great commercial potential.

1.2 Aims

Data Gathering

This project should be able to gather data actively via the sensor, the data gathered by the sensor will be processed by the Arduino board and passed via network to the Database/ Ubuntu system on the Raspberry pi.

Data Storage

This project will store the data that has been passed on from the Arduino board and sensor to the Raspberry Pi and store it in a manner which is easily analysed and able to be accessed at a later date

Data Presentation and Analysis

The project should be able to give users access to the raw data and graphically displayed data that is easy to analyse and draw conclusions from. there should be processes that can display the data compared to other data stored on the system.

1.3 Technologies

There will be many different technologies used in the making of this project.

The hardware used in this project is the Arduino Uno R3 circuit board this is the main controller of the sensor for detecting carbon monoxide gas, this is the MQ-7 sensor which will take readings from the environment and pass it on to the Arduino board where it is pre-processed into serial output and sent on via serial port to the raspberry pi. This will read the serial output of the Arduino board and process it into a SQL query that saves it to the database on the raspberry pi. the raspberry pi is running on Raspbian which is a Debian Linux operating system specifically built for raspberry pi's and on that is the python script and the MySQL server. multiple technologies are to be made use of on the pi such as screen

SSH and rsync prior to the database being made accessible over the internet. The main technology used in the production of the app is R and its many packages such as shiny and shiny dashboard for creating the web content ggplot and plotly for the creation of the plots of the data and DBI, dbConnect and RMySQL for connecting to and querying from the database.

Definitions, Acronyms, and Abbreviations

AB	Arduino board
RP	Raspberry Pi

2 System

2.1 Introduction

this project will have many functional requirements and multiple types of users the prospective users should fall into two categories

General User

The general user will want access to the data via a graphical model e.g. using R Shiny to create graphic representations of the data to display to them in a manner that doesn't exceed the technical knowledge user and their knowledge of the data requested and its format

Advanced User

this type of user will require access to the raw data stored on the database before it has gone through any processing for analysis. these users may require the data to make a case study of their own and manipulate it to their needs allowing them to get precisely what they want from the data.

2.2 Requirements

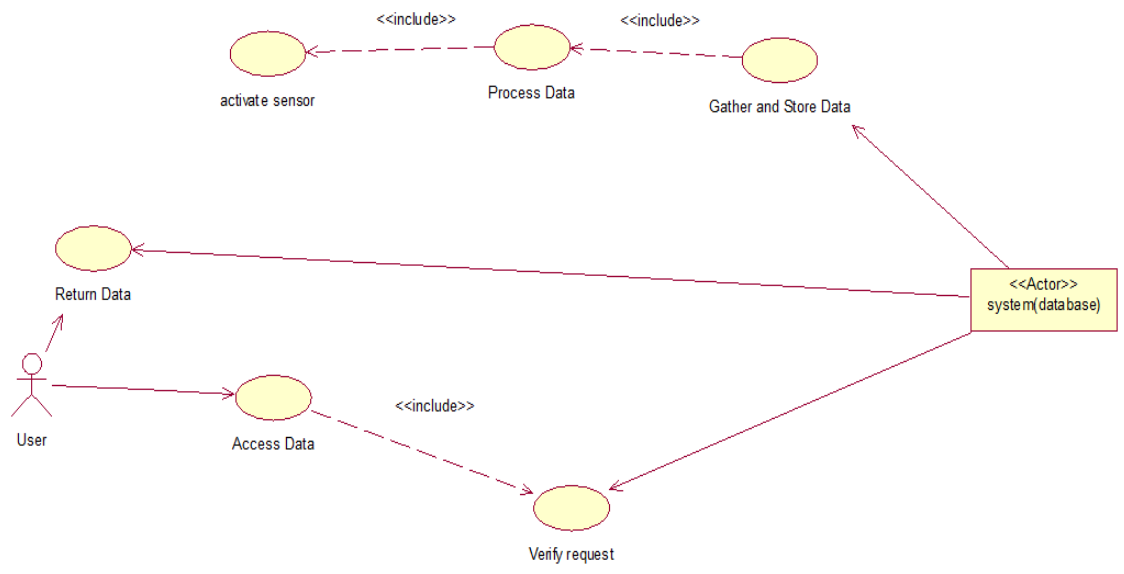
Depending on what users access the system they should have access to either the raw data stored on the database or have access to the graphically represented data. the main device of the project should be portable and require minimal setup, once setup they shouldn't require any regular maintenance and should work autonomously.

Functional requirements

- The Sensor must be able to be activated to take readings from the atmosphere and pass that data onto the AB where it will be processed
- The AB must be able to receive the data from the sensor and process it into a format in which it can be stored.
- the RP must be able to take this processed data and store it to be viewed and requested at a later date

- the User must be able to make a request of some data from the system
- the system will verify the request from the user and gather the correct data
- the system will return the data to the user

Use Case Diagram



activate Sensor

Description & Priority

This case is essential because if the sensor wasn’t able to be activated there would be no data to store and analyse.

Scope

The scope of this use case is to activate the sensor so that it takes a reading of the atmosphere around it and passes the gathered data back to the AB

Description

This use case describes activating the sensor to take a reading

Use Case Diagram

Refer to previous Use Case diagram.

Flow Description

Precondition

all items needed to make the connection are present and connected correctly and the system is setup correctly.

Activation

This use case starts when the system makes a request from the AB to take a reading from the sensor.

Main flow

- The system makes a request for the sensor to take a reading
- the sensor makes reading
- the sensor then passes the gathered data to the AB

Alternate flow

A1: sensor activated at intervals

- The system is set to make request at certain intervals e.g. 5 mins
- The sensor makes the reading at the timed intervals
- The use case continues at position 3 of the main flow

A2: continuous sensor activation

- The system sets the sensor to constantly make readings of the atmosphere
- The will make readings constantly
- The use case continues at position 3 of the main flow

Termination

the sensor has passed data gathered from reading.

Post condition

The system goes into a wait state

Arduino Board takes in and processes data

Description & Priority

This requirement is where the data is gathered by the sensor is processed and formatted to be stored and is important as it acts as a bridge between the sensor and the main sensor

Scope

The scope of this use case is to take in data from the sensor, process it and pass it onto the RP.

Description

This use case describes the AB taking in the sensors reading and processing them and passing them on

Use Case Diagram

Refer to previous Use Case diagram.

Flow Description

Precondition

The sensor has made a reading from the atmosphere and passed the data to the AB.

Activation

This use case starts when the system requests a reading.

Main flow

- The sensor passes the data to the AB
- The AB takes in the data and processes it
- The AB passes on the processed and formatted data to the RP

Termination

The AB has passed the processed data to the RP

Post condition

The system goes into a wait state

Raspberry pi takes in data and stores it in a database

Scope

The scope of this use case is for the Raspberry pi takes in the processed data passed from the AB and stores it in a database

Description

This use case describes taking in and storing the processed data from the AB.

Use Case Diagram

Refer to previous Use Case diagram.

Flow Description

Precondition

Processed data has been passed to and taken in by the RP.

Activation

This use case starts when the processed data is returned by the AB to the RP.

Main flow

- The RP receives the data from the AB.
- The RP ensures the data is in the correct format.
- The RP then stores the data for later analysis.

Termination

the RP has stored the data in the database.

Post condition

The system goes into a wait state

User makes a request of the system

Scope

The scope of this use case is for the User to be able to request data that is stored on the system

Description

This use case describes the user making a request from the system for certain data

Use Case Diagram

Refer to previous Use Case diagram.

Flow Description

Precondition

The user has access to the system and there is data stored on the system

Activation

This use case starts when the user makes a request from the system for some data.

Main flow

- The user accesses the system
- The user selects what data it wants
- The user sends off the request to the system

Termination

The user has sent a request to the system

Post condition

The user goes into a wait state

Verify Request

Description & Priority

This case is essential because the RP needs to know what data to return to the user or if the user even made a valid request

activate Sensor

Scope

The scope of this use case is to for the system to verify the request of data from the user and return the correct data

Description

This use case describes verifying the request sent by the user

Use Case Diagram

Refer to previous Use Case diagram.

Flow Description

Precondition

The user has sent a request to the system and there is valid data stored on the system to be sent back.

Activation

This use case starts when the user makes a request of the system for data.

Main flow

- The system receives the request
- The system verifies the request and finds it valid. <else A1>
- The system finds the requested data and returns it

Alternate flow

A1: Invalid request

- The system finds the request by the user invalid
- The system returns an error to the user because of invalid request

Termination

the system has returned a response to the user

Post condition

The system goes into a wait state

Show response to user

Description & Priority

This case is essential because if the user cannot get a response the service and data is useless.

System response

Scope

The scope of this use case is show the system response to the user.

Description

This use case describes showing the response to the user.

Use Case Diagram

Refer to previous Use Case diagram.

Flow Description

Precondition

all items needed to make the connection are present and connected correctly and the system is setup correctly.

Activation

This use case starts when the system makes a request from the AB to take a reading from the sensor.

Main flow

- The user receives the response from the system.
- the responded data can be manipulated and analysed by the user

Alternate flow

A1: Error response

- The user is returned an error due to invalid request
- The user can append their request to make it valid

Termination

the user has received a response from the system

Post condition

The system goes into a wait state

Data requirements

- The data should be stored in a SQL server on the RP
- The data should show when it was measured
- The data should be accessible to the user with the device

User requirements

From the customer's point of view the data gathered should be valid and have no errors. The data should be secure but easily accessed and should be able to be viewed in a manner that would allow them to interpret it and draw conclusions from it. The clients also want reliable hardware that doesn't stop working at randomly at intervals and can be setup with minimal effort.

Environmental requirements

The environmental requirements are that the device can be used anywhere power and shelter is provided, also to get accurate readings the device should be in an enclosed space.

Usability requirements

The device should be able to be used autonomously, so that once set up in a certain area it will continue to run and scan the atmosphere until interrupted or stopped. The data should also be able to be accessed when you have access to the device.

Non-Functional Requirements

Performance/Response time requirement

the system must respond in a timely fashion when it receives a response from a user.

Availability requirement

the system should be reasonably available as the data will be stored on the RP and wherever that is the data should be accessible as of now there are no plans to allow access over the internet.

Reliability requirement

the system must be reliable and return valid readings on the atmosphere of the area in which its place otherwise the data gathered is invalid and no conclusions can reasonably be made from it.

Maintainability requirement

the system shouldn’t need much to be maintained as its and enclosed system it doesn’t even need internet access, power should be sufficient to keep it running. however, storage may be an issue as the data will be stored on the RP which only has an SD card, which long term can run out of space.

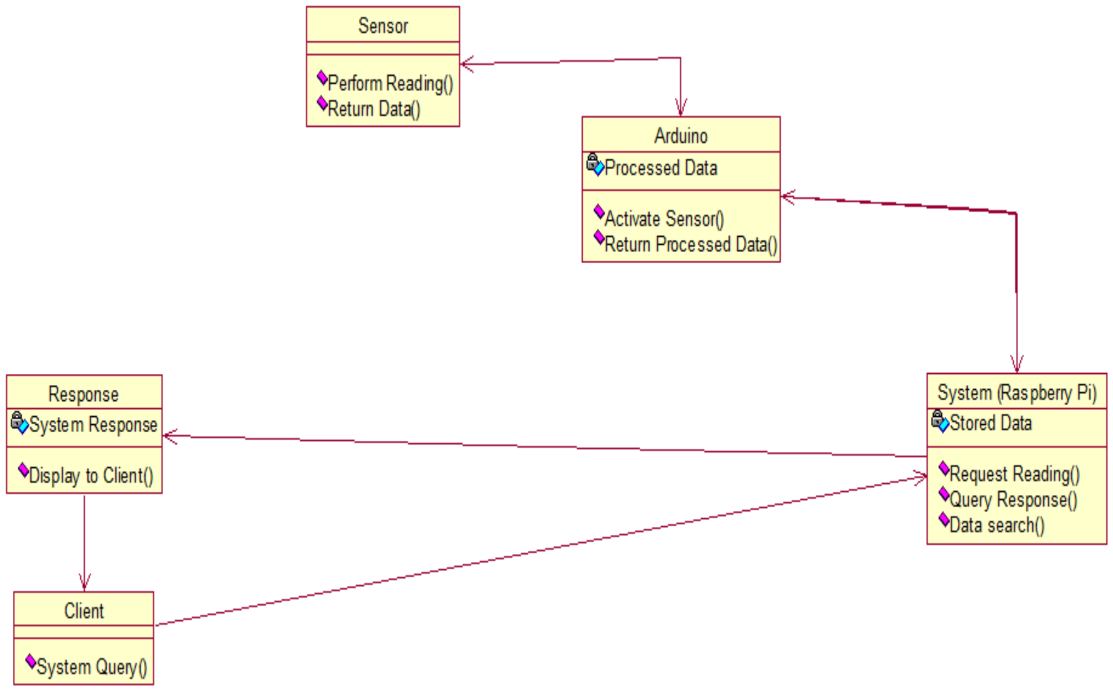
Portability requirement

the hardware involved in the system must be portable and should be able to be used in multiple locations.

Extensibility requirement

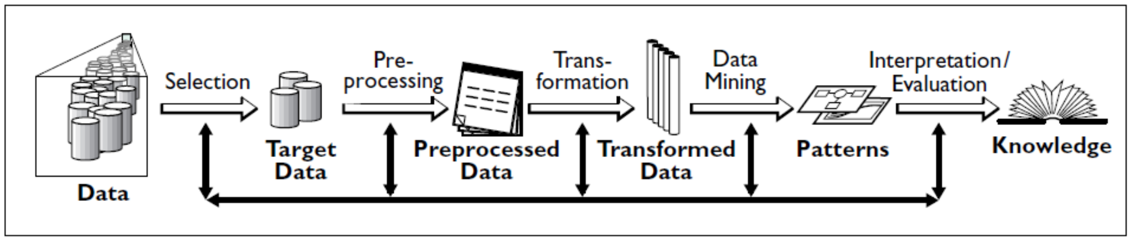
the system is reasonably extendable as adding more Arduino boards with different sensors can allow you to gather more data to draw more conclusions from.

2.3 Design and Architecture



KDD

Figure 1. Overview of the steps constituting the KDD process



Data

In this project the Data will be stored on the RP running on ubuntu Linux and will be within a sql server from which it can be accessed by admins and users

Target Data

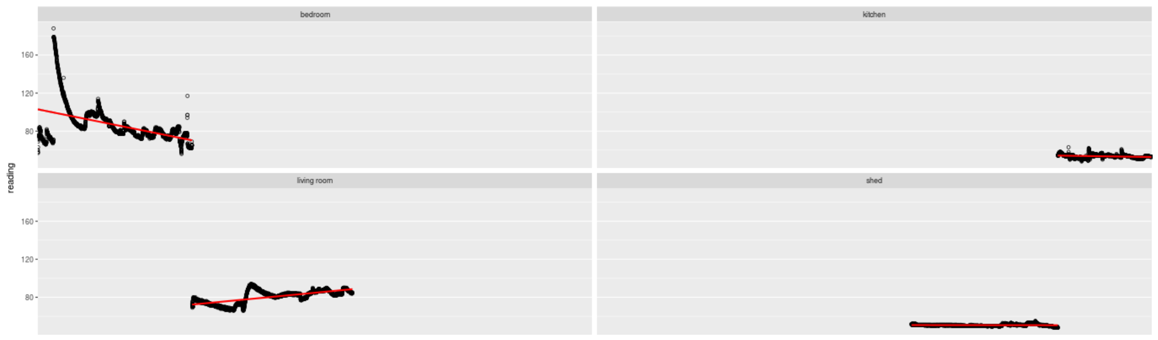
The target data in this project is any data stored on the system which has been requested by any type of user, the target data will be validated by the system to ensure that only the correct data is being returned when requested.

Processed Data

The Processing of the data in this project is done on the AB, the board takes in the data from the sensor processes it and passes it onto the RP to be stored. This means that when data is requested from the system only minimal processing and changes have to be made to it for the user to view it

Transformed Data

The data from the project has minimal transformation made to it when passed onto the advanced user. When a general user requests data there are many more transformations made to the target data such as changing it to be viewed using R Shiny so that they can analyse and draw conclusions from it. an example of data that has been graphed using R is as follows.



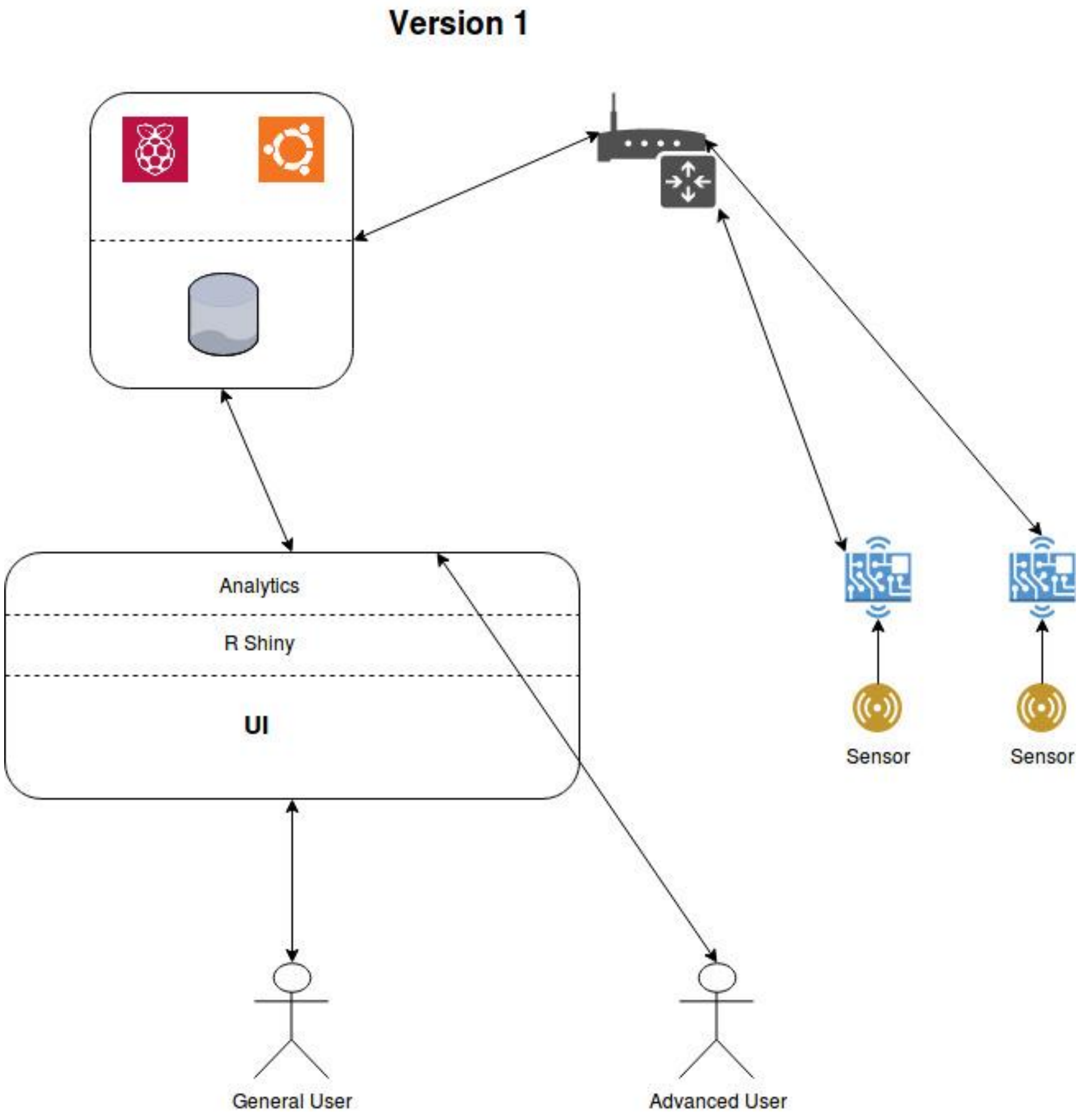
Patterns

At this point in the project the data is gathered by the user and comparisons are drawn an example of what patterns might be looked for is, At what atmospheric conditions are office workers most productive with these patterns drawn the user can be lead to new knowledge.

Evaluation/Knowledge

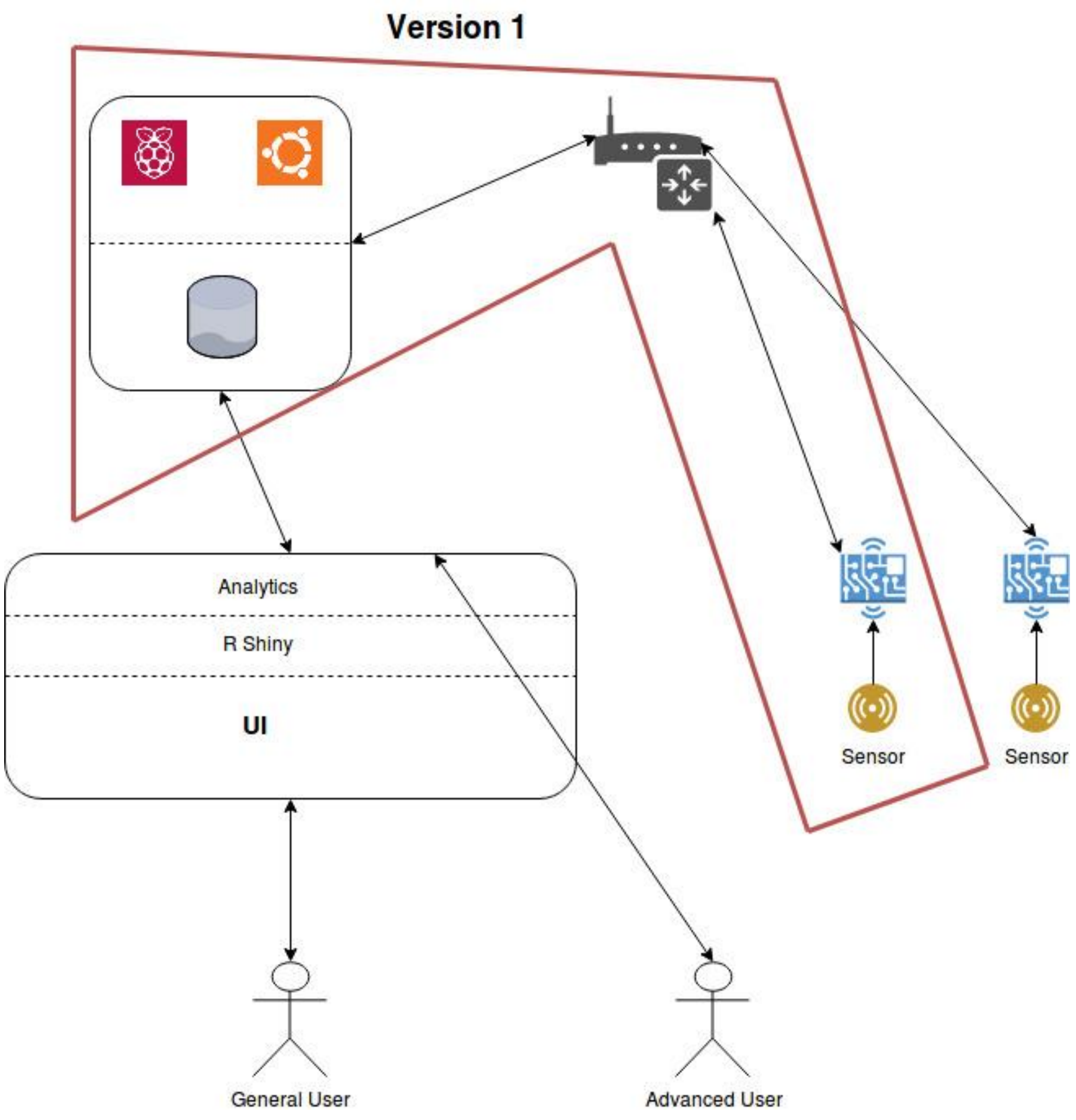
There are many evaluations that can be made using the target data in this project such if areas are up to legal code for the atmospheric conditions e.g. CO levels in a business or home. The Data can be compared to nearly all occurrences such as when a room is full or when it is empty, when the humidity spikes or falls, such data to companies who wish to be as efficient as possible and avoid any large costs such as fines, and home owners would want to know if their own home is safe to stay in.

2.4 Implementation



When the project is fully operational the whole system will work like this with the sensors attached to the AB's and the boards connected to a network of some sort and through this network they will connect to the RP which will be operating on a ubuntu distribution, this will act as a database/Server and store the data and handle the requests from the users

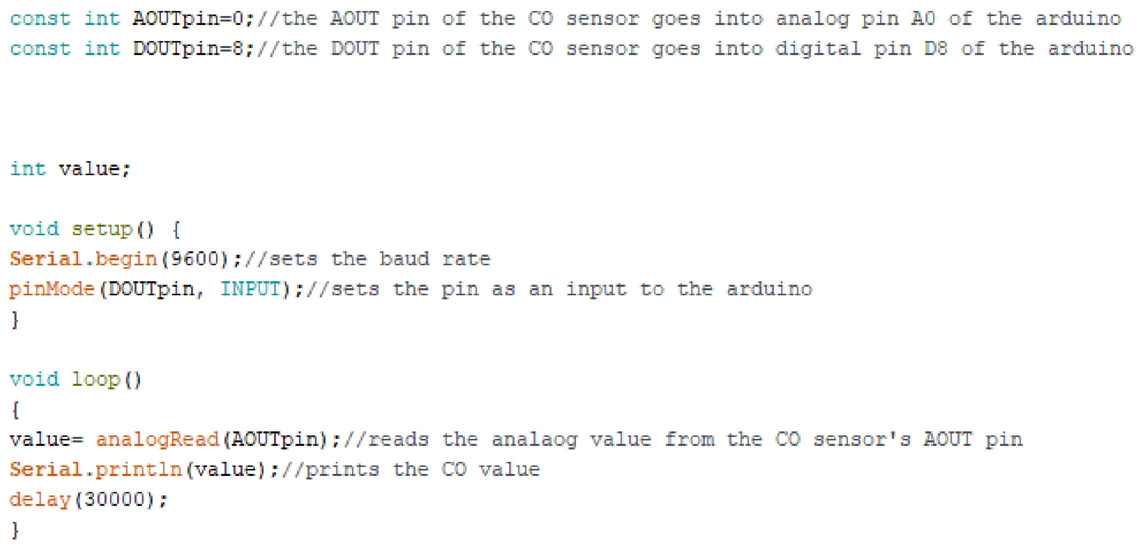
Version 1



The initial version of this project will feature a single sensor connected to a single AB and in turn connected to the RP/database. the sensor will run once and pass the data to the AB and from there it will be processed for storage. The processed data will then be passed to the RP and stored appropriately. access to the data will be limited to those who have access to the internal of the database and will not have any analytic functions or graphical displays built in at this point.

Arduino Board

The AB is set up with a sketch to make a reading from the sensor every 30 seconds. This reading is then printed through the serial port, this will continue in a loop as long as there is power going to the device. The pins the code reference are the pin ports the jumper cables are plugged into from the sensor

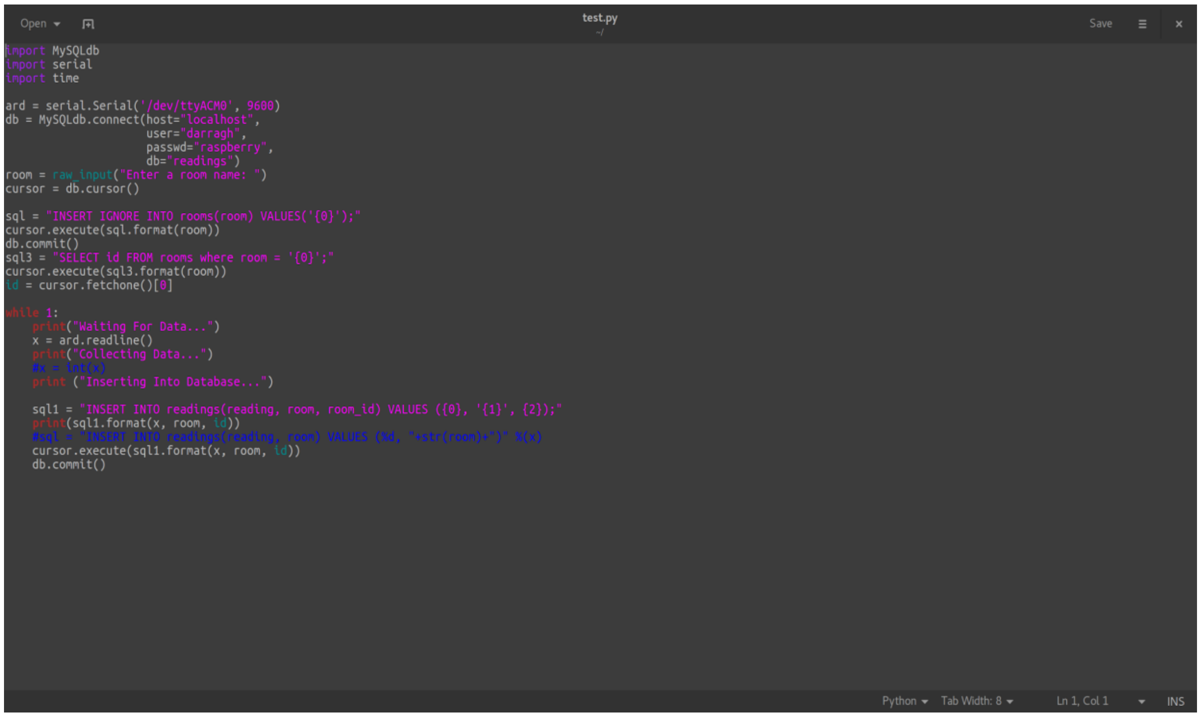


```
const int AOUTpin=0;//the AOUT pin of the CO sensor goes into analog pin A0 of the arduino
const int DOUTpin=8;//the DOUT pin of the CO sensor goes into digital pin D8 of the arduino



int value;

void setup() {
Serial.begin(9600);//sets the baud rate
pinMode(DOUTpin, INPUT);//sets the pin as an input to the arduino
}

void loop()
{
value= analogRead(AOUTpin);//reads the analaog value from the CO sensor's AOUT pin
Serial.println(value);//prints the CO value
delay(30000);
}
```

without the prompting of any other devices the AB will continue to cast its readings through its serial port even if the raspberry pi or any other device connected is reading its serial prints. the AB works at a standard baud rate of 9600 bps. This baud rate was chosen because speed was not of the essence in taking these readings and 9600 bps is one of the more common baud rate used across devices. The mq-7 sensor is sensitive between 20 and 2000 ppm when detecting carbon monoxide. It can give an analog reading of the carbon monoxide between 0 and 1023

Python script

The RP is running a Linux system(Raspbian) the system is lightweight and processes the data gathered from the AB. Initially the data is gathered through a python script.



```python
import MySQLdb
import serial
import time

ard = serial.Serial('/dev/ttyACM0', 9600)
db = MySQLdb.connect(host="localhost",
                     user="darragh",
                     passwd="raspberry",
                     db="readings")
room = raw_input("Enter a room name: ")
cursor = db.cursor()

sql = "INSERT IGNORE INTO rooms(room) VALUES('{0}');"
cursor.execute(sql.format(room))
db.commit()
sql3 = "SELECT id FROM rooms where room = '{0}';"
cursor.execute(sql3.format(room))
id = cursor.fetchone()[0]

while 1:
    print("Waiting For Data...")
    x = ard.readline()
    print("Collecting Data...")
    #x = int(x)
    print ("Inserting Into Database...")

    sql1 = "INSERT INTO readings(reading, room, room_id) VALUES ({0}, '{1}', {2});"
    print(sql1.format(x, room, id))
    #sql = "INSERT INTO readings(reading, room) VALUES (%d, "+str(room)+")" %(x)
    cursor.execute(sql1.format(x, room, id))
    db.commit()
```

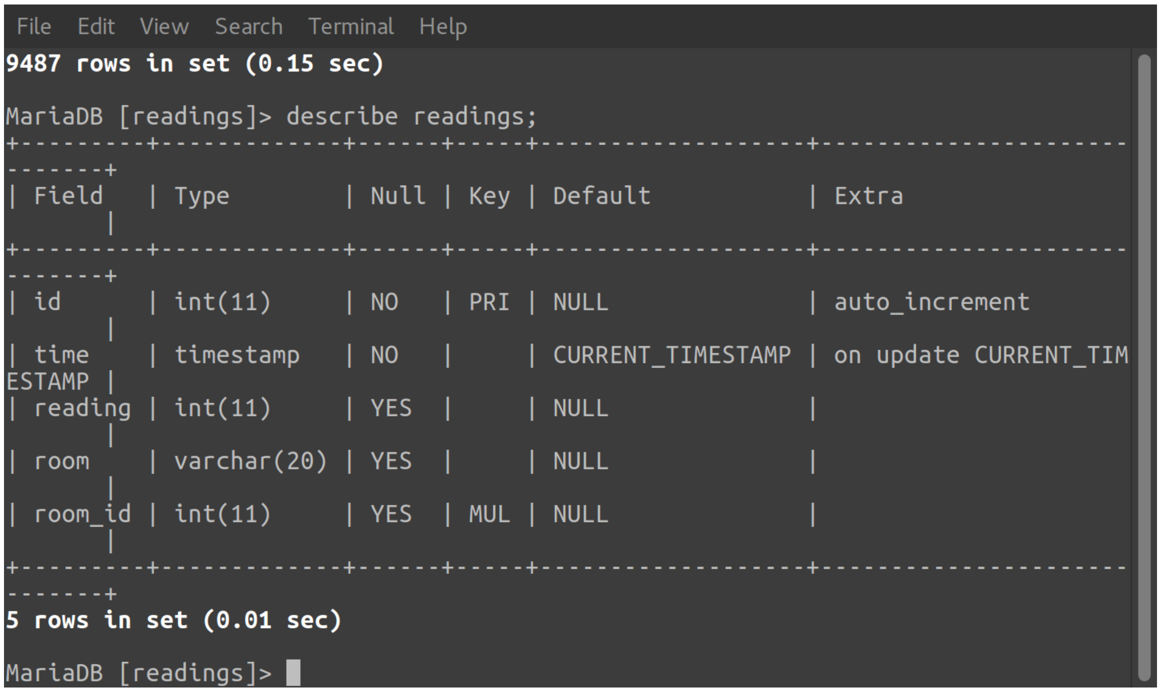
The correct serial port of the Arduino board was found using the command

```
$ python -m serial.tools.list_ports
```

this lists the ports available for python to be accessed. Using the import serial, the script was set to listen on the port /dev/ttyACM0 at the same baud rate of 9600 that the AB was printing its readings at. Then came one of the first software issues of the project, it came in the form of the import MySQL dB this import cannot be used with python 3 therefore when running the script, it is necessary to run with the command python 2.7 once this was established the script had obtained a connection with the SQL server installed on the Raspberry pi and connected to the DB readings. when run the console will ask the user to enter a room name e.g. bedroom and with this the initial insert into the Rooms table with the parameters to ignore any duplicate room names. After that the script will query the rooms table with the name entered by the end user and return its room id for later use. The main purpose of the script is then executed upon the AB's serial print the script gathers that data and processes it into a proper SQL command and enters it into the reading table along with the room name entered by the user and the room id gathered by the script.

MySQL Database

Initially the SQL database was created to only have one table for storing readings. it would store the reading an ID and room name, however to make querying the database less computationally strenuous it was decided to add a room table and room_id as a foreign key. However, when this change was made the timestamp for over half of the entries was updated to the current timestamp of when that change was made making much of the data gathered not eligible for time series analysis or graphing, this mistake was because in the database the timestamp is set to be updated when the row is rather than when it is created.

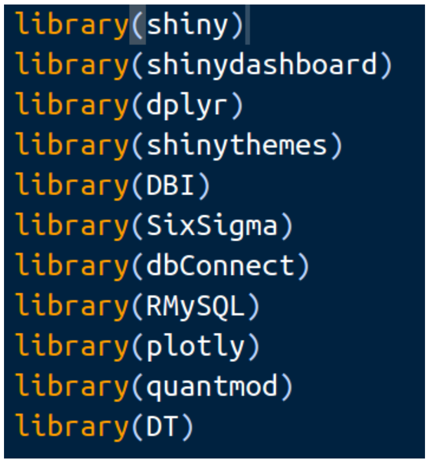


```
9487 rows in set (0.15 sec)

MariaDB [readings]> describe readings;
+---------+------------+------+-----+-------------------+---------------------
-------+
| Field   | Type       | Null | Key | Default           | Extra
       |
+---------+------------+------+-----+-------------------+---------------------
-------+
| id      | int(11)    | NO   | PRI | NULL              | auto_increment
       |
| time    | timestamp  | NO   |     | CURRENT_TIMESTAMP | on update CURRENT_TIM
ESTAMP |
| reading | int(11)    | YES  |     | NULL              |
       |
| room    | varchar(20) | YES  |     | NULL              |
       |
| room_id | int(11)    | YES  | MUL | NULL              |
       |
+---------+------------+------+-----+-------------------+---------------------
-------+
5 rows in set (0.01 sec)

MariaDB [readings]>
```

There are almost ten thousand rows of data gathered from four different environments. along with the design and implementation of the database it was required to be accessible not only on localhost or even local network, it had to be accessible via the web from anywhere as long as it had internet access. Therefore, the RP required a static IP and ports forwarded to it. This required networking and sysadmin work on both the main router in the household and the RP itself. The static IP given was 192.168.0.39 and the ports forwarded to this IP was port 3306. and with this a user was created who had access to the readings table from any IP allowing for Immediate updates on the web app. along with allowing access to the MySQL database SSH access was also enabled from anywhere in the world by enabling SSH on the RP and forwarding port 22 to the pi's IP. this along with the use of net gear plug links allowed access to the RP's internal system and allowed it to be deployed almost anywhere in the house. To keep the Raspberry pi secure Fail2Ban was installed on it, this will automatically block IP's that try to SSH into it and repeatedly get the password wrong in a brute force method of hacking into the raspberry pi. This will help protect the Raspberry Pi from malicious attacks and from becoming part of the botnet.

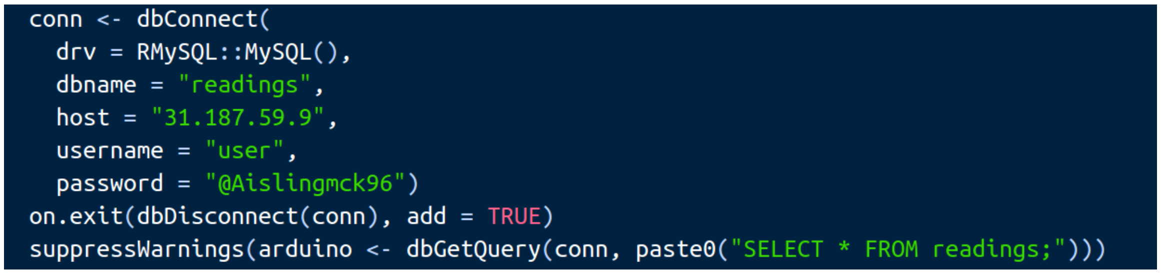
R Shiny

For This section R Shiny was chosen for the base of the web app because it is versatile and can handle large datasets like the one being used for this with over 10000 rows Many packages are required to make the web app



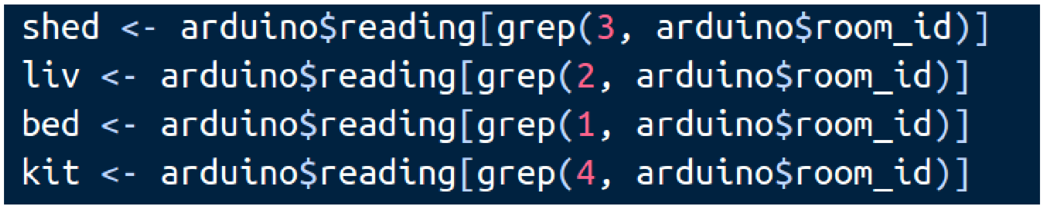
```
library(shiny)
library(shinydashboard)
library(dplyr)
library(shinythemes)
library(DBI)
library(SixSigma)
library(dbConnect)
library(RMySQL)
library(plotly)
library(quantmod)
library(DT)
```

Packages such as shiny and shiny dashboard are required to produce the html and CSS that the page is built on, shiny dashboard is similar to shiny but produces a cleaner looking web app with less clutter and gives you the ability to build the web app with each page layered on top of one another and allows you to place links to switch through these pages in a collapsible sidebar built in. shiny also gives you the ability to build more reactive web pages and graphs that can be changed on the fly. packages such as DBI dbConnect and RMySQL allow the web app to connect to interface with and query the database stored on the raspberry pi which it accesses through public IP address and user credentials with the right to access the readings database from any IP address.



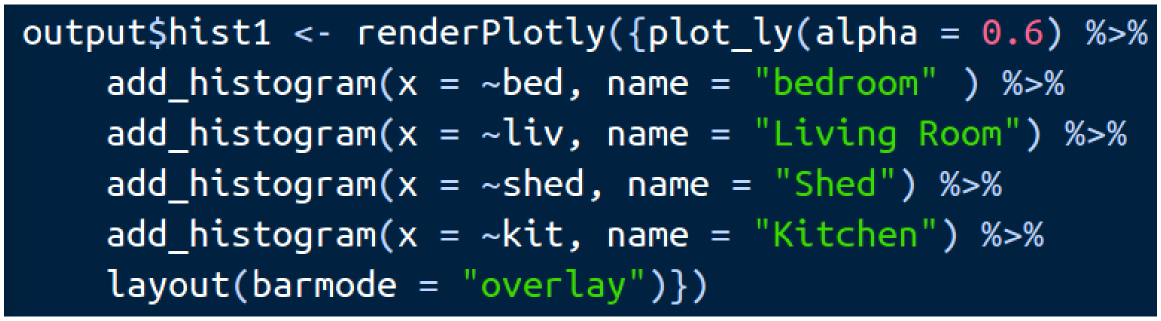
```
conn <- dbConnect(
  drv = RMySQL::MySQL(),
  dbname = "readings",
  host = "31.187.59.9",
  username = "user",
  password = "@Aislingmck96")
on.exit(dbDisconnect(conn), add = TRUE)
suppressWarnings(arduino <- dbGetQuery(conn, paste0("SELECT * FROM readings;")))
```

This code accesses the database and gathers all data from the table readings and stores it in a dataset called arduino. This code runs every time the web app is refreshed allowing the user to have the most up to date data from the database. Other packages such as plotly dplyr and DT are for the graphing and representing of data.



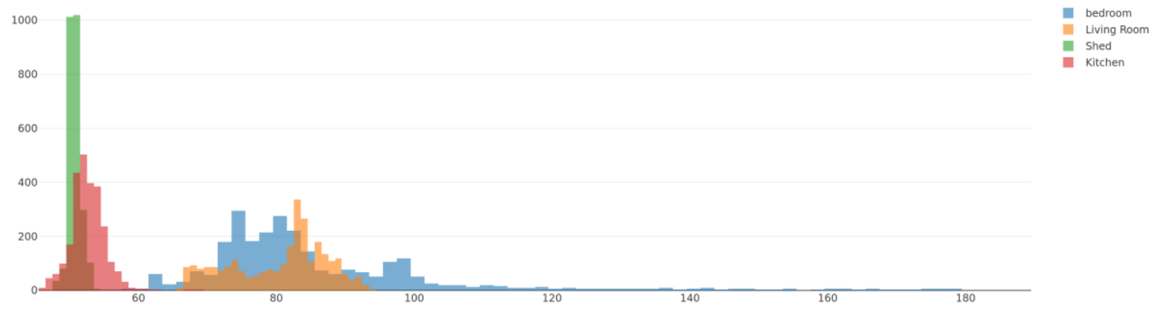
```
shed <- arduino$reading[grep(3, arduino$room_id)]
liv <- arduino$reading[grep(2, arduino$room_id)]
bed <- arduino$reading[grep(1, arduino$room_id)]
kit <- arduino$reading[grep(4, arduino$room_id)]
```

This code separates the rooms readings and places them into their own datasets, this would not be necessary if it weren’t for the issue with timestamps. once the readings are separated they can be graphed easily.

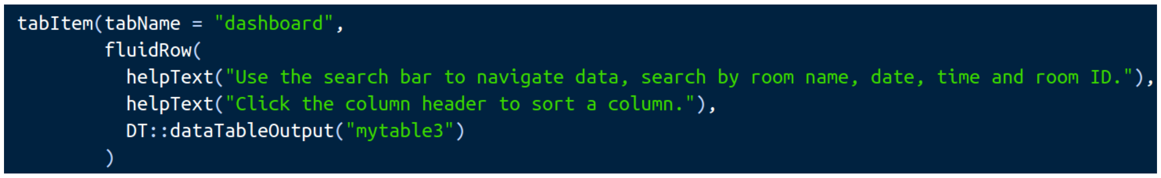


```
output$hist1 <- renderPlotly({plot_ly(alpha = 0.6) %>%
    add_histogram(x = ~bed, name = "bedroom" ) %>%
    add_histogram(x = ~liv, name = "Living Room") %>%
    add_histogram(x = ~shed, name = "Shed") %>%
    add_histogram(x = ~kit, name = "Kitchen") %>%
    layout(barmode = "overlay")})
```

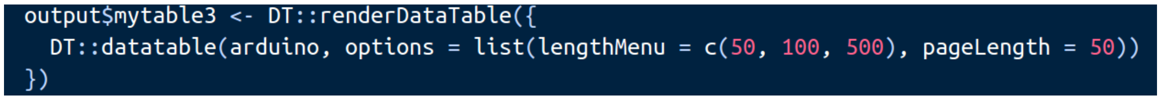
this code uses the plotly package to create histograms for the reading data.



This shows the spread of the data readings and clearly shows which rooms had a larger spread and gives a better idea of which datasets are statistically normal as with histograms the closer to a perfect bell curve it is, is an indicator of statistic normality. other uses for the Shiny package and DT package are to allow the user to completely search through the database using any parameter.

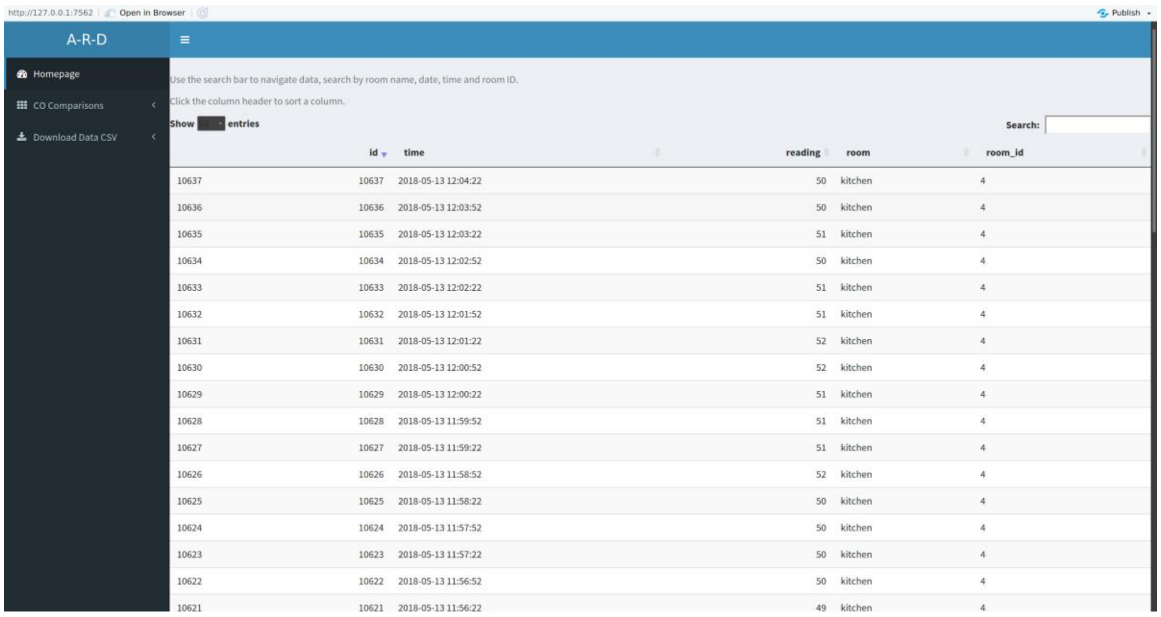


```
tabItem(tabName = "dashboard",
        fluidRow(
          helpText("Use the search bar to navigate data, search by room name, date, time and room ID."),
          helpText("Click the column header to sort a column."),
          DT::dataTableOutput("mytable3")
        )
```



```
output$mytable3 <- DT::renderDataTable({
  DT::datatable(arduino, options = list(lengthMenu = c(50, 100, 500), pageLength = 50))
})
```

this code will create a data table that will display all data pulled from the database and allow the user to sort through the data in any method they choose.



Use the search bar to navigate data, search by room name, date, time and room ID.

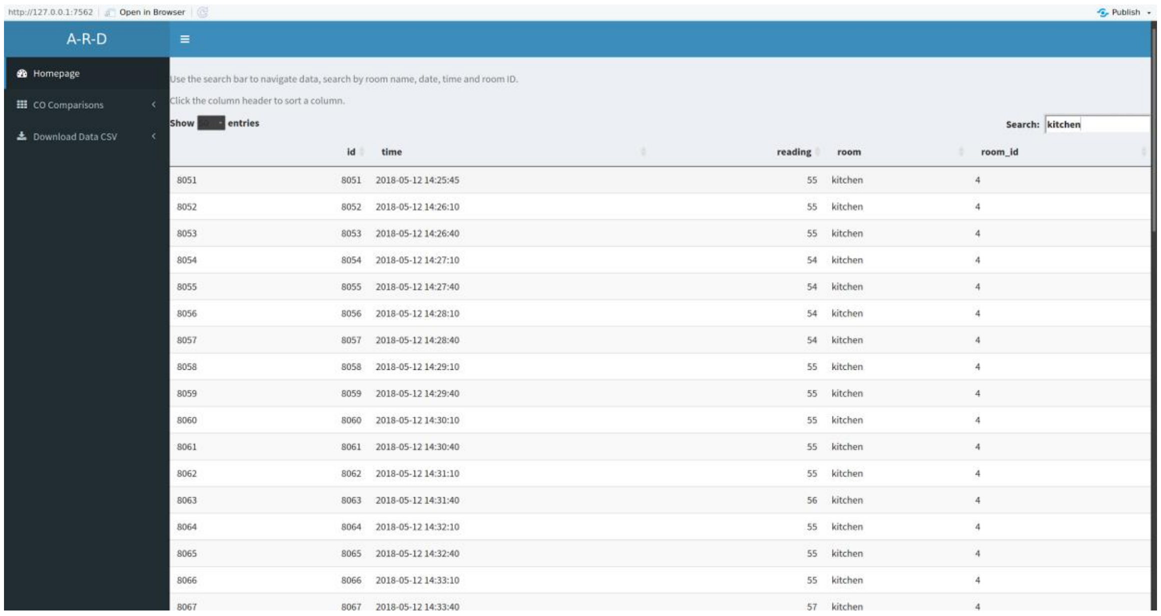
Click the column header to sort a column.

Show entries

Search:

	id	time	reading	room	room_id
10637	10637	2018-05-13 12:04:22	50	kitchen	4
10636	10636	2018-05-13 12:03:52	50	kitchen	4
10635	10635	2018-05-13 12:03:22	51	kitchen	4
10634	10634	2018-05-13 12:02:52	50	kitchen	4
10633	10633	2018-05-13 12:02:22	51	kitchen	4
10632	10632	2018-05-13 12:01:52	51	kitchen	4
10631	10631	2018-05-13 12:01:22	52	kitchen	4
10630	10630	2018-05-13 12:00:52	52	kitchen	4
10629	10629	2018-05-13 12:00:22	51	kitchen	4
10628	10628	2018-05-13 11:59:52	51	kitchen	4
10627	10627	2018-05-13 11:59:22	51	kitchen	4
10626	10626	2018-05-13 11:58:52	52	kitchen	4
10625	10625	2018-05-13 11:58:22	50	kitchen	4
10624	10624	2018-05-13 11:57:52	50	kitchen	4
10623	10623	2018-05-13 11:57:22	50	kitchen	4
10622	10622	2018-05-13 11:56:52	50	kitchen	4
10621	10621	2018-05-13 11:56:22	49	kitchen	4

here the user is sorting the entries by id descending, the user can also use the search bar to search through the data by an parameter e.g. time, date , room_id, room name and even reading



Use the search bar to navigate data, search by room name, date, time and room ID.

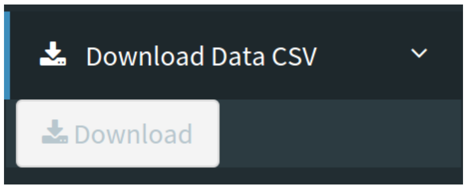
Click the column header to sort a column.

Show entries

Search: kitchen

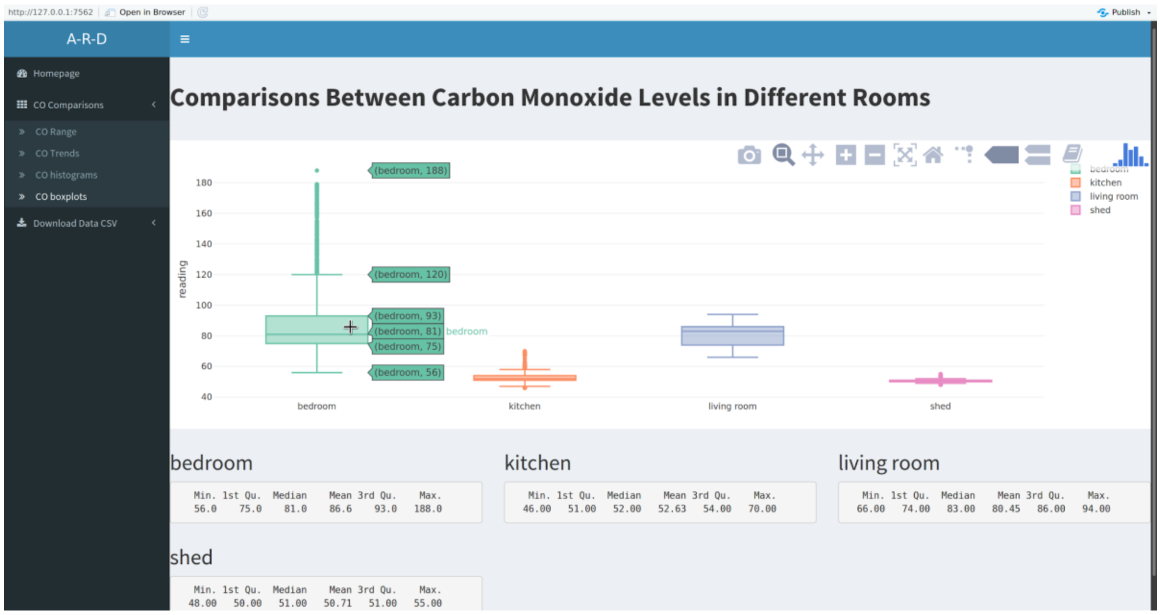
	id	time	reading	room	room_id
8051	8051	2018-05-12 14:25:45	55	kitchen	4
8052	8052	2018-05-12 14:26:10	55	kitchen	4
8053	8053	2018-05-12 14:26:40	55	kitchen	4
8054	8054	2018-05-12 14:27:10	54	kitchen	4
8055	8055	2018-05-12 14:27:40	54	kitchen	4
8056	8056	2018-05-12 14:28:10	54	kitchen	4
8057	8057	2018-05-12 14:28:40	54	kitchen	4
8058	8058	2018-05-12 14:29:10	55	kitchen	4
8059	8059	2018-05-12 14:29:40	55	kitchen	4
8060	8060	2018-05-12 14:30:10	55	kitchen	4
8061	8061	2018-05-12 14:30:40	55	kitchen	4
8062	8062	2018-05-12 14:31:10	55	kitchen	4
8063	8063	2018-05-12 14:31:40	56	kitchen	4
8064	8064	2018-05-12 14:32:10	55	kitchen	4
8065	8065	2018-05-12 14:32:40	55	kitchen	4
8066	8066	2018-05-12 14:33:10	55	kitchen	4
8067	8067	2018-05-12 14:33:40	57	kitchen	4

the user here is searching through the table using room name. The table is reactive and will search upon each character entry into the search bar. The user can also download the entire readings table in CSV format from the dashboard menu.



```
    filename = function() { paste("readings", '.csv', sep = '') },
    content = function(file) {
      write.csv(arduino, file, row.names = FALSE)
    }
  )
  outputOptions(output, 'downloadData', suspendWhenHidden=FALSE)
```

by selecting the download button in the dashboards sidebar, the user will be prompted into downloading a csv of the entire readings table.



Comparisons Between Carbon Monoxide Levels in Different Rooms

bedroom

```
 Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 56.0    75.0    81.0    86.6    93.0   188.0
```

kitchen

```
 Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
46.00   51.00   52.00   52.63   54.00   70.00
```

living room

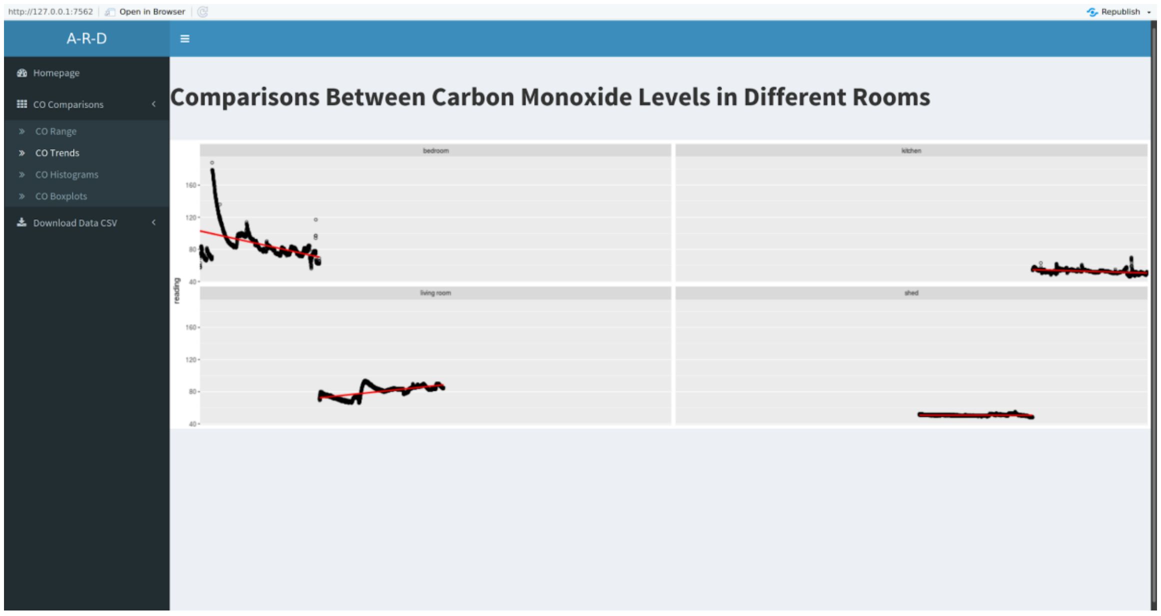
```
 Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
66.00   74.00   83.00   80.45   86.00   94.00
```

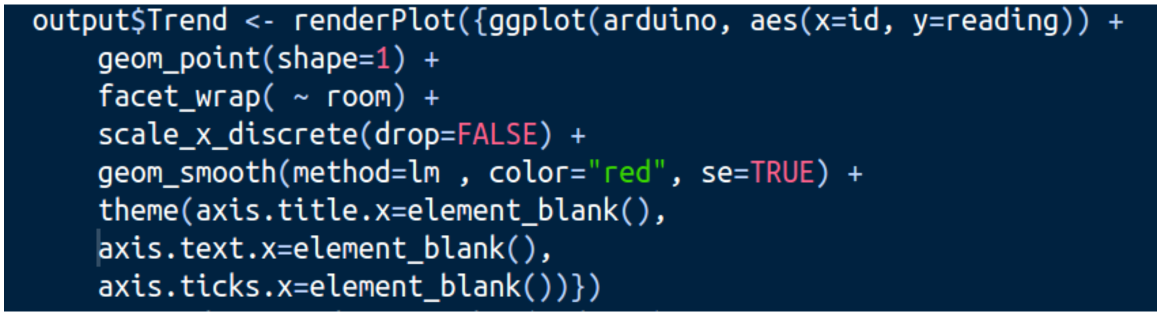
shed

```
 Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
48.00   50.00   51.00   50.71   51.00   55.00
```

```
output$box <- renderPlotly({plot_ly(arduino, y = ~reading, color = ~room, type = "box")})
```

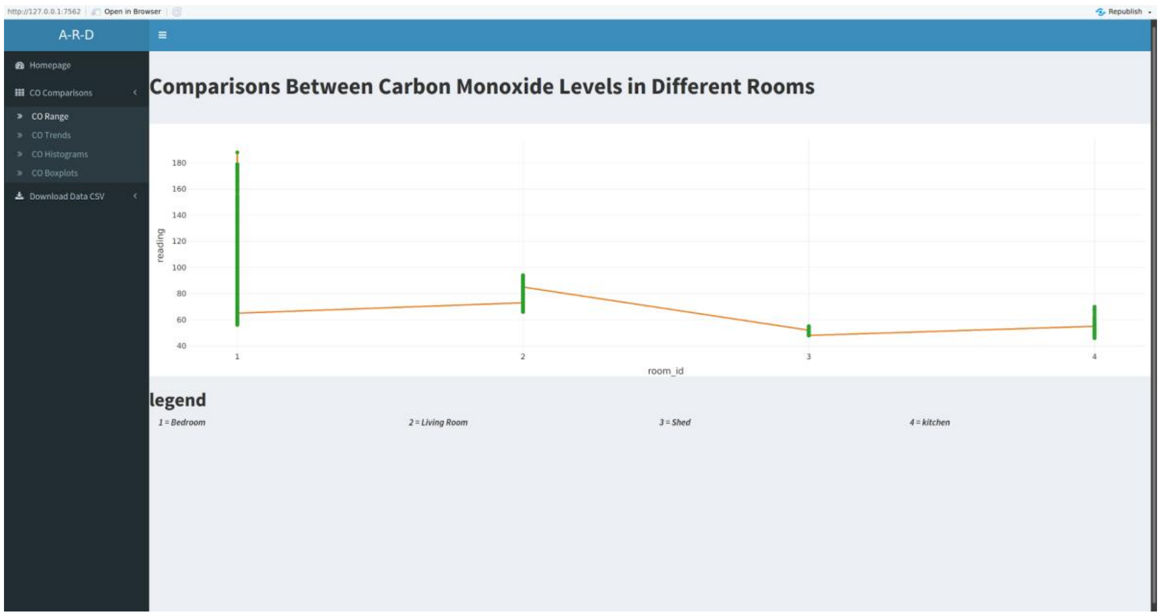
The use of R and plotly can provide strong statistical data in the form of interactive plots and summary statistics generated for the data, which can lead to end users gaining knowledge from the data gathered from their environment.

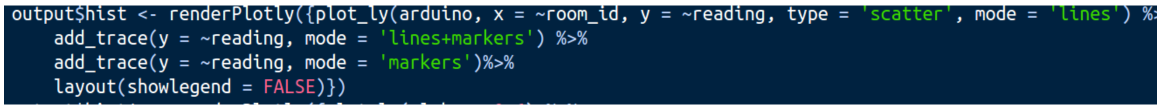




```
output$Trend <- renderPlot({ggplot(arduino, aes(x=id, y=reading)) +
     geom_point(shape=1) +
     facet_wrap( ~ room) +
     scale_x_discrete(drop=FALSE) +
     geom_smooth(method=lm , color="red", se=TRUE) +
     theme(axis.title.x=element_blank(),
     axis.text.x=element_blank(),
     axis.ticks.x=element_blank())})
```

Here the readings of carbon monoxide are placed in order and trends are drawn from them using the GGplot package allowing for inferences to be made about the future data that could be gathered in each particular environment.





```
output$hist <- renderPlotly({plot_ly(arduino, x = ~room_id, y = ~reading, type = 'scatter', mode = 'lines') %
    add_trace(y = ~reading, mode = 'lines+markers') %>%
    add_trace(y = ~reading, mode = 'markers')%>%
    layout(showlegend = FALSE)})
```

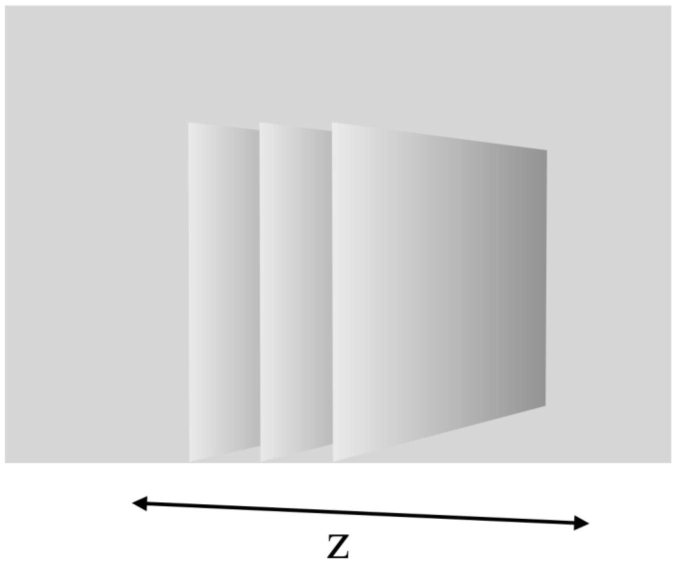
Here the Plotly package is used again to represent the range of values gathered in each environment showing that bedroom by far has the largest range of values

Headless Raspbian

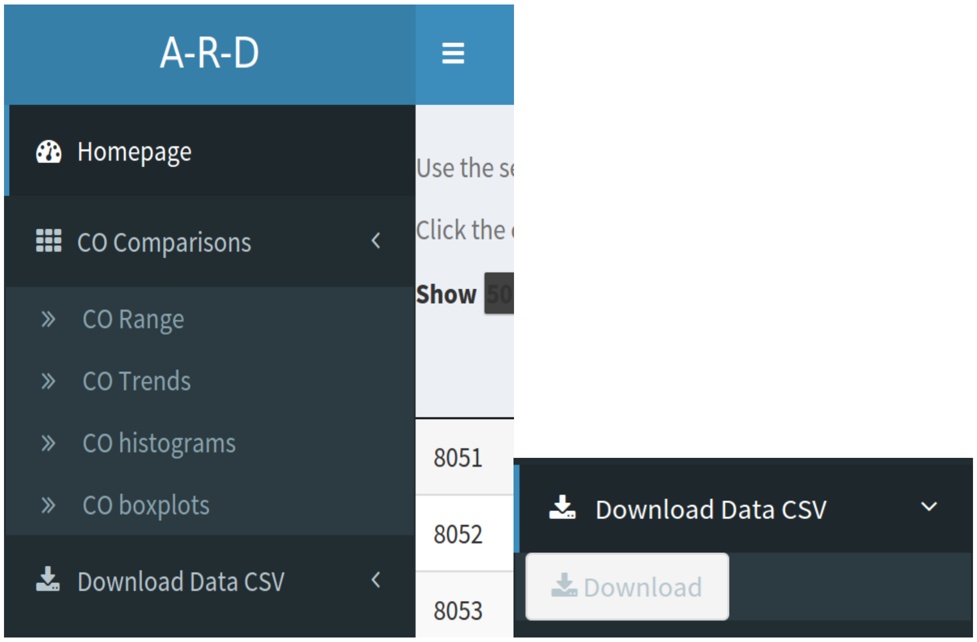
due to the nature of this project and the fact that the Raspberry pi needed to be mobile the majority of interactions with the pi were through SSH and command line. Apart from the initial setup all interactions with the raspberry pi took place over SSH and in due to that reason the sessions would end and the Python script taking readings and entering them to the database would stop running in order to solve this problem and to ensure session continuity the screen command was used this command can allow sessions to persist even after disconnection. On most Linux systems screen comes installed by default but due to the lightweight nature of the OS screen had to be installed with $ sudo apt-get install screen. when connected to the Raspberry pi over SSH initializing a screen session with the command $ screen. Screen could hold many terminal instances a new would be created by ^A then c and ^n could let you switch to the next terminal session. Sessions could be persisted with ^a then ^d this session would continue even after disconnection and could be recalled with the command $ screen -r

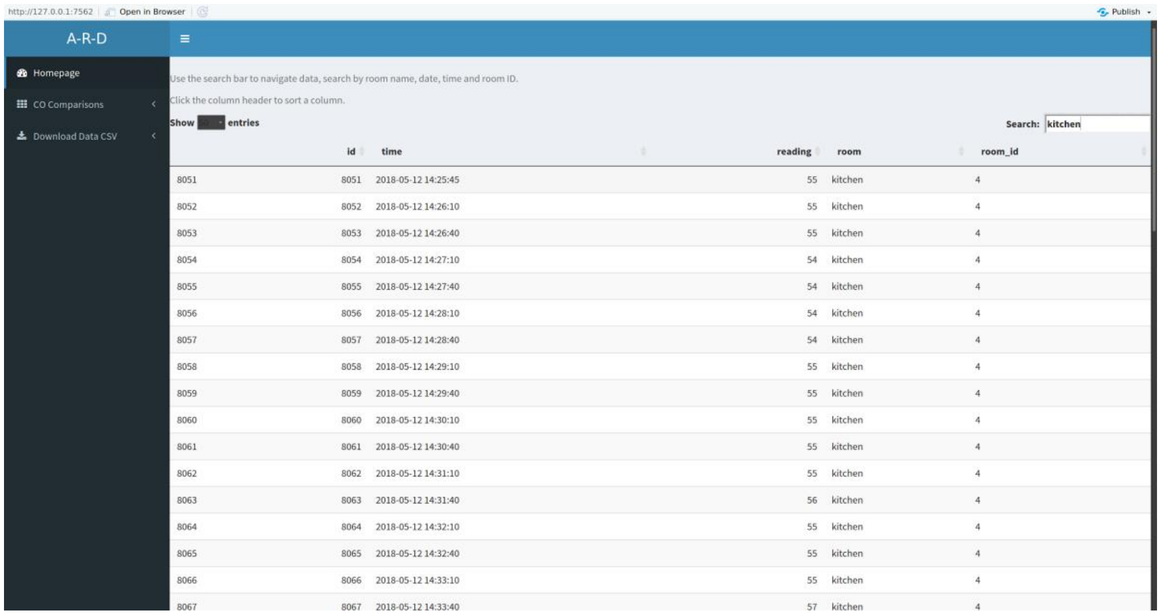
2.5 Graphical User Interface (GUI) Layout

The Graphical user interface along with the graphs of data is generated using the Shiny and Shiny dashboard package in R. All pages are accessible from the collapsible sidebar the pages are all generated together and the links in the sidebar inform the system of which page to show the user. The pages are generated layered on top of one another in a manner such as this.



The first page the user opens on is the data table page and from their all navigation is made through the sidebar which itself has collapsible menu items





Use the search bar to navigate data, search by room name, date, time and room ID.

Click the column header to sort a column.

Show entries

Search: kitchen

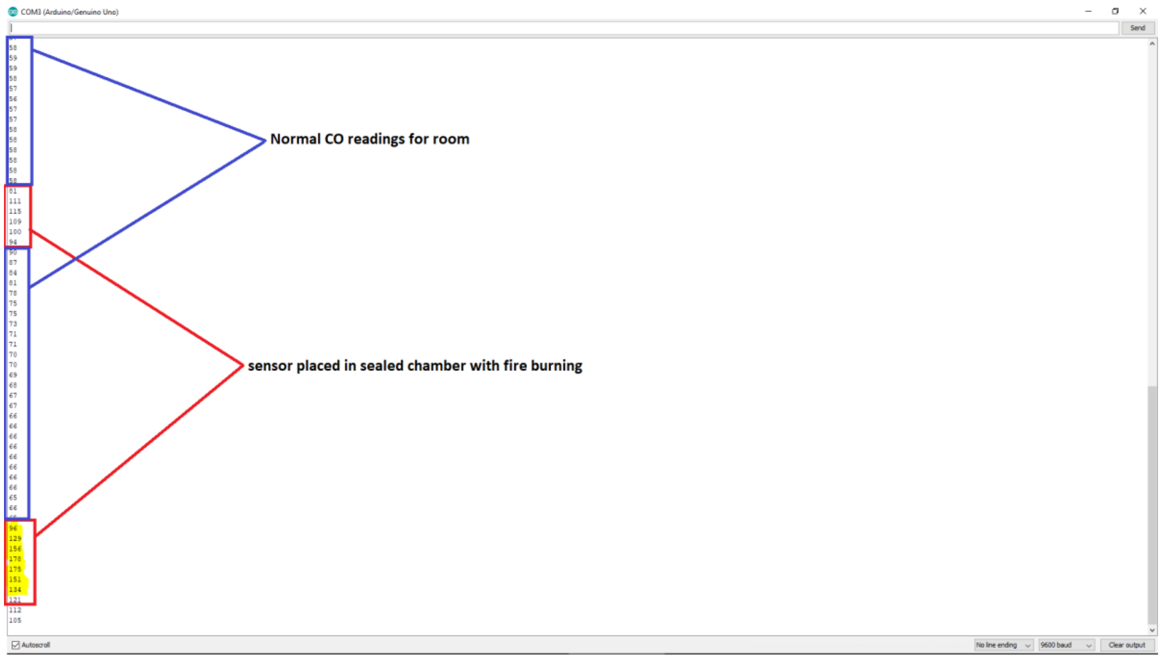
	id	time	reading	room	room_id
8051	8051	2018-05-12 14:25:45	55	kitchen	4
8052	8052	2018-05-12 14:26:10	55	kitchen	4
8053	8053	2018-05-12 14:26:40	55	kitchen	4
8054	8054	2018-05-12 14:27:10	54	kitchen	4
8055	8055	2018-05-12 14:27:40	54	kitchen	4
8056	8056	2018-05-12 14:28:10	54	kitchen	4
8057	8057	2018-05-12 14:28:40	54	kitchen	4
8058	8058	2018-05-12 14:29:10	55	kitchen	4
8059	8059	2018-05-12 14:29:40	55	kitchen	4
8060	8060	2018-05-12 14:30:10	55	kitchen	4
8061	8061	2018-05-12 14:30:40	55	kitchen	4
8062	8062	2018-05-12 14:31:10	55	kitchen	4
8063	8063	2018-05-12 14:31:40	56	kitchen	4
8064	8064	2018-05-12 14:32:10	55	kitchen	4
8065	8065	2018-05-12 14:32:40	55	kitchen	4
8066	8066	2018-05-12 14:33:10	55	kitchen	4
8067	8067	2018-05-12 14:33:40	57	kitchen	4

navigation through the dashboard for the user is intuitive all pages are accessible through the same place and can be collapsed and hidden if hindering the viewing and use of the page. all the available pages are all the same in basic layout with the navbar at the side and the main body of the page to the right of it making it seem more like an application than a web page.

The majority of the graphs in the application are produced using the plotly package creating interactive histograms box plots and scatter graphs

2.6 Testing

In an enclosed space a candle is lit, and the Arduino sensor is running while in that enclosed space for testing purposes the sensor read time is reduced from 30 seconds to 1 second and the data is not passed to the database it is merely monitored using the Arduino IDE and its monitoring tools. As the readings show CO levels detected by the sensor spiked and slowly dropped afterward this is indicative of this sensor, when it detects carbon monoxide the readings spike when the source of carbon monoxide is removed the electricity passing through the sensing plate burns off any remaining chemicals resulting in the slow return to normal carbon monoxide levels


COM3 (Arduino/Genuino Uno)
Send
Normal CO readings for room
sensor placed in sealed chamber with fire burning
Autoscroll
No line ending
9600 baud
Clear output

```
58
59
59
58
57
56
57
57
58
58
58
58
58
58
81
111
115
109
100
94
90
87
84
81
78
75
75
73
71
71
70
70
69
68
67
67
66
66
66
66
66
66
66
66
65
66
96
129
156
178
175
151
134
121
112
105
```

2.7 Customer testing

for the customer testing the page was displayed to people familiar with technology and software development and asked for their feedback and any changes that could be made to the app that would make it better and more intuitive and even any features that would be better to add to the web app to improve it. Some of the feedback I received which I was able to implement was simple things like not having text in the right position and having no capitalization where there should be. Other feedback that was given was the use of the time data which could not be changed due to the issues had with the database and the timestamp updating to the new time. Other than that, I received feedback about having the data be accessible for advanced users to perform their own tests on this in turn led to the adding of the Download button to download a .csv file of the current data.

2.8 Evaluation

In this section there will be a review on the project as a whole and the results it obtained e.g. the readings and whether they are valid data to be tested. This project focused heavily Data gathering side of Data analytics and the KDD method. Therefore, a review of the circumstances surrounding the data, the data from the bedroom. This bedroom an attic bedroom with little to no ventilation, the day of recording the data was one of the hottest days of the year so far, the downward trend in the data is evidence of the temperature's effect as the reading was started during the day and ran into the day and morning, the following day was much cooler and as such the levels of carbon monoxide dropped

the living room is the largest of all four rooms tested however it is not well ventilated leading to mid to high readings of carbon monoxide levels It averages a lower temperature than the bedroom by a high margin, thus explaining the lower readings.

The shed is the coolest and most well-ventilated environment in the study by far it is as close to getting outside as one could get when testing the environment. the readings in this environment were very low and steady, which is suspected to be a result of the lack of human traffic, low temperatures and extreme ventilation.

The kitchen should have a lower than average temp as it is closed off from most of the heat generated by the sun, there is also good ventilation in this room. However, the kitchen receives a lot of human traffic in the household and is subject to environmental fluctuations due to cooking leading to some spikes in the readings.

Overall from the evaluation of the non-recorded situational data, the readings seem accurate and seem to be affected by many factors such as human traffic, temperature, humidity and ventilation. The readings seem accurate and follow a predictable pattern based on these factors

3 Conclusions

The advantages of this project in particular are its wide skill range requirements. Leading to large amounts of research and acquiring new skills such as directly working with Arduino boards and the large amount of sysadmin work involved in making the project work and be usable, the opportunities for the project are near endless as the different types of data that can gathered with the hardware involved, however there is a large caveat and that is that the amount of success with a project such as this and the project limits are near directly correlated with the amount of money to be invested in hardware to expand the limits of the data gathering in this project. one of the main disadvantages to the project is the reliance on hardware. Some other data gathering projects would use software to gather the needed data leaving near limitless options for attaining its goal, this project is both freed to create its own data by the hardware required and limited in its methodology by its reliance on certain hardware.

4 Further development or research

Without a limit on resources this project could further develop into many different things. the types of data that could be gathered would be limited by the hardware resources, e.g. the evaluation that temperature and humidity play a large role in the amount of carbon dioxide in the environment could be thoroughly investigated where limits on the hardware not there. along with investigating causes of the current data gathered new environmental data could be gathered and its reason for change and fluctuation could further be investigated endlessly. Advancing further on the path of KDD beyond the mere gathering and processing of data many more tests could be carried out and new knowledge obtained from it if time weren't a limiting factor. The further scale of the data gathering could be limitless were further development and investment of time placed in the project. The data gathering could be branched out much further with the addition of more of the necessary hardware.

5 References.

MQ-7 Carbon Monoxide Sensor Circuit Built with an Arduino. 2018. *MQ-7 Carbon Monoxide Sensor Circuit Built with an Arduino*. [ONLINE] Available at: http://www.learningaboutelectronics.com/Articles/MQ-7-carbon-monoxide-sensor-circuit-with-arduino.php. [Accessed 13 May 2018].

Raspberry Pi: the Perfect Home Server | Linux Journal. 2018. *Raspberry Pi: the Perfect Home Server | Linux Journal*. [ONLINE] Available at: https://www.linuxjournal.com/content/raspberry-pi-perfect-home-server. [Accessed 13 May 2018].

R Graphing Library | Plotly. 2018. *R Graphing Library | Plotly*. [ONLINE] Available at: https://plot.ly/r/. [Accessed 13 May 2018].

The R Graph Gallery. 2018. *The R Graph Gallery – Inspiration and Help with R Graphics*. [ONLINE] Available at: https://www.r-graph-gallery.com/. [Accessed 13 May 2018].

Shiny. 2018. *Shiny*. [ONLINE] Available at: https://shiny.rstudio.com/. [Accessed 13 May 2018].

Shiny Dashboard. 2018. *Shiny Dashboard*. [ONLINE] Available at: https://rstudio.github.io/shinydashboard/. [Accessed 13 May 2018].

GitHub. 2018. *How to suppress typecasting warnings · Issue #37 · r-dbi/RMySQL · GitHub*. [ONLINE] Available at: https://github.com/r-dbi/RMySQL/issues/37. [Accessed 13 May 2018].

Stack Overflow. 2018. *How do I open up my MySQL on my Raspberry Pi for Outside / Remote Connections? - Stack Overflow*. [ONLINE] Available at: https://stackoverflow.com/questions/18733802/how-do-i-open-up-my-mysql-on-my-raspberry-pi-for-outside-remote-connections. [Accessed 13 May 2018].

6 Appendix

6.1 Project proposal

Project Proposal

Atmospheric analyser

Student Name: Darragh McKernan

Number: X14516637

email address: darraghmckernan@gmail.com

Degree Programme Name: BSc (Hons) in Computing

Specialisation: Data Analytics

Date: 5th Nov 2017

1. Objectives

Data Gathering

I want to be able to gather data actively with this device using the hardware attached. I also want to process that data and store it in an organized manner and have it easily readable.

Functioning hardware

In this project I will be connecting hardware manually via soldering, one of the objectives is that the connections between the multiple devices in the whole system will work together correctly.

Data Presentation and Analysis

The data that is gathered and stored should be able to be presented in an understandable manner the presentation of the data should be modifiable. An analysis should be able to be drawn from the presented data giving the viewer the ability to make assumption from it.

2. Background

The idea of this project came to me when I was in work and wondered what the most ideal atmospheric conditions are for working in which led to the idea of gathering the data, I then realized that I would need to store the data to be able to properly find out the ideal conditions. I also found that companies today strive for the most efficient approach to any task and if they could monitor their employees work rates and find correlation in ideal atmospheric conditions this information would be very valuable to them. I decided that the most important thing to the companies would be the information gathered not the product itself making this an interesting concept for my data analytics project.

3. Technical Approach

sensor

A sensor will be connected to an Arduino board there will be multiple sensors involved the main sensor will be for detecting carbon monoxide and passing the data gathered onto the Arduino board

Board

The board I will get that will be connected to the sensor will be an Arduino board it will receive the information from the sensor and will process that information and pass it on via usb cable to a raspberry pi where it will be stored.

Raspberry Pi

This will act as the server/ database for the project and will gather the data to be stored and perform further processing to it if required and store the data in an organized manner such as SQL or JSON to be viewed and presented using something like tableau.

4. Special resources required

Hardware

There are pieces of hardware required for this project to work and be produced. the majority of the hardware that I don't already own such as the raspberry pi will be purchased from

https://www.sparkfun.com/

This included the Arduino board and the sensors and any wires cables and solder supplies that will be needed.

5. Technical Details

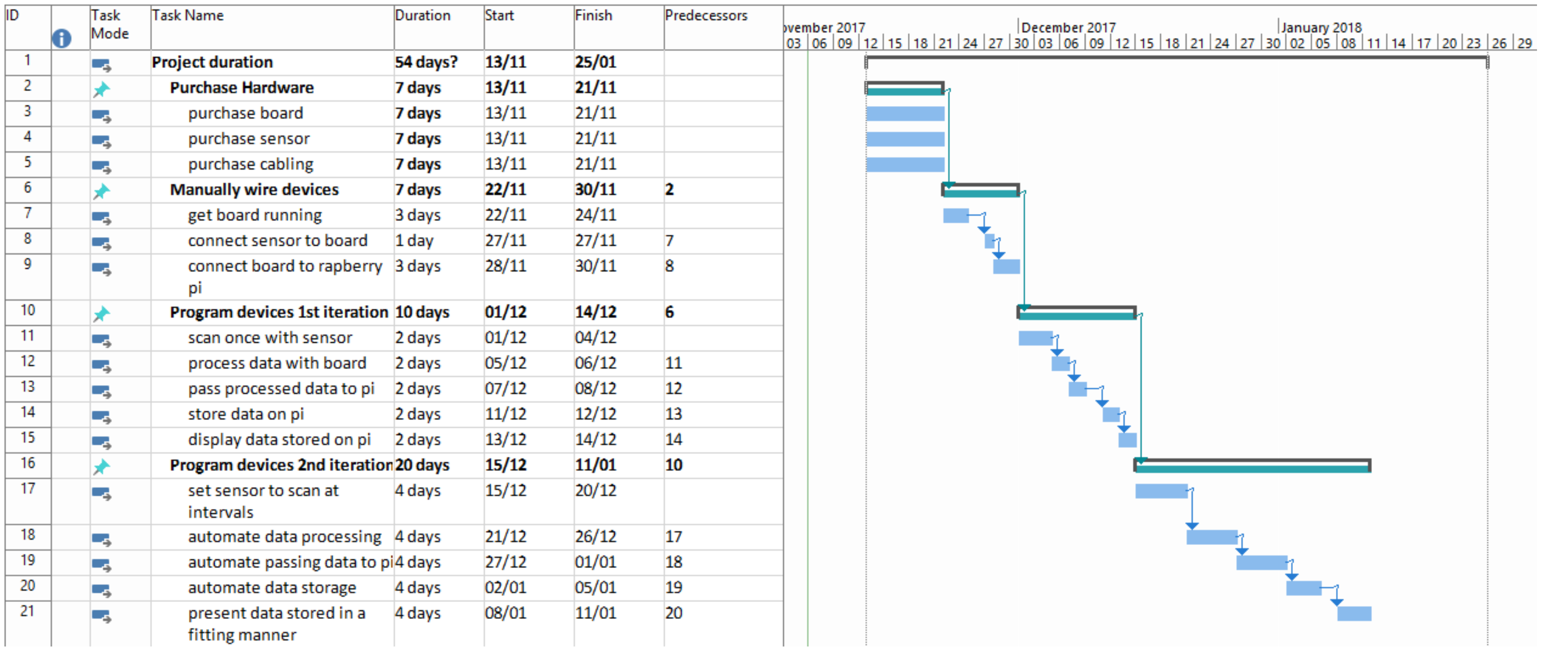
the main languages I will be using will be Python, C and possibly SQL, JSON and R. If stored using a SQL database, the data will be displayed using tableau.

C will be used on the Arduino board, python will be used on the raspberry pi and the storage and presentation types will be used on the raspberry pi.

6. Evaluation

The device will be placed in different rooms we know the schedule to around the college e.g. the Kelly theatre and other rooms around the college for a period of time and the data that is gathered will be monitored and checked to see if certain events bring about measurable change that we can observe through the device. I will try to analyse the data gathered to prove that hypotheses can be proved using the data gathered.

6.2 Project Plan



ID		Task Mode	Task Name	Duration	Start	Finish	Predecessors
1			**Project duration**	**54 days?**	**13/11**	**25/01**	
2			**Purchase Hardware**	**7 days**	**13/11**	**21/11**	
3			purchase board	**7 days**	13/11	21/11	
4			purchase sensor	**7 days**	13/11	21/11	
5			purchase cabling	**7 days**	13/11	21/11	
6			**Manually wire devices**	**7 days**	**22/11**	**30/11**	**2**
7			get board running	3 days	22/11	24/11	
8			connect sensor to board	1 day	27/11	27/11	7
9			connect board to rapberry pi	3 days	28/11	30/11	8
10			**Program devices 1st iteration**	**10 days**	**01/12**	**14/12**	**6**
11			scan once with sensor	2 days	01/12	04/12	
12			process data with board	2 days	05/12	06/12	11
13			pass processed data to pi	2 days	07/12	08/12	12
14			store data on pi	2 days	11/12	12/12	13
15			display data stored on pi	2 days	13/12	14/12	14
16			**Program devices 2nd iteration**	**20 days**	**15/12**	**11/01**	**10**
17			set sensor to scan at intervals	4 days	15/12	20/12	
18			automate data processing	4 days	21/12	26/12	17
19			automate passing data to pi	4 days	27/12	01/01	18
20			automate data storage	4 days	02/01	05/01	19
21			present data stored in a fitting manner	4 days	08/01	11/01	20

6.3 Monthly Journals

Reflective Journal

Student name: Darragh McKernan

Programme: BSc in Computing

Month: September

My Achievements

This month, I was able to research the hardware requirements of my project, finding the equipment that will be necessary for the actual implementation of my project idea. I now have a firmer grasp on what my final project at least should be, which is a device that takes readings on certain atmospheric conditions in a room e.g. CO2, CO, temperature and humidity.

My contributions to the projects included getting my actual project Idea approved in the project presentation.

My Reflection

I felt, it worked well to research the hardware requirements as I need to know these and have them before I can start the project. I have yet to devote enough attention to the project and what it requires as the other modules also have a very busy CA and upload schedule.

Intended Changes

Next month, I will try to finish the project proposal document and get started on the requirements specification document. If all goes well the requirements specification document will be at least half completed by the end of next month. I also want to decide on what readings and data that I will be gathering for the project whether it's all or just a select few as proof of concept as I do not want to overload myself with work.

I realised that I need to keep up to date on the documentation uploads and completions as I have been getting caught up in completing assignments and Cas for other modules.

Supervisor Meetings

I have yet to be assigned a project supervisor, I will be assigned a project supervisor next week, I hope to get someone with experience in manipulating and graphing the data I want to gather with this project.

Reflective Journal

Student name: Darragh McKernan

Programme: BSc in Computing

Month: October

My Achievements

This month I continued research the requirements of the project and what parts were needed.

My Reflection

I felt I worked well in the research phase of the project, but I was hindered again by the lack of parts necessary for the project work to commence.

Intended Changes

Next month, I will try to order the equipment and parts for work on the project to proceed, I realised that I need to finalize exactly what parts are needed prior to the ordering of any. also have to prepare for the midpoint presentation and the midpoint technical doc that are due next month

Reflective Journal

Student name: Darragh

Programme: BSc in Computing

Month: November

My Achievements

This month I completed the midpoint tech document that was due, this also allowed me to narrow my understanding and get a better view of what this project will turn out to be. I also completed my midpoint presentation.

My Reflection

I felt I worked well at chipping away at the workload of the tech document and not leaving it until last minute. I also ordered the parts of the project that are required to start the project itself, however I was unable to complete a working prototype due to not having the necessary parts.

Intended Changes

Next month I will try to work on assembling the hardware of the project as I should finally have them in hand

Supervisor Meetings

Items discussed: technical document and points on the presentation

Action Items: was given tips on the wording of the document and how the presentation should play out.

Reflective Journal

Student name: Darragh McKernan

Programme: BSc in Computing

Month: January

My Achievements

This month I was able assemble some of the hardware the Arduino board and the MQ-7 sensor as they arrived over Christmas. I was able to get serial readings from the Arduino.

My Reflection

I felt that I worked well in beginning the assembly of the sensor and Arduino board and getting the first readings out of them.

Intended Changes

I need to continue working on learning how to control the sensor and Arduino board so I can set it to pass the data onto the raspberry pi. I also need to do further research on how to setup the raspberry pi so that it can receive and store the readings from the sensor.

Reflective Journal

Student name: Darragh McKernan

Programme: BSc in Computing

Month: February

My Achievements

This Month i was able to make progress on the Raspberry pi, I installed the operating system on it, its running on Raspbian. This is a Linux based operating system similar to ubuntu, it is a Debian based system and with this the raspberry pi can work as both the server and the SQL database for the data gathered. I was also successful in the controlling of when the Arduino takes a reading.

My Reflection

I felt, it worked well to make progress with the raspberry pi and get to know more about how it's setup and works, it was also good to be able to control the rate at which the sensor takes a reading. However, I was not successful in passing the data from the Arduino to the raspberry pi.

Intended Changes

Next month I need to make sure I can pass the readings to the Raspberry pi and save them in a database.

Reflective Journal

Student name: Darragh McKernan

Programme: BSc in Computing

Month: March

My Achievements

This month I was successful in passing readings to the raspberry pi from the Arduino board using a serial import for python I haven’t been able to save it to a database yet though.

My Reflection

after further research I have chosen not to use the original idea of using tableau for the graphs, I have looked into it and because of issues with licensing it would be more trouble than it’s worth. I have decided to go with R Shiny which my supervisor had informed me of earlier on in the year

Intended Changes

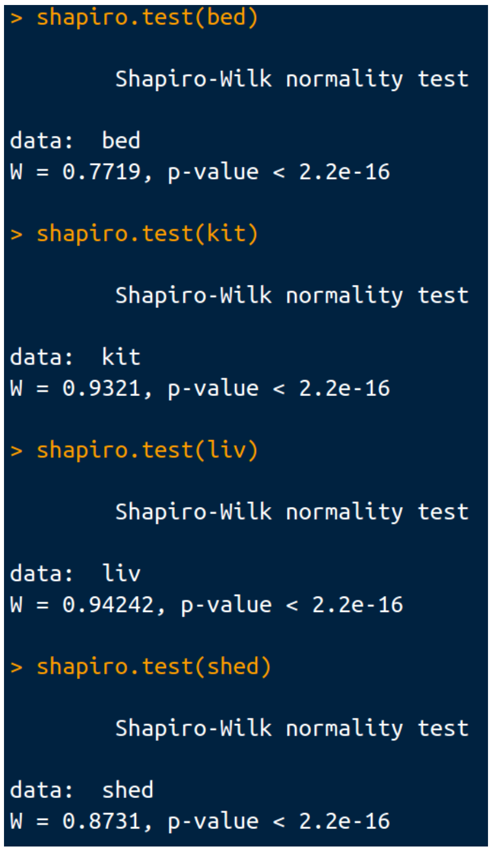
Next month I plan to set the database up and research further into R Shiny and see how I can make the best use of it for my project.

6.4 Other Material Used

Statistical Tests

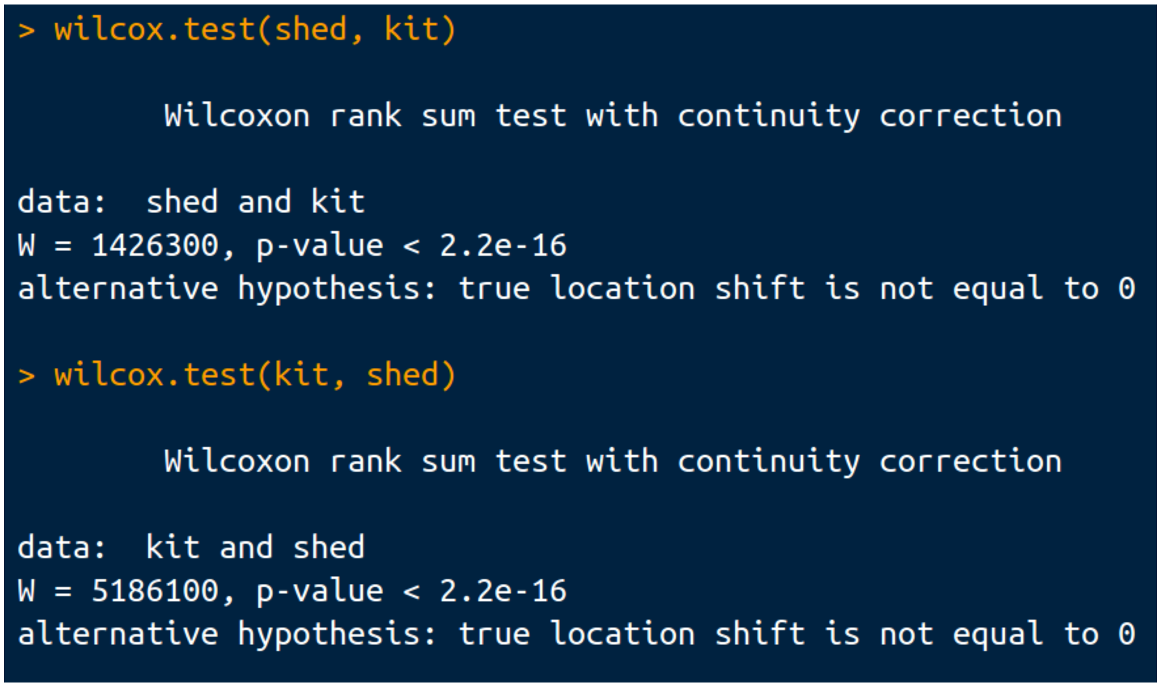
Mann-Whitney U Test

tests for normality



```
> shapiro.test(bed)

        Shapiro-Wilk normality test

data:  bed
W = 0.7719, p-value < 2.2e-16

> shapiro.test(kit)

        Shapiro-Wilk normality test

data:  kit
W = 0.9321, p-value < 2.2e-16

> shapiro.test(liv)

        Shapiro-Wilk normality test

data:  liv
W = 0.94242, p-value < 2.2e-16

> shapiro.test(shed)

        Shapiro-Wilk normality test

data:  shed
W = 0.8731, p-value < 2.2e-16
```

From these tests we can tell that all the datasets are far from normal by the fact the p-values for tests for normality we can reject the null hypotheses in favour of the alternatives that the datasets are not of normal distribution, therefore all apply to being used in non-parametric tests for significant differences in the Median. let’s try two data sets that were the closest in their similarity and readings, shed and kitchen



```
> wilcox.test(shed, kit)

        Wilcoxon rank sum test with continuity correction

data:  shed and kit
W = 1426300, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0

> wilcox.test(kit, shed)

        Wilcoxon rank sum test with continuity correction

data:  kit and shed
W = 5186100, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0
```

Again, the P values are incredibly low in both orders therefore we the null in favour of the alternative hypothesis that there is a difference in the medians of both data sets