Humber Campus Navigator

**Anoopjot Kaur Dhallu**

**Bettin Jacob**

**Ishan Khuttan**

**Humber College, Toronto**

**Winter-2019**

# Authorship Statement

We, Anoopjot Kaur Dhallu, Bettin Jacob & Ishan Khuttan, confirm that this work submitted for assessment is our own and is expressed in our own words. Any uses made within it of the works of any other author, other than us, in any form (ideas, equations, figures, texts, tables, programs), are properly acknowledged at the point of use. A list of the references used is included.

I, Anoopjot Kaur Dhallu was responsible for maintaining and programming Raspberry Pi and the programs inside it.

I, Bettin Jacob was responsible for developing the Android mobile application and the firebase database design and maintenance, and the GitHub account associated with the project.

I, Ishan Khuttan was responsible for designing the PCB and the enclosure of the hardware part of this project. I also helped Anoopjot Kaur Dhallu in the programming of Raspberry Pi.

Signature: Anoopjot Kaur Dhallu

: Bettin Jacob

: Ishan Khuttan

# Abstract

The technical report is about developing an Android smartphone application along with its required hardware components. This project is a new idea to solve the problem of navigation inside a group of buildings or campus. To be specific, this is an attempt to resolve the in-campus navigation of Humber College North Campus. As Humber College campus is vast and it has a lot of buildings, it's hard to navigate within the school. This technical report will elaborate on the Software and Hardware requirements of the project and gives a detailed description of how the application works. We make use of QR codes and Dijkstra's algorithm inside the app, Firebase database, Raspberry Pi and three sensors namely LSM9DS0, TMP-36, and ST7735 to accomplish the aim of the project.

Table of Contents

[Authorship Statement 2](#_Toc2940195)

[Abstract 3](#_Toc2940196)

[List of Illustrations 7](#_Toc2940197)

[Introduction 9](#_Toc2940198)

[Additional Comments 10](#_Toc2940199)

[Methodology 11](#_Toc2940200)

[Build Instructions & Troubleshooting for Individual Sensors 11](#_Toc2940201)

[LSM9DS0 11](#_Toc2940202)

[Project Group Work (Integration) 23](#_Toc2940203)

[Connecting LSM9DS0 Sensor with Raspberry-Pi 23](#_Toc2940204)

[Project Description 27](#_Toc2940205)

[Requirements Specification 29](#_Toc2940206)

[Software 29](#_Toc2940207)

[Database 31](#_Toc2940208)

[Firmware 32](#_Toc2940209)

[Hardware 33](#_Toc2940210)

[Interface boards and sensors 34](#_Toc2940211)

[Other Accessories and Enclosure 35](#_Toc2940212)

[Conclusion 37](#_Toc2940213)

[Recommendations 39](#_Toc2940214)

[Bibliography 41](#_Toc2940215)

[Appendix 43](#_Toc2940216)

# List of Illustrations

# Introduction

Humber Campus Navigator is a project made specifically for navigation within a group of buildings or a campus because for a group of building's like a college campus, and it is tough to navigate through it and find the room numbers, washrooms or any point of interest. We are trying to solve this issue in Humber North Campus with an Android application that works on an Android smartphone. For instance, when we were new to this college, we used to roam inside the campus to find our destination point within the school. It was tiresome and frustrating. There was no much help as we don't know who is new and who to seek help.Moreover, everyone was busy. Even this semester it was difficult for us in finding the location of NB part of the college. To solve that, we are making an application which will work within the campus. As a sample project, we are first trying to take five points and trying to navigate within them. After taking Professor's suggestions, time available and our experience into consideration, we decided to solve this real-world problem. Once we are successful in completing this project, we have a plan to expand it to the entire campus oh Humber College, which will be very helpful for Humber students and its visitors, and it will be much easier for anyone to find a point or destination on campus.

## Additional Comments

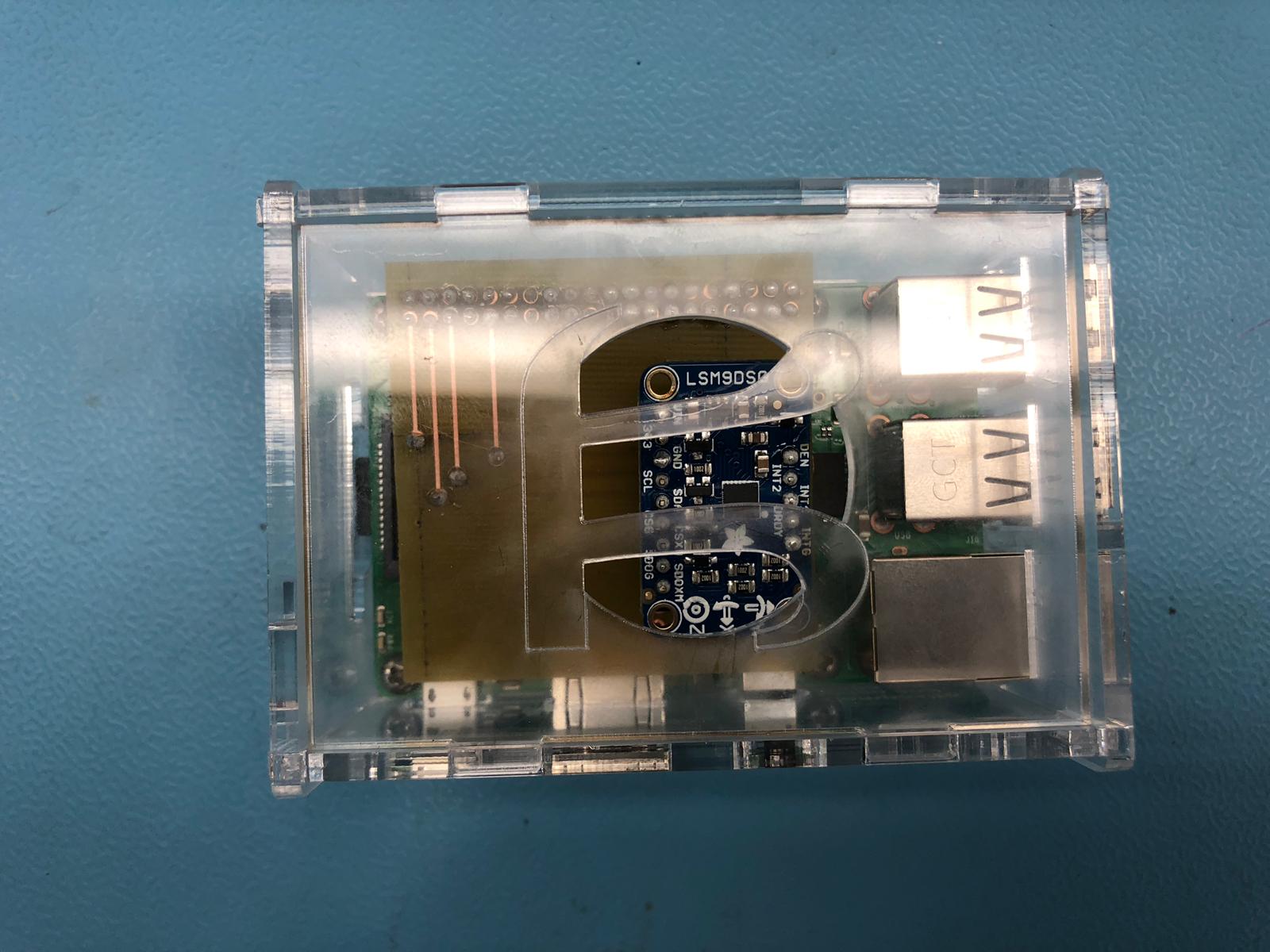
The first thing that comes to someone’s mind when talking about navigation is GPS which helps to find a location and to get a direction between any two points. Instead of using GPS signals for location finding and getting the direction, we are developing an application which can work even offline with the help of QR codes (to locate the user) and Dijkstra’s algorithm (to get a direction between two points).

# Methodology

## Build Instructions & Troubleshooting for Individual Sensors

### LSM9DS0

#### Build Instructions for using 'LSM9DS0 9-axis Accelerometer, Magnetometer, Gyroscope & Temperature Sensor' on a Raspberry-Pi

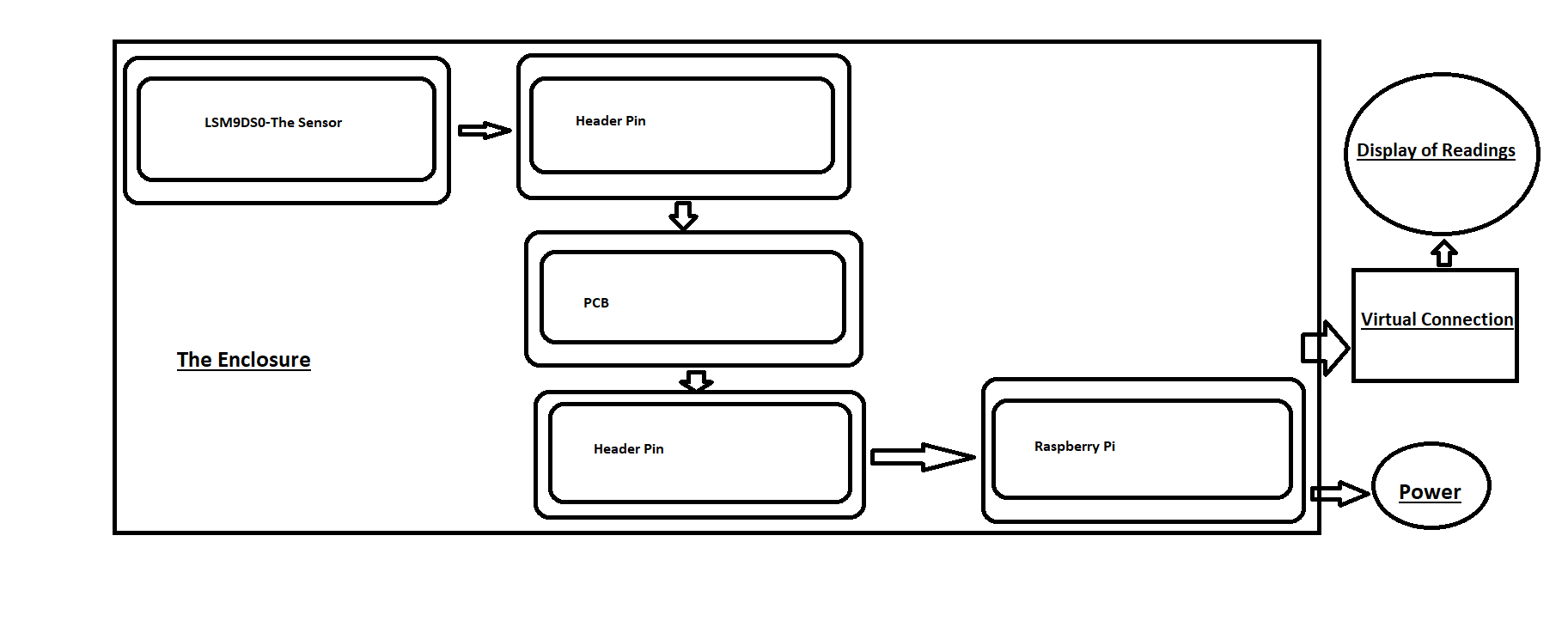


In this Project, the aim is to give step by step instructions to build a hardware unit that measures the values coming from the LSM9DS0 sensor. I will be discussing the components needed to build the unit, how much it might cost, various tools that we might need while building up a unit, how to use that tools, and the key factors that should be taken care of .

Following these six instructions for a student, it might take a couple of days for completion of a hardware unit. But the shipping time of hardware parts will increase the time needed for completion for a student. If it is for mass production, one can use the design and codes readily available with the instructions, and build around 15 units in a single day, provided all parts are readily available.

We will start with the system diagram,

#### System Diagram of the Hardware

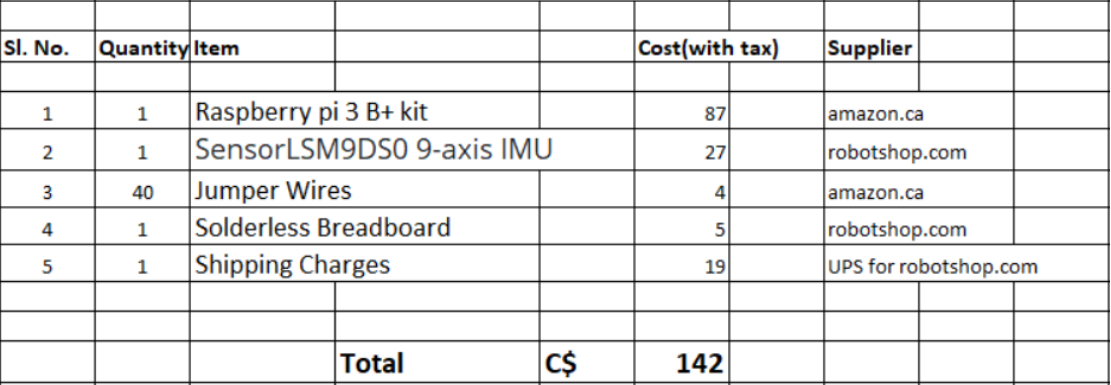


The above diagram explains how LSM9DS0 sensor can be used to build an hardware unit which can be used to record the readings coming from the sensor using a Raspberry Pi. The sensor should be soldered to an header pin and the header pin's other end should be soldered to a PCB. The PCB in turn is soldered to another header pin on the opposite side where the sensor is attached. The second header pin is then attached to the I/O pins of the Raspberry Pi. This whole unit is enclosed in an acrylic case for safety. The Raspberry Pi needs power and should be connected to a Virtual connection (wired/wireless) for operating Raspberry Pi remotely by which we can read the values coming from the sensor.

#### The Components needed to make the Unit and its Cost

If you are a student of a college or University, you most probably will have a Parts Kit which can be used here. You don't need to buy all those things, that you already have, which reduces the budget significantly. I am including all the components needed for the project, excluding items that are expected to be with students.

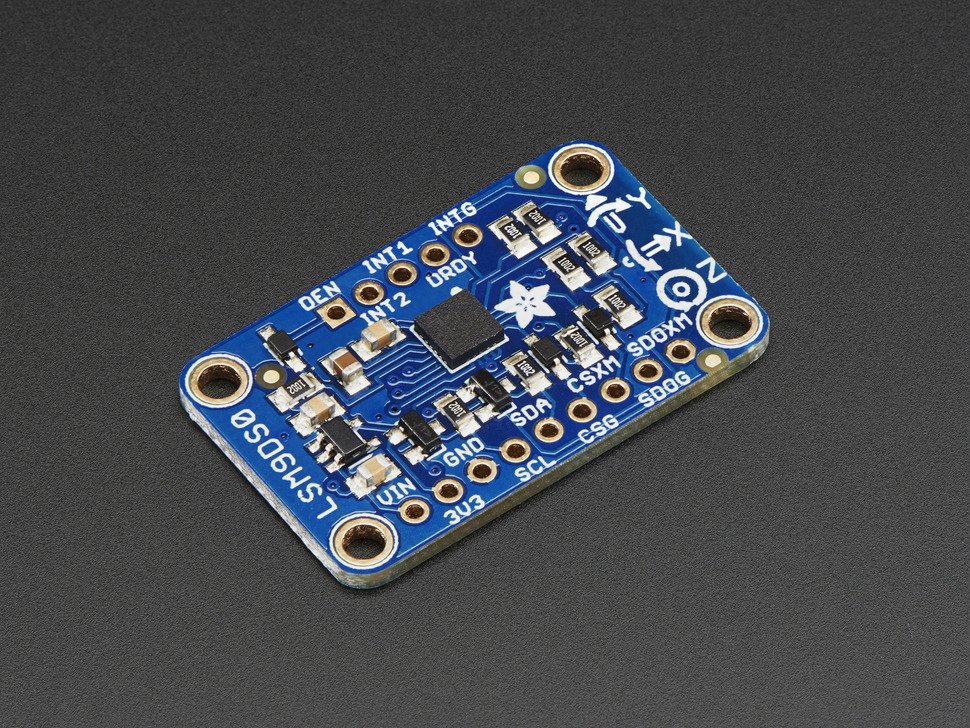
All prices are in Canadian Dollars and include Taxes where ever applicable.



You will also need a Laptop or a Desktop to set up the Virtual Connection and operate the Raspberry Pi remotely. Also for printing a 'PCB' and an enclosure (Hard case that holds the entire unit) you might need the help of a Prototype Lab, the cost of which I haven’t included here. You can get the Raspberry Pi, breadboard & Jumper Wires from Amazon.ca and the LSM9DS0 sensor from robotshop.com. There are several online portals that sell these items, you can shop around and find the cheapest one. This data is just for an overall idea about the budget.

You can see the pictures of the major components below,

#### The Sensor





## 

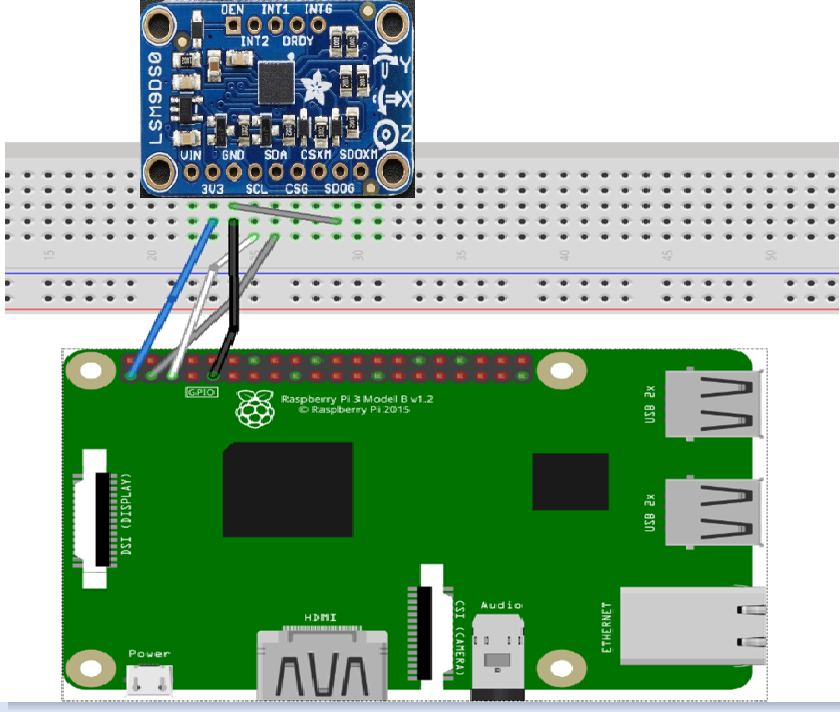
#### Step by Step Instructions for building the Unit.

##### Step-1

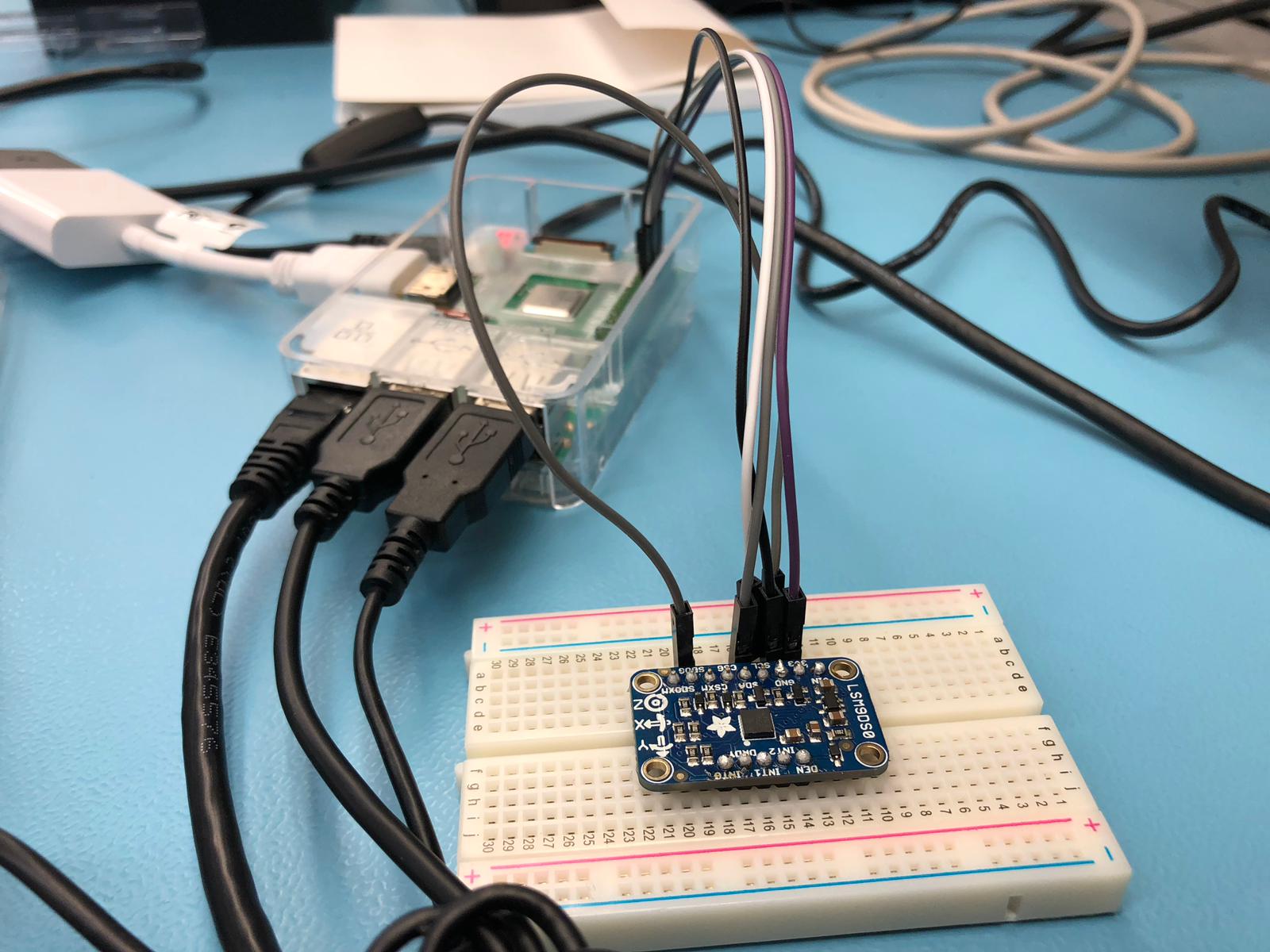
The first step is to set up the Raspberry Pi. Raspbian is the Operating System for the Raspberry Pi. We will flash Raspbian OS on a 16 GB SD card and insert it into the Raspberry Pi. Raspbian OS is free and can be downloaded online. The flashing instructions can be found here and if the flashing was successful, when giving power to Raspberry Pi and connecting it to a display, will detect and boot the Raspbian OS automatically. For the first time, we need a mouse and a keyboard connected to Raspberry Pi to set up the Virtual Connection (using VNC Viewer) for Raspberry Pi, by giving input and Wi-Fi credentials. For setting up a virtual connection for remote operation of the unit we need to install VNC Viewer on Raspberry Pi and on the Desktop/Laptop which we use to remotely control Raspberry Pi. The details of how to install and setup VNC viewer can be found by clicking here.

##### Step-2

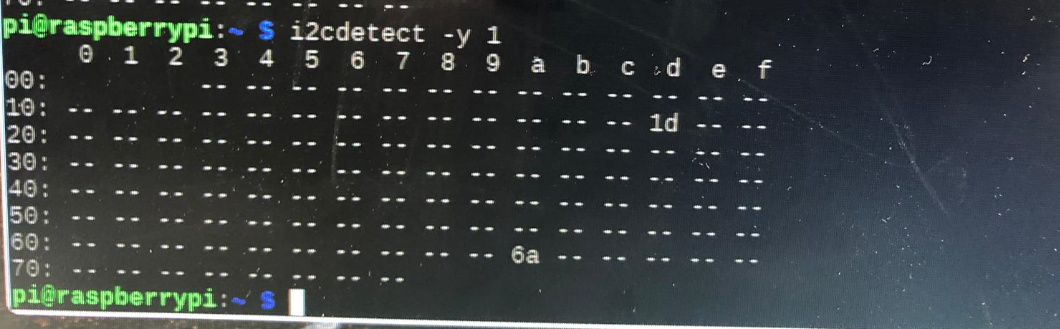
Second step is to check whether the sensor is in working condition. First we should solder a 9 pin header pin on to the sensor, so that the sensor can be inserted into a breadboard. Before start soldering please read this document for your own safety. We need to get the raw readings from the sensor on the specific I2C address of the Raspberry Pi. For that we need to find out the wiring diagram of the Sensor output to the specific pins on the Raspberry Pi. We are expected to get readings in two addresses after the wiring of sensor and Raspberry Pi as shown below. The addresses where we get readings are '6a' and '1d' of the Raspberry Pi. To get the readings after connection open a terminal on the Raspberry Pi and give the command 'i2cdetect -y 1'. Connections: - Pin 1 from Raspberry Pi to 3V3 of Sensor - Pin 2 from Raspberry Pi to SDA of Sensor - Pin 3 from Raspberry Pi to SCL of Sensor - Pin 5 from Raspberry Pi to GND of Sensor - SDOG from Sensor to GND of Sensor



###### Sensor wired with Raspberry pi

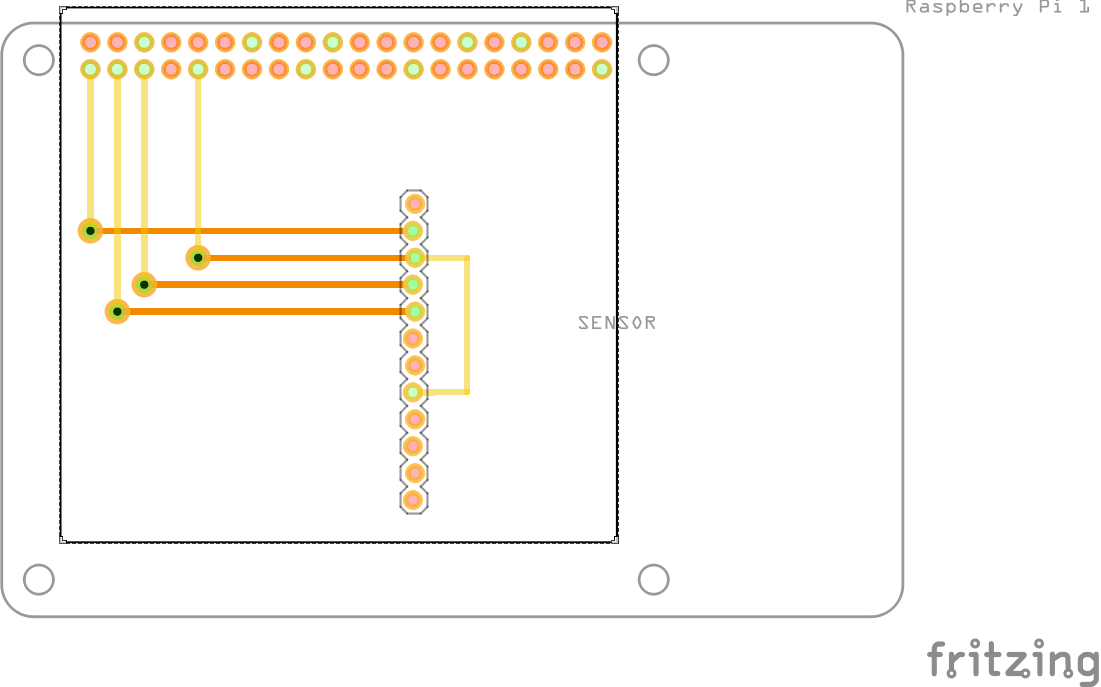


###### Readings as Expected

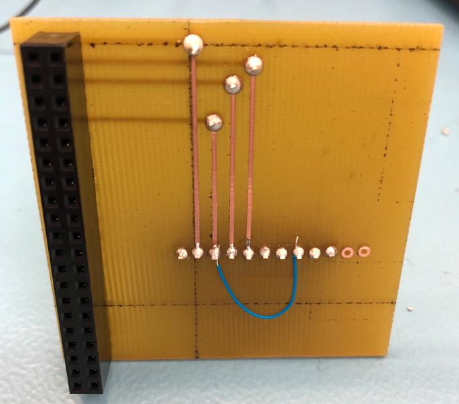


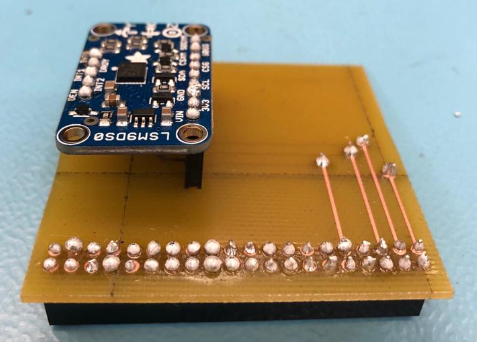
##### Step-3

Once we make sure that our Raspberry Pi and sensor is working, we can start the designing of PCB and soldering (Please read safety measures) of components. Designing a PCB can be done using fritzing software. Click here to get instructions on how to use fritzing. This is the image of the fritzing file generated specifically for this sensor and Raspberry Pi. You can download the below fritzing file from this link in zip format.



Once you are done with the PCB design, you should take this file to a lab where PCB's can be printed out and get it printed out. Once you get the PCB board printed out, you should solder two header pins onto the PCB. One stack header pin of <a href="https://canada.newark.com/adafruit/2223/40-pin-pi-gpio-stacking-header/dp/31AC4582?gclid=EAIaIQobChMIo4ejjLOW3wIVQbjACh2MeQRJEAYYBCABEgISn\_D\_BwE&CAGPSPN=pla&CAWELAID=120185770002227709&CAAGID=23354969332&CMP=KNC-GCA-GEN-SHOPPING&CATCI=pla-294680686006">40 pins (two 20 pins in parallel) to be soldered to PCB so that the PCB can be securely attached to the Raspberry Pi. Another single line 10 pin stack header pin to be soldered on the other side of the PCB where the sensor can be attached to the PCB securely. The following images will make the idea clear.

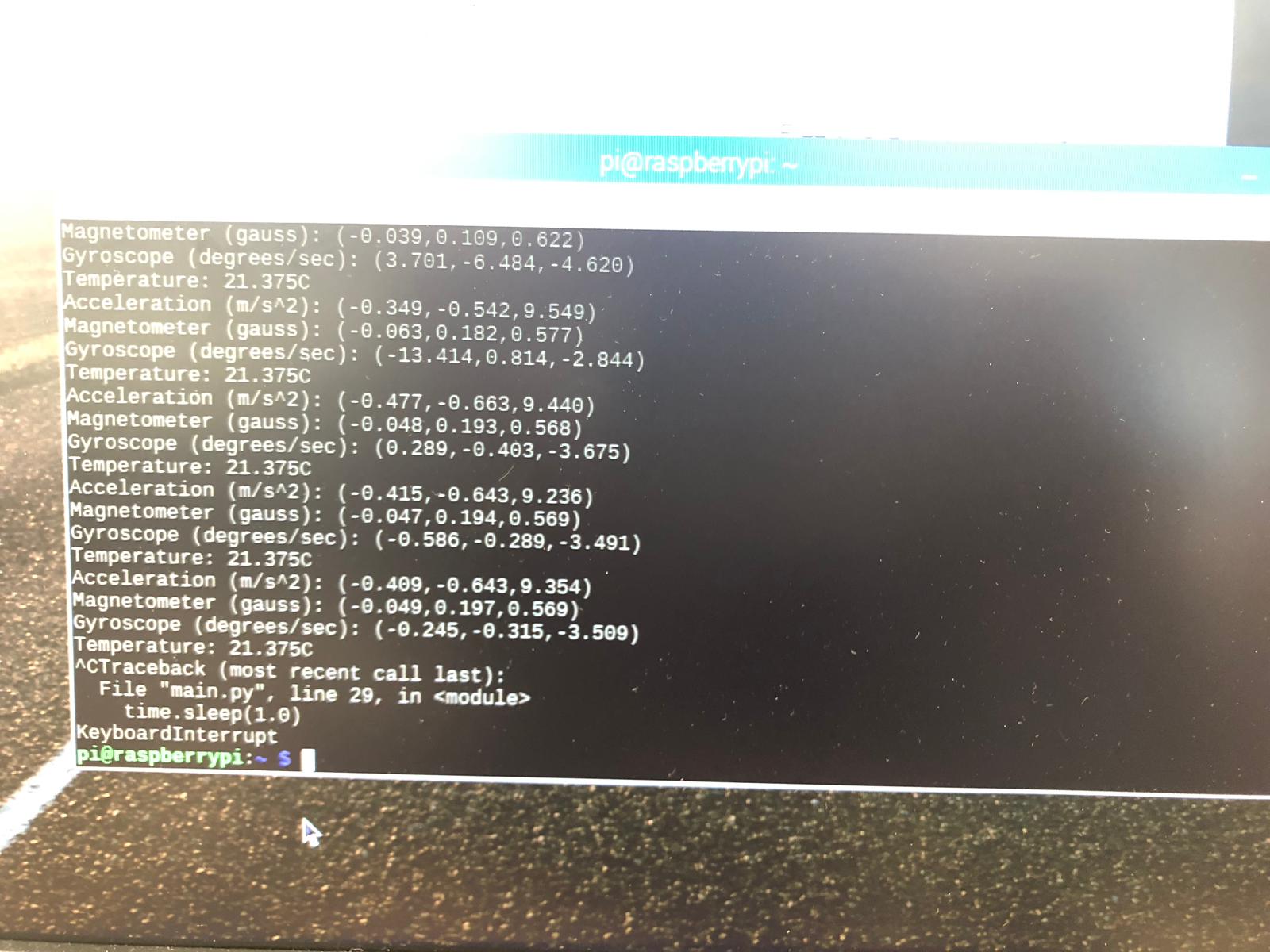






##### Step-4

After attaching the sensor on to the PCB and then the PCB on to the Raspberry Pi will take out all wirings and the sensor is now connected to the Raspberry Pi with the help of PCB and header pins which are all soldered each other appropriately to make a single unit. Now in this step we need to install a python software specific for our sensor in the Raspberry Pi so that the software will process the raw signals coming from the sensor into human readable values. The measured values of Accelerometer, Gyroscope, Magnetometer and Temperature are expected. The python software for this is readily available from this link. All the necessary instructions for installing the software is given in this link. This software expects an output in the address '6b'. Since we are expecting the reading in '6a' address, a minor change need to be done before running the software. I changed the address from '6b' to '6a' in the file 'adafruit\_lsm9ds0.py' and the software detected the readings and outputted the values as shown below. We need to do this because output of the sensor has been changed from '6b' to '6a' by making the SDO\_G terminal of the sensor low. If SDO\_G terminal of the sensor is not made low, we don't need to make this change to this software. The software file I changed is located at "usr/local/lib/python3.5/dist-packages/adafruit\_lsm9ds0.py" on my Raspberry Pi. The readings will come when running 'main.py' inside the software.



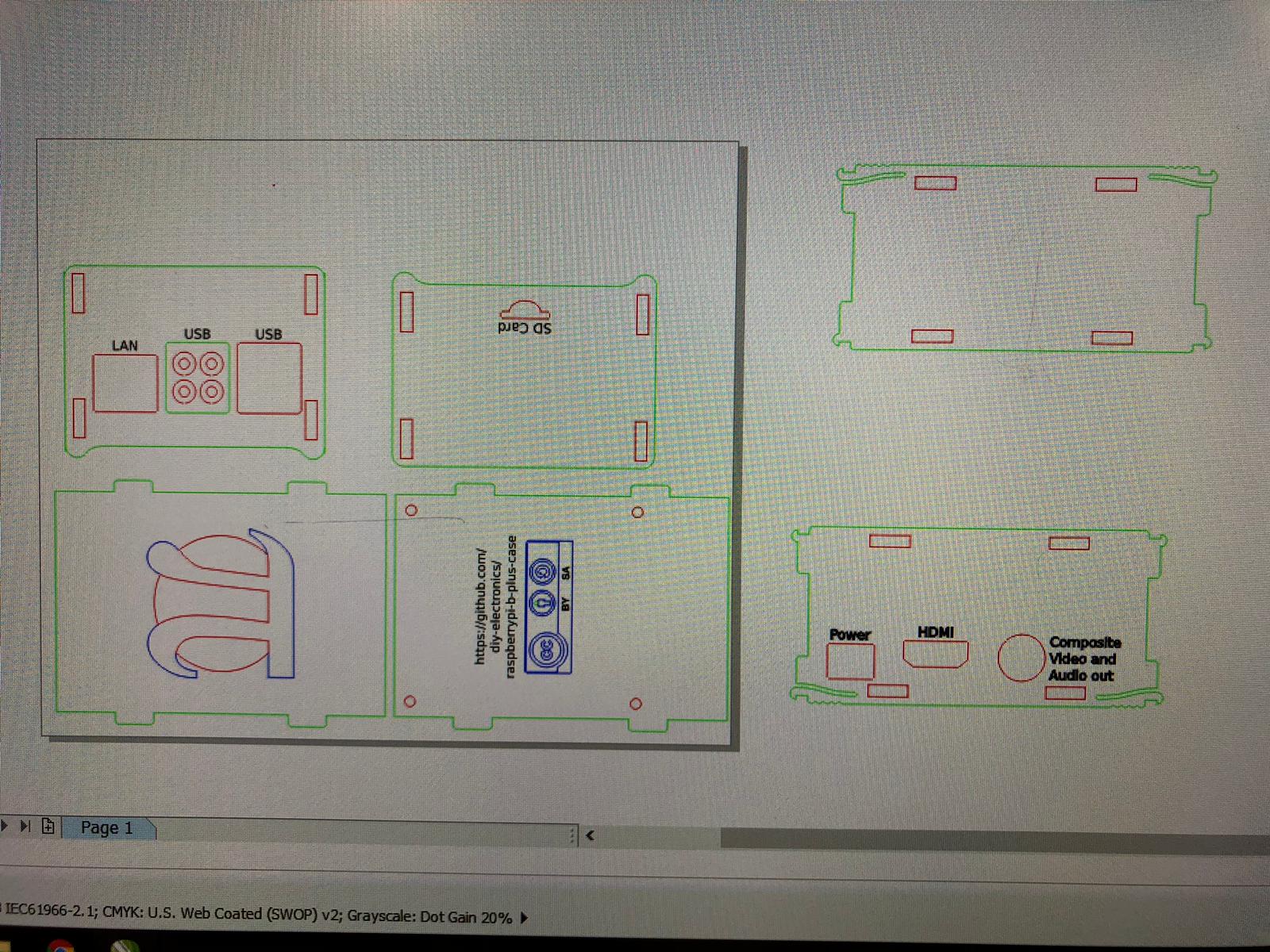
##### Step-5

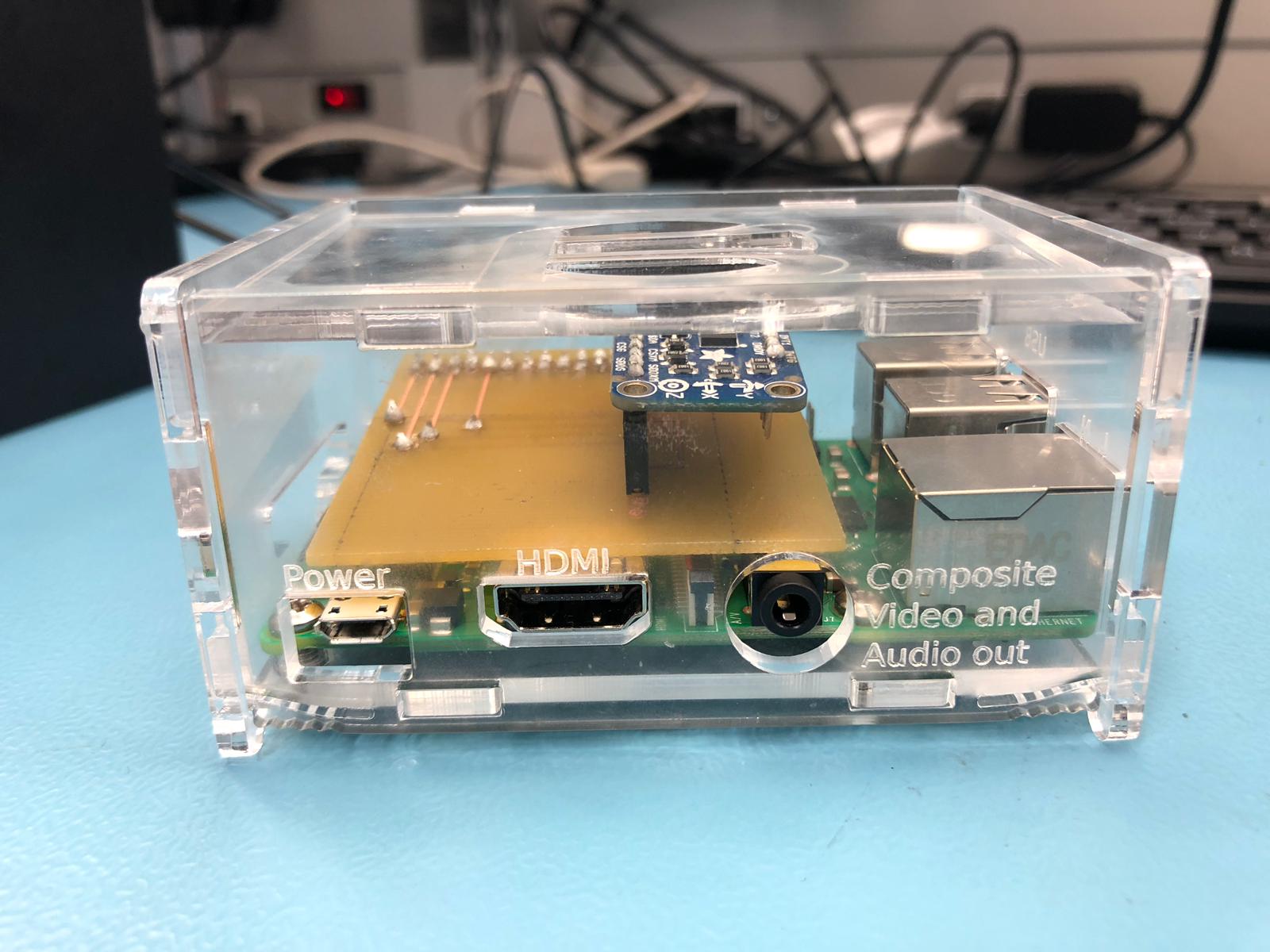
Now the next step is the enclosure. It simply means to protect the working piece of your hardware get enclosed in a secure case to protect the unit from external damages. So to do that, you need to design an acrylic case for the Raspberry Pi which encloses the sensor attached to the PCB. To design the case, you can use CorelDraw software.

Here for this unit there are 6 faces and each faces are designed separately with relative measurements taken for locks that holds the unit as an whole. Before assembling the faces, don't forget to screw the Raspberry Pi to the base piece so that the unit is firm with the whole case.

This is the link to download the CorelDraw codes for the below design. https://github.com/HumberCampusNavigator/HumberCampusNavigation/blob/HumberCampusNavigator-CoralDraw/Pi2CaseX6.cdr

The design of the case using CorelDraw software





##### Step-6

The last step is to test the unit after the enclosure so that we can make sure that our hardware unit works perfect. So the testing of the components at the beginning, then after soldering and then at the last are helpful to make sure that our unit building is progressing fine. This also helps to eliminate errors as we go.

## Project Group Work (Integration)

### Connecting LSM9DS0 Sensor with Raspberry-Pi

The next step of our project was to connect the primary sensor (LSM9DS0) with the firebase database. The best way to do that is to write some python codes in Raspberry-Pi which will help to write the temperature readings directly into firebase database. For that, firstly we need to import python-firebase library into Raspberry-Pi. This can be done by executing the following two commands in command terminal.

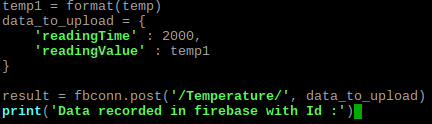


Retrieved from <https://ozgur.github.io/python-firebase/>

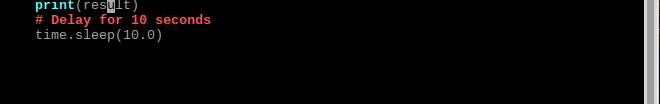
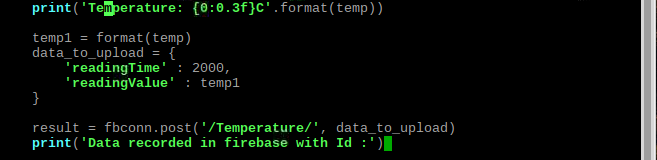
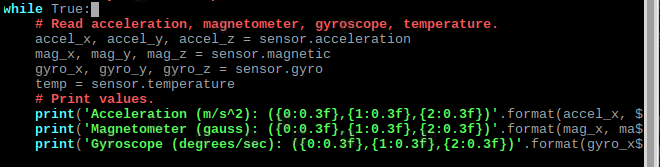
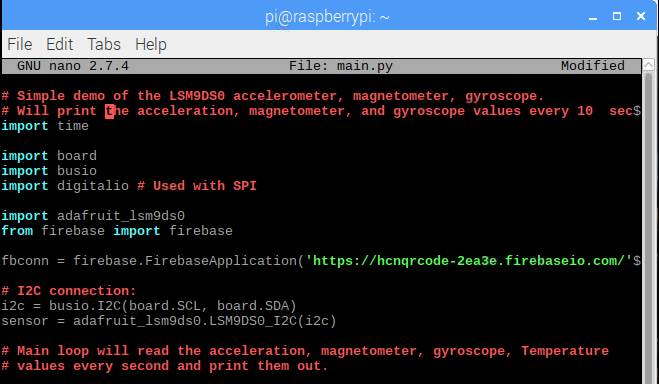
After this step we need to write the following codes into the python program(here main.py),

From firebase import firebase

Fbconn = firebase.Firebaseapplication(‘your unique firebase database url’,None)



With the above codes, into the main.py program, sensor readings will be recorded into the firebase database directly. A screenshot of the entire program is given below for reference. One thing I learned while doing this is, one should be very carefully in the indenting of the codes, otherwise python codes will not work in loops. Python program determines the codes inside a loop by indentation of codes. Further including the firebase library in the same folder where the program (main.py) is executed is necessary.



# Project Description

We are designing an app to make the navigation within the Humber campus, comfortable and convenient. For students, especially who are new to the school, it is hard to navigate within the college. This application will help students to explore and navigate within the campus from any point to any point in a short time and less distance. The app works with the help of the QR code and finding the shortest path using Dijkstra's Algorithm. Unique QR codes are pasted to each door of the classrooms or washrooms, or where ever we want within the campus. The user needs to scan the nearest QR code using the mobile application and type in or select the destination point from the map or a drop-down list, and the application will show the easiest route possible, and this will be accomplished with the help of Dijkstra's Algorithm. In-campus-navigation is the primary function of the app, other than that it tells the temperature of the current point where we are standing, and the normal average of temperature at that point. It also shows the current and historical average temperature of the destination point where the user wants to go. All values are taken with the help of our sensors and are be stored within our database.

The internet connected hardware will include a Raspberry Pi with a custom PCB with the following sensors and actuators

1. LSM9DS0 9-axis accelerometer, Gyroscope, Magnetometer, and Temperature sensor.
2. 1.8 TFT Color Display ST7735
3. Tmp-36 Temperature sensor

The database will store the temperature readings from LSM9DS0. We need to design a PCB board which can stack all the sensors on a single board so that the integration is complete. So the temperature readings from the LSM9DS0 sensor will also be displayed on ‘1.8 TFT Colour Display ST7735' screen, and the temperature readings from LSM9DS0 will be verified by the temperature readings from Tmp-36 Temperature sensor, which also will be displayed on the ‘1.8 TFT Colour Display ST7735' screen alongside. The temperature readings from the LSM9DS0 sensor stored in the database are made use by the Android application also.

# Requirements Specification

### Software

#### Mobile Application

Bettin Jacob will be responsible for this part. We are developing the application in Android platform with support for API key 21 and above which supports 95% of Android users worldwide. The SDK we make use of is the Android studio of Google for developing the application. The high point of this application is that it will work without GPS. Instead of using GPS based real-time navigation maps we are using navigation maps without real-time navigation. We could implement that with the use of QR codes to find current location and Dijkstra’s Algorithm which helps to solve the shortest path between any two points on the map. So QR code and Dijkstra’s algorithm, are the backbone of this project. For scanning QR code and finding the current location we are using the camera of the smartphone, for detecting the shortest path we are using Dijkstra's algorithm and for displaying the way, we use the path array coming from the algorithm. Another attraction of the app is that, in case of an emergency, we can directly call security, as there is a specific option to make a call named "call security" on the home page. Then, as an extra feature, we have an address book which has phone numbers of all the major academic schools and facilities, so a student or a visitor or any interested person can get the number, and they can call and get the information of their interested areas. This application will help in promoting Humber College, and everyone can explore Humber Campus easily on a single platform.

In the application, first the user has the option whether he wants the navigation, or he wants to call security, or he wants the access to phone book of major schools, or he wants to check the temperature values at that point.

For navigation, the user needs to scan the nearest QR code through which the app identifies the current location of the user and the user can then type in the destination point, and on clicking a button, it will take the user to the next page, which highlights the shortest/most comfortable path to reach his destination, along with the other available tracks as well. If the user wants to go using another way or the user can follow the highlighted route on the map, it is up to the user to decide.

While using the navigation, the user can have a look at the temperature values of the place where he stands and the temperature of the room/point where he wants to go.

If user clicks ‘Call Security’ button from the main page, app will take the user to a page, where the app needs a kind of confirmation from the user, if he really wants to call the security, and if user again taps ‘call security' button on that page, it will call security directly, else user can go back to the main page. The app also displays the number to which the user is going to make the call.

On Selecting the ‘Phone Book' option from the home page of the App, it shows the names of all the major schools, and by tapping the interested school, it will display the phone number of that school, and if the user touches on that number and select the call icon from the bottom, it will call chosen school. All extensions are included within that number, so the user need not type in any of the extension.

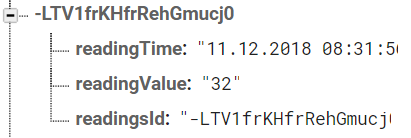
For Temperature values of a room, we have a button named "Temperature Readings”. If the user scans a QR code, the application will display the temperature of that room and also the historical average of temperature of that room.

At last, we have the ‘Quit' button, which will quit from app completely.

### Database

Bettin Jacob will take care of the Database. We make use of the Firebase database provided for free by Google. The database handles two datase. One is recording the data coming from the sensor in each room, and other is storing the info of the rooms/nodes. Sensor data has a time stamp with room number, date and time in the 24-HR format with precision to a second. This time stamp will carry a corresponding reading from the sensor of each room so that we can understand what the temperature readings were at each point of time was. The other data that is held by the application are the information about all the nodes of the graph. The nodes will be room numbers, and each room number will carry a unique id for the application to identify the nodes.

A sample screenshot of temperature reading recorded in the database,



### Firmware

We make use of Raspberry Pi to stack our sensors, and the raspberry pi uses of Raspbian as the operating system. The readings from the sensor are handled using Python programs installed in Raspberry Pi. The Python programs convert the signals from the sensor into human-readable decimal values, and that values are being written into the database. We can decide the interval between two readings to be recorded to the database. It can be read every 10 minutes or every hour or however we want.

### Hardware

#### Development Platform

Anoopjot Kaur Dhallu and Ishan Khuttan will be responsible for maintaining and developing the Hardware development Platform for Humber Campus Navigator which is a Raspberry Pi-3 model B+, which is the latest production of Raspberry Pi 3 featuring a 64-bit quad core processor running at 1.4 GHz and has 1 GB RAM. It supports Wi-Fi and Bluetooth connectivity, along with these features, it can support other devices using the interfaces, I2C and SPI. In the Humber Campus Navigator project, we are connecting our sensors (LSM9DS0, TMP36 & ST7735 1.8 TFT Color Screen) with Raspberry Pi using the i2c interface. The primary function is to display the temperature of a particular place using LSM9DS0 sensor on the android application. LSM9DS0 is a 9-axis accelerometer, and Temperature sensor which reads temperature and the reading is sent to Raspberry Pi, then the python code will take the readings and will convert them into human-readable format and records into a database and also displayed on ST7735 1.8 TFT Color screen. We are cross-checking our temperature readings, with the help of TMP36 temperature sensor which is also displayed on the ST7735 1.8 TFT colour screen. Raspberry Pi is acting as a platform here, to record, convert & display the reading and store them in the database.

### Interface boards and sensors

#### Interface Board

Anoopjot Kaur Dhallu and Ishan Khuttan will be responsible for designing the PCB which will hold all the sensors. The PCB will be directly attached to Raspberry Pi. LSM9DS0 and ST7735 are interacting directly with raspberry pi, but TMP36 needs a microcontroller (ATtiny85) to communicate with raspberry pi, as it an analog device, it cannot interact directly with raspberry pi. For TMP36, TMP36 is connected to ATtiny85 over SDA & SCL CLK lines, and ATtiny85 is connected to Raspberry Pi. The three sensors, LSM9DS0, TMP36 & ST7735 are combined on a single platform, Raspberry Pi, and they will be interacting, using i2c interface.

#### Sensors

LSM9DS0 9-axis accelerometer, Gyroscope, Magnetometer, and Temperature Sensor will take temperature readings and display on ST7735, and these readings will be recorded in Database also.

ST7735- It is a 1.8 TFT Color Display Screen, on which both the temperature readings will be displayed.

TMP-36 Temperature Sensor -> It is an analog sensor, it will be verifying the temperature readings with the LSM9DS0 sensor.TMP36 sensor will take the readings, and send them to ATtiny85 and the ATtiny85 will convert them into a human-readable format and will send to raspberry pi to display on the screen, upon request.

### Other Accessories and Enclosure

The development platform will require a power source, display, and network connection.

The enclosure should be designed in CorelDraw and laser cut from acrylic (Responsibility: Ishan Khuttan). It may be further improved by 3D printing (responsibility: Mechanical Engineering Technology collaborators- if available).

# Conclusion

# Recommendations

# Bibliography

# Appendix