Computer Systems and Networks Project.

Student: TJ Fitzpatrick

Student Number: 20027865

Lecturer: Frank Walsh.

Course: H.Dip in Computer Science.

Contents

1. Introduction.

The purpose of this project was to demonstrate the capabilities of the IOT, and to gain a better understanding of the issues and challenges of a connected world (IOT Infrastructure).

2. Block Diagram.

The following Block Diagram was this project started, the idea was to read analogue pins of the Arduino, transport that data to the raspberry pi, then print it up on some IOT platform.

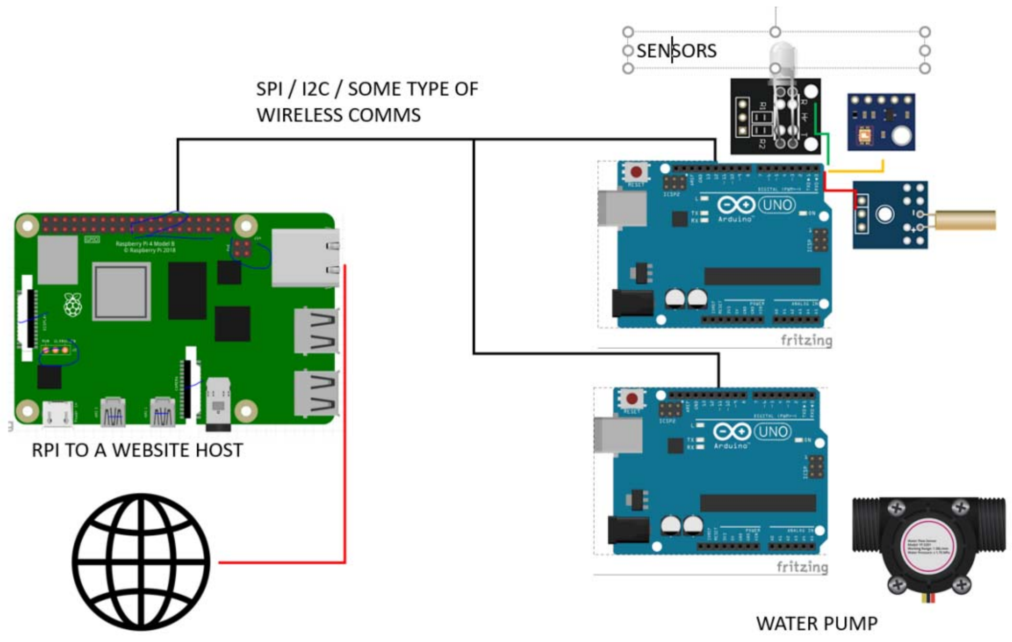


Figure 1 - Block Diagram of Project.

Some of the Sensors I bought in include, DHT11 Temperature and humidity, A soil moisture sensor, a water pump, an LDR and many more I was hoping to build a prototype IOT configuration for.

3. Research.

Whilst building this project I tested each component individually.

3.1 Button Tests.

This simple circuit is to test a button, the button is pressed the LED will light, and a logic 1 value will print up to the serial port, when the button is not pressed a logic 0 is printed to the serial port.

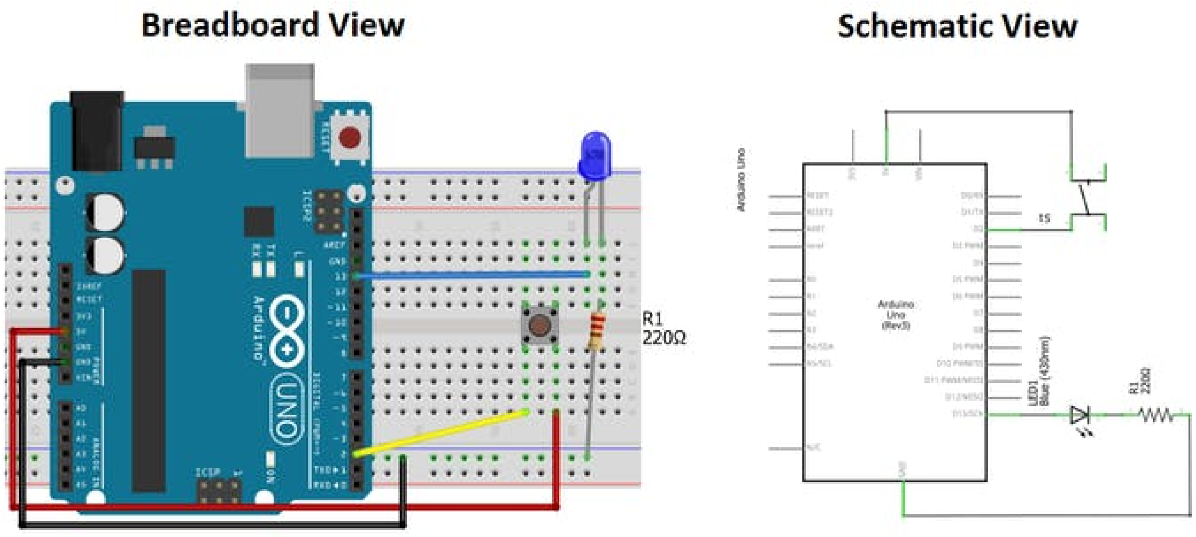
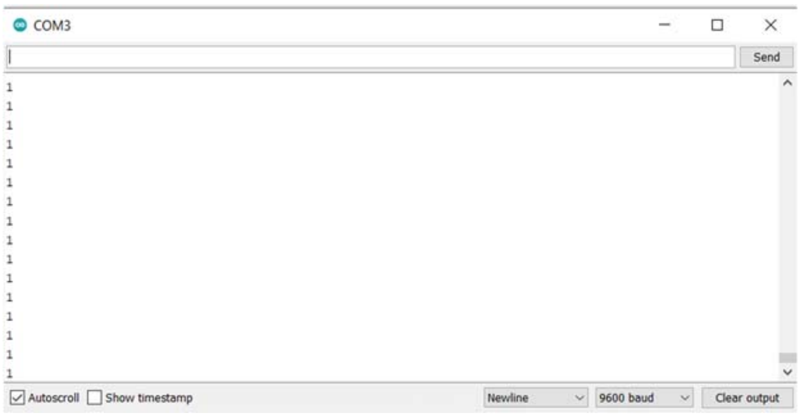
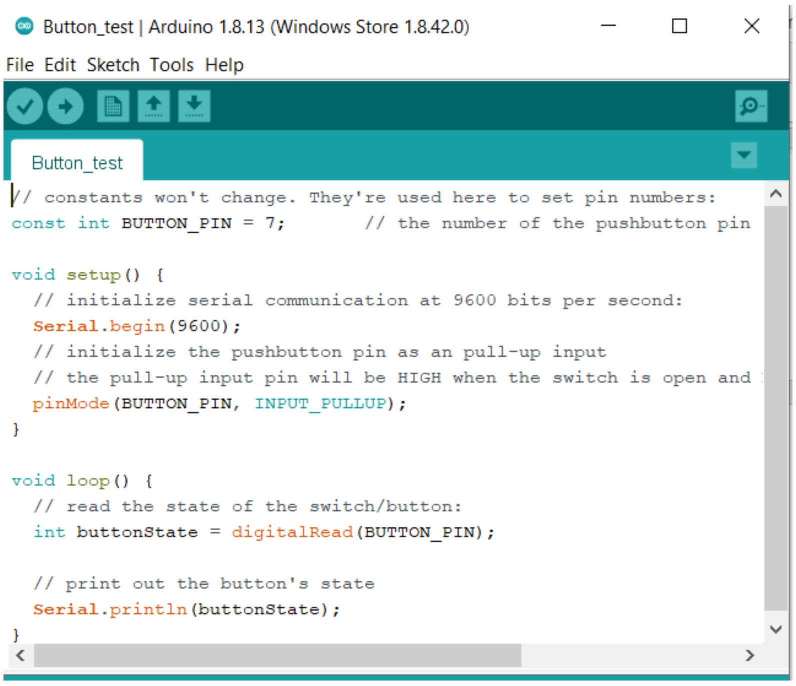


Figure 2 - Arduino Button Test.



```
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
```

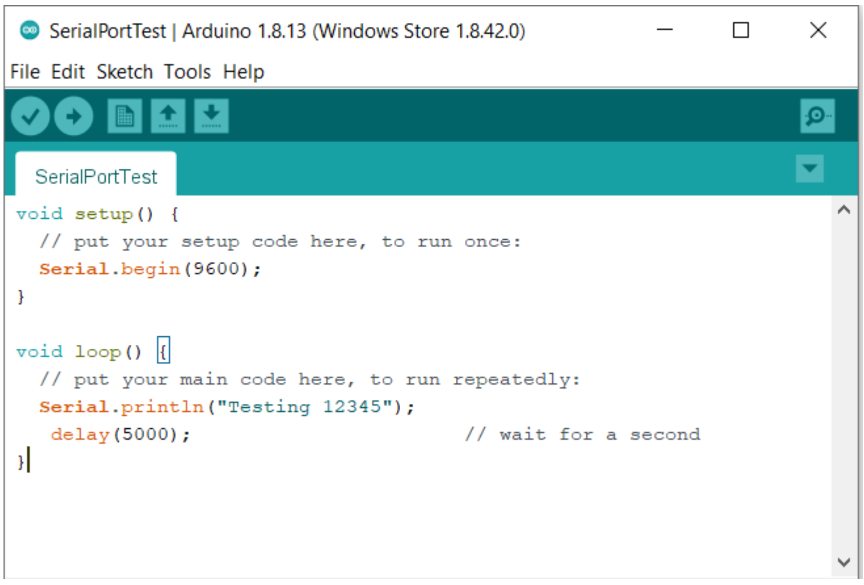
Figure 3 -Serial Port Button Test.


Button_test | Arduino 1.8.13 (Windows Store 1.8.42.0)
File Edit Sketch Tools Help
Button_test

```
// constants won't change. They're used here to set pin numbers:
const int BUTTON_PIN = 7;       // the number of the pushbutton pin

void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
  // initialize the pushbutton pin as an pull-up input
  // the pull-up input pin will be HIGH when the switch is open and
  pinMode(BUTTON_PIN, INPUT_PULLUP);
}

void loop() {
  // read the state of the switch/button:
  int buttonState = digitalRead(BUTTON_PIN);

  // print out the button's state
  Serial.println(buttonState);
}
```

Figure 4 - Button Test Code

3.2 Serial Port Test.


SerialPortTest | Arduino 1.8.13 (Windows Store 1.8.42.0)
File Edit Sketch Tools Help
SerialPortTest

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  Serial.println("Testing 12345");
   delay(5000);                        // wait for a second
}
```

Figure 5 - Serial Test Code.

3.3 Capacitive Soil Moisture Sensor Test.

This sensor is placed inside soil up to each grove on each side, it then sends back an analogue value based on how moist the soil is.

This soil moisture sensor measures soil moisture levels by capacitive sensing rather than resistive sensing like other sensors on the market. It is made of corrosion resistant material which gives it an excellent service life.

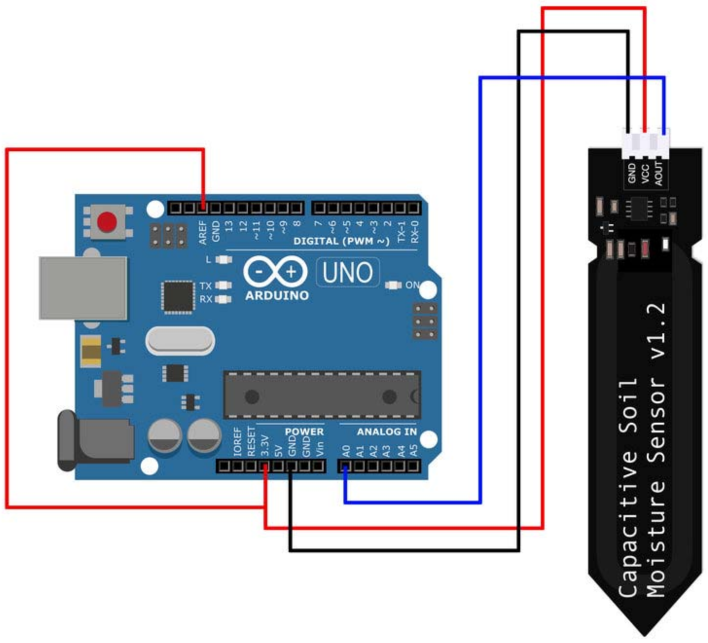


Figure 6 -Capacitive Sensor Module.

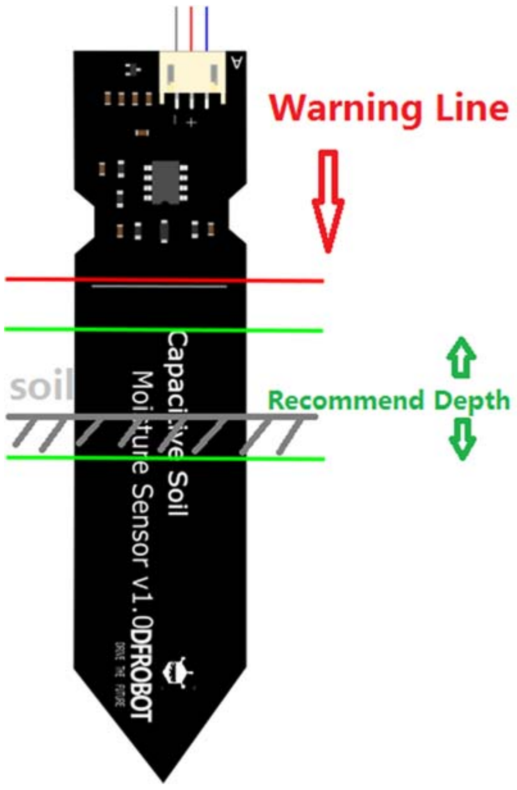
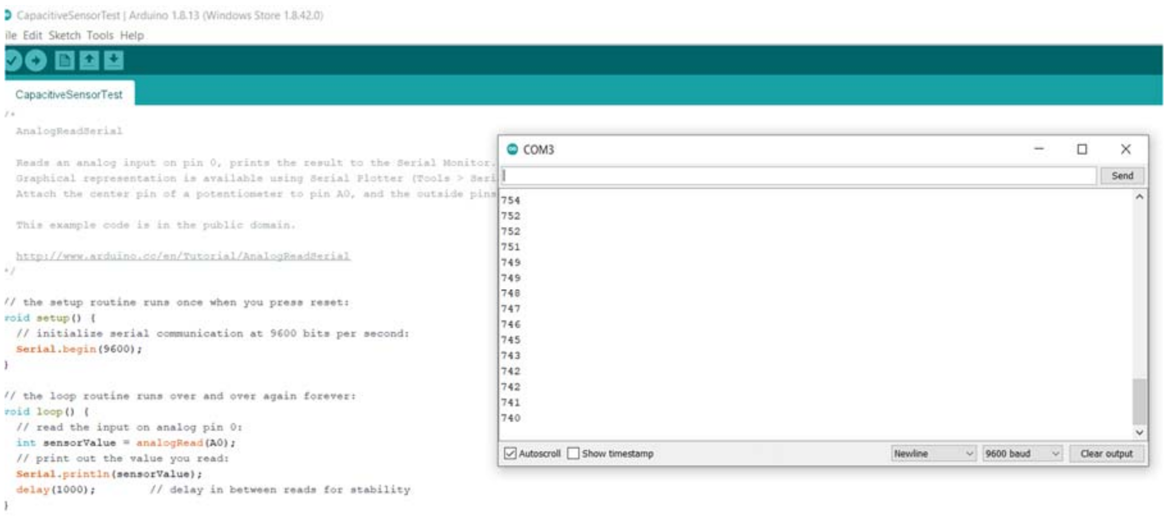


Figure 7 - Recommended Soil Depth.


CapacitiveSensorTest | Arduino 1.8.13 (Windows Store 1.8.42.0)
ile Edit Sketch Tools Help
CapacitiveSensorTest
COM3
Send
Autoscroll
Show timestamp
Newline
9600 baud
Clear output

```
/*
  AnalogReadSerial

  Reads an analog input on pin 0, prints the result to the Serial Monitor.
  Graphical representation is available using Serial Plotter (Tools > Seri
  Attach the center pin of a potentiometer to pin A0, and the outside pins

  This example code is in the public domain.

  http://www.arduino.cc/en/Tutorial/AnalogReadSerial
*/

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  delay(1000);        // delay in between reads for stability
}
```

```
754
752
752
751
749
749
748
747
746
745
743
742
742
741
740
```

Figure 8 - Arduino Comm Port Reading Soil Module.

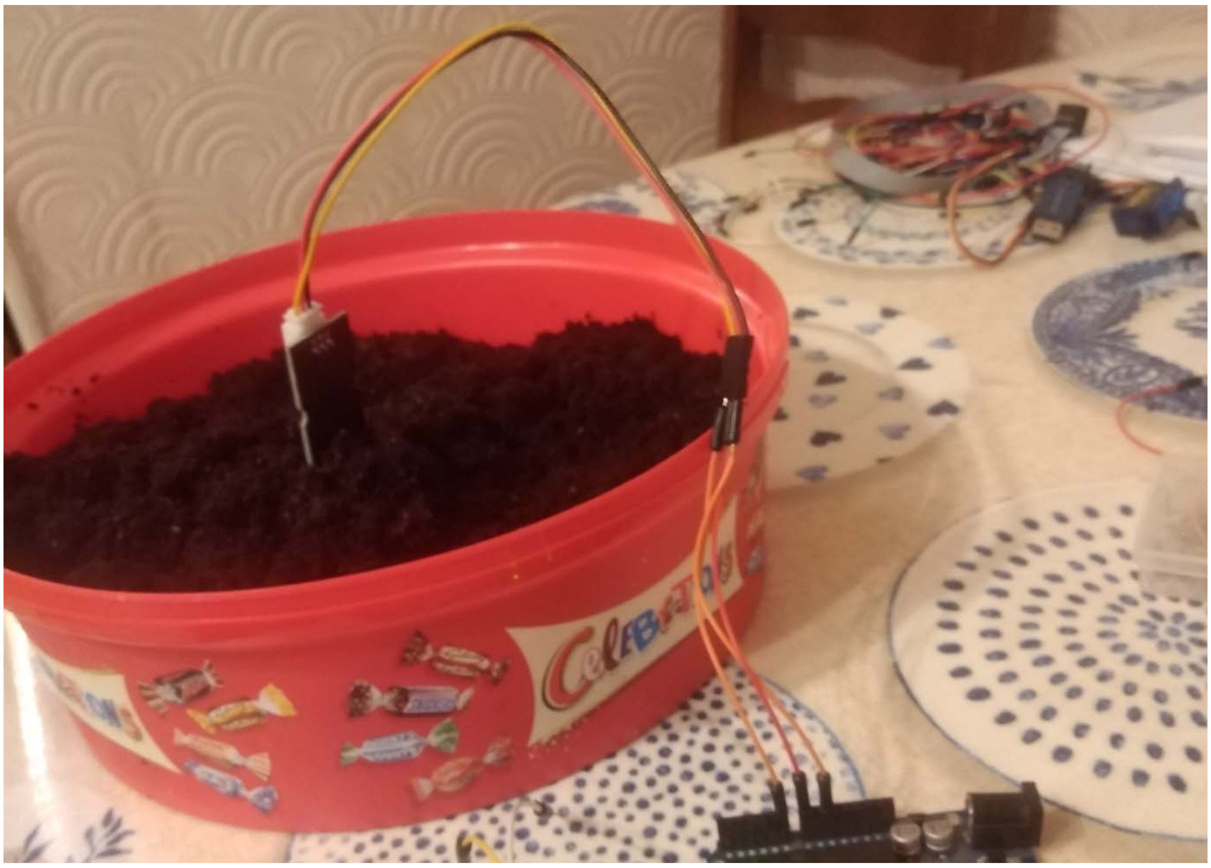


Figure 9 - Actual Image of Soil Module.

3.4 Relay Test.

A relay is just a switch, it can be used to switch on and off high voltage applications.

In my case I replaced a bulb with an LED and I turned it on and off.

I Believe there is a video showing this.

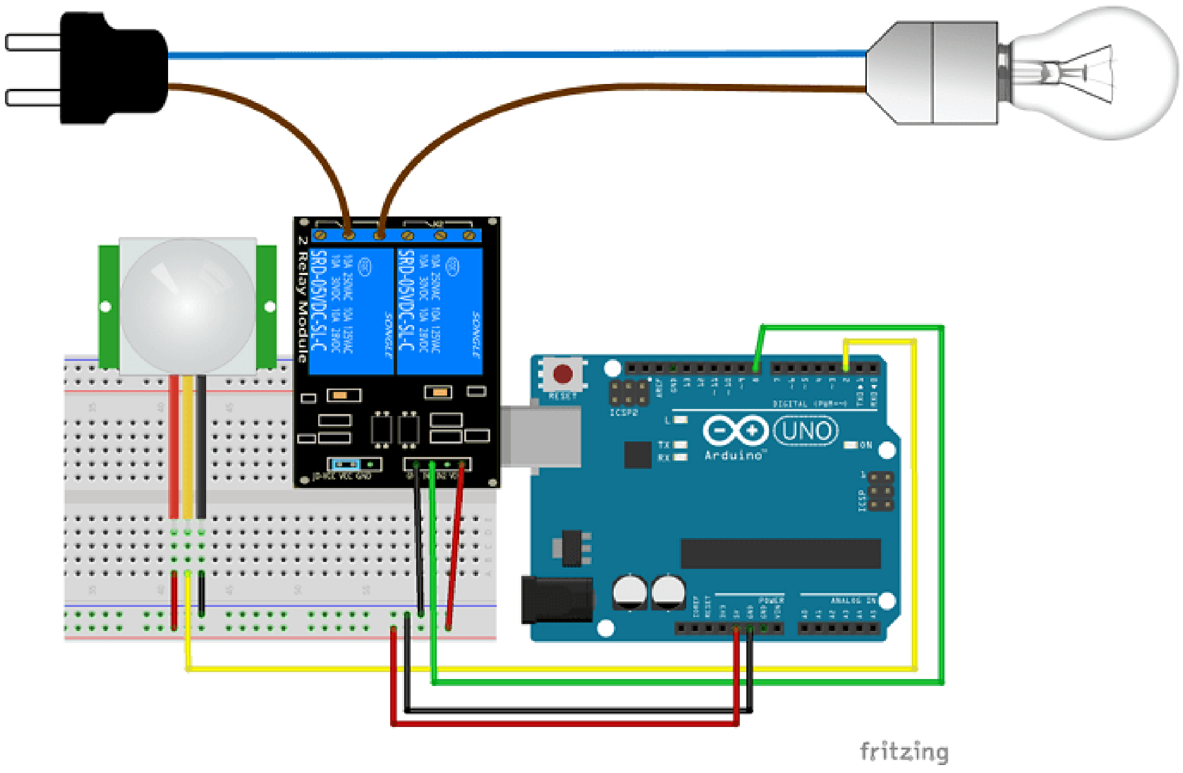


Figure 10 – Arduino Relay Module.

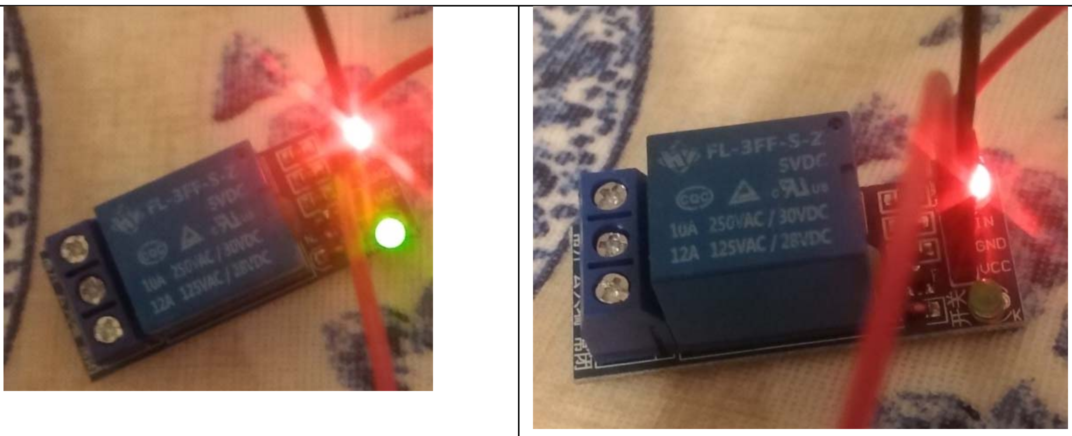


Figure 11 - Demonstrating A Relay.

3.5 Water Pump Test.

This is a low voltage water pump, that turns on and off using 5v.

I use the relay to turn it on and off and I recommend turning it on for 100 milli seconds.

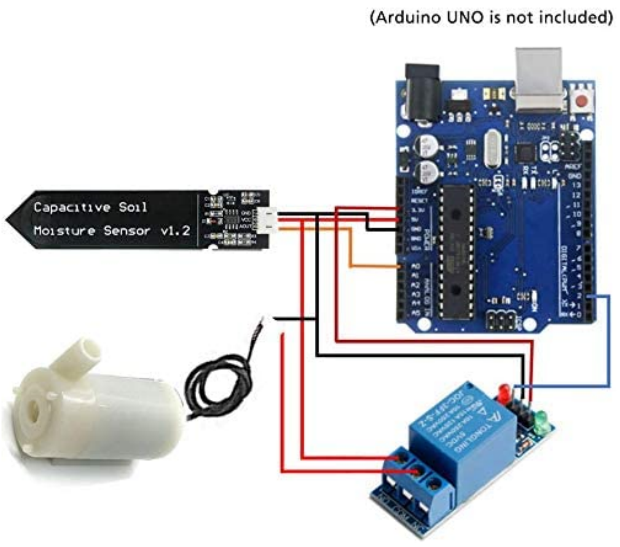


Figure 12 - Water Pump Configuration.

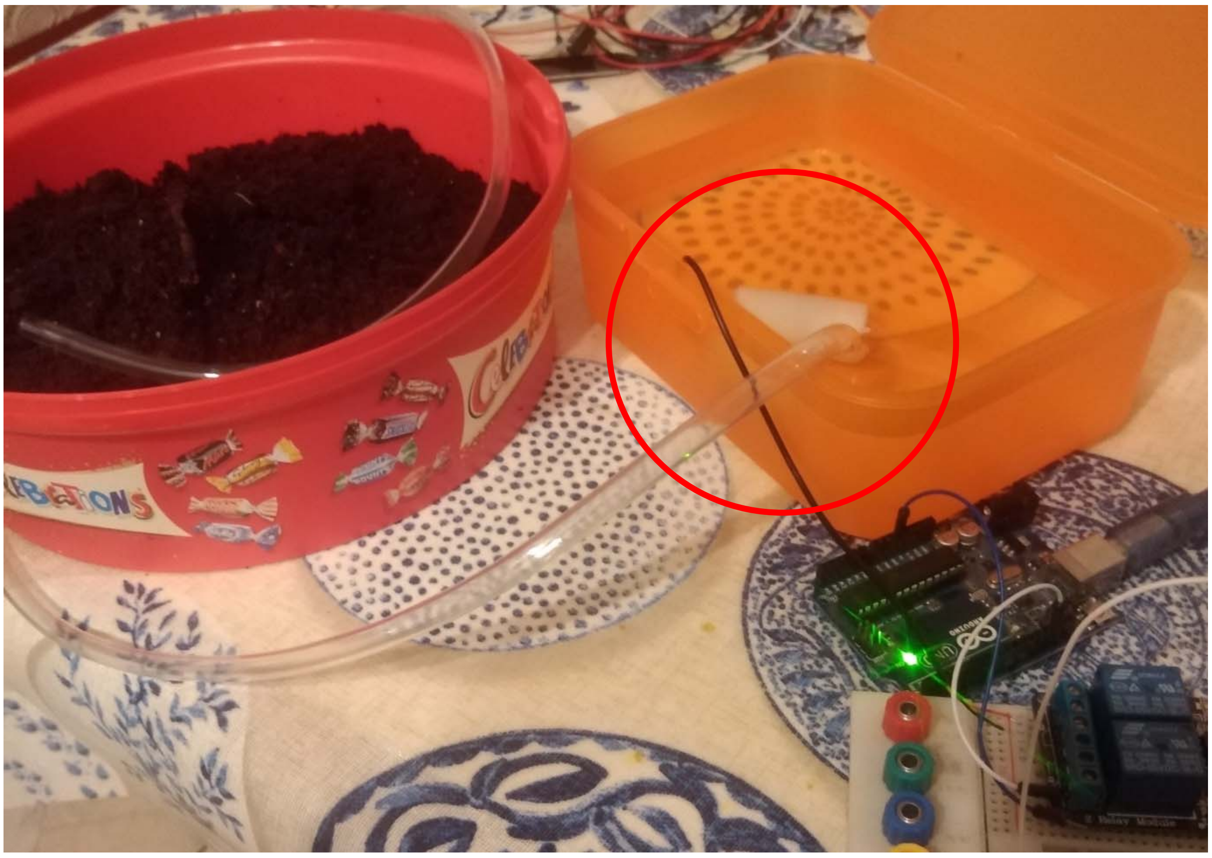


Figure 13 – Water pump Setup.

3.6 Pi Communication Tests to Arduino.

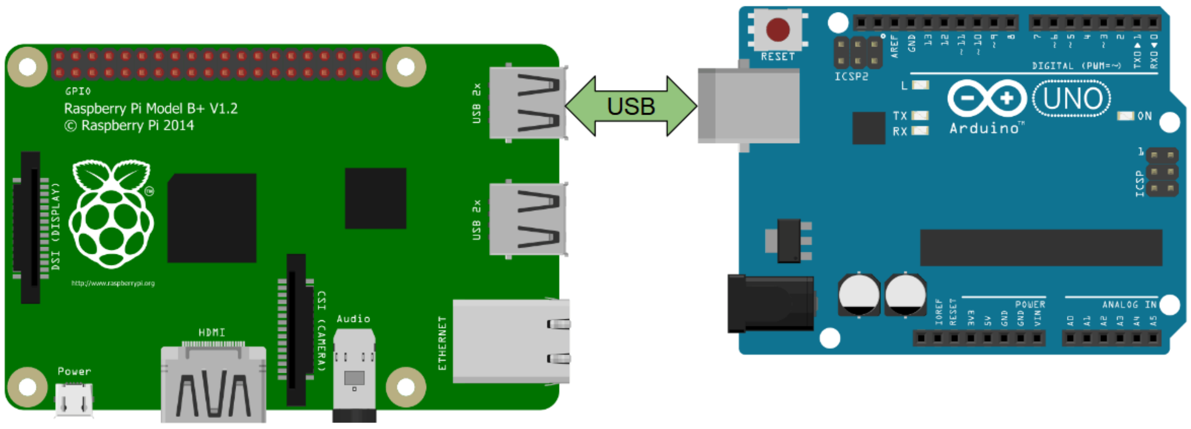


Figure 14 - Arduino to Raspberry Pi Serial Communication

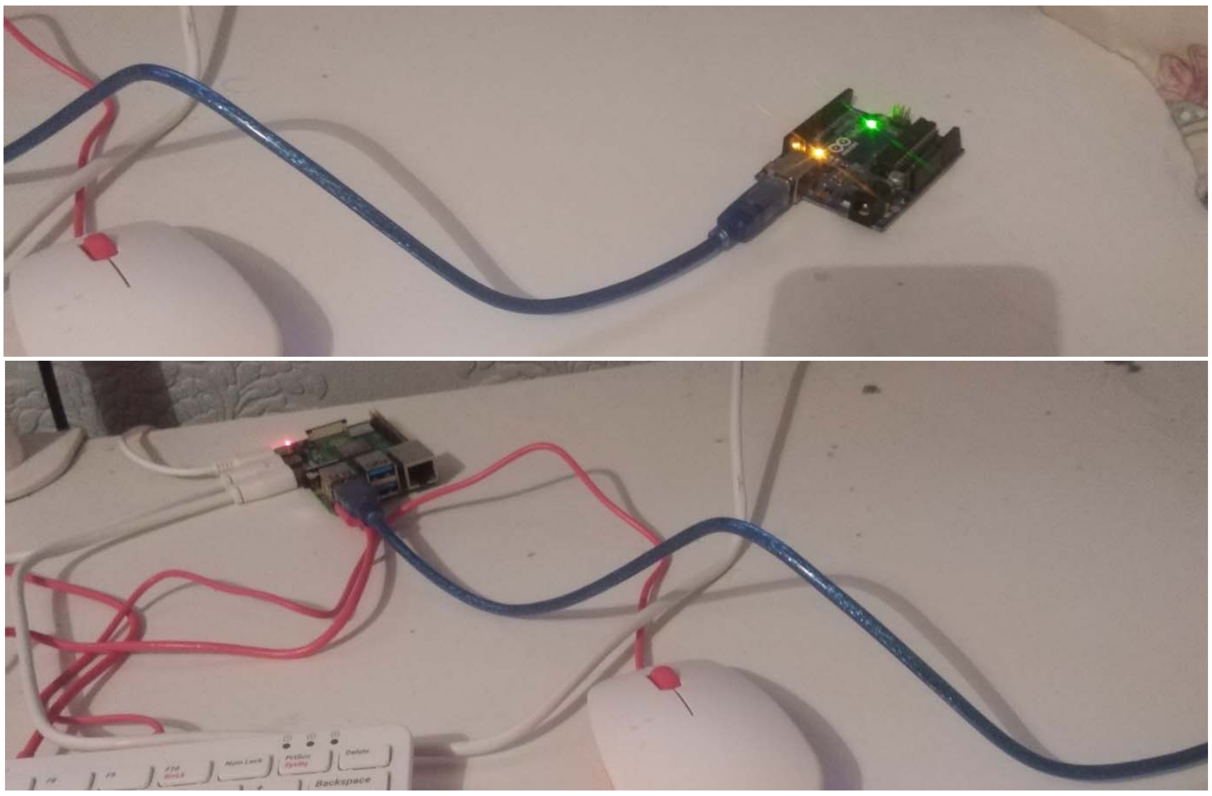
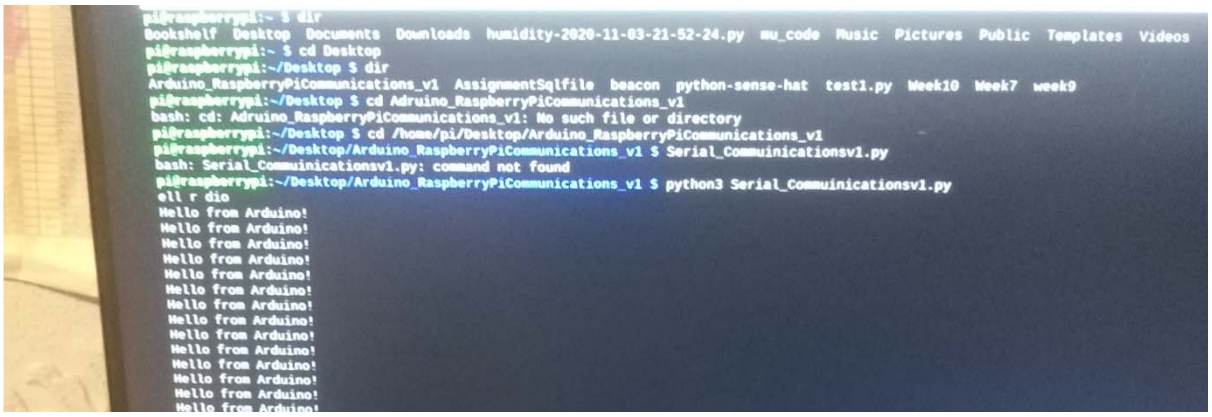
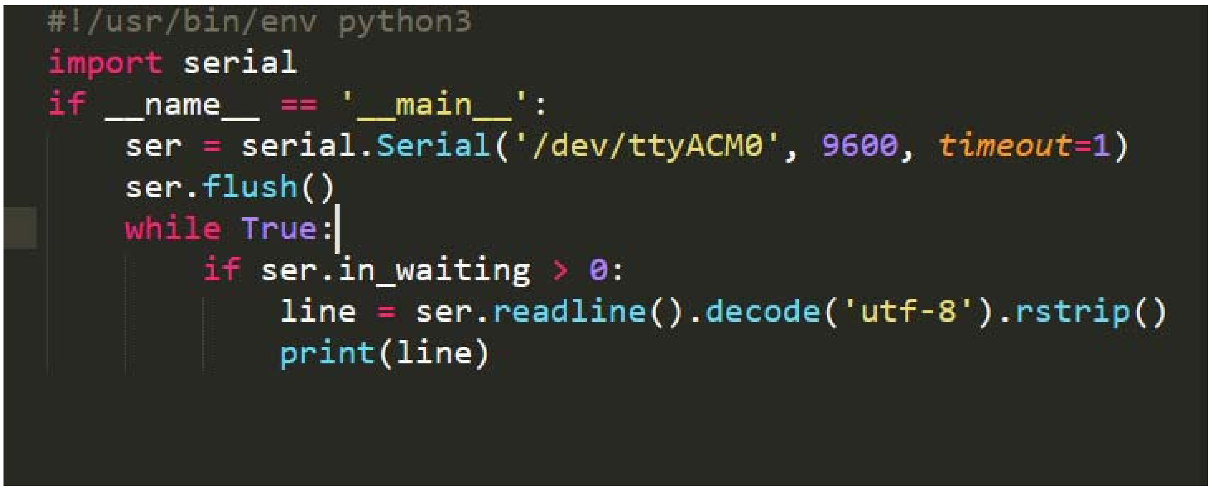


Figure 15 - Arduino And Raspberry Pi Attached Serially Over USB.



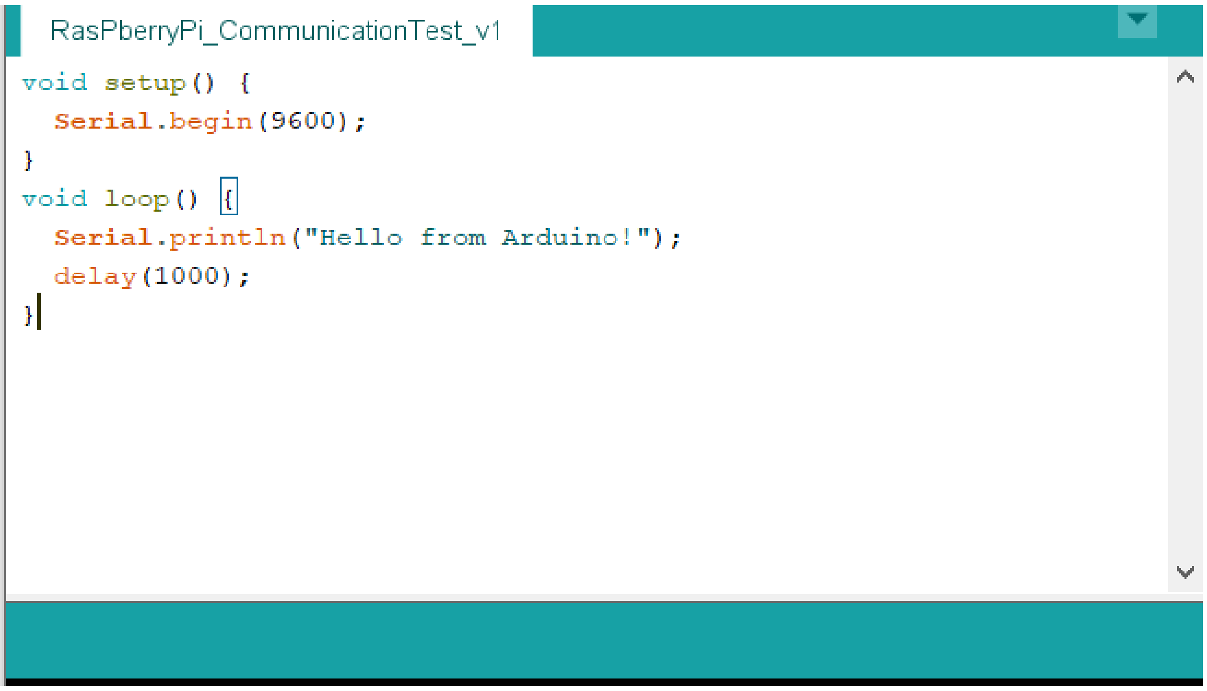
```
pi@raspberrypi:~ $ dir
Bookshelf  Desktop  Documents  Downloads  humidity-2020-11-03-21-52-24.py  mu_code  Music  Pictures  Public  Templates  Videos
pi@raspberrypi:~ $ cd Desktop
pi@raspberrypi:~/Desktop $ dir
Arduino_RaspberryPiCommunications_v1  AssignmentSqlfile  beacon  python-sense-hat  test1.py  Week10  Week7  week9
pi@raspberrypi:~/Desktop $ cd Adruino_RaspberryPiCommunications_v1
bash: cd: Adruino_RaspberryPiCommunications_v1: No such file or directory
pi@raspberrypi:~/Desktop $ cd /home/pi/Desktop/Arduino_RaspberryPiCommunications_v1
pi@raspberrypi:~/Desktop/Arduino_RaspberryPiCommunications_v1 $ Serial_Commuinicationsv1.py
bash: Serial_Commuinicationsv1.py: command not found
pi@raspberrypi:~/Desktop/Arduino_RaspberryPiCommunications_v1 $ python3 Serial_Commuinicationsv1.py
ell r dio
Hello from Arduino!
Hello from Arduino!
Hello from Arduino!
Hello from Arduino!
Hello from Arduino!
Hello from Arduino!
Hello from Arduino!
Hello from Arduino!
Hello from Arduino!
Hello from Arduino!
Hello from Arduino!
Hello from Arduino!
Hello from Arduino!
Hello from Arduino!
```

Figure 16 - Hello from Arduino.



```python
#!/usr/bin/env python3
import serial
if __name__ == '__main__':
    ser = serial.Serial('/dev/ttyACM0', 9600, timeout=1)
    ser.flush()
    while True:
        if ser.in_waiting > 0:
            line = ser.readline().decode('utf-8').rstrip()
            print(line)
```

Figure 17 - Raspberry Pi Code.



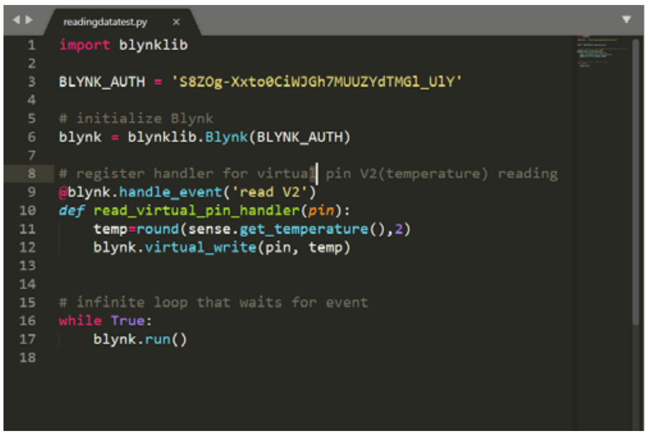
RasPberryPi_CommunicationTest_v1

```cpp
void setup() {
  Serial.begin(9600);
}
void loop() {
  Serial.println("Hello from Arduino!");
  delay(1000);
}
```

Figure 18 - Arduino Raspberry Pi Communication.

3.7 Pi Communication Tests to Blynk Website.

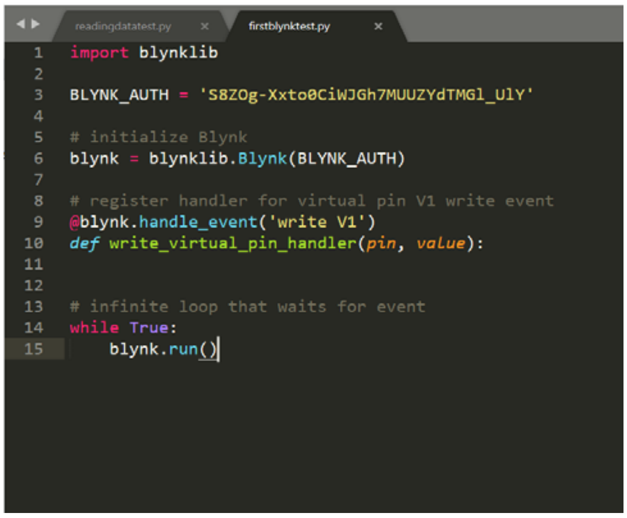
The following is a simple code I used in a test lab I had worked on previously; it allows the raspberry pi to communicate with the Blynk website:



```python
import blynklib

BLYNK_AUTH = 'S8ZOg-Xxto0CiWJGh7MUUZYdTMGl_UlY'

# initialize Blynk
blynk = blynklib.Blynk(BLYNK_AUTH)

# register handler for virtual pin V2(temperature) reading
@blynk.handle_event('read V2')
def read_virtual_pin_handler(pin):
    temp=round(sense.get_temperature(),2)
    blynk.virtual_write(pin, temp)


# infinite loop that waits for event
while True:
    blynk.run()
```

Figure 19 - Raspberry Pi Read Test

It allows the Raspberry Pi read the temperature sensor from the sense hat and communicate it to the Blynk app.
The following code was also used from another lab using the raspberry Pi to write data on a digital output pin.



```python
import blynklib

BLYNK_AUTH = 'S8ZOg-Xxto0CiWJGh7MUUZYdTMGl_UlY'

# initialize Blynk
blynk = blynklib.Blynk(BLYNK_AUTH)

# register handler for virtual pin V1 write event
@blynk.handle_event('write V1')
def write_virtual_pin_handler(pin, value):


# infinite loop that waits for event
while True:
    blynk.run()
```

Figure 20 - Raspberry Pi Write Test

4. Design Methods.

The whole idea with this project was to design an IOT application rapidly over a short period of time.

The process I used was Rapid Prototyping.

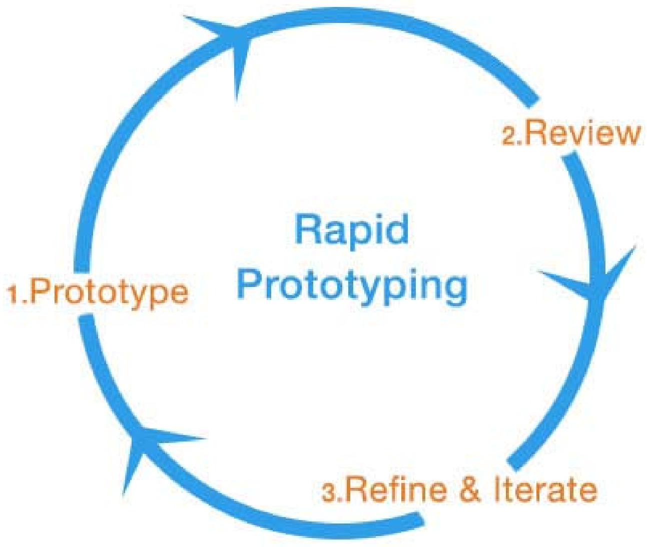


Figure 21 - Rapid Prototyping.

I've broken the project down into 4 parts like the following:

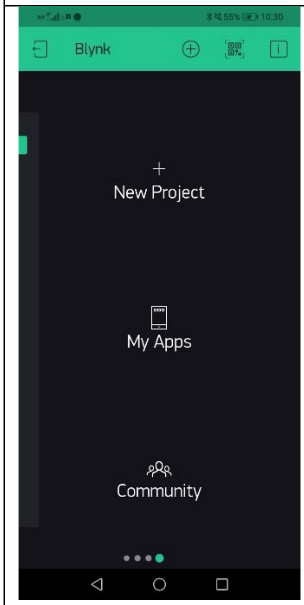
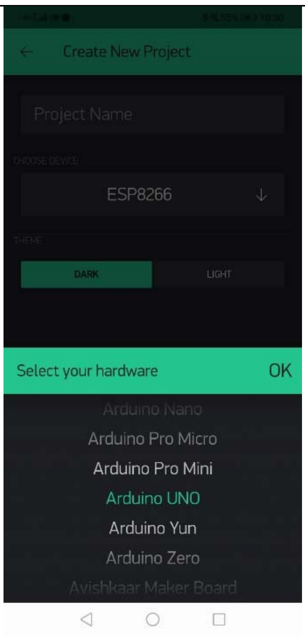
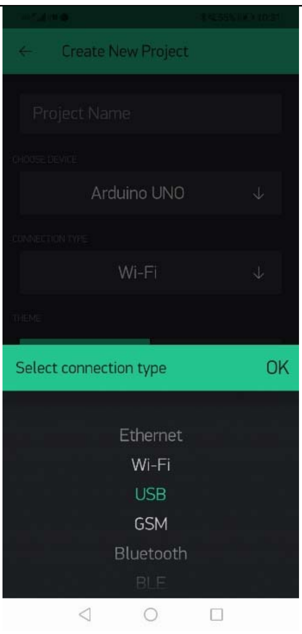
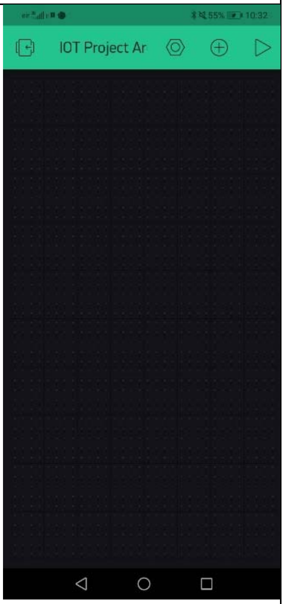
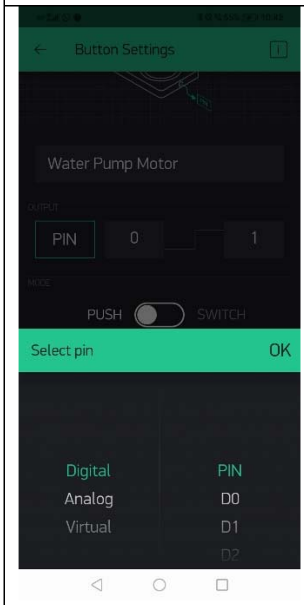
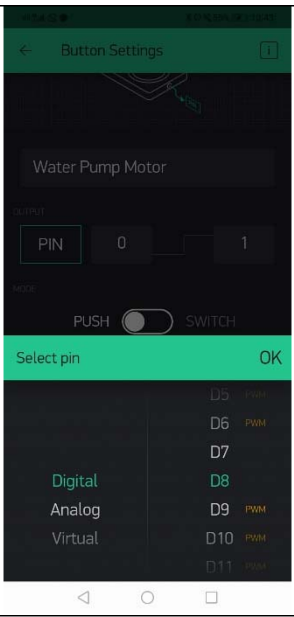
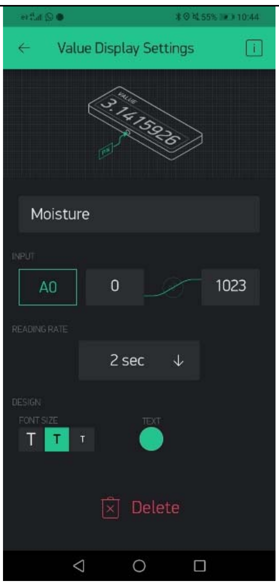
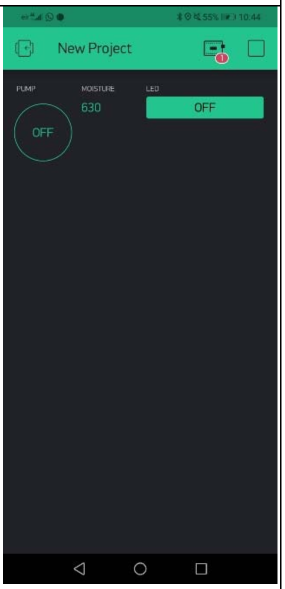
ARDUINO AND PI COMMUNITCATIONS	PI PYTHON COMMUNITCATIONS.
SENSORS TO ADRUINO COMMUNICATIONS	PYTHON TO BLYNK COMMUNICATIONS.

Due to having a lot of the application prototypes individually all I have to really do is combine the individual prototypes.

5. Design.

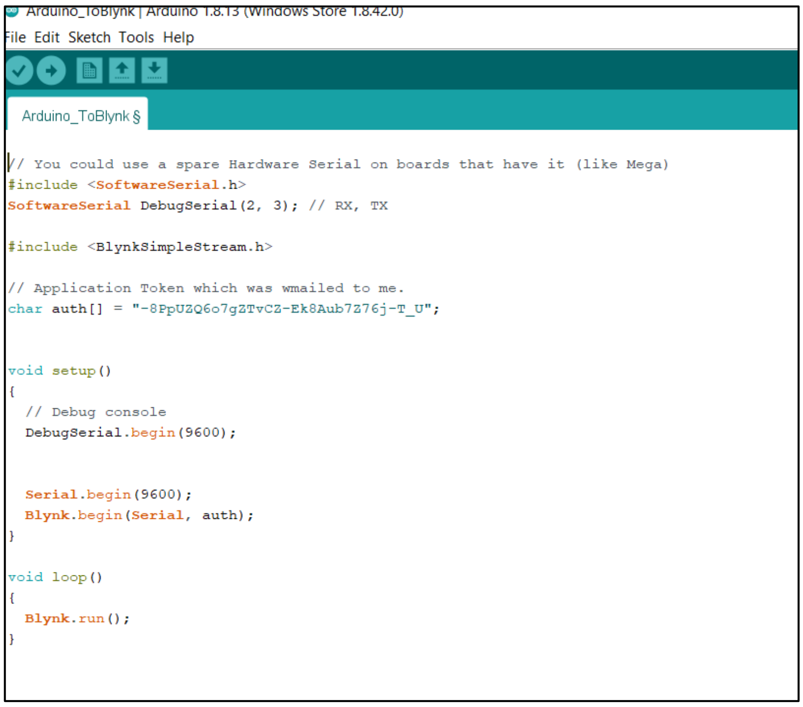
5.1 Arduino To Blynk.

5.1.1 Setting up Blynk.

Step 1	Step 2	Step 3	Step 4
			
A project token was then sent to my email address.			
Setting up the water pump and moisture sensor.			
			
Step 5	Step 6	Step 7.	Step 8.

5.1.2 Setting Up Arduino.

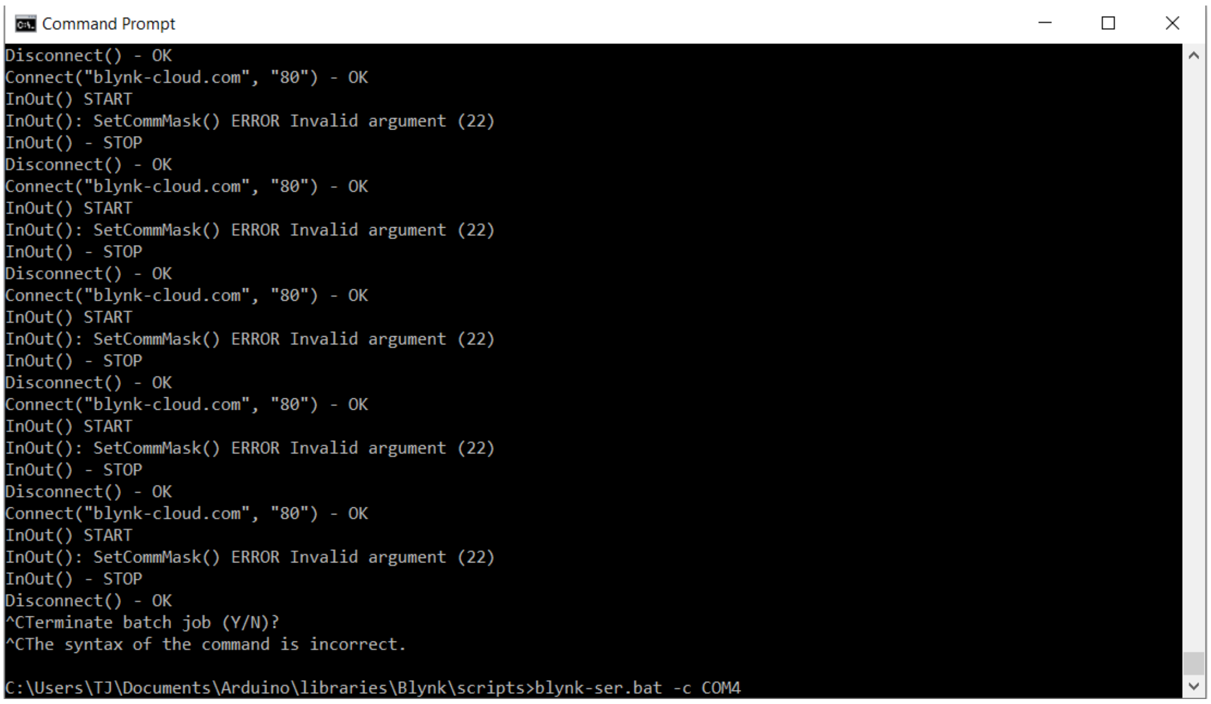
The Code I'm using to run on the Arduino is called: ArduinotoBlynk, this must be loaded onto the Arduino before we proceed.


Arduino_ToBlynk | Arduino 1.8.13 (Windows Store 1.8.42.0)
File Edit Sketch Tools Help
Arduino_ToBlynk §

```
// You could use a spare Hardware Serial on boards that have it (like Mega)
#include <SoftwareSerial.h>
SoftwareSerial DebugSerial(2, 3); // RX, TX

#include <BlynkSimpleStream.h>

// Application Token which was wmailed to me.
char auth[] = "-8PpUZQ6o7gZTvCZ-Ek8Aub7Z76j-T_U";


void setup()
{
  // Debug console
  DebugSerial.begin(9600);


  Serial.begin(9600);
  Blynk.begin(Serial, auth);
}

void loop()
{
  Blynk.run();
}
```

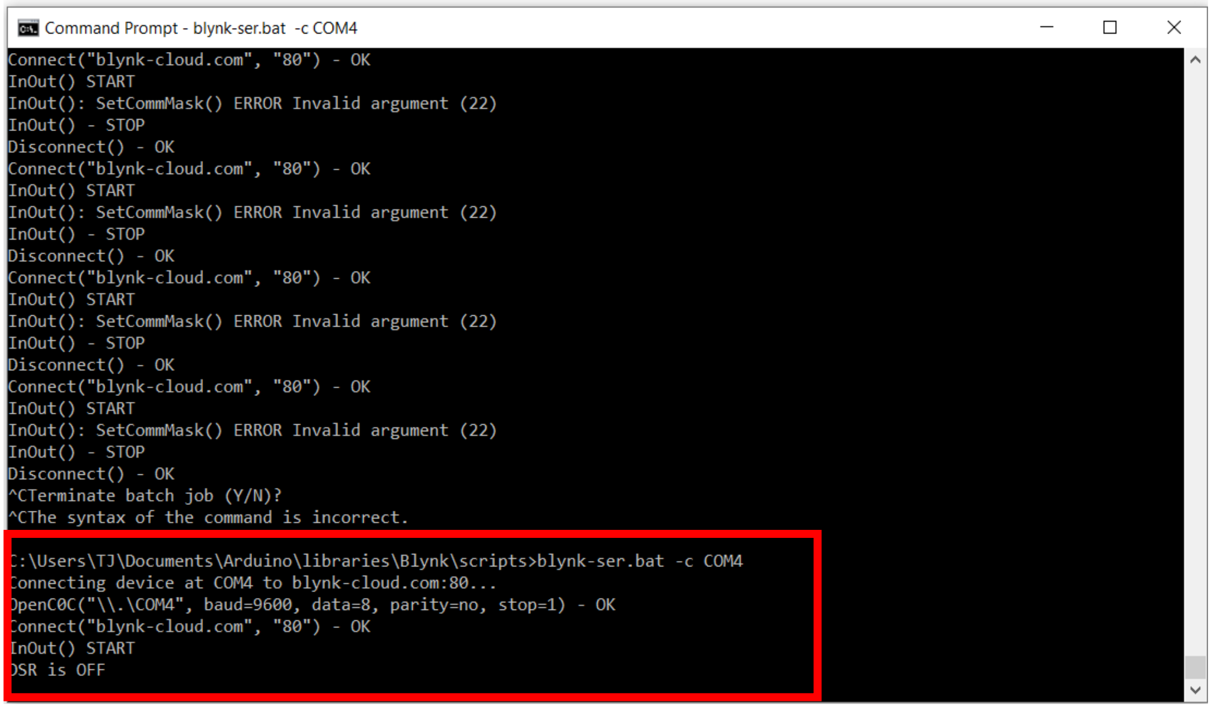
Figure 22 - Arduino To Blynk.

We then must navigate to our library folders and execute the "blynk-ser" bat file.



```
Disconnect() - OK
Connect("blynk-cloud.com", "80") - OK
InOut() START
InOut(): SetCommMask() ERROR Invalid argument (22)
InOut() - STOP
Disconnect() - OK
Connect("blynk-cloud.com", "80") - OK
InOut() START
InOut(): SetCommMask() ERROR Invalid argument (22)
InOut() - STOP
Disconnect() - OK
Connect("blynk-cloud.com", "80") - OK
InOut() START
InOut(): SetCommMask() ERROR Invalid argument (22)
InOut() - STOP
Disconnect() - OK
Connect("blynk-cloud.com", "80") - OK
InOut() START
InOut(): SetCommMask() ERROR Invalid argument (22)
InOut() - STOP
Disconnect() - OK
Connect("blynk-cloud.com", "80") - OK
InOut() START
InOut(): SetCommMask() ERROR Invalid argument (22)
InOut() - STOP
Disconnect() - OK
^CTerminate batch job (Y/N)?
^CThe syntax of the command is incorrect.

C:\Users\TJ\Documents\Arduino\libraries\Blynk\scripts>blynk-ser.bat -c COM4
```

Figure 23 - Blynk-ser bat file.



```
Command Prompt - blynk-ser.bat  -c COM4
Connect("blynk-cloud.com", "80") - OK
InOut() START
InOut(): SetCommMask() ERROR Invalid argument (22)
InOut() - STOP
Disconnect() - OK
Connect("blynk-cloud.com", "80") - OK
InOut() START
InOut(): SetCommMask() ERROR Invalid argument (22)
InOut() - STOP
Disconnect() - OK
Connect("blynk-cloud.com", "80") - OK
InOut() START
InOut(): SetCommMask() ERROR Invalid argument (22)
InOut() - STOP
Disconnect() - OK
Connect("blynk-cloud.com", "80") - OK
InOut() START
InOut(): SetCommMask() ERROR Invalid argument (22)
InOut() - STOP
Disconnect() - OK
^CTerminate batch job (Y/N)?
^CThe syntax of the command is incorrect.

C:\Users\TJ\Documents\Arduino\libraries\Blynk\scripts>blynk-ser.bat -c COM4
Connecting device at COM4 to blynk-cloud.com:80...
OpenC0C("\\.\COM4", baud=9600, data=8, parity=no, stop=1) - OK
Connect("blynk-cloud.com", "80") - OK
InOut() START
DSR is OFF
```

Figure 24 - Blynk-ser Bat file being executed.

The Arduino is not interfaced to the Blynk app, and is controlling the motor and reading the moisture from the pot of soil.

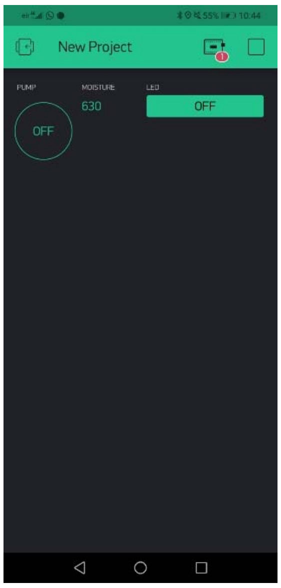


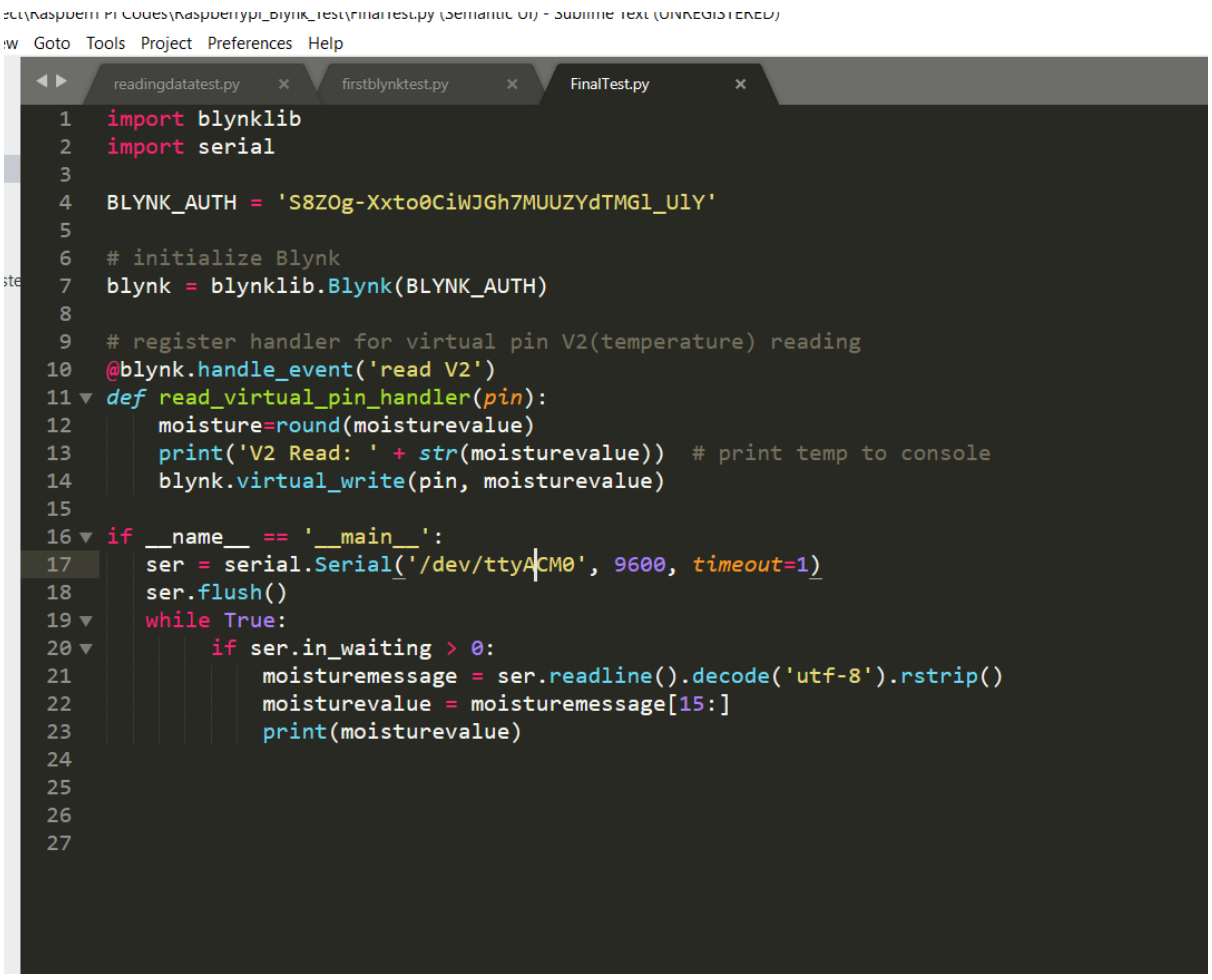
Figure 25 - Blynk App And Arduino.

There is a video documenting this progress called “Arduino to Blynk.mp4” I have also included the Arduino libraries could not be submitted as Git would not accept them.

5.2 Arduino And Raspberry Pi To Blynk.

5.2.1 Setting Up Both Arduino and Pi.

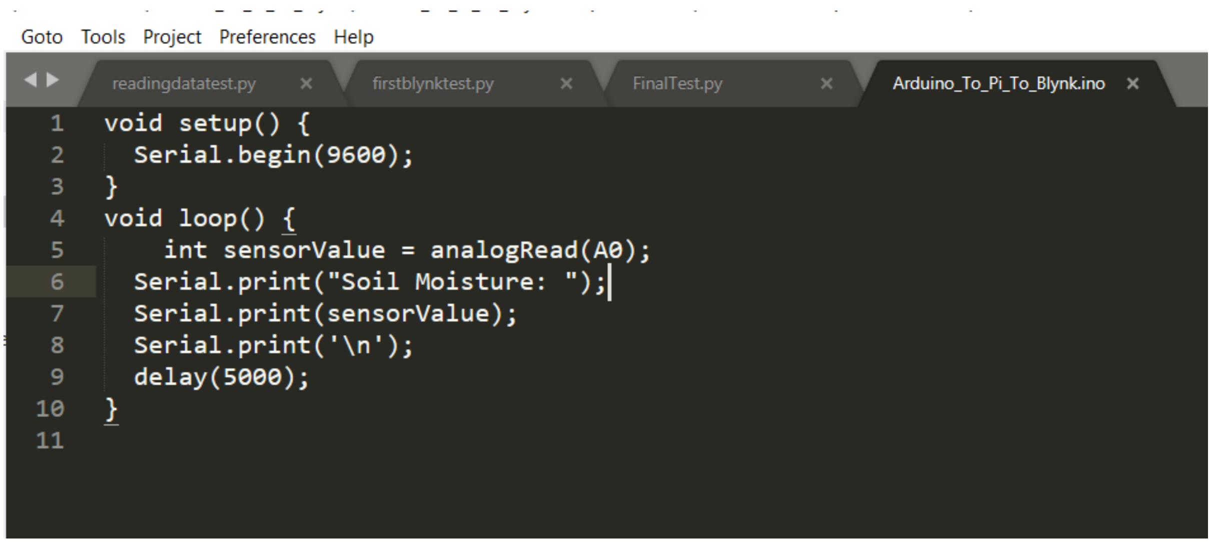
Communication between Arduino and Pi was setup serially over usb correctly, unfortunately I could not get the Pi to communicate to the Blynk website when the serial port was open, so I'm actually receiving the analogue measurements on the Raspberry pi side of things, but I couldn't get these measurements up onto the Blynk Website.



ect\Raspberri Pi Codes\Raspberrypi_Blynk_Test\FinalTest.py (Semantic UI) - Sublime Text (UNREGISTERED)
w Goto Tools Project Preferences Help
readingdatatest.py
firstblynktest.py
FinalTest.py

```python
import blynklib
import serial

BLYNK_AUTH = 'S8ZOg-Xxto0CiWJGh7MUUZYdTMGl_UlY'

# initialize Blynk
blynk = blynklib.Blynk(BLYNK_AUTH)

# register handler for virtual pin V2(temperature) reading
@blynk.handle_event('read V2')
def read_virtual_pin_handler(pin):
    moisture=round(moisturevalue)
    print('V2 Read: ' + str(moisturevalue))  # print temp to console
    blynk.virtual_write(pin, moisturevalue)

if __name__ == '__main__':
   ser = serial.Serial('/dev/ttyACM0', 9600, timeout=1)
   ser.flush()
   while True:
        if ser.in_waiting > 0:
            moisturemessage = ser.readline().decode('utf-8').rstrip()
            moisturevalue = moisturemessage[15:]
            print(moisturevalue)
```

Figure 26 - Raspberry Pi Failed Python Code


Goto Tools Project Preferences Help
readingdatatest.py
firstblynktest.py
FinalTest.py
Arduino_To_Pi_To_Blynk.ino

```
void setup() {
  Serial.begin(9600);
}
void loop() {
    int sensorValue = analogRead(A0);
  Serial.print("Soil Moisture: ");
  Serial.print(sensorValue);
  Serial.print('\n');
  delay(5000);
}
```

Figure 27 - Arduino Failed Code.

So my whole idea was I had a little hand shake going between the Pi and Arduino, the sensor moisture was read and sent to the Pi over the serial port and the actual measurement was to be from character 15 onwards.

This actually worked up to a point then the Raspberry Pi would just freeze.

Half the messages would come up as a printf statement for example a two digit code, when it should have been 3 digit.

Unfortunately, I ran out of time to debug.

6. Results.

Results	
Arduino With Blynk	Working
Raspberry Pi with Blynk.	Working
Arduino and Raspberry Pi with Blynk.	Not Working.

Unfortunately, I ran out of time during this project to get the full pi and Arduino fully functioning but I'm very sure that if I was to get more time, I'm sure there could be a way they can communicate to one another.