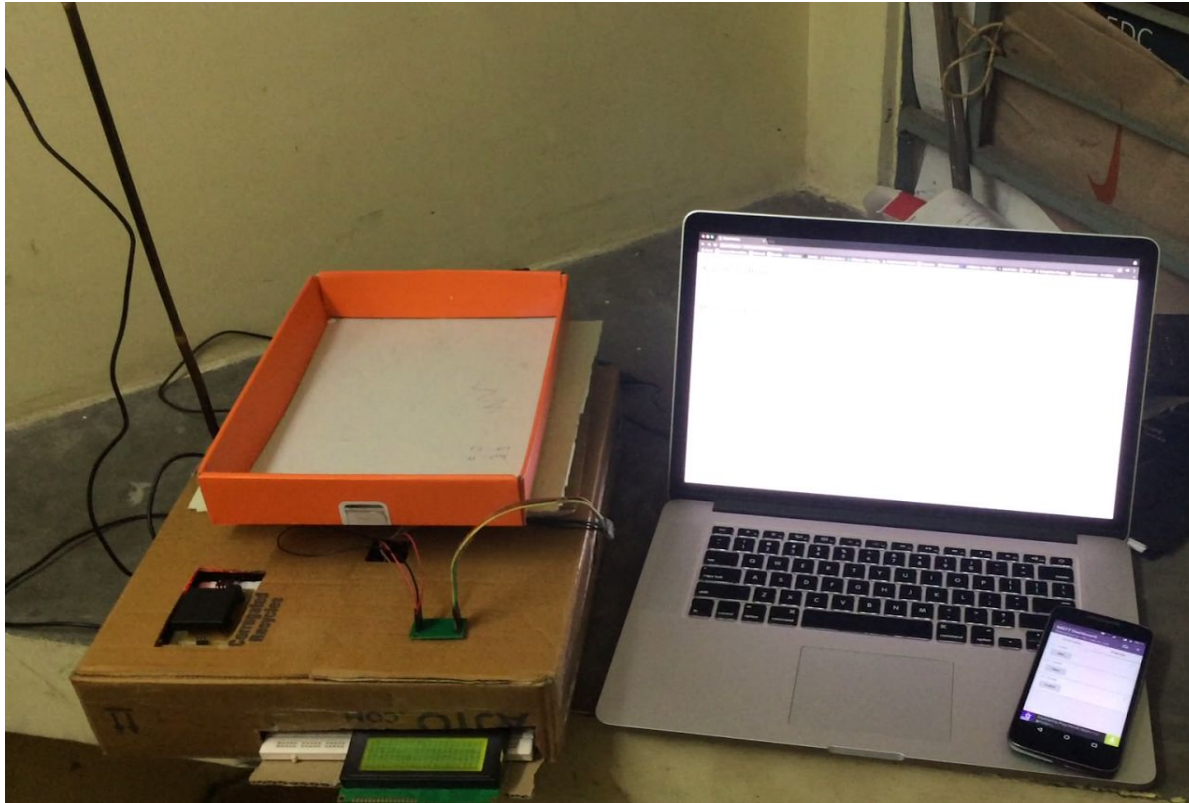


AUTOMATED WEIGHING MACHINE

**Guided by:**

Prof. S.B. Nair

Group Members:

Bharath Chandra(150101066)

Sai Shobith(150101032)

Rajendhar Reddy(150101040)

Harshavardhan (150101075)

INDEX:

1. Introduction.....	3
2. Hardware Modules.....	3
3. Software Packages.....	6
4. Timeline.....	10
Week 1.....	10
Week 2.....	10
Week 3.....	11
5. Interfacing Sensors.....	11
Buzzer circuit.....	12
Load cell.....	13
Rfid reader & Tag.....	15
16x2 LCD screen.....	19
Web cam.....	23
Wifi adapter	23
6. Features.....	24
7. Working.....	24
Mode 1.....	24
Mode 2.....	26
Mode 3.....	26
8. Web application.....	26
9. Limitations.....	30
10. Precautions.....	30
11. Further Scope.....	31

Introduction:

The project is automated weighing machine. The aim of the project is to weigh an object based on its RFID tag. The machine can also be controlled by mobile using MQTT protocol. Machine also captures the image of the object and stores in the database. The machine can run in 3 different modes

Hardware Modules

i) Raspberry pi:

Why?

Raspberry pi used to integrate all interfaced sensors and web application with each other. We can't use a microcontroller like arduino because we need database and regular queries to database are required. We also have USB webcam which can't be connected to arduino. We also need to work with mqtt through wifi and raspberry can easily support these all features. So we used raspberry pi.

ii) Load cell and Hx711:

Why?

As the project is smart weighing machine we need to take weight of the objects, load cell is used to calculate the weight of the object based on the amount of strain and Hx711 is used to connect the raspberry and load cell

Working

A load cell is a sensor or a transducer that converts a load or force acting on it into an electronic signal. We used resistive load cell

Resistive load cells work on the principle of piezo-resistivity. When a load/force/stress is applied to the sensor, it changes its resistance. This change in resistance leads to a change in output voltage when a input voltage is applied.

There are four strain gauges which are connected in wheatstone bridge when the load is applied two of these gauges will be in compression and other two are in tension as a result there will be change in resistance which results in change in current (usually about 20 mv).As this current is very small Hx711 amplify this current and gives us the amplified signal which can be detected by raspberry pi and used for weight calculation.

iii) Buzzer circuit:

Why?

We used buzzer circuit to indicate that processing a particular item is done and you can place the other item. In mode 3 buzzer is used to indicate that the quantity is less than required value

Working

Piezo buzzer is the handy sound generator used in electronic circuits to give audio indication. A Piezo buzzer has a Piezo disc and an oscillator inside. When the buzzer is powered, the oscillator generates a frequency around 2-4 kHz and the Piezo element vibrates accordingly to produce the sound. An ordinary Piezo buzzer works between 3 – 12 volts DC.

iv) 16x2 LCD screen:

Why?

We used screen to display the live weight of the object and to indicate that item already exists or similar item was found.

Working

An LCD is an electronic display module which uses liquid crystal to produce a visible image. The 16x2 LCD display is a very basic module commonly used in DIYs and circuits. The 16x2 translates o a

display 16 characters per line in 2 such lines. In this LCD each character is displayed in a 5×7 pixel matrix.

v) Rfid reader and tags:

Why?

In order to identify or know the objects easily RFID is used. Every object has unique id (i.e RFID tag), RFID reader is used to read that particular item/object

Working

RFID tags contain an integrated circuit and an antenna, which are used to transmit data to the RFID reader (also called an interrogator). The reader then converts the radio waves to a more usable form of data. RFID (Radio Frequency Identification) uses electromagnetic fields to read, monitor and transfer data from tags attached to different objects. It is not necessary that the cards are to be in visibility of the reader, it can be embedded in the tracked object. The tags can be actively powered from a power source or can be passively powered from the incoming electromagnetic fields.

vi) Wifi dongle for pi:

Why?

As we have included mqtt using wifi we need to connect raspberry pi to the wifi, so wifi dongle is used to make the system wireless.

vii) USB Webcam:

Why?

In order to take image of the given item webcam is used.

viii) Resistors:

Why?

1. To adjust the contrast and LED of the LCD.

Software Packages

i)mqtt

we have used paho-mqtt for connecting Raspberry Pi to mqtt broker. To install it in Python run the following command.

```
$ pip install paho-mqtt
```

we have installed mqtt broker in Raspberry Pi itself. In order to communicate with Raspberry Pi we need another mqtt device. So, we have installed `MQTT Dashboard` app from Android store.

create username and password for mqtt broker. we have used `username` and `passwd` as username and password for mqtt broker. mqtt broker uses `1883` as the port for communication.

since mqtt broker is also in raspberry pi, the python code can communicate with mqtt broker using `127.0.0.1` as hostname, `1883` as port, username and password.

we used wifi adapter to connect Raspberry pi to `hotspot` and connected android phone to the same hotspot. The ip address given by the hotspot to raspberry pi is `192.168.43.108`. MQTT Dashboard can communicate with mqtt broker using `192.168.43.108` as hostname, `1883` as port, username and password.

for Python to communicate with mqtt broker we need to import a package.

```
import paho.mqtt.client as mqtt
```

then we need to start a client

```
test_client = mqtt.Client()
```

then we need assign some functions to the client as follows

```
test_client.on_publish = on_publish
```

```
test_client.on_message = on_message
test_client.on_connect = on_connect
```

`on_publish()` does nothing

```
def publish(mosq, userdata, mid):
    pass
```

`on_connect()` prints a message to console and connect to the topic `topic_1`.

```
def on_connect(client, userdata, flags, rc):
    print("Connected with result code "+str(rc))
    client.subscribe("topic_1")
```

we need to provide username, password and then connect using ip and port.

```
test_client.username_pw_set("username", password="passwd")
test_client.connect("127.0.0.1", 1883, 60)
```

then we loop forever. what this does is start listening to mqtt messages asynchronously.

```
test_client.loop_forever()
```

when we connect to mqtt broker, on_connect() function is executed.

whenever we publish something, on_publish() function is executed.

whenever we receive a message, on_message() function is executed.

to publish a message the following must be done.

```
test_client.publish("topic", "message")
```

where `topic` and `message` must be replaced with your own topic and message.

The weighing machine has 3 modes and there is a function for each one of them.

```
mode_one() # for mode one.
mode_two() # for mode two.
mode_three() # for mode three.
```

mode one is for adding a single object or viewing the details of a single object. It publishes weight and RFID tag to the topic-`topic_2`.

mode two is for getting the number of items, if weight of single item of same type exists in the database. It published number of items to topic-`topic_2`.

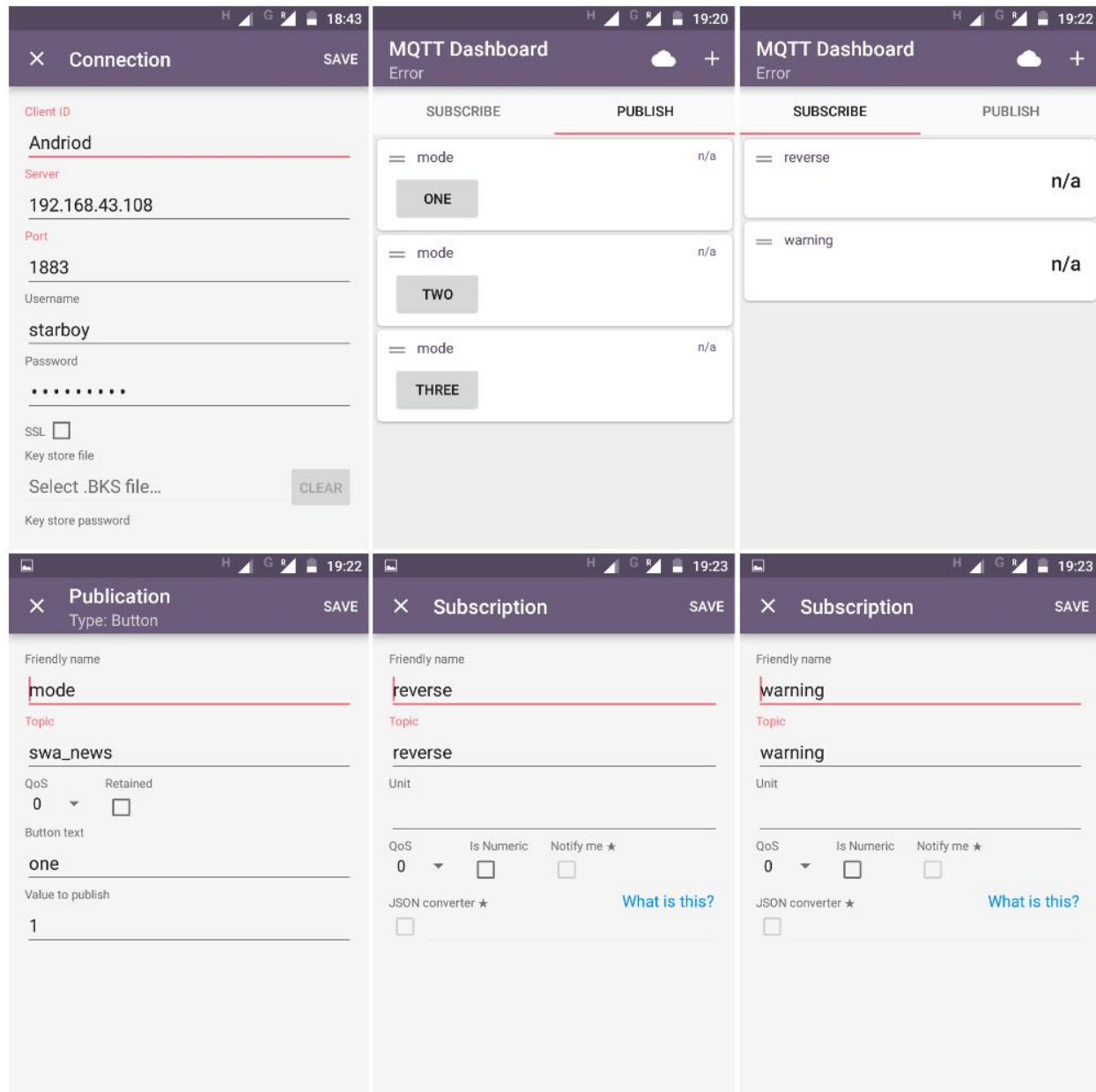
mode three is for monitoring the weight of something. A Threshold value to used by this mode. If its weight is more than the value, buzzer doesn't make any noise and the message `safe` is published to the topic `topic_3`. If its weight is less than the value, buzzer stats making noise and the message `critical` is published to the topic `topic_3`.

the messages that can be sent to the topic-`topic_1` are `1`, `2` and `3`.

When `topic_1` receives message, it triggers `on_message()` function in the python and runs mode 1 or 2 or 3 accordingly.

```
def on_message(client, userdata, msg):
    print("[ "+msg.topic+": "+str(msg.payload))
    a_string = str(msg.payload)
    if (a_string == "1"):
        mode_one()
    elif (a_string == "2"):
        mode_two()
    elif (a_string == "3"):
        mode_three()
```

Settings for MQTT Dashboard App to connect it to mqtt broker:



ii)Apache, Mysql and Php

To install Apache we must install the Metapackage apache2. This can be done by searching for and installing in the Software Centre, or by running the following command.

```
$sudo apt-get install apache2
```

To install MySQL you must install the Metapackage mysql-server. This can be done by searching for and installing in the Software Centre, or by running the following command.

```
$sudo apt-get install mysql-server
```

To install PHP we must install the Metapackages php5 and libapache2-mod-php5. This can be done by searching for and installing in the Software Centre, or by running the following command.

```
$sudo apt-get install php5 libapache2-mod-php5
```

iii)Python Serial:

We need python serial package to read from serial port of raspberry pi. Our os had Py serial so we didn't install. But if it is needed for some OS Run the following command below.

```
$sudo pip install pyserial
```

iv)Fswebcam:

We need a package to take image from webcam in raspberry for that we have installed fswebcam package. It can be installed using the following command.

```
$sudo apt-get install fswebcam
```

Timeline:

We have explained our project progress during 3 weeks below.

Week 1:

In first week we tried to use 3 wire load cell. We have made suitable connections for 3 wire load cell but loadcell was unable to work. We then tried using 4 wire load cell and able to work with it properly. We have interfaced RFID reader with raspberry pi.

Week1 conclusions:

1. Interfaced load cell
2. Interfaced RFID reader.

Week 2:

In second week we worked with buzzer circuit and webcam . We have tried to work with mysql database. While working on it some error took place because of which mysql didn't work. Later we switched to new os(Ubuntu

mate) in order to work with mysql and wifi. At the end of week 2 we have integrated all individual sensors with the system and we have implemented working modes for the system. We haven't integrated RFID reader because it requires MySQL database which was broken in our case.

Week2 Conclusions:

1. Interfaced buzzer circuit.
2. Worked with webcam.
3. Worked with MySQL and it was broken.
4. Integrated all interfaced sensors into one system.
5. Implemented working modes of the system.

Week 3:

In third week we came to know that raspbian os can't support wifi and our mysql database was also broken in our current OS we switched to new OS(ubuntu mate). This week we have implemented mqtt subscription messages for each mode. We have also implemented control from mobile to machine using mqtt. We implemented mySql database and integrated database with the main code and web application which used to explore the items present in the database. We have written timeout handling and serial exception handling for RFID reader.

Week 3 Conclusions

1. Interfaced 16x2 LCD screen.
2. Connected using wifi dongle.
3. Interfaced all sensors with the system.
4. Implemented exception handling and timeout for RFID reader.
5. Build a web application for overview of database.
6. Implemented mqtt for both controlling and receiving messages of system
7. Implemented an encoding scheme to have only one image of several similar objects in order to save disk space.
8. Build a case to enclose the system.

Interfacing Sensors:

Explained interfacing sensors and schematic diagrams

buzzer circuit:

It has 3 connections to raspberry pi.

They are ground, VCC, I/O.

connections:

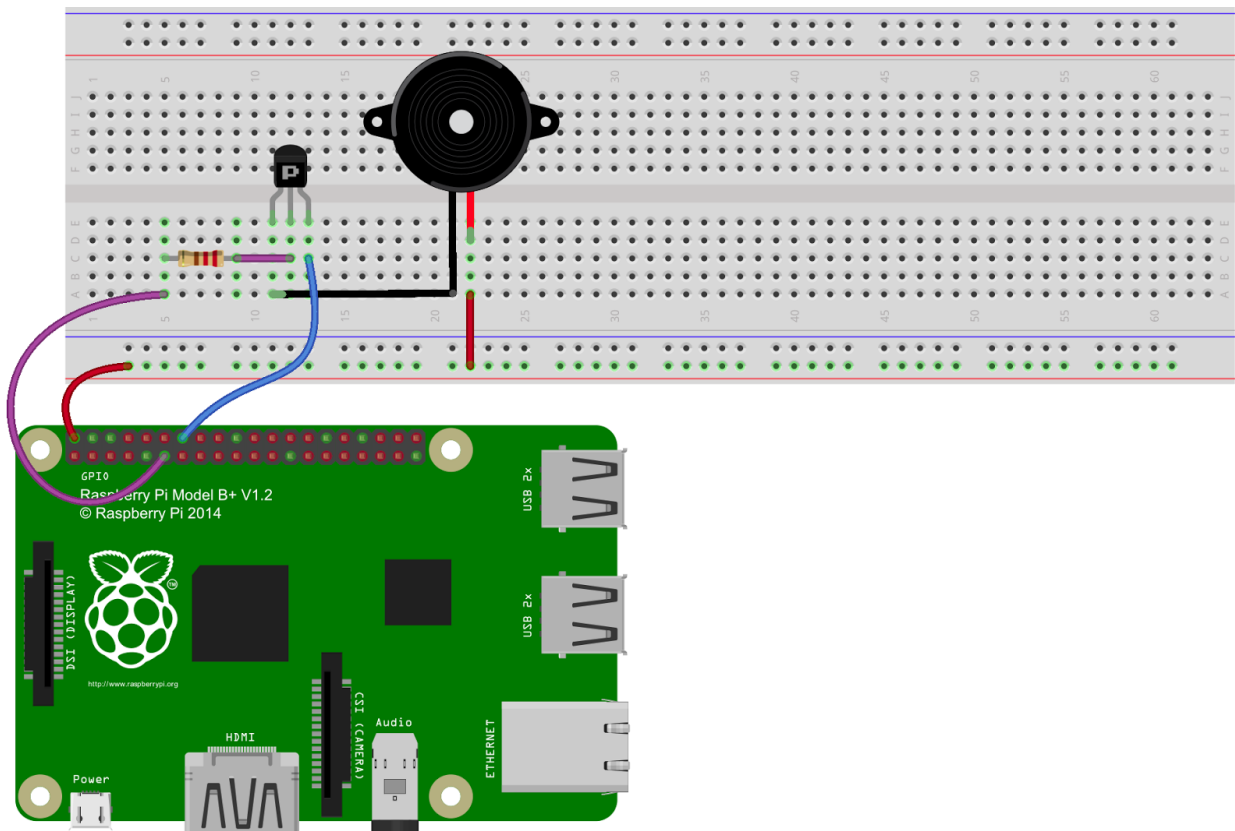
ground - ground pin.

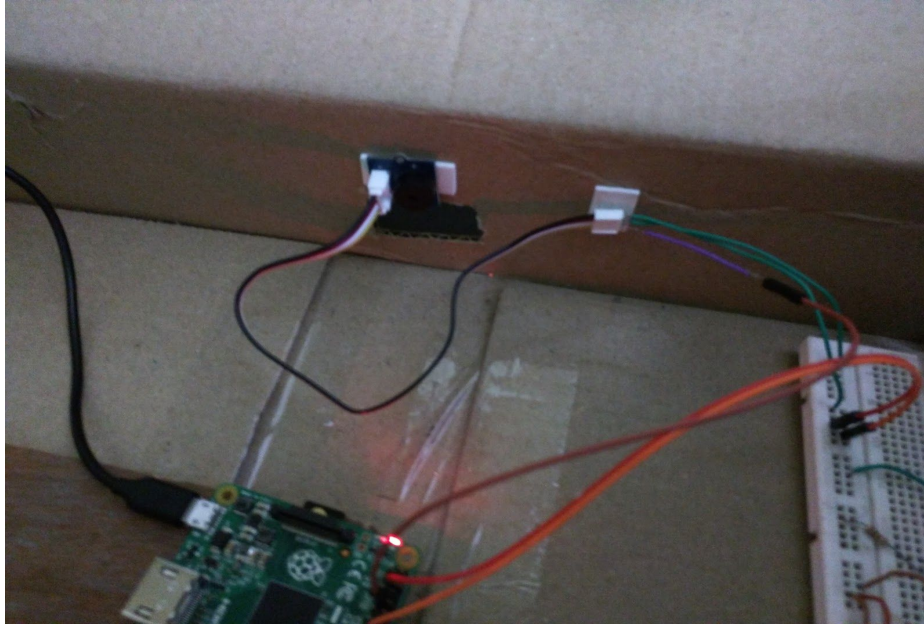
VCC - 5 volts.

I/O - GPIO4.

The working of buzzer circuit is simple. It starts making noise when I/O (GPIO4) is high and doesn't make any noise when I/O (GPIO4) is low.

There need to be a time gap between making I/O high and then low for the buzzer to make noise.



**load cell:**

The most important thing to build your own scale is a “load cell”, which is a metal bar with a hole in the center. This is available for different weight classes (up to 1kg, up to 5kg, up to 50kg, etc.). Even though some have a different form, all are provided with four cables. To read out the values, the HX711 weight sensor is also required. We have used the green HX711.

Before the load cell is connected to the HX711 weight sensor, it should be mounted on the two plates.

If the construction is complete, we can go to the HX711. The four cables of the Load Cell must be connected to the weight sensor. The green HX711, however, has six connections, of which we only need four for the cables.

The connection is as follows:

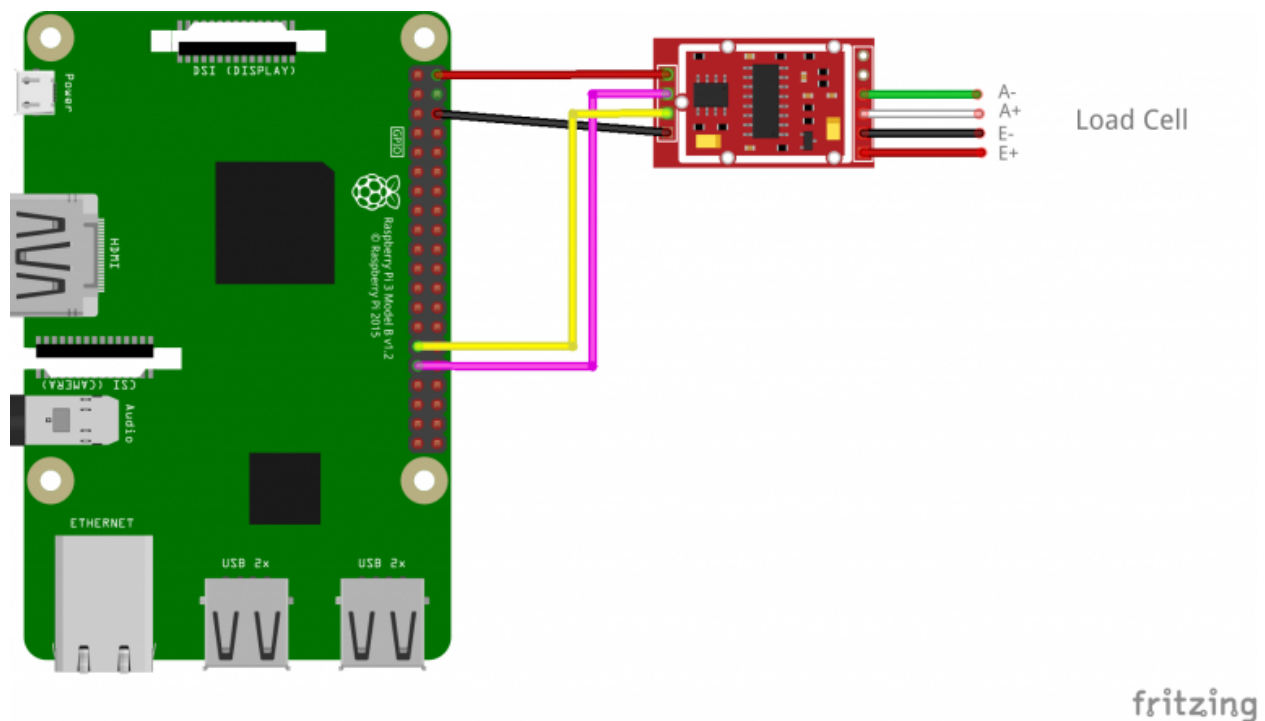
- Red: E+
- Black: E-
- Green: A-
- White: A+

The pins labeled B+/B- remain empty. Apparently there are versions of the sensor. Where the pins are labeled S+/S- instead of A+/A-.

Now you just have to connect the sensor to the Raspberry Pi. Since this also has only four connections, the wiring is quite simple:

- VCC to Raspberry Pi Pin 2 (5V)
- GND to Raspberry Pi Pin 6 (GND)
- DT to Raspberry Pi Pin 29 (GPIO 5)
- SCK to Raspberry Pi Pin 31 (GPIO 6)

Schematically, the connection to a Raspberry pi 2 then looks as follows:



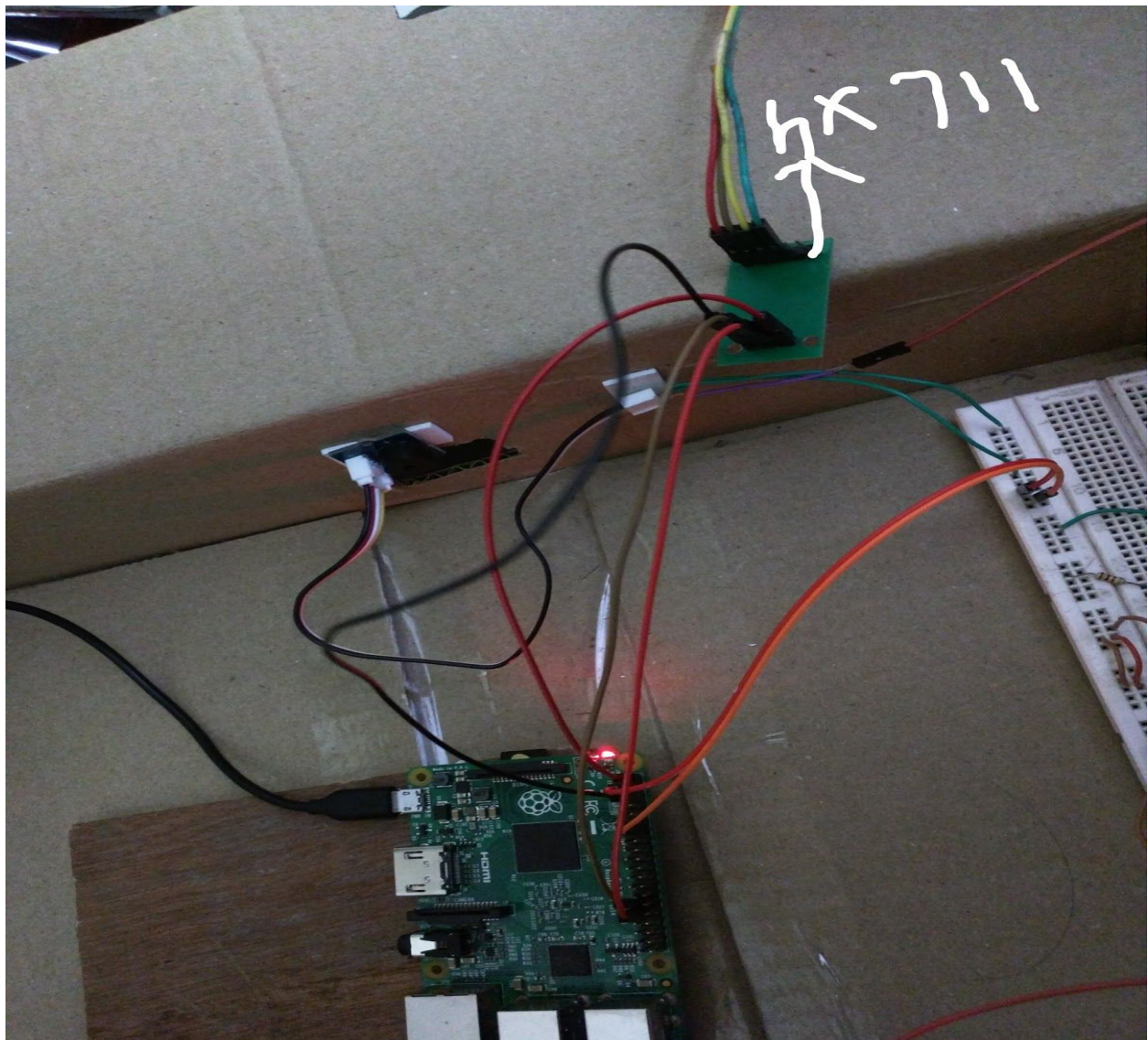
Calibrating load cell:

The correct calibration of the weight sensor and the Raspberry Pi balance is crucial. For this we need a comparison object whose weight we know. For example, We have taken known weight of 150gms, since it is recommended to choose an average value of the maximum. Place it on the scale and run it again with `sudo python final.py`. The displayed values can be positive as well as negative. In my case were displayed at 150

grams values around -60150. My reference value is thus $-60150 \div 150 = -401$.

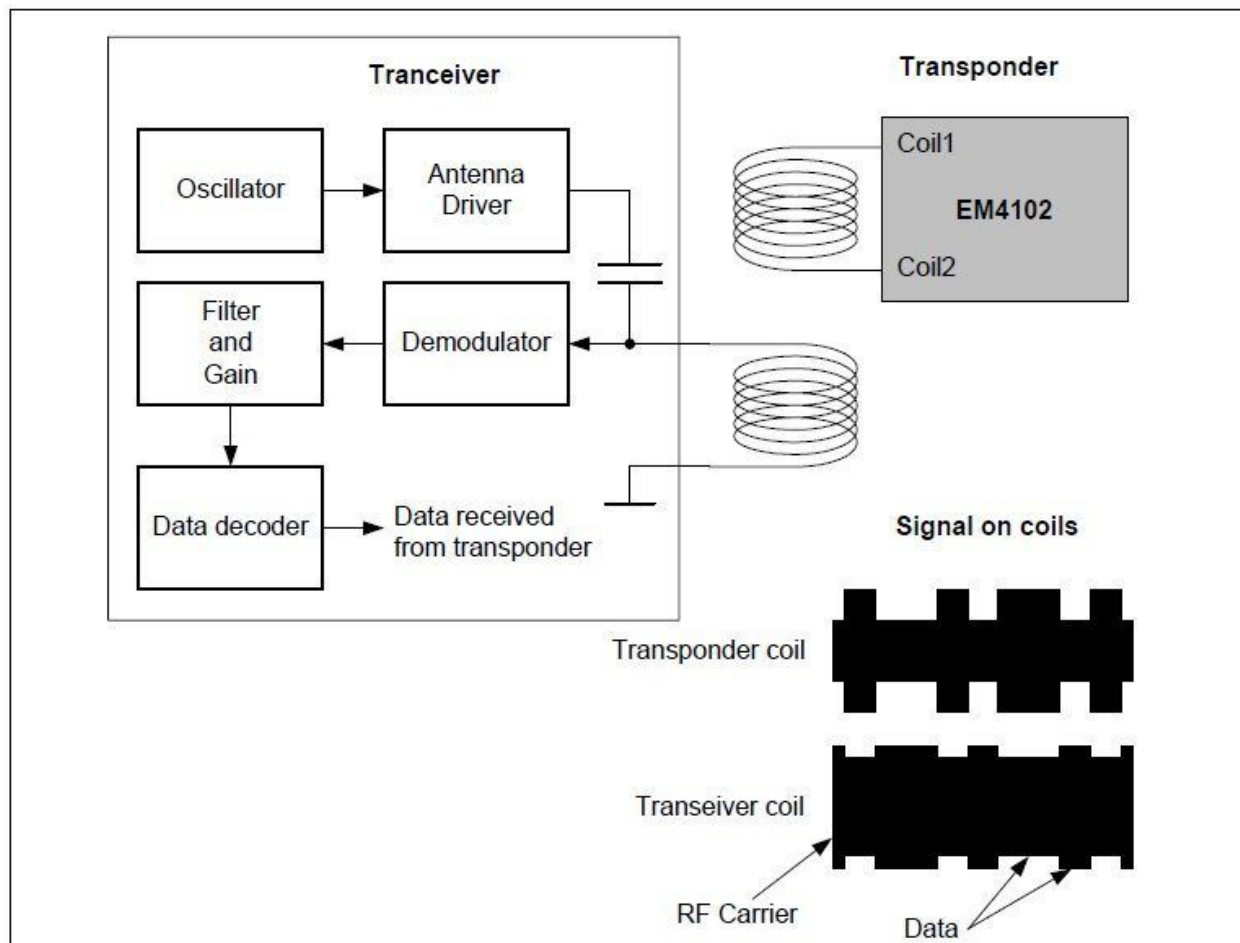
We then edit the sample file in the same way as above described , removed the comment hashtag and enter this value accordingly.

Connect hx711 and raspberry pi image:

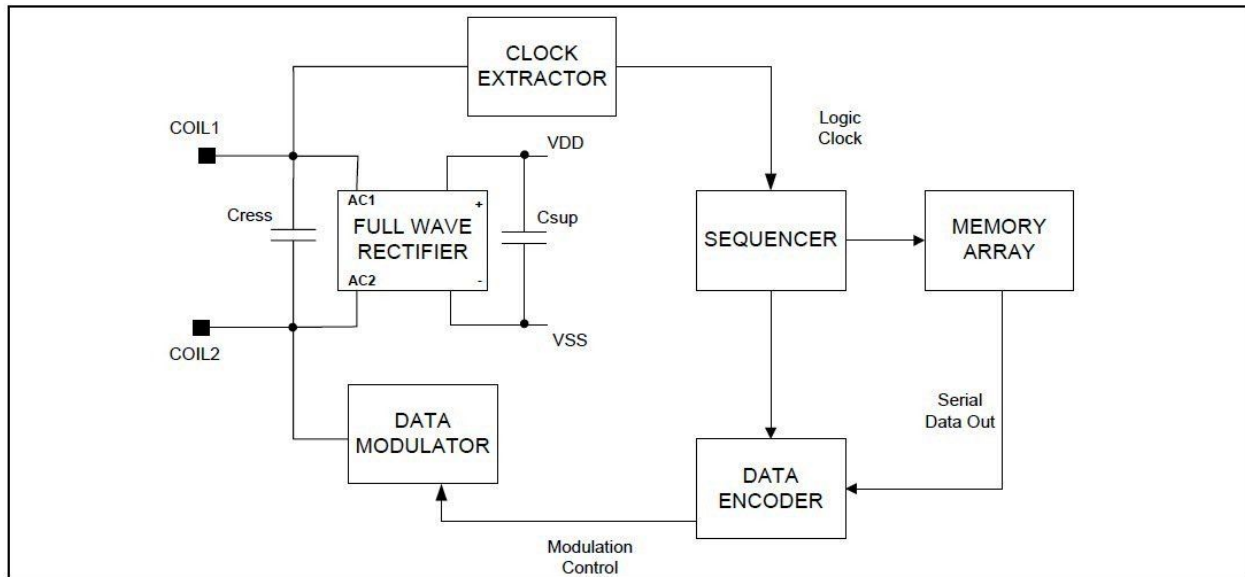


RFID reader & Tag:

EM-18 RFID reader module is the one we used and can read any 125KHz tags. It features low cost, low power consumption, small form factor and easy to use. It provides both UART and Wiegand26 output formats. It can be directly interfaced with microcontrollers using UART and with PC using an RS232 converter.

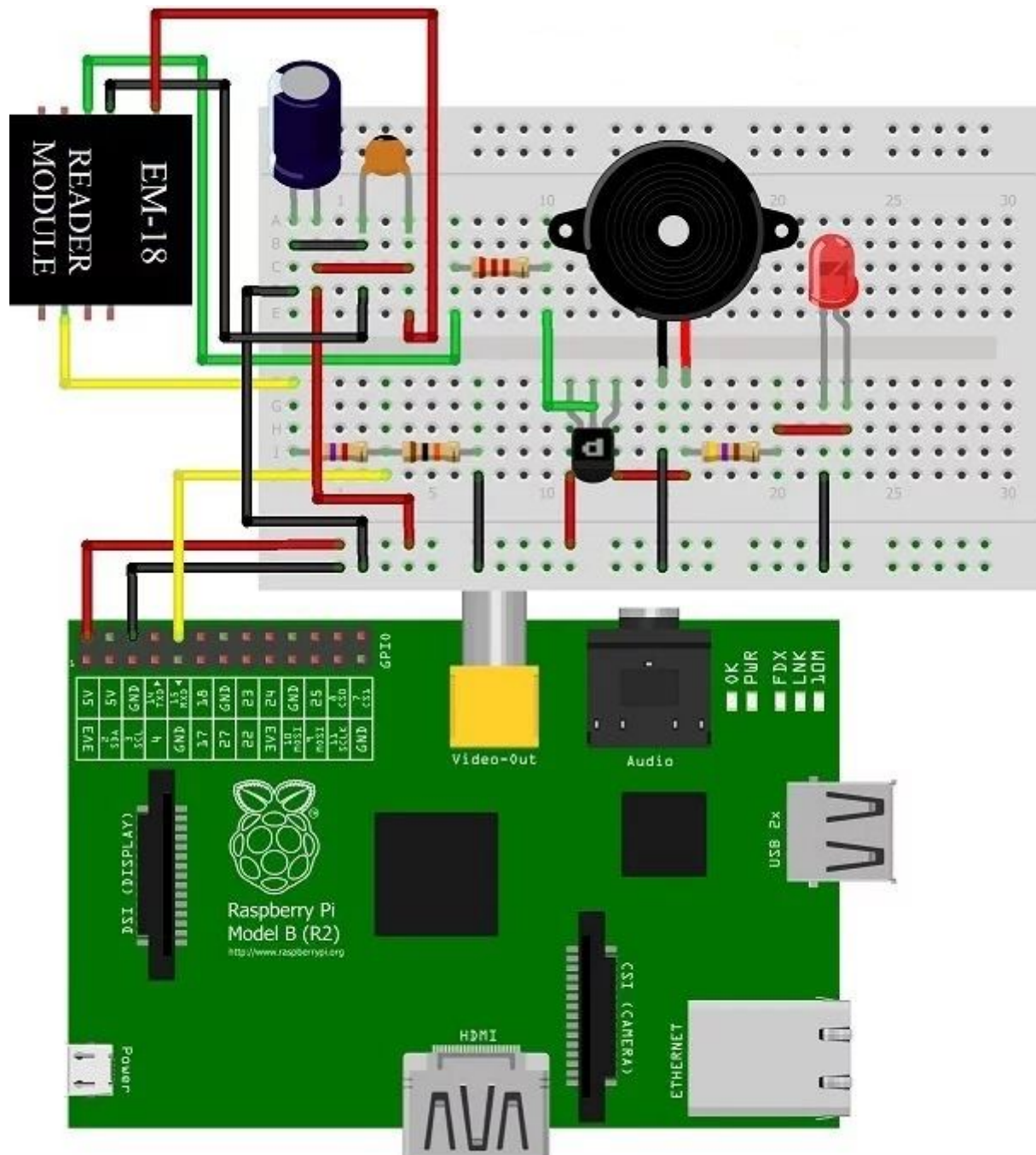


Rfid system principle:



Passive Rfid Tag

The UART TX output of EM-18 is of 5v. The input pin of Raspberry Pi GPIO is rated at 3.3v. So 5v cannot be directly given to the unprotected 3.3v input pin. Therefore we need a voltage divider circuit using appropriate resistors to bring down the voltage to 3.3V.



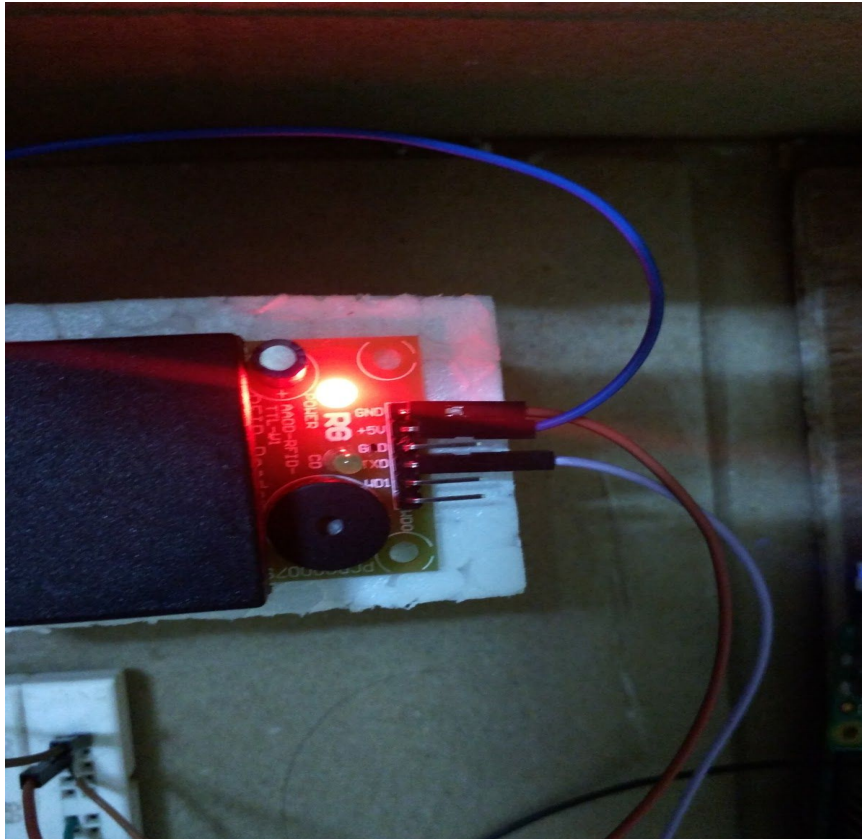
As the given rfid reader also consists the voltage divider we only have 3 pins to connect.

VCC - 5 volts.

ground - ground pin.

Tx - connect to the raspberry pi serial pin to transmit data.

The output consists of 12 character ASCII data, where first 10 bits will be the tag number and last 2 bits will be the XOR result of the tag number which can be used for error correction. For eg : If the RFID tag number is 500097892E, output of EM-18 Reader will be 500097892E60, where 60 is $50 \text{ xor } 00 \text{ xor } 97 \text{ xor } 89 \text{ xor } 2E$.



16x2 LCD display:

We have used 16×2 alphanumeric LCD module. These modules are cheap and easy to interface to the Raspberry Pi. They have 16 connections but we only need to use 6 GPIO pins on your Pi and rest are used to power up and adjust contrast.

The pinout of the module is :

1. Ground
2. VCC (Usually +5V)
3. Contrast adjustment (VO)

4.Register Select (RS).

RS=0: Command, RS=1: Data

5.Read/Write (R/W).

R/W=0: Write, R/W=1: Read

6.Enable

7.Bit 0 (Not required in 4-bit operation)

8.Bit 1 (Not required in 4-bit operation)

9.Bit 2 (Not required in 4-bit operation)

10.Bit 3 (Not required in 4-bit operation)

11.Bit 4

12.Bit 5

13.Bit 6

14.Bit 7

15.LED Backlight Anode (+)

16.LED Backlight Cathode (-)

Usually the device requires 8 data lines to provide data to Bits 0-7.

However the device can be set to a “4 bit” mode which allows us to send data in two chunks (or nibbles) of 4 bits. This reduced the number of GPIO connections we require when interfacing with your Pi.

Here is how we wired up LCD :

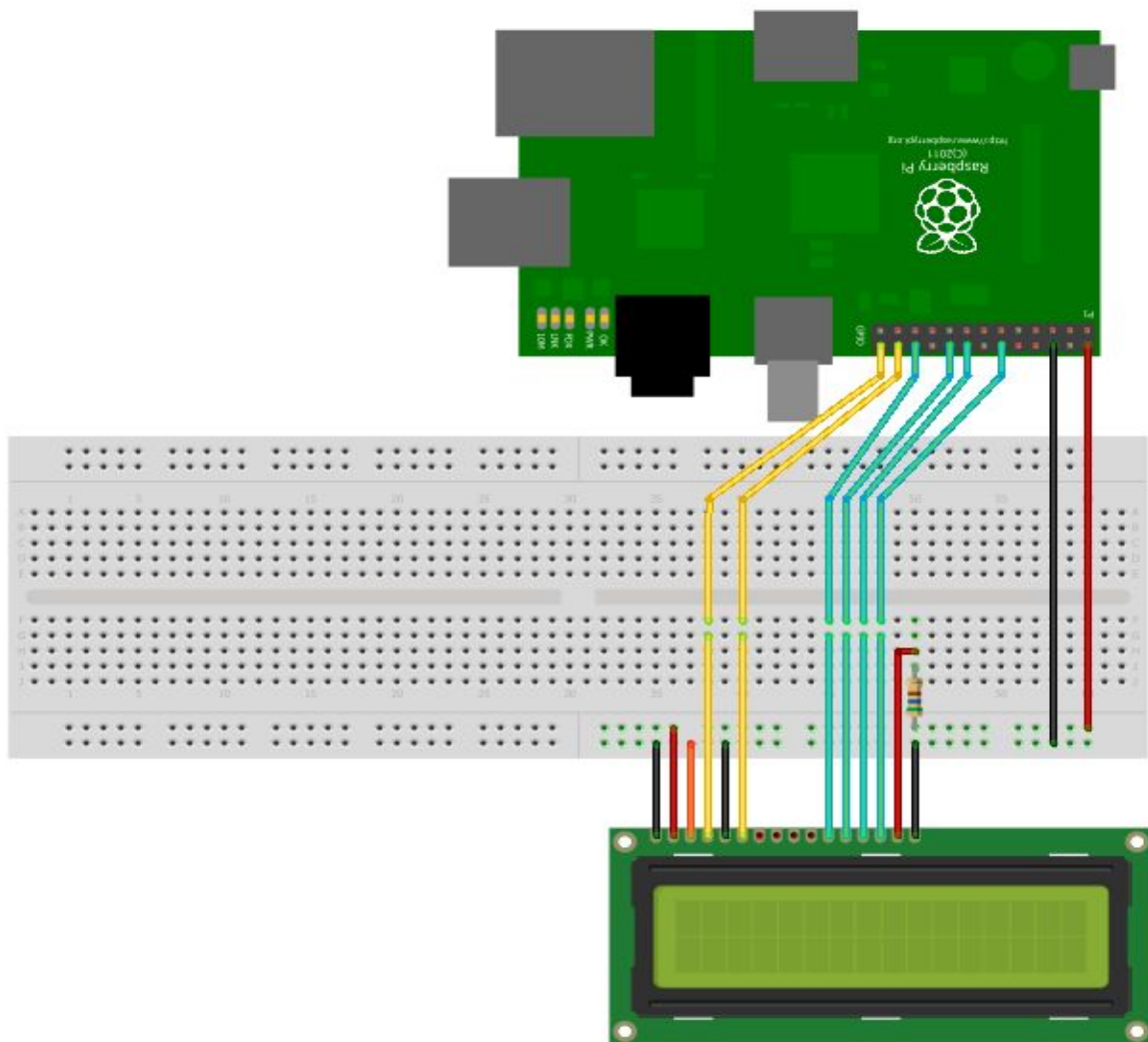
LCD Pin	Function	Pi Function	Pi Pin
01	GND	GND	P1-0 6
02	+5V	+5V	P1-0 2
03	Contrast	GND	P1-0 6

04	RS	GPIO7	P1-2 6
05	RW	GND	P1-0 6
06	E	GPIO8	P1-2 4
07	Data 0		
08	Data 1		
09	Data 2		
10	Data 3		
11	Data 4	GPIO25	P1-2 2
12	Data 5	GPIO24	P1-1 8
13	Data 6	GPIO23	P1-1 6
14	Data 7	GPIO18	P1-1 2
15	+5V via 560ohm		
16	GND		P1-0 6

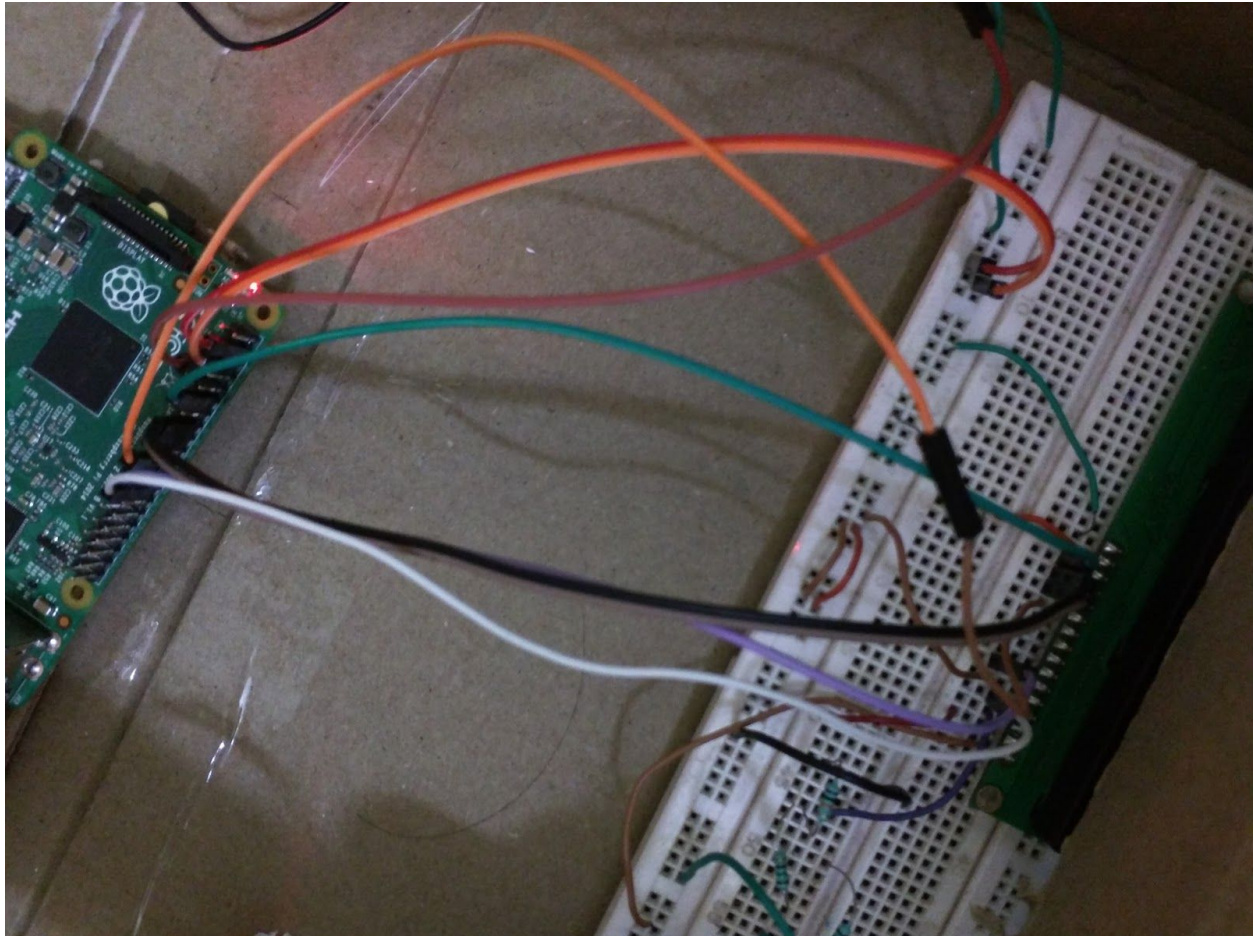
The RW pin allows the device to be put into read or write mode. We wanted to send data to the device but did not want it to send data to the Pi so We tied this pin to ground. The Pi can not tolerate 5V inputs on its GPIO

header. Tying RW to ground makes sure the device does not attempt to pull the data lines to 5V which would damage the Pi.

In order to control the contrast we can adjust the voltage presented to Pin 3. This must be between 0 and 5V. As we don't have a potentiometer, we used two resistors in series, connected one side to 5V and another to GND. The voltage between those two identical resistors would be 2.5V, which we connected to the contrast pin of LCD display.



Pin 15 provides 5V to the backlight LED. We thought connecting this pin directly to 5V will damage LCD. So we placed a 1k ohm resistor in series with the pin.

**Webcam:**

We have used logitech CX170 webcam which we connected to pi using USB. The following command was called to take an image from the webcam.

```
fswebcam --fps 15 -S 8 -r 320*240 "image name"+"jpg"
```

Wifi adapter:

connected to USB port of Raspberry Pi 2.

mqtt messages are sent to Raspberry Pi 2 through wifi.

Features:

- i)Can take weight,image of the item and assign an rfid to store in database.
- ii)Can find objects of similar kind using the encoding of rfid(i.e based on first 5 characters).
- iii)Add or remove an item from database based on rfid.
- iv)Count the number of items of particular kind.
- v)To indicate the minimum quantity of particular item and alert the user.

Working:

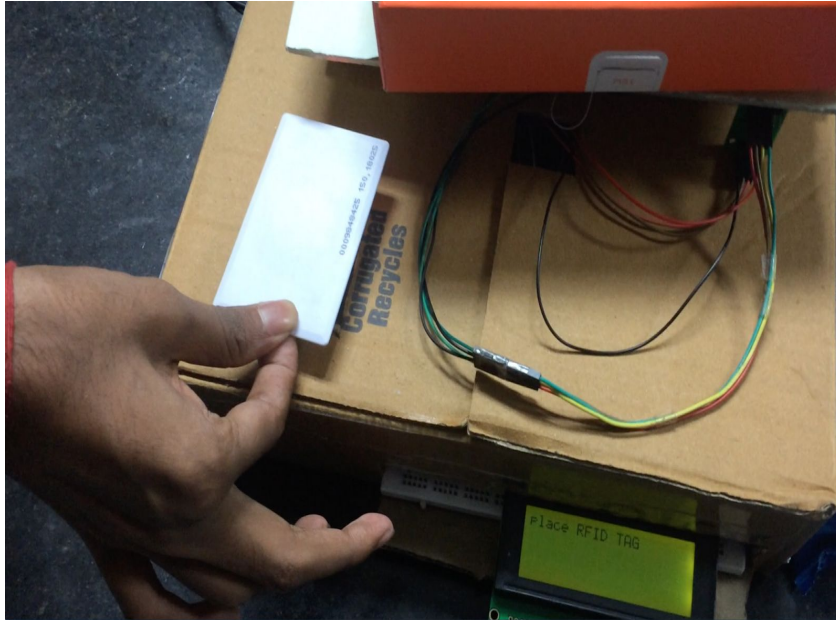
This smart weighing machine consists of three modes of operation.Mode1 to note the new object or to identify the existing object,Mode2 to count the number of items of particular kind given the tag of one known item,Mode3 to indicate minimum quantity of the particular item is not present.

Mode1:

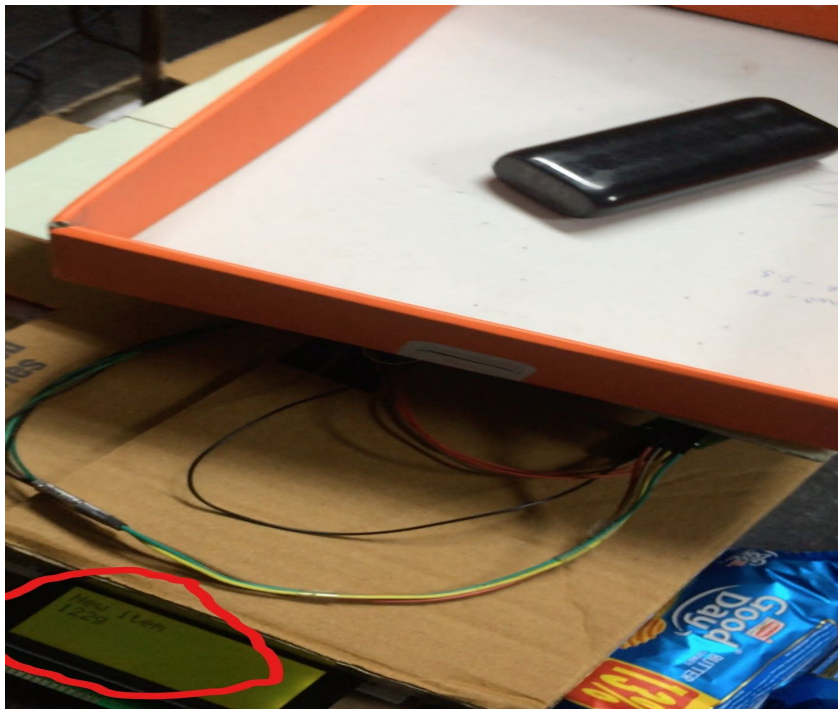
When Mode1 is selected the weighing machine prompts to place the tag on RFID reader and the item on the weighing pan to note the weight.Then machine searches the RFID in the database if the object is present then no weight is noted and machine shows that the object is found on the lcd screen.If same object was not found then it search for objects of similar kind present in the database(i.e based on the encoding scheme given to similar objects).If similar kind of object already exists then machine takes the weight of the object and add the previously existing image to this new object as the object of similar kind is already present,it is shown on the lcd screen.If the object of similar kind not exist in database then the machine allot the new tag to the given object and waits until the weight of the object gets stabilized(i.e until we get constant weight for 6 times) then the weight is noted,image is taken and stored in database then buzzer will be triggered to indicate that processing is done.

Images to be added

1.place the rfid on screen image



2.whenever the new item is added



Mode2:

In this mode it counts the number of items of similar kind. If we place n number of items on the weighing pan and we place the tag of one known item of its kind then the machine returns the number of items present on the weighing pan



Mode3:

This mode indicates the whether the item is present above particular quantity or not, this can used mainly in industries to restrict the lubricant to particular lower limit to run machines smoothly. When the lubricant amount is less than a particular value then system starts alarm to indicate that the required quantity is less that alarm stops when the lubricant is added more than the lower limit

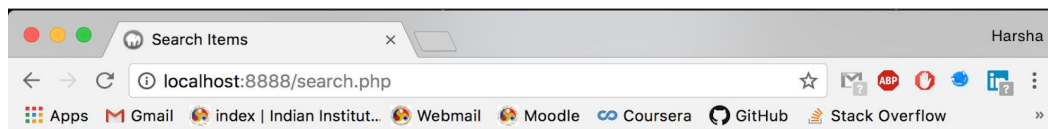
Web application and database:

We have used php as a server-side scripting language and MySQL database management system to make the web application.

The following are the pages created:

The main page has

- Search item
- Delete item
- Show all item
- Mode 3



Search the Item

Delete the Item

MODE 3

SHOW ALL ITEMS

The search item and show all items direct to page showing searched items and all items respectively.




Items are displayed in following way

192.168.43.108/3.php

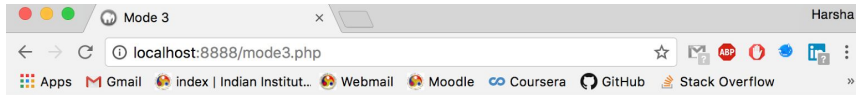
Not Secure | 192.168.43.108/3.php

Gmail Index | Indian Institut... Webmail Moodle Coursera GitHub Stack Overflow SPOJ.com - Problem... OPERATING SYSTEMS Topcoder TechCrunch

Connected successfully

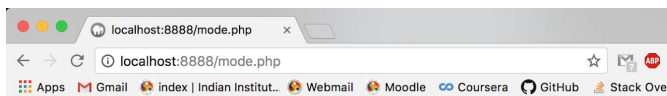
id	weight (gms)	image
0900964668	77	
0900964669	77	
0902964669	122	

The mode 3 redirects to following page :



Input the limiting weight

Upon submit in mode 3 page it redirects to following page:



Current Weight 100

Limiting Weight 0

SAFE

Difficulties Faced:

i) First we tried to use a 3 wire load cell. A three wired load cell alone can't be connected to HX711 as it has only 3 wires.

We needed four wires to use HX711 (wheatstone bridge). So, we used two 1K ohm resistors and 3 wire load cell to get four outputs to be connected to HX711. It didn't seem to be working. (our circuit with 2 resistors and load cell was correct. HX711 was damaged, which we found out later)

So, we have used a four wired load cell.

ii) The RFID reader didn't work as smooth as we expected. It sometimes unable to read or send the data and sometimes it thrown serial exceptions.

We finally make it work by setting timeout conditions, exception handling and resetting output buffer of rfid reader.

iii) In order to control the contrast we can adjust the voltage presented to Pin 3. This must be between 0 and 5V. This can be done using potentiometer. As we don't have a potentiometer, at first we connected it to GND. The contrast was very low. Then we used two resistors in series, connected one side to 5V and another to GND. The voltage between those two identical resistors would be 2.5V, which we connected to the contrast pin of LCD display.

iv) We first used raspbian os for raspberry pi and tried to connect to wifi using wifi adapter, we were failed despite of many efforts. We later found that raspbian can not connect to wifi. We then switched our OS to ubuntu mate.

Limitations:

1. It can't automatically detect the placed object. It can weigh the object as soon as it is placed but it will record only when the weight stabilizes.
2. Every time after the one mode completed the user needs to select one of the 3 modes available.
- 3.

Precautions:

1. The load cell and its wires are too sensitive. We have to take care while placing it and making connections. We need to double check if their is any weak connections are there or not.
2. LCD screen has almost 10 connections to load cell out of which are 6 GPIO connections making the connections complex. We have to check the proper connections and it is better to use breadboard instead of directly connecting to pi.

Further Scope:

- I. The weighing machine can be fully automated by adding a conveyor belt to it, so that the placing of object and removing of object can be done using the conveyor belt.
- II. Object detecting mechanism can be added so that it could directly start its process.