GNG 1103

# Design Project User and Product Manual

## Dust Detection System

Submitted by:
Dynamic Dusters C9
Luke Beausoleil 0300244213
Nicholas Martins 0300306097
Harrison Meeds 0300306567
Michael Mekalopolos 0300239862

April 11, 2023
University of Ottawa

# Table of Contents

# List of Figures

# List of Tables

# 1 Introduction

This User and Product Manual (UPM) provides the information necessary for the clients and other workers for a brewery to effectively use our dust detection system and for prototype documentation. This document will be going over everything to do with the prototype. It will give an overview of how it works, our initial thought process of starting the prototype, describing how it works, and as well any maintenance or support required for the prototype. All of this and as well more will be reviewed in this document. The privacy and security of the brewery's information that they have provided will be taken into consideration and we will not discuss anything so important that it will cause harm to the brewery and reveal anything important to the public.

# 2 Overview

A need exists for a safe, cost-effective dust detection system that can pre-emptively measure varying industrial quantities of both organic and sedimentary dust and record this information periodically. This process should be easily maintained without substantial risks or changes to brewery operations. This detection system was designed for a brewery where a great quantity of malt dust was often encountered which could create blockages through dust "slugs". These components posed a threat to the efficacy of the piping throughout the auger network and therefore affected productivity.

With respect to fundamental user needs, the following have been obtained:

- **Sophisticated monitoring of dust**
  Monitoring, as the primary function of our device, involves the systematic notification of dust buildup prior to there becoming a process-wide blockage or similar issue.

- **Easy maintenance and operation**
  Maintenance and operation refer to the cleaning of dust from the monitor and/or vent fan and the education and training an employee would need to efficiently operate the device.

- **Cost-effectiveness**
  Cost-effectiveness emphasizes keeping within the design budget needs.

- **Handleability of all encountered dust**
  Dust handleability indicates that the system must be able to detect dust of both forms—sedimentary and organic.

- **Maintained safety standards**
  In reference to safety, the current brewery safety policies and procedures will not be undermined.

- **Retained transportation systems**
  Dust transportation is the process of dust movement throughout the piping system, which will not be compromised.

These describe the essential user needs that will always be covered through this design. It is important to note; However, excess needs may only be partially covered by this model and modifications may be necessary.

There are two main differences that this design has in comparison to contemporary industrial dust monitoring systems: pre-emptive measurement and quantization. Firstly, with the measurement aspect, the dust is continually measured at intervals of approximately 30 seconds, rather than removing the entirety of dust in a receptacle and measuring concentrations occasionally. Therefore, the system can provide accurate, systematic readings periodically so regular investigation can occur. This feature was completely negated previously with other models. This provides a critical

advantage in safety and efficiency by providing employees with up-to-date information on the dust make-up. With respect to Quantization, each dust extraction (for concentration determination) is, in effect, “quantized” by a multiplicity factor of 19.63%. This means that by calibrating run-time and empirical analysis, the removed sample is massed and multiplied by (100/19.63) and then divided by the total mass rate to obtain a % mass/mass dust concentration. This is enormously advantageous because it saves money, time, and energy. Instead of removing an entire load, only a discrete, small amount is required to determine a dust concentration with high precision. There are several other, smaller differences in this design as well, such as the small volume of the actual machinery, and the simplicity of the software. Hence, this dust detection system is unique among classical models.

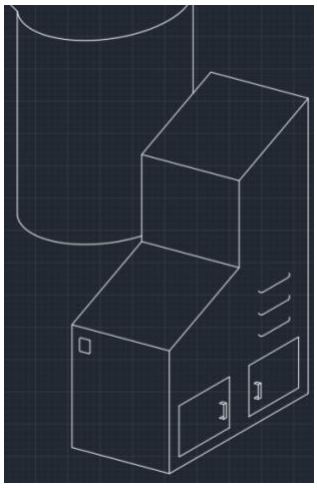

**Figure 1. Final Design**

**Figure 2. Final Prototype**

The main concept of this design is to remove a quantity of dust, mass it, and provide a concentration. This is largely accomplished via a vacuum of industrial grade that is able to physically suction 19.63 % of dust flowing through a 4-inch vertical pipe. This is an important dimension because the multiplicity factor changes with the physical conditions of operation (ie pipe dimension and placement). So, once vacuuming has concluded after 5 seconds, the dust falls below into a chute (see diagrams) where a tared load sensor masses the amount of extracted dust. This is then multiplied by the multiplicity factor of 19.63% by the software. This, finally, is then divided by the total mass rate to obtain a concentration.

The casing, which is the system's external covering, was constructed from wood for prototype purposes. In reality, the best determined material was ¼ inch galvanized steel as it is not overly expensive, is strong, and protects against corrosion. The wood casing (for the prototype) was built using 3/5-inch plywood as it is very readily available and inexpensive, while also having an acceptable level of durability. The software component is an Arduino IDE 2-part system. The first is the relay system which allows the vacuum to be powered while including a periodicity necessity. This periodic nature would not be possible without part 1. Part 2 of the software is an Arduino load sensor that transmits to a computer screen and converts its measurement (with the multiplicity factor) to a % mass/mass concentration. These two systems are housed adjacent to the main vacuum/chute area within the casing.

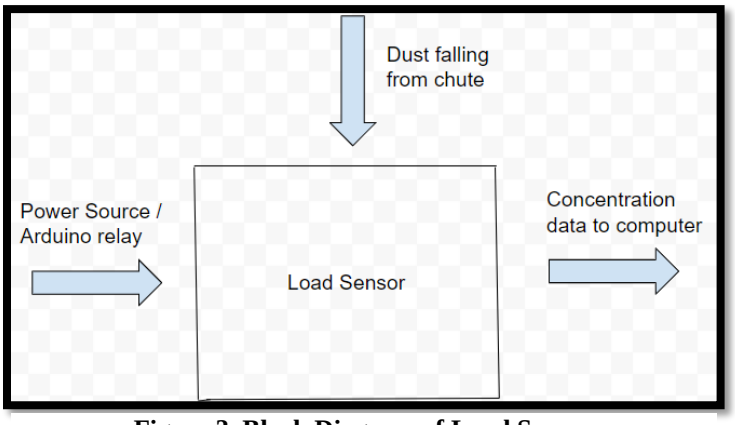


**Figure 3. Block Diagram of Load Sensor**

## 2.1 Conventions

All sections are sub headed and are there to emphasize importance. No other considerations are necessary.

## 2.2 Cautions & Warnings

The only caution with this devcice is to ensure there is adequate installation AND running space. The casing itself (see casing and installation) is relatively small, however, it is needed that there are no blockages to the system. Therefore, the area around it must be free from debris and other large objects in order for the arduino and vaccuum equipment to work properly.

# 3 Getting started

Before setting up the system, it is first necessary to install a hole in the silo transfer pipe the same size as the vacuum inlet, which is 7 cm in diameter. This hole must be centered 22” above ground level. Then, to begin setting up the system, install a fitted sieve 60 filter around the vacuum inlet, as per Figure 4. Then, mount the vacuum atop a 23 cm tall, insulated box, with a hole in the top to allow the vacuum outlet to pass through. The vacuum inlet should be in-line with the hole in the transfer pipe.

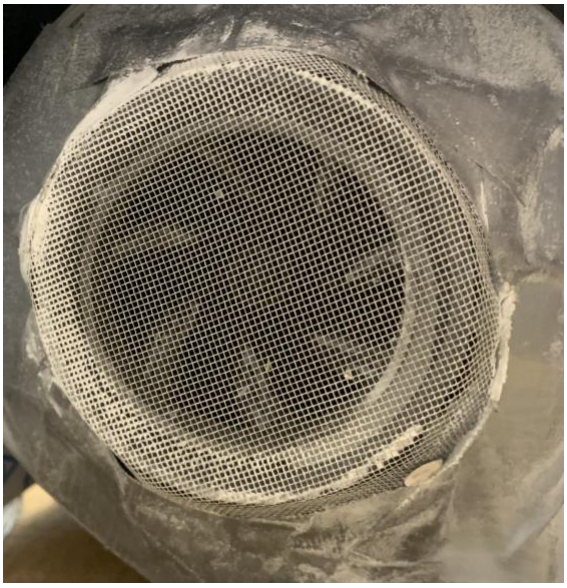

**Figure 4. Sieve Attached to Inlet of Vacuum**

Next, underneath on the bottom of that box, install the load cell and the load cell platform. Then, attach the dirt bag to the outlet of the vacuum. Cut a 1x1 inch hole in the back corner of the box and pass the vacuum cord and load cell wires through that hole.

Then, attach a second insulated box adjacent to the first one, on the same side that the hole is cut. Cut a 1x1 inch hole in the second box and pass the wires through. In this box, place a relay-configured electrical outlet and an Arduino Uno Mini, and wire them all together. The details of this wiring will be explained Section 6.2.3 of this report. Cut a final 1x1 inch hole in the back corner of the second box to allow the Arduino wire to exit.

All holes cut should resemble those found in **Error! Reference source not found.**.

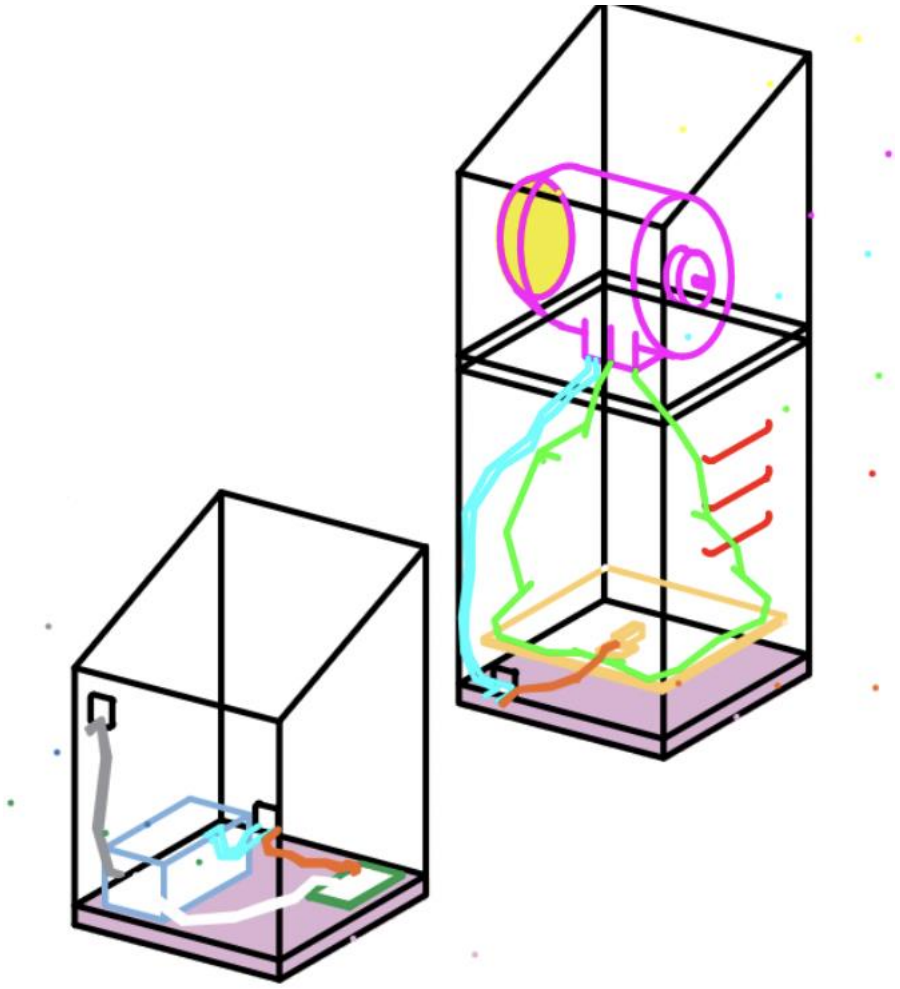

**Figure 5. Design Concept Diagram Showing the Insides**

Finally, seal all boxes with doors on each face to allow access to the insides and connect the Arduino wire to a computer. Then, attach roofing above both boxes (this includes being above the vacuum)

To operate the prototype, run the code found in Section 6.2.3 (or Appendix II) with Arduino IDE. This will turn the vacuum on and off cyclically and measure the composition of dust after each cycle. This code will run automatically and does not require any additional user input.

It should be mentioned that to operate this system with an HMI interface, additional setup beyond the scope of this report will be required. The HMI system will need to be able to read the output composition from the Arduino IDE and relay this information to the dust filtration system.

Finally, maintenance will be required in the form of changing/emptying the dust bag one time per day. It is recommended to turn the system off when changing the dust bag to prevent dust from being expelled into the box/environment.

## 3.1 Configuration Considerations

The system should be placed on the ground next to the vertical transfer pipe. The vacuum must be connected directly to the transfer pipe. Furthermore, the box containing the electrical components must be placed on any side of the vacuum box that is not the same as the side of the transfer pipe, otherwise there won't be enough space.

Orientation as per Figure 6 will allow the setup to fit. This also makes the setup more compact and take up less space.

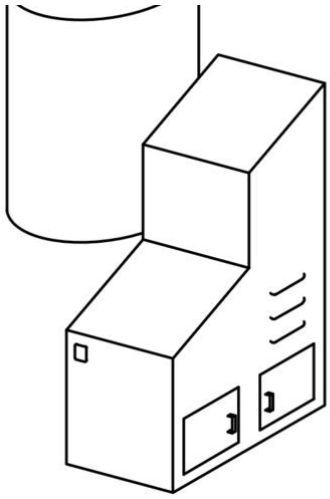

**Figure 6. Setup of System Attached to Vertical Transfer Pipe**

Securing the vacuum to the transfer pipe is not necessary, as it can sit inside the transfer pipe and will not move. However, if securing it is desired, it can be done using screws or a sealant.

## 3.2 User Access Considerations

Potential users with access to this system include Mill Street Brewery employees and managers, as well as government health inspectors.

First, employees should have access to the physical component of the system to perform its daily maintenance. However, they do not necessarily need access to the code or dust composition that the code outputs. This information should be kept confidential and should be at the discretion of the managers to disclose.

Second, managers should have access to the physical and software component of the system. This allows them to perform general checkups, and any needed maintenance on all components.

Finally, government health inspectors should have access to only the physical components of the system. These components include all casing and material that malt could possibly touch. This specifically includes the sieve filter, as malt is coming in direct contact with it.

## 3.3 Accessing/setting up the System

After finishing the construction of the final prototype, it is important to ensure security across the system. After construction, check all joints and attachments to ensure all important attachments such as the casing, load cell, dust bag, and pipe are attached firmly and sealed.

If the user plans on running different size malt or particles in general, it is very important to change the sieve to an appropriate size. Changing the sieve size should be done whenever the particle composition is noticeably changed to prevent inaccuracy and possible damage to the vacuum. Once the appropriate size of sieve has been acquired, properly size it to the 7 cm requirements. This sieve can then easily replace the existing sieve by detaching the attached sieve and reattaching the new size sieve the same way. It is highly recommended that if the user is planning on changing the size of sieve regularly that they attach their sieves using industrial tape or screws. While welding the sieve to the vacuum would be the most secure method of attachment, actively replacing the sieve with this method used would be difficult and wasteful.

As the vacuum system knowingly cannot collect all the dust that passes by, a multiplicity factor will be required to determine the full dust mass at any point. A multiplicity factor has already been pre-tested for Mill Street Brewery's needs, however in a situation where this prototype is to be used in a different environment, this factor is subject to change. This value can change slightly based on malt flow rate and composition, so it is most accurate to collect your own value for your situation. To do this, insert a measured amount of dust through the attached pipe with the system running, and use the measurement of dust remaining to calculate the percentage of dust collected by the system. Once this percentage is calculated insert the value into its respective line of code shown in figure 7. This percent value should be converted to a decimal value with 1.0 representing 100% dust collection. If such a test is unrealistic with the user's setup, it is recommended that the user use the multiplicity factor given for Mill Street Brewery of 19.63% as a base value, and to change in small increments based on the semblance of accuracy observed.

```
8   float multiplicity_factor = 0.1963; //percentage of dust that is sucked through the vacuum
9   float flow_rate = 2000; //mass flow rate of malt (in kg/hr)
10  float run_time = 2; //vacuum run-time in one cycle (in seconds)
11  float off_time = 6; //time between vacuum run-times (in seconds)
12  float composition; //dust composition in malt flow stream (mass/mass %)
```

**Figure 7. Multiplicity Factor in System Code**

The flow rate of malt passing by the system is an important value, as this can greatly vary the amount of mass passing by the system. In the code, it is important the flow rate is represented by a value that can be input into calculations. If the user has plans on using the prototype in different conditions than Malt Street Brewery and its 2000kg/hr malt flow, it is important to determine this value to input into the code as shown in figure 8 below. If the exact value cannot be found, an approximation is still useable, however it is important to note that the more inaccurate the approximation, the more inaccurate the system.

```
8   float multiplicity_factor = 0.1963; //percentage of dust that is sucked through the vacuum
9   float flow_rate = 2000; //mass flow rate of malt (in kg/hr)
10  float run_time = 2; //vacuum run-time in one cycle (in seconds)
11  float off_time = 6; //time between vacuum run-times (in seconds)
12  float composition; //dust composition in malt flow stream (mass/mass %)
```

**Figure 8. Flow Rate in System Code**

The vacuum system is designed to operate cyclically, turning itself off and on to conserve the energy that would be wasted by operating nonstop. The current design and recommendation is having the vacuum on for 5 seconds, and off for 15 seconds, giving three 20 second cycles every minute. However, if these times do not fit with any user specifications, they can easily be changed. To change the time of cycles, input the lengths of time desired for the vacuum to be on/off into the code as shown in figure 9 below.

```
8   float multiplicity_factor = 0.1963; //percentage of dust that is sucked through the vacuum
9   float flow_rate = 2000; //mass flow rate of malt (in kg/hr)
10  float run_time = 2; //vacuum run-time in one cycle (in seconds)
11  float off_time = 6; //time between vacuum run-times (in seconds)
12  float composition; //dust composition in malt flow stream (mass/mass %)
```

**Figure 9. System Run Time in System Code**

## 3.4 System Organization & Navigation

The system is composed of various systems working in unison. These components are a vacuum system, a dust bag, a load cell, a load cell platform, a relay system, casing and insulation, and Arduino code. These will each be discussed in detail in this section.

### 3.4.1 Vacuum System

The vacuum is the fan and motor component of a Hoover U4537-930 legacy vacuum. Its inlet is placed directly in the malt transfer pipe hole, which is designed tight enough to secure it. The sieve filter is attached industrial grade tape; there is not much stress applied to the sieve and thus using tape will hold. The vacuum outlet passes through the top platform of the box. The roofing of the casing is tight enough that it secures the vacuum in place. The wiring for the vacuum is connected inside the box near the vacuum outlet and is then passed through a hole. This wiring is also taped to the inner side of the box to prevent it from resting atop the load cell platform.

### 3.4.2 Dust Bag

The dust bag is a paper bag that is used in a Hoover U4537-930 Legacy vacuum. It is directly attached to the outlet of the vacuum using strong elastic bands. The bag is not suspended in the air, rather it is resting atop the load cell platform. Therefore, there is little tension applied to the elastic bands, thus they are very unlikely to come off.

### 3.4.3 Load Cell and Load Cell Platform

The load cell is a Bolsen Tech Digital Load Cell Weight Sensor 1 kg. This load cell does not have a large enough maximum load capacity for industrial use and would need to be replaced with a 35 kg maximum load cell. There is a piece of wood screwed in to bottom of the load cell on the right side, as per figure 10. This suspends the load cell partially in the air, allowing it to measure the torque of any weight that bends it. The, another piece of wood as well as an MDF load cell platform are screwed into the top of the load cell on the left side, as per figure 10. The load cell is also screwed into the bottom of the box, allowing it to be secured. Finally, the load cell wiring is attached to an Arduino to allow it to be controlled.

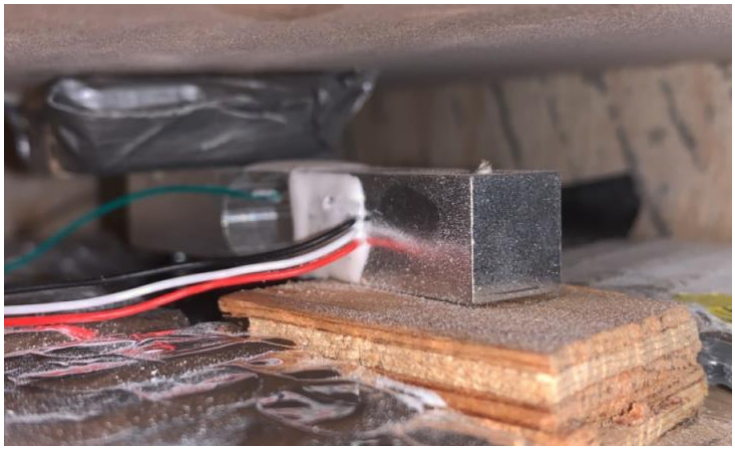

**Figure 10. Load Cell and Platform Attachment**

### 3.4.4 Relay System

The vacuum is powered from a regular wall outlet, but it is controlled with an Arduino and a relay. The relay is connected exactly as done in the Scott Campbell's guide. In brief, a power strip wire was stripped of its sheathing to reveal the hot, neutral, and ground wires. The neutral and ground wires were attached to an electrical outlet. Then, a piece of the hot wire was cut off. The non-cut off end of the hot wire was inserted into the C terminal of the relay. Then, one end of the cut off piece of the hot wire was inserted into the NO terminal of the relay, and the other end was attached to the electrical outlet. This the relay was then wired to an Arduino to allow it to be controlled.

The wire from the vacuum plugs into the electrical box. As well, the power strip wire is connected to a wall outlet. This allows the relay to open and close the circuit cyclically as needed.

All images needed to understand the connection can be found in Scott Campbell's guide, which can be found in the References of this report.

### 3.4.5 Casing and Insulation

The first box, which contains the load sensor is made of 1.5 cm thick wood that was sawed and screwed together. The second box, which contains the electrical components, is made of carboard, and is taped together. If this prototype is used for industrial use, the casing will need to be replaced by galvanized steel.

Each box is lined with a layer of pipe wrap insulation along the inner-bottom face. In an industrial application of this prototype, this pipe wrap will need to be replaced with a layer of 1/8" silicone rubber. The insulation is taped to the box using electrical tape.

### 3.4.6 Arduino Code

The code is a simple cyclical program that turns the vacuum on and off for set amounts of time. These times were set to 2 seconds on and 6 seconds off for the purpose of demonstrating the capabilities. However, these values should ideally be set to 5 seconds on and 25 seconds off for a real application. It should be mentioned that running the vacuum for more than 5 seconds is not useful, as when the vacuum is running, malt will obstruct the sieve and block dust from passing through. Therefore, any run time longer than 5 seconds will not accumulate much more dust than the amount accumulated in 5 seconds. Furthermore, the multiplicity factor of 19.63% was experimentally determined for a 5 second run time, and thus any run time longer will have a different factor.

The Arduino code also determines the dust composition by controlling the load cell. First, it tares the load cell to 0, then after the vacuum is done running one cycle, it measures the mass of dust accumulated. Then, it calculates the composition from equation 1.

$$composition = \frac{(mass) \Big/ (multiplicityfactor)}{(flowrate) * (runtime)} \quad (1)$$

This equation depends on the mass, which is measured by the load cell, and it also depends on the multiplicity factor, flow rate, and run time, all of which can be set by the user. This allows variable operating conditions to be used while still being able to use the dust detection system.

Then, after the dust composition is calculated, the cycle repeats itself. This gives a dust composition value every 30 seconds, which will allow the dust filtration system to operate effectively.

## 3.5 Exiting the System

To put the system away, it is necessary to remove the vacuum inlet from the transfer pipe. This can be done simply by pulling the vacuum away from it. This will, however, leave an opening in the pipe, which will need to be covered. Once the vacuum is removed from the transfer pipe, it is necessary to unplug the power strip cable from the wall outlet.

Then, the entire system can be removed/transported because everything is attached together and thus can move together. When transporting, it is important not to drop the system, as that could damage the load cell or the Arduino.

When not in use, the system should be stored in a dry location at temperatures close to 20ºC. This will ensure that damage to the system is virtually impossible.

# 4 Using the System

After completing set up, the system is designed to be very simple, requiring very limited user input to operate the system. Once your appropriate values have been input and malt is flowing past the system, open the Arduino code to start it up. Whether the code is being accessed directly at the system or connected from somewhere else such as the brewery, all that must be done is for your code to be verified and ran. As shown in figure 11, this is as simple as pressing two buttons at the top left of the Arduino application, first the blue checkmark, and then the blue arrow.

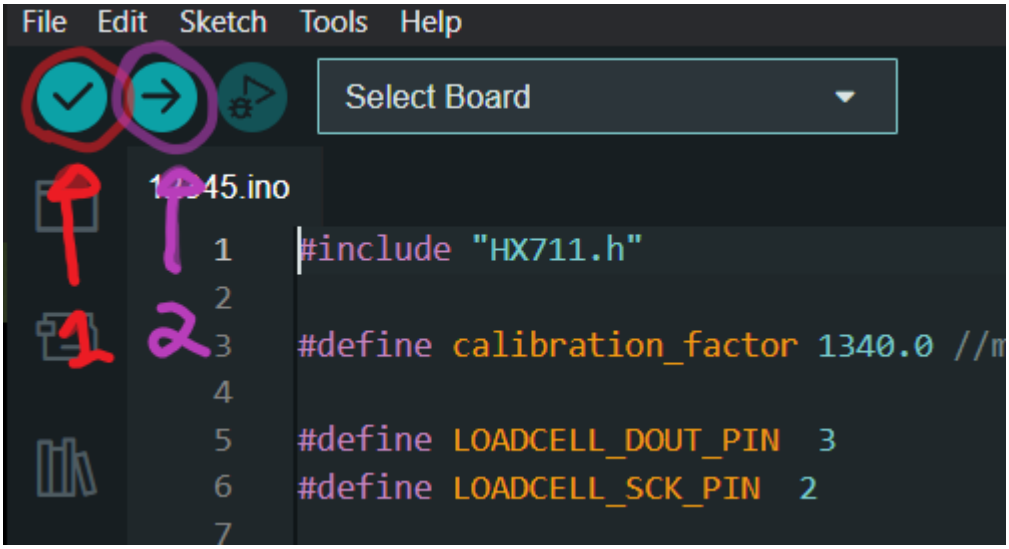


**Figure 11. Starting Up System Code**

This will upload the code to the Arduino, commencing the system. Once the code is in the Arduino, it begins the cycle of both powering off/on the vacuum and operating the load cell. Simultaneously, the information being gathered by the load cell is being run through the proper calculations to determine the dust composition. Each time a dust composition has been calculated the monitor will print out “The dust composition is X”, with X being the calculated dust composition. To turn off the system, the code can either be cancelled the same way, or power can be removed from the computer. Besides shutting it off, the system requires no other user inputs to run, operating self-sufficiently once the code is uploaded to the computer.

# 5 Troubleshooting & Support

This section will identify and clarify areas of the design that may be prone to malfunctioning and therefore could require adjustment. This product has been thoroughly tested and used, so the information provided is accurate and complete.

## 5.1 Error Messages or Behaviors

There are two main areas of this design that are particularly prone to malfunction or breakage. These are the load cell and the vacuum Arduino. In the case of the load cell, the wire that attaches to the load cell (see figure 11.), which relays the information to a computer, can become compromised if not handled well or is left unprotected during transportation. To ensure this wire is not removed unintentionally, make sure to insulate thoroughly and only disturb it when essential. With the vacuum Arduino, again the connector cable can become removed from the computer if not handled carefully, so make sure to hold it when transporting the prototype and once stationary, do not touch it. These are the only two major regions to "watch out" for with this design prototype.

## 5.2 Special Considerations

The only special consideration to have is when testing the device. The easiest way to test is to have the system upright and on a raised platform (i.e., a table). This will ensure all areas can be accessed and mended if necessary. It also allows for easier data collection because of the better visibility when in this position and the ability to record observations by writing or typing them on a hard surface.

## 5.3 Maintenance

The only regular needed maintenance is the emptying of the dust collection bag. Of course, this bag is a semi-permeable receptacle allowing air to pass through when running; However, changes may be required. These cases (for prototyping and testing purposes) include Creating a small tear in the bag, non-proper connection to chute, or the bag becomes full. In reality, for the full system, the bag needs to be changed once daily, or whenever full, and that should be the only reason for the system to require any maintenance at all – apart from routine inspections.

## 5.4 Support

Contact information of people who may offer support is found in table 1.

**Table 1. Support Staff Contact Information**

| Support Staff | Contact Information |
|---|---|
| Luke Beausoleil | lbeau021@uottawa.ca |
| Nick Martins | nmart120@uottawa.ca |
| Harrison Meeds | hmeed072@uottawa.ca |
| Michael Mekalopolos | mmeka067@uottawa.ca |

When running into difficulty with the device, consult this manual as a primary resource prior to external contacting. However, should the need arise for more detailed assistance, the individuals above designed and created the original prototype and can all offer help in the event of required guidance. The way to reach any of these people is through the emails listed above.

To report a system problem, simply email a brief description of the issue to one of the identified contacts. A response will be obtained by the questioner, as soon as possible with complete instructions on how to resolve the problem. A picture or diagram is also beneficial to include to allow for a better understanding on the part of the receiver.

Make sure to treat all technological system components with care – they are fragile. Should a question come up regarding the security of such items, follow the instructions above.

# 6 Product Documentation

The prototype design was separated into three subsystems: vacuum system, Arduino system, and the casing. Furthermore, any needed classifications and calculations made will be discussed in their relevant subsection of this report.

## 6.1 Vacuum System

The vacuum system refers to all components of the prototype that directly interact with the vacuum. This includes the vacuum itself, the sieve, and the dust bag.

### 6.1.1 BOM (Bill of Materials)

Table 2 is the bill of materials for all items used in the vacuum system design. Any entry in the price column marked with a hyphen indicates that there was no associated cost.

**Table 2. Vacuum System Bill of Materials**

| Part | Price | Retrieved From |
|---|---|---|
| Vacuum Fan | $10.00 | https://www.facebook.com/marketplace |
| Cord for Vacuum | - | *Came with vacuum fan* |
| Dust bag | - | *Came with vacuum fan* |
| 2x Elastic Band | - | House |
| Sieve Filter | $14.66 | https://www.walmart.ca/en |
| Duct Tape | - | House |

### 6.1.2 Equipment list

The needed equipment to construct the vacuum system is as follows:

- Scissors

### 6.1.3 Instructions

First, if necessary, disassemble the Hoover U4537-930 Legacy vacuum to retrieve the motor/fan unit from the insides. If the fan/motor is already extracted, ignore this step.

Next, secure the sieve around the vacuum inlet using duct tape. Then, wrap the dust bag around the vacuum outlet and secure it with elastic bands. Then, take the vacuum cord and plug the female end into the prongs on the vacuum cleaner. Then, take the male end of the cord and plug it into the electrical outlet that will be designed in section 6.2 of this report. Place the vacuum as needed in the hole of the top casing platform in which it fits.

The vacuum should sit upright in the top of the casing; however, if the vacuum is unstable, small pieces of woods can be placed on the underside of the non-balanced portion to provide additional support.

### 6.1.4 Testing & Validation

There were many tests done throughout this prototype, beginning with the testing of the vacuum. The purpose of this testing phase was to see how much dust can be pulled from the vacuum. We used in this test flour and popcorn to represent the malt and dust. This would be pulled into the vacuum passing through a porous plate that we made. After making the testing, we concluded an average amount of dust (flour) that was passed through was 9.22%. There were three trials made to determine this value. The results are listed below:

**Table 3. First Prototype Multiplicity Factor Calculation**

| Trial # | Mixture Before Vacuuming | | | Mixture After Vacuuming | Flour Removed | |
|---|---|---|---|---|---|---|
| | Popcorn (g) | Flour (g) | Total (g) | Total Mass (g) | Mass (g) | % of Flour |
| 1 | 156 | 70 | 226 | 221 | 5 | 7.14 |
| 2 | 153 | 70 | 223 | 217 | 6 | 8.57 |
| 3 | 152 | 67 | 219 | 211 | 8 | 11.94 |
| Average | | | | | | 9.22 |

## 6.2 Arduino System

The Arduino system refers to all components of the prototype that are connected to an Arduino. This includes the relay setup and the load cell.

### 6.2.1 BOM (Bill of Materials)

Table 4 is the bill of materials for all items used in the Arduino system design. Any entry in the price column marked with a hyphen indicates that there was no associated cost.

**Table 4. Arduino System Bill of Materials**

| Part | Price | Retrieved From |
|---|---|---|
| Arduino Uno Mini Clone | $17.00 | https://makerstore.ca/ |
| Arduino USB Wire | - | *Comes with Arduino* |
| Dual Relay Module | $3.50 | https://makerstore.ca/ |
| Electrical Box | $13.39 | https://www.homedepot.ca/ |
| Arduino Jumper Wires | $15.81 | https://www.amazon.ca/ |
| Power Strip | $27.11 | https://www.amazon.ca/ |
| HX711 Load Sensor 1 kg | $16.94 | https://www.amazon.ca/ |
| HX711 Amplifier Chip | - | *Comes with Load Sensor* |
| Header Pins | - | *Comes with Load Sensor* |
| Electrical Outlet | $2.98 | https://www.amazon.ca/ |
| Protoboard | - | Garbage |
| Arduino IDE | - | https://wiki-content.arduino.cc/en/software |
| HX711 Library | - | https://github.com/bogde/HX711 |

The power strip was chosen because it is 12-feet long and thus gives a lot of room to strip the wires back as needed, while still being long enough to connect to a wall outlet form a distance. Other power strips would be equally as functional and could be substituted if needed. Furthermore, the electrical box was chosen because it was convenient, the right size, and not made from a conducting material. This box could easily be replaced as needed by any other box that fits. Finally, the 1-kg HX711 load sensor was chosen because it is Arduino compatible, and there was already a repository online for how to code with it in Arduino IDE. This load sensor could be changed to any different one, provided it is also compatible with Arduino, and it is known how to write code for it.

### 6.2.2 Equipment list

The needed equipment to construct the Arduino system is as follows:

- Wire cutters
- Micro screwdrivers
- Soldering iron
- Digital multimeter

### 6.2.3 Instructions

The first step in setting up the Arduino is to set up the relay subsystem. Following, Scott Campbell's guide, this process should be very simple. First, using wire cutters, cut the end off the power strip and remove 4 inches of the sheathing to reveal the three inner wires. Using a digital multimeter, determine which inner wire corresponds to each of the three prongs at the end of the wire. This will determine which is the hot wire, neutral wire, and ground wire. Then, pass the cut end of the power strip wire through a hole on the electrical box. It is important to pass the wire through right now because this cannot be done later.

Once the three wires have been identified, strip back about 1" of the sheathing from the neutral and ground wires. Then, cut about 3" of the hot wire off. On the hot wire piece that was cut off, strip back about ½" and 1" of sheathing on either end. Insert the ½" end of the exposed copper into the NO terminal of the relay and screw the terminal closed using a micro screwdriver. At this point, the setup should resemble figure 12.

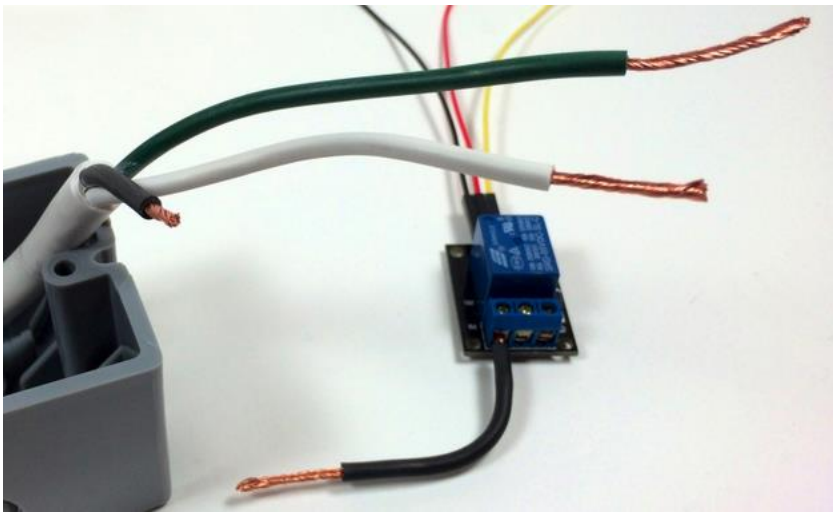

**Figure 12. Intermediate Stage of Relay Setup**

Then, take the remaining piece of hot wire still attached to the power strip wire and insert it into the C terminal of the relay and screw this terminal closed. Then, take the electrical outlet, and connect the unattached hot wire, neutral and ground wire to the gold, silver, and green screws on the electrical outlet, respectively. Tighten these screws to secure the wires in place. It is important to note that the neutral and hot screws need to be on the same side of the electrical outlet (either both on the top or both on the bottom).

Next, take three Arduino female-male jumper cables and attach the female ends to the VCC, IN, and GND pins on the relay. Then, into the Arduino, connect the male ends of those three wires to the 5V, 13, and GND digital pins, respectively.

Now the relay is fully connected and needs to be secured. Thus, place the relay at the bottom of the electrical box, then place the electrical outlet on top of it. After placing both into the box, ensure none of the wires came out of their terminals or pins. Then, screw the outlet into the box. The final product should resemble figure 13.

**Figure 13. Final Relay Setup**

The next step in setting up the Arduino system is to connect the load sensor to the Arduino. First, solder the header pins to the amplifier chip. Then solder four more header pins to one row on a protoboard. Next, take the load sensor and identify its four wires as per figure 14.

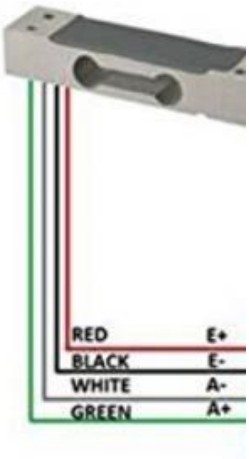


**Figure 14. Wiring Identification of Load Sensor**

Take each of the wires and solder them into a row on the protoboard adjacent to the header pins. Ensure the wires and the header pins are soldered together to secure a connection between them. Then, take four female-female Arduino jumper wires (preferably the same colour as the wires on the load sensor) and attach them from the protoboard pins to the amplifier chip, as per figure 15.

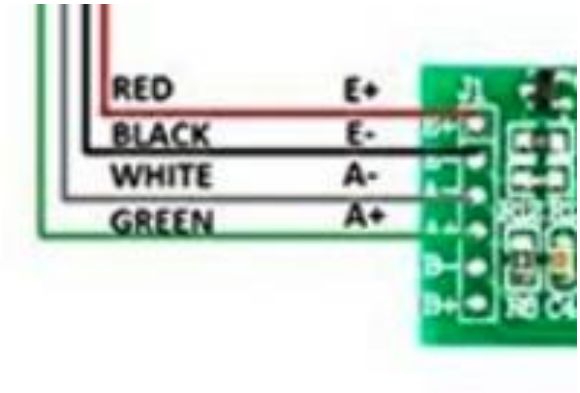


**Figure 15. Connection Between Protoboard and Amplifier Chip**

Then, take four female-male Arduino jumper cables and connect the female ends to the four header pins on the opposite side of the amplifier chip. These pins should be labelled as GND, DT, SCK, and VCC. Next, connect the male ends of those wires to the Arduino in the GND, 3, 2, and 5V digital pins, respectively.

Once this is done, the load sensor is fully wired to the Arduino and can be controlled with it. If desired, the wires can be taped together to decrease the clutter they cause, however, this step is not necessary as the wiring is functional with or without it.

Finally, the last step to complete the Arduino is to create the code that the Arduino runs on. First, on Arduino IDE, install Bogde's HX711 library form GitHub. Then, open a new sketch and paste in the code found in figure 16.

```
#include "HX711.h"

#define calibration_factor 1340.0 //unique to every load sensor

#define LOADCELL_DOUT_PIN  3
#define LOADCELL_SCK_PIN  2

float multiplicity_factor = 0.1963; //percentage of dust that is sucked through the vacuum
float flow_rate = 2000; //mass flow rate of malt (in kg/hr)
float run_time = 5; //vacuum run-time in one cycle (in seconds)
float off_time = 25; //time between vacuum run-times (in seconds)
float composition; //dust composition in malt flow stream (mass/mass %)

HX711 scale;

void setup() {
 pinMode(LED_BUILTIN, OUTPUT); //initialize the Arduino-relay connection
 Serial.begin(9600);
 scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
 scale.set_scale(calibration_factor);
 scale.tare(); //set scale to 0
}

void loop() {
 digitalWrite(LED_BUILTIN, LOW); //turns fan on
 delay(run_time * 1000); //fan stays on for run_time
 digitalWrite(LED_BUILTIN, HIGH); //turns fan off
 delay(off_time * 1000); //fan stays off for off_time
 composition = ((scale.get_units(), 2) / multiplicity_factor) / (flow_rate * (1/36) *
    run_time); //calculate dust composition
 Serial.print("Dust Composition is: ");
 Serial.print(composition * 100);
 Serial.println("%");
 scale.tare(); //set scale to 0
}
```

**Figure 16. Arduino IDE Code to Operate System**

Finally, plug the Arduino into a computer using the Arduino USB and upload the code. It is important to note that the code introduces variables for flow rate, run time, off time, and multiplicity factor. This allows these values to be set and easily changeable in the code as needed. Furthermore, the calibration factor is a value unique to each load sensor, and thus must be determined before

using the system. Therefore, follow DegrawSt's guide, and use their code, to calibrate the load sensor as needed.

Then, the Arduino system is fully functional and complete.

### 6.2.4 Testing & Validation

The testing done for the Arduino system is twofold: first, testing if the Arduino can properly control the vacuum; and second, testing if the Arduino can control load sensor and measure accurately.

First, to test if the Arduino can control the vacuum, the vacuum (or any other electrical appliance) is plugged into the electrical outlet. Then, the code is uploaded to the Arduino. If the vacuum (or other appliance) turns on and off cyclically for the duration stated in the code, it means the Arduino is properly controlling the vacuum (or other appliance). It is recommended to change the run time and off time to 2 seconds and 5 seconds, respectively, during this test, as that will greatly reduce the time it takes to validate this test.

Second, to test if the Arduino is getting accurate measurements from the load sensor, create a mixture of sunflower seeds and flour to emulate malt and dust (or get an actual mixture of malt and dust). Then, upload the code to the Arduino while passing the mixture by the vacuum inlet to emulate malt flow in a transfer pipe. After the vacuum runs one cycle, read the dust composition that it measured. Then, compare the measured dust composition with the actual dust composition and check the error using equation 2.

$$\%\ error = \left|\frac{(real\ composition) - (measured\ composition)}{(real\ composition)}\right| * 100\% \qquad (2)$$

This test was performed on three different compositions, and each of them yielded less than 5% error, as per table 5. Therefore, the system is considered to have passed the test, as 95% accuracy is an excellent accuracy, which will function properly in industrial use.

**Table 5. Load Sensor Test Results**

| Trial | **Real Flour %** | **Measured Flour %** | **% Error** |
|---|---|---|---|
| **1** | 7.69 | 7.32 | 4.81 |
| **2** | 6.15 | 6.38 | 3.74 |
| **3** | 3.85 | 3.70 | 3.90 |
| **Average** | | | 4.15 |

## 6.3 Casing

The Arduino system refers to all components of the prototype that are involved in the building casing of the boxes. This includes the outer casing, the insulation, and the platform of the load cell.

### 6.3.1 BOM (Bill of Materials)

Table 6 is the bill of materials for all items used in the Arduino system design. Any entry in the price column marked with a hyphen indicates that there was no associated cost.

**Table 6. Casing Bill of Materials**

| Part | Price | Retrieved From |
|---|---|---|
| 1.5 cm wood | - | House |
| Cardboard | - | *Comes with Arduino* |
| Flex tape | $22.59 | https://www.canadiantire.ca/en.html |
| Screws | - | House |
| 1/8” MDF Sheet | $3.00 | https://makerstore.ca/ |

### 6.3.2 Equipment list

The needed equipment to construct the casing is as follows:

- Screwdriver
- Electric drill
- Handsaw

### 6.3.3 Instructions

Building the case for this prototype is not complicated and the steps taken to do will be shown. The first step of the building process was to gather all the material, the materials needed are listed above. The dimensions were needed to determine how large the casing needs to be. For the wood portion which holds the load sensor and the bag which the dust goes into, the dimensions of those were 7.5 x 8”.

After figuring out the dimensions needed, it was now time to begin putting the casing together. For the wood, we would use a handsaw to cut the wood to the dimensions needed, there were 5 pieces of wood cut to make a box, the front being open so people could see what was happening in the inside. For the final prototype, the front would be closed in the cardboard and wood portion to fully protect everything, but in this circumstance, we left the front open. To connect all the pieces together, an electric drill was needed to drill everything together. This was used to make sure everything is stable. Also inside of the wooden box, there was a wooden slate inserted to

put the load sensor from the bag above. There were two little wooden slabs below the load sensor as well so that the load sensor will be able to detect the dust going into the bag. For the cardboard box, we just inserted the Arduino, and the outlet to turn on the vacuum.

After this was completed, we would cut cardboard pieces where it could make a roof for both the wood and cardboard box. These roofs were on angles on purpose, so it can withstand weather conditions. It would use galvanized steel for the industrial prototype, but for time and cost's sake, cardboard would do. The roof was put together by using tape.

The final step for the casing was to make a hole between the cardboard box and the wooden box so that the load sensor was able to connect to the Arduino. The purpose of this was so we were able to calculate the dust composition.

### 6.3.4 Testing & Validation

After completing this phase of the testing, we moved on to finding what type of casing and insulation we wanted for our prototype. To determine these materials, we would do a series of tests to find these materials. For the casing portion of the prototype, we wanted to see galvanized steel, PVC, and aluminum which would fit best for the final product. We determined that galvanized steel would be the best material for the casing. We found this result by comparing the yield strengths, if it can withstand certain weather conditions, and the cost of it. The table of all these values are found below:

**Table 7. Strength Tests and Costs of Possible Casing Materials**

| Material | Yield Strength (MPa) | Survives Snow ($4.17 \times 10^{-4}$ MPa) | Survives Wind ($7.26 \times 10^{-4}$ MPa) | Cost ($/m$^2$) |
|---|---|---|---|---|
| **Galvanized Steel** | 520 | Yes | Yes | 28.58 |
| **Aluminum** | 270 | Yes | Yes | 41.23 |
| **Polyvinyl chloride (PVC)** | 55.2 | Yes | Yes | 53.77 |

For the insulation part of the testing, we wanted to see which insulation would fit best for our final design. The three insulators we were deciding on were EDPM rubber, silicone rubber, and

QEP Cork Plus. In the end of this, we determined that silicone rubber would be the best fit. We determined this by looking at two factors: the materials specifications and looking at the load cell temperatures by using the different kinds of insulators. The graphs represent load cell temperatures vs the ambient temperature of a certain type of insulator. The table below just shows some information regarding the insulator, including cost, thermal conductivity, and thickness. All these diagrams and tables are below:

**Table 8. Thermal Conductivity, Thickness, and Cost of Possible Insulators**

| Material | Thermal Conductivity Coefficient (W/m/K) | Thickness (mm) | Cost |
|---|---|---|---|
| **EDPM Rubber** | 0.465 | 6.35 | $5.44/sq.ft |
| **Silicone Rubber** | 0.165 | 3.175 | $10.27/sq.ft |
| **QEP Cork Plus** | 0.38 | 6 | $65.00 |

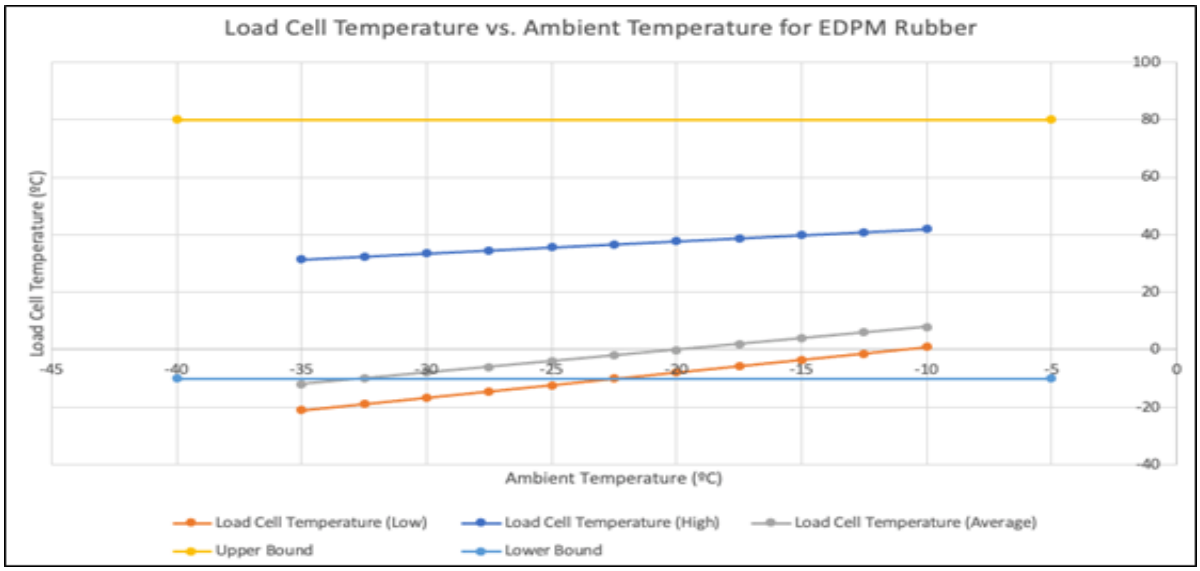


**Figure 17. Load Cell Temperature vs. Ambient Air Temperature for EDPM Rubber**

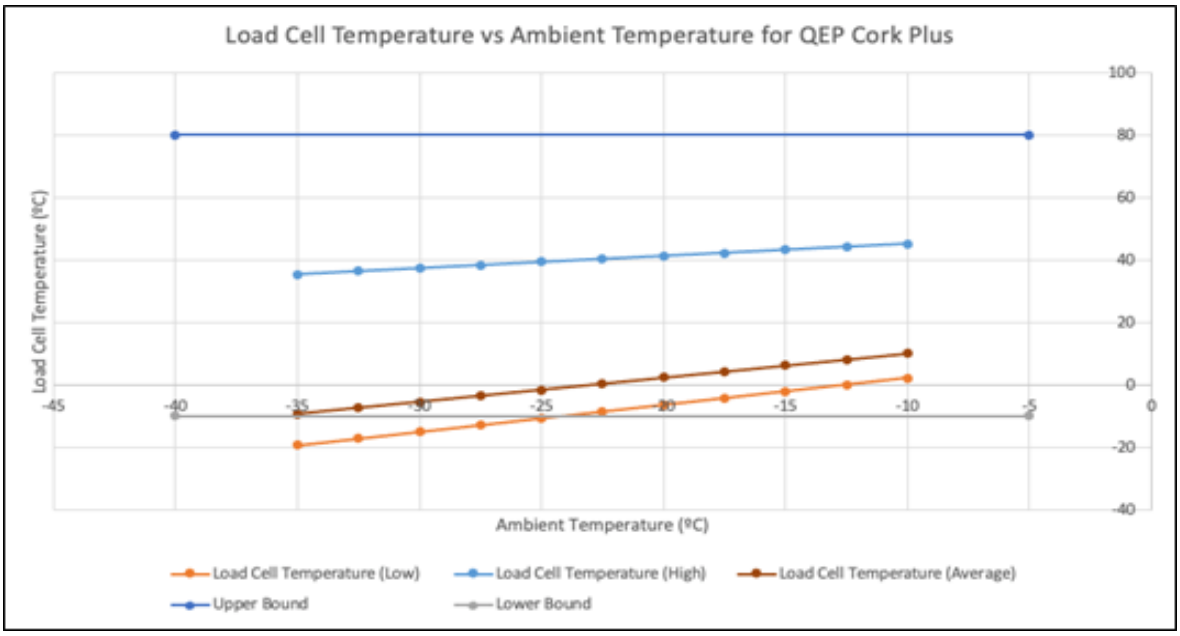


**Figure 18. Load Cell Temperature vs. Ambient Air Temperature for QEP Cork Plus**

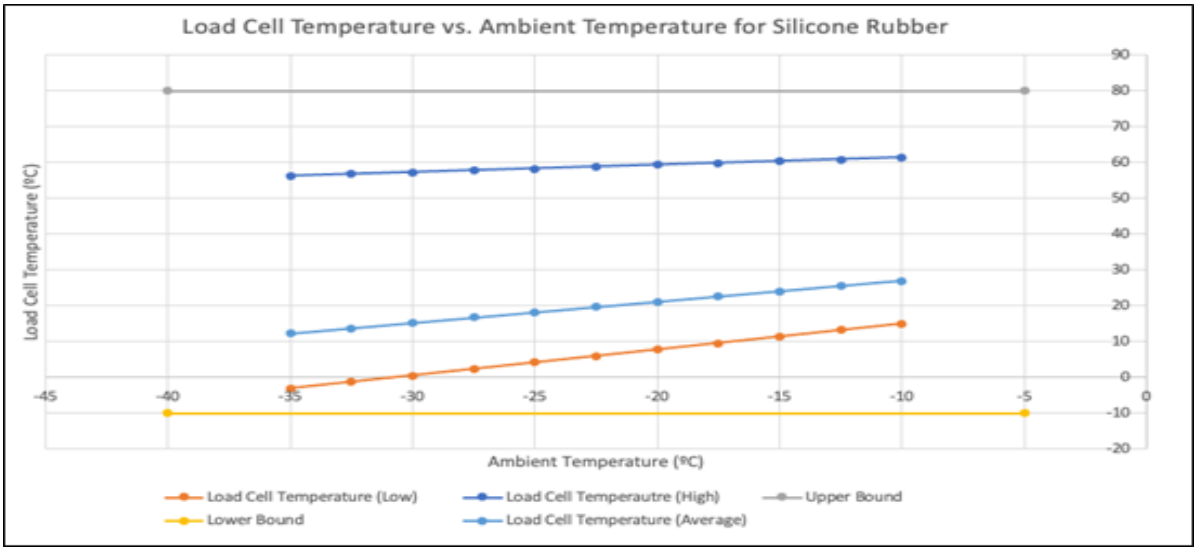


**Figure 19. Load Cell Temperature vs. Ambient Air Temperature for Silicone Rubber**

# 7 Conclusions and Recommendations for Future Work

The most important lesson to take away from this project is that communication is key. With having proper communication, you will have a team that works well together which will ultimately lead to success. The use of good communication allows team members to know what everyone is doing and if a person in the group needs assistance or won't be able to finish something on time, those members of the group will be able to help and try to solve the issue. If we had a few more months to work on this project, one of the components we would have changed the casing of the prototype. Due to time constraints, we had to have the casing of our final prototype with cardboard and some wood. Ideally, we would have used wood to represent the galvanized steel that would have been used in the final prototype. It would have made the final design look better and more professional. Also, the design of the bag being attached would have changed. Since just the bag is attached via elastic, so we would have found a much better way to attach the bag. Finally, we would implant a system in the code where to alarms people that the system is not working properly to let them know if it needs any maintenance.

# 8 Bibliography

*Bolsen Tech digital load cell weight sensor 1KG portable electronic kitchen scale + HX711 weighing sensors ad module for Arduino*. Amazon.ca: Industrial & Scientific. (n.d.). Retrieved February 20, 2023, from https://www.amazon.ca/Bolsen-Tech-Portable-Electronic-Weighing/dp/B071YN832G/ref=pd_ci_mcx_mh_mcx_views_0?pd_rd_w=9PYCP&content-id=amzn1.sym.51ce09d4-a5e1-4c26-916a-dd527820dcd6&pf_rd_p=51ce09d4-a5e1-4c26-916a-dd527820dcd6&pf_rd_r=ZRJ5689A3ZP86J5AESCW&pd_rd_wg=L33Mm&pd_rd_r=8275b459-6bf5-41a9-a0ad-4a31970b5ce8&pd_rd_i=B071YN832G

Campbell, S. (2021, November 14). *Turn any appliance into a smart device with an Arduino controlled power outlet*. Circuit Basics. Retrieved March 13, 2023, from https://www.circuitbasics.com/build-an-arduino-controlled-power-outlet/

DegrawSt, & Instructables. (2020, October 22). *Arduino scale with 5kg load cell and HX711 amplifier*. Instructables. Retrieved March 13, 2023, from https://www.instructables.com/Arduino-Scale-With-5kg-Load-Cell-and-HX711-Amplifi/

# APPENDICES

# 9 APPENDIX I: Design Files

This document is tentatively the final work for the Dynamic Dusters's dust detection project. This user manual finalizes all work completed throughout the project deliverables, as found in table 9. Each of these deliverables can be found through the MakerRepo link to the project, which is:

https://makerepo.com/lbeau021/1652.gng-1103-group-c9-dynamic-dusters

If any more information about the project, that for some reason cannot be found through MakerRepo, is desired, please contact a member of the support staff as listed in table 1.

**Table 9. Referenced Documents**

| Document Name | Document Location and/or URL | Issuance Date |
|---|---|---|
| Deliverable A | https://makerepo.com/lbeau021/1652.gng-1103-group-c9-dynamic-dusters | 26/01/2023 |
| Deliverable B | https://makerepo.com/lbeau021/1652.gng-1103-group-c9-dynamic-dusters | 29/01/2023 |
| Deliverable C | https://makerepo.com/lbeau021/1652.gng-1103-group-c9-dynamic-dusters | 05/02/2023 |
| Deliverable D | https://makerepo.com/lbeau021/1652.gng-1103-group-c9-dynamic-dusters | 12/02/2023 |
| Deliverable E | https://makerepo.com/lbeau021/1652.gng-1103-group-c9-dynamic-dusters | 19/02/2023 |
| Deliverable F | https://makerepo.com/lbeau021/1652.gng-1103-group-c9-dynamic-dusters | 05/03/2023 |
| Deliverable G | https://makerepo.com/lbeau021/1652.gng-1103-group-c9-dynamic-dusters | 12/03/2023 |
| Deliverable H | https://makerepo.com/lbeau021/1652.gng-1103-group-c9-dynamic-dusters | 26/03/2023 |
| Deliverable I (part 1) | https://makerepo.com/lbeau021/1652.gng-1103-group-c9-dynamic-dusters | 29/03/2023 |
| Deliverable I (part 2) | https://makerepo.com/lbeau021/1652.gng-1103-group-c9-dynamic-dusters | 29/03/2023 |
| Deliverable J | https://makerepo.com/lbeau021/1652.gng-1103-group-c9-dynamic-dusters | 13/03/2023 |
| All Design Concepts | https://makerepo.com/lbeau021/1652.gng-1103-group-c9-dynamic-dusters | 05/04/2023 |
| Code | https://makerepo.com/lbeau021/1652.gng-1103-group-c9-dynamic-dusters | 26/03/2023 |

# 10 APPENDIX II: Arduino IDE Code

```
#include "HX711.h"

#define calibration_factor 1340.0 //makes the load sensor measure in grams

#define LOADCELL_DOUT_PIN  3
#define LOADCELL_SCK_PIN  2

float multiplicity_factor = 0.1963; //percentage of dust that is sucked through the vacuum
float flow_rate = 2000; //mass flow rate of malt (in kg/hr)
float run_time = 2; //vacuum run-time in one cycle (in seconds)
float off_time = 6; //time between vacuum run-times (in seconds)
float composition; //dust composition in malt flow stream (mass/mass %)

HX711 scale;

void setup() {
  pinMode(LED_BUILTIN, OUTPUT); //initialize the Arduino-relay connection
  Serial.begin(9600);
  scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
  scale.set_scale(calibration_factor);
  scale.tare(); //set scale to 0
}

void loop() {
  digitalWrite(LED_BUILTIN, LOW); //turns fan on
  delay(run_time * 1000); //fan stays on for run_time
  digitalWrite(LED_BUILTIN, HIGH); //turns fan off
  delay(off_time * 1000); //fan stays off for off_time
  composition = ((scale.get_units(), 2) / multiplicity_factor) / (flow_rate * (1/36) * run_time); //calculate dust composition
  Serial.print("Dust Composition is: ");
  Serial.print(composition * 100);
```

```
  Serial.println("%");
  scale.tare(); //set scale to 0
}
```