# ECTE350

**Project:** FitSmart Juicer

**Deliverable:** Final Project Report

**Submission Date:** 19th May 2019

**Project Manager:** Umm Kulsoom Emad

**Sponsor:** University of Wollongong in Dubai

**Team Name:** The Firm Logistics

**Team Details:**

| | Name | Student ID |
|---|---|---|
| 1. | Mohammad Owais Imtiaz | 5653770 |
| 2. | Tanmay Khachane | 5570980 |
| 3. | Ahmed Abdou | 5292074 |
| 4. | Talha Khan | 5244511 |
| 5. | Umm Kulsoom Emad | 5529657 |

# Contents

# 1. Executive Brief

This report is about a product that calculates calories and nutritional facts which include carbohydrates, sugars, protein and vitamins, which can be used by the people who are looking after their diets and working on their macros. Moreover, this report covers the main topics about marketing and competition, design, testing and its results, and its finances put together to show whether the project is cost efficient and is beneficial.

The customer may be interested in such a product because of the unique design and computerized functionality with an application which is user-friendly (easy-to-use) and most importantly is informative. FitSmart Juicer is a product which recommends different juices and calculates their nutritional facts. In addition, it monitors the stock by notifying the user when the stock of a fruit runs low. All these features of this product can be used with the aid of our easy to use app. The functionality is achieved by using load cells which determine the weight of the fruit used to prepare the juice, and calculate its carbohydrates, calories, etc. These details and the remaining stock on are displayed on the app.

As technology is advancing in almost every area of this era and a lot of people have become conscious about their health, this product is a perfect blend of both. There are lots of competitors out there but FitSmart juicer is unique and would be a great addition to a kitchen or even a gym, as the product on its own is very feasible and user-friendly. It helps in maintaining a healthy lifestyle and keeping track of the stock, with great convenience as everything is available in the users' mobile phone. A user manual is also provided with this product.

# 2. Overview

## a. Project Background

The product would be a part of the trendy technology technologies because of its features, which includes automatic calorie count and restocking. The system calculates the nutritionals details like amount of sugar, carbohydrates, minerals, vitamins, and gives the total calories of the juice made, promoting healthy lifestyle amongst the people. It also has valve which can be used to dispense either water to change the thickness of the juice, or with milk to make a smoothie. Moreover, when the stock reaches low, a message is sent to the user notifying him that reordering of a specific fruit is required, minimizing the waste of resources. The nutrition facts of the juice and the available stock is displayed on the screen. The specifications of the product are mainly achieved by the use of weight sensors and a microcontroller. This product helps juice companies to manage their resources more efficiently as well as increase the appeal towards diet conscious consumers thus boosting sales on an industrial scale. It can be also used at home to enjoy a healthy lifestyle and keep track of your stock.

## b. Aim of the project

The aim is to create a device which not only calculates the nutrition values of the juice made in order to promote healthy lifestyle, but also automatically restocks the fruits by notifying the user so that ordering fruits in bulk is avoided in order to save resources and not exceed the budget unnecessarily.

## c. Project Scope

The system would be able to calculate the size of portions, calories and intricate nutrition values, separated and identified properly of the fruit used for making the juice. In addition, it helps the consumer to have the required fruit automatically restocked by notifying the customer with the use of IoT (Internet of Things). This product can manage resources more efficiently and have much better managed resources which in turn contributes to boost sales on an industrial scale and contribute to promote a healthy lifestyle on a personal usage.

Adding ice cubes instead of water and making the blender itself would be out of scope, only the system that displays calories and automatically restocks the fruits is part of the project.

## d. Objectives or work packages

1. Hardware: Designing the prototype, building the prototype and also the testing of each component and the complete the hardware section of the prototype.
2. Software: Finding a compatible software for the microcontroller and making the code according to the functionality of the prototype.
3. Combining Software and Hardware: Combining the software and the hardware and testing them together.
4. IOT (Internet of Things): Integrating the microcontroller, software and hardware by using Bluetooth to create an app for the user to be used on the mobile phone for ease.
5. Report and Presentations: The theoretical details of the project.

## e. Deliverables

*Table 1: Work packages, deliverables and due dates*

| Work Package | Deliverable | Due Date |
|---|---|---|
| **1. Hardware** | | |
| **1.1. Design** | 1.1 Multisim File | 20th Nov 2018 |
| **1.2. Parts** | 1.2 List of Parts to be ordered | 23rd Nov 2018 |
| **1.3. Building the circuits** | 1.3 Breadboard Connections | 9th April 2018 |
| **1.4. Testing** | 1.4 Test Results | 11th April 2019 |
| **2. Software** | | |
| **2.1. Choose software** | 2.1 Syncing with microcontroller | 20th Dec 2018 |
| **2.2. Designing the code** | 2.2 Flowchart | 21st Dec 2018 |
| **2.3. Writing the codes** | 2.3 Codes | 17th April 2019 |
| **2.4. Testing** | 2.4 Working code | 23rd April 2019 |
| **3. Combining the Hardware and Software** | | |
| **3.1. Combining circuits** | 3.1 Connection between hardware & software. | 17th April 2019 |
| **3.2. Adding the software component** | 3.2 Programmed microcontroller | 24th April 2019 |
| **3.3. Testing both together** | 3.3 Test Results | 5th May 2019 |
| **4. IoT** | | |
| **4.1. Configuration** | 4.1 Connection between the app and microcontroller. | 28th April 2019 |
| **4.2. Software** | 4.2 Functional IoT. | 28th April 2019 |
| **4.3. Compiling** | 4.3 Working code. | 29th April 2019 |
| **4.4. Testing** | 4.4 Test Results | 29th April 2019 |

| | | |
|---|---|---|
| **5. Report and Presentation:** | | |
| **5.1 Project Charter Form** | 5.1 Report | 7th Oct 2018 |
| **5.2 Feasibility Report** | 5.2 Report | 16th Oct 2018 |
| **5.3 Project Plan** | 5.3 Report | 28st Oct 2018 |
| **5.4 Design Chapter** | 5.4 Report | 12th Nov 2018 |
| **5.5 Project Technical Presentation** | 5.5 PowerPoint | 4th Dec 2018 |
| | 5.6 Demonstration | 23rd April 2019 |
| **5.6 Progress Demonstration** | 5.7 Demonstration | 7th May 2019 |
| **5.7 Technical Demonstration** | 5.8 Report | 18th May 2019 |
| **5.8 Final Report** | 5.9 Report | 14th May 2019 |
| **5.9 Reflections** | 5.10 Poster | 13th May 2019 |
| **5.10 Final Poster** | 5.11 Video | 20th May 2019 |
| **5.11 Project's Video** | 5.12 Demonstration | 19th May 2019 |
| **5.12 Innovation Fair** | | |

### f. Project Team

1. Mohammad Owais Imtiaz
2. Tanmay Khachane
3. Ahmed Abdou
4. Talha Khan
5. Umm Kulsoom Emad

## 3. Market and Competitor Research

What would this product provide in terms of businesses for the people who purchase? Is it different from the other fruit juicer? Yes, our perspective market targets companies that need a device which calculates caloric in-take and make it look healthier to attract people that are on a diet. Not only companies would want to buy this, but it is also great for personal use at work or even at home, families do enjoy a trendy machine which goes beyond expectations when looking after their health. The companies can benefit from the machine by providing the customer with details on the shake or drink they make them based on how many fruits they have blended together

Currently 'FitSmart Juicer' has two main competitors which may include *Ronco Smart Juicer* and *Kurvings Smart Juicer* health friend smart juicer. The way *Kurvings Smart Juicer* operates is by taking users preferences in fruit juice and providing with healthy recipes for juices and/or smoothies. The only innovative aspect of the system is the app that it informs users of how much fruit in grams to add to the fruit juicer in order to obtained their personalized recipe keeping it under the calorie limit, the *Ronco Smart Juicer* is not an accessory rather it is an entire fruit juice system that offers its users more convenience through its easy to use 2-steps set up juicing machine and the machine can clean itself in 2-steps as well the juicer is called a smart juicer just because of the utility aspect of it and does not inform its users of recipes or calories what so ever.

The way that FitSmart Juicer is better is that rather than having the calories and weight of the fruit juicer predetermined in the system and having the user cut the fruit to size accordingly as in the case of *Kurvings Smart Juicer*, FitSmart notifies the user of exactly how much calories they have added to their juice as soon as they pick up the fruit off of the fruit basket through the weight sensors not only that but the system also comes with a restocking option that allows for

notifications sent to the user or fruit sellers of the user's restocking needs. currently the only existing juice-based firm that implemented weight sensors to estimate calories and notify users of restocking is fit smart. The juice market is estimated to be around 143 billion dollars and mostly consists of giant fruit juice manufacturers like Florida's natural, minute maid, Tropicana etc. With the use of FitSmart Juicer, the users are able to get fresh juice made from fresh ingredients at home with a handy app that keeps the user to keep track of their calorie intake.

# 4. Product Design

## a. System Design

### i. Overall Diagram of the system

The following figure shows the overall diagram of the system.

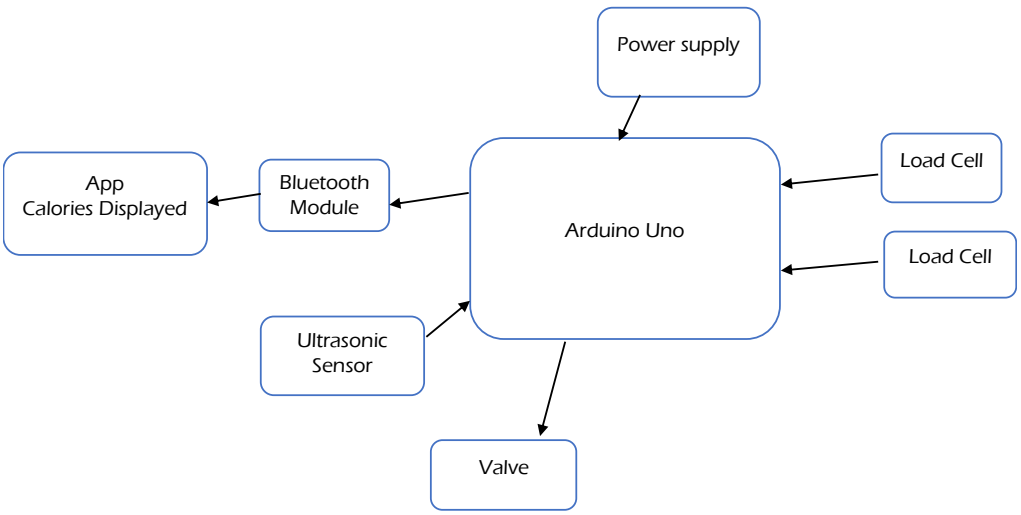


*Figure 1: Overall Diagram of the system*

### ii. Block diagram of the system

The following is the detailed block diagram of the system with explanation of each component and its functionality.

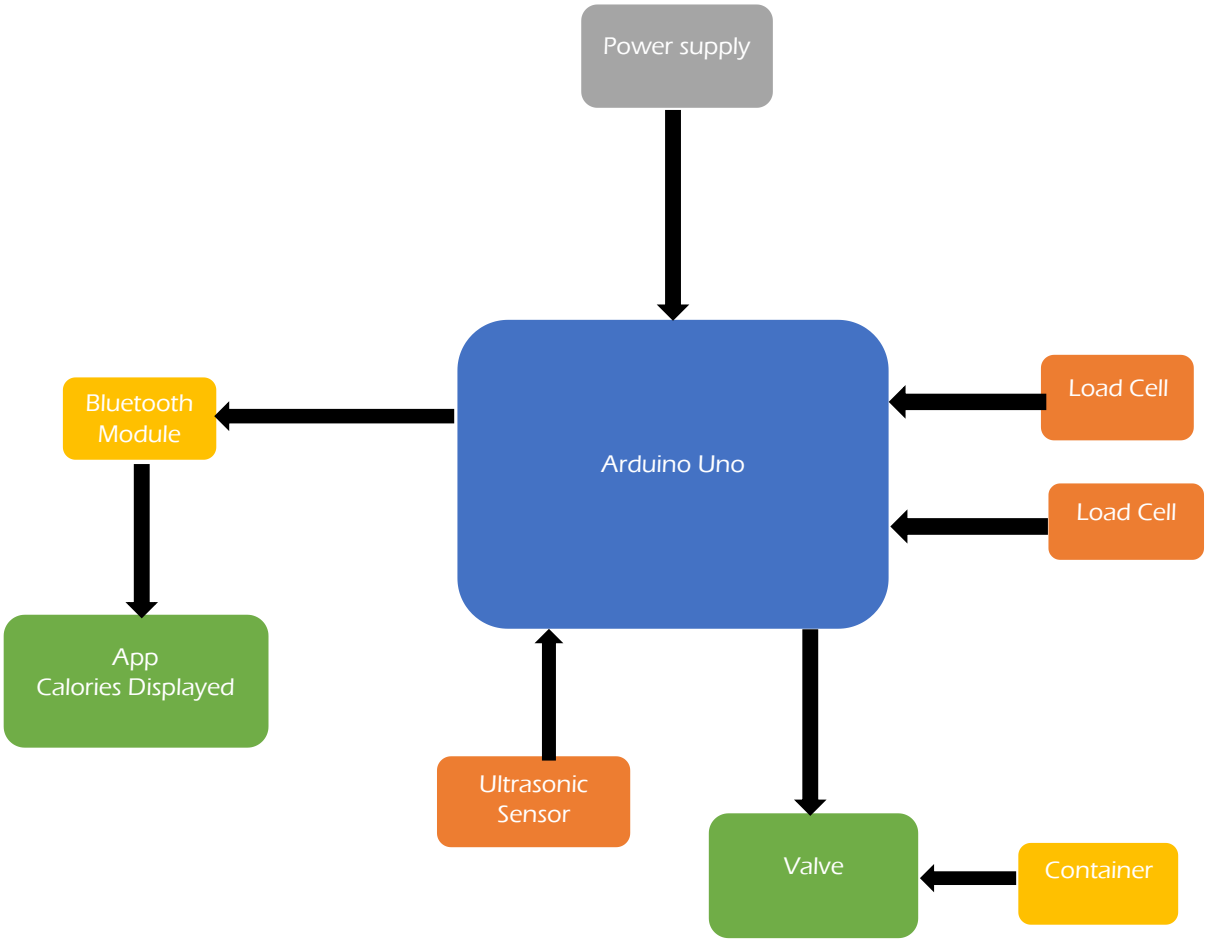


*Figure 2: Block Diagram of the system*

**The power supply is shown in grey color*
**The microcontroller is shown in blue color*
**The components to help the outputs are in yellow*
**The inputs are shown in orange color*
**The outputs are shown in green color*

- The ***power supply*** to the Arduino Uno is provided by a USB cable connected to a laptop.
- The ***load cells*** send data about the weight removed of the fruits to the Arduino Uno. Each load cell is predefined a specific fruit.
- The ***Arduino Uno*** is the microcontroller which controls the entire system. It gives output on the app and the valve based on the inputs from the load cells and ultrasonic sensor. It calculates the nutritional facts based on the weight removed (explained further in the software design section) and controls the status of the valve for it to be open or close.
- The ***Bluetooth Module*** acts as an interface between the *Arduino Uno* and the *App.*
- The ***App*** displays the output of the system, the nutritional facts and the stock of the fruits, are displayed on the mobile phone. It also notifies the user when restocking for a certain fruit is required.

- The ***ultrasonic sensor*** is used to detect the presence of a cup in order to control the *valve*, to open it or close it for dispensing *container*'s fluid.

### iii. 3D Model of the system

The following figure shows the 3D model of the system.

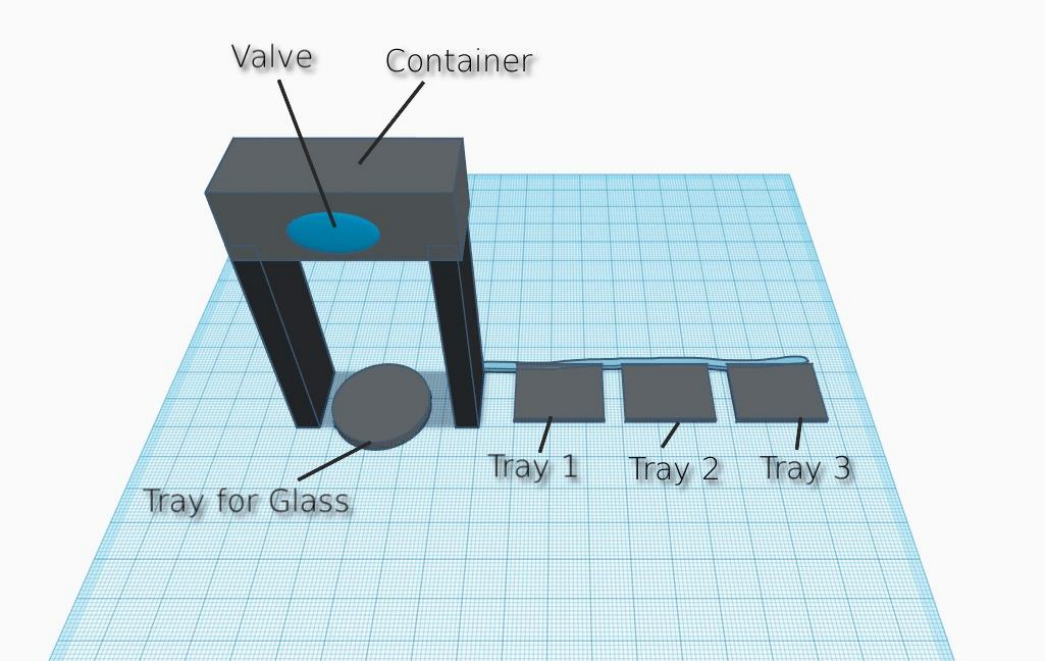


*Figure 3: 3D model of the system*

### iv. Final Model of the system

The following figure shows the final and actual model of the system of the load cells without the valve system.

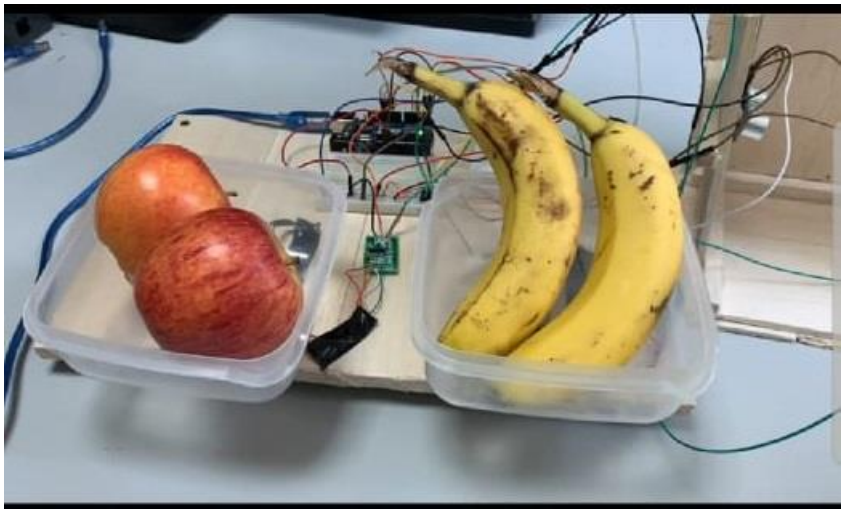

*Figure 4: Final Model of the System*

## b. Hardware Design

The following hardware components were used to achieve the system functionality and the model.

### i. Arduino Uno

Reason for Choice with comparison to other microcontrollers:
Arduino Uno is the preferred processor for this project, the Arduino is a user-friendly processor that is able to meet all the requirements and requires fewer man hours to program compared to Raspberry pi. Furthermore, on average Raspberry pi costs more than the Arduino Uno, and requires an operating system to be installed while the Arduino does not, thereby the Arduino is much more efficient for use.

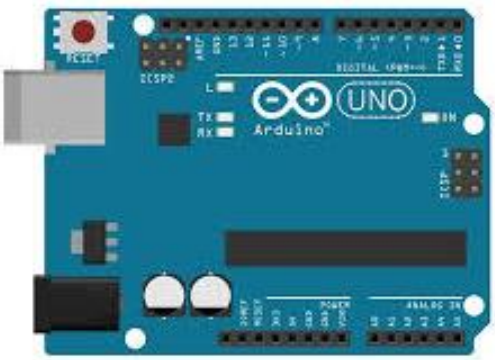


*Figure 5: Arduino Uno*

Description and Schematics
The Arduino UNO board is a microcontroller which has 14 input/output pins. 6 of these pins can be used as PWM outputs. The UNO board is powered by a USB-B and is hugely popular for its simple yet effective applications in the real world.
The UNO board is powered by a USB-B which is connected to a computer or a power supply. This connection also enables the transfer of the code for the function to be carried out.

Pin layout of the UNO board:

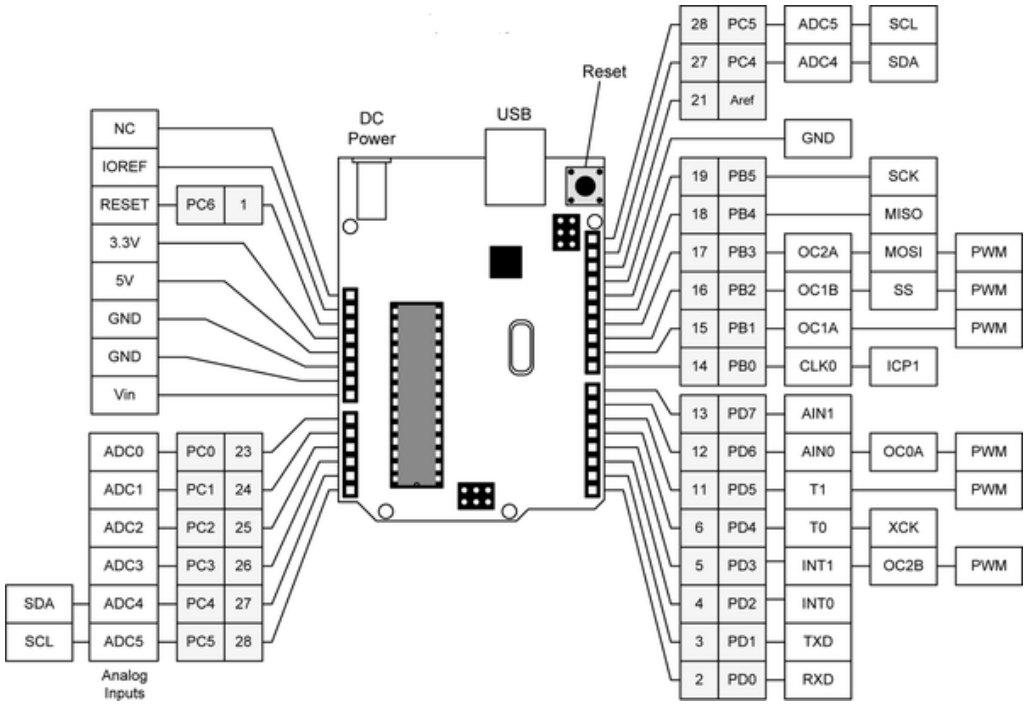


*Figure 6: Pin Layout of Arduino Uno*

- GND, short for ground, is used to ground the circuit.
- 5V and 3.3V pins supply either 5 or 3.3 volts of power to the board. Most components connected to the board generally run on either voltage with no reported problems.
- 6 Analog pins, A0 to A5, read the signal from a connected analog sensor to a digital value.
- Digital pins, from 0 to 13 on the UNO can be used for both digital input and output.
- Certain pins with a tilde (~) next to them are PWM pins. These pins can simulate analog output.
- There is also a reset button on the UNO board which can be used to temporarily connect the reset pin to the ground and restart any uploaded code.

### ii. Load Cell

Reason for choice:

Load cell was compared with other similar sensor in the market such as strain weight sensor, which required at least two sensors to be used to be able to measure the weight which is costly compared to load cell. Furthermore, strain weight sensor is not suitable to be used with Arduino Uno, so load cell was considered to be the better option in both, cost and functionality. Load cell is used with 'HX711 Amplifier' to amplify the sensitivity of the load cell. The maximum valve the load cell can handle is up to 5kg load.

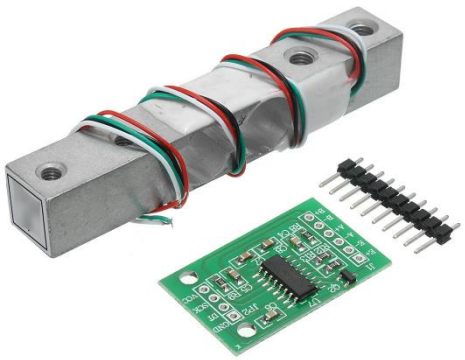

*Figure 7: Load Cell with Hx711 amplifier*

Functionality:
Load cell is used for measuring the weight of the stock of fruits. When a certain weight is removed, that value is used to calculate the nutritional values of the juice in which it is used. The remaining available value helps in determining whether restocking is required or not.
Two load sensors are used in our project, each is pre-assigned a specific fruit to monitor.

Testing of Load cell

- Testing Connections:
  Red wire, black wire, white wire and green wire of load cell was connected to E+ port, E- port, A- port and A+ port respectively of HX711. Then the output from HX711, port DT was connected to digital pin3 and SCK port is connected to digital PIN2 of Arduino, Vcc and GND of HX711 are connected to Vcc and GND of Arduino Uno, which was powered by the USB Cable of 5Volts.

- Testing Code Snippet:

```
#include "HX711.h"

// CIRCUIT:
// HX711 DOUT   TO PIN 2
// HX711 PD_SCK  TO PIN 3

const int LOADCELL_DOUT_PIN = 2;
const int LOADCELL_SCK_PIN = 3;

HX711 scale;

void setup() {
  Serial.begin(57600);
  //scale.begin(D2, D3);
  scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
}

void loop() {

  if (scale.is_ready()) {
    long reading = scale.read();
    Serial.print("HX711 reading: ");
    Serial.println(reading);
  } else {
    Serial.println("HX711 was not found or not ready.");
    Serial.println("HX711 not found.");
  }

  delay(1000);
  }
```

- Testing Procedure:
  After the connections being made and code uploaded on the Arduino Uno, weight was applied on one side of the load cell. Depending upon the bend, the readings given by the load cell varied. The following figure shows the readings before the calibration.

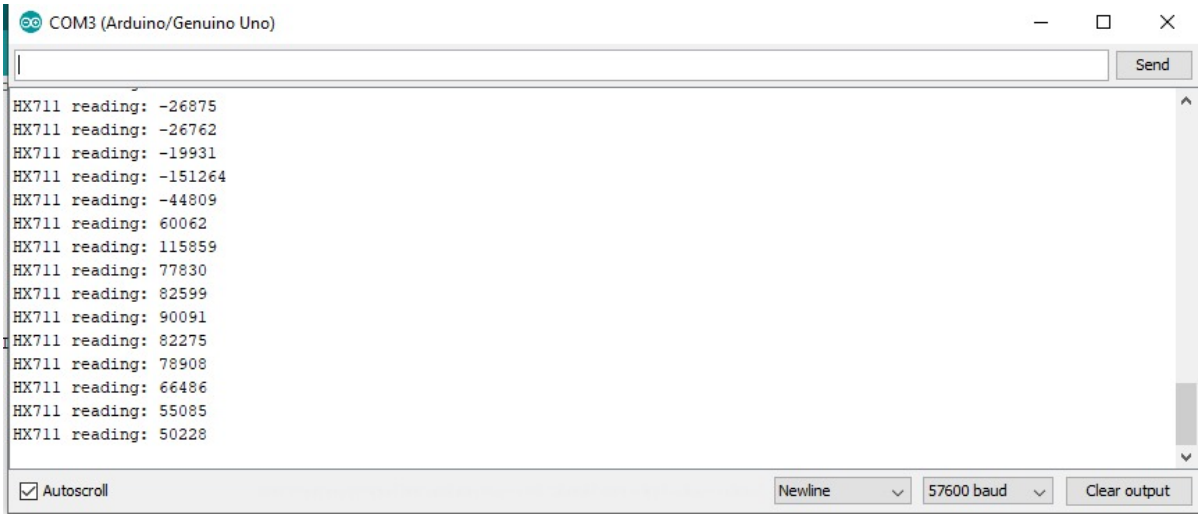


*Figure 8: Result of Load Sensor Testing*

- Calibration

  The calibration of the load cells is very important as the load cells are not capable of detecting actual weight without being calibrated as shown the figure above. The following code was used to calibrate the load cells.

```
#include "HX711.h"
HX711 scale;
float calibration_factor =90000; // this calibration factor is
adjusted according to my load cell

float units;
float ounces;
void setup() {
  Serial.begin(9600);
   scale.begin(5,4);
  Serial.println("HX711 calibration sketch");
  Serial.println("Remove all weight from scale");
  Serial.println("After readings begin, place known weight on
scale");
  Serial.println("Press + or a to increase calibration factor");
  Serial.println("Press - or z to decrease calibration factor");
  scale.set_scale();
  scale.tare();  //Reset the scale to 0
  long zero_factor = scale.read_average(); //Get a baseline reading
  Serial.print("Zero factor: "); //This can be used to remove the
need to tare the scale. Useful in permanent scale projects.
  Serial.println(zero_factor);
}
void loop() {
  scale.set_scale(calibration_factor); //Adjust to this calibration
factor
  Serial.print("Reading: ");
  units = scale.get_units(), 10;
  if (units < 0)
  {
    units = 0.00;
  }
  ounces = units * 0.035274;
  Serial.print(units);
  Serial.print(" grams");
  Serial.print(" calibration_factor: ");
  Serial.print(calibration_factor);
  Serial.println();

  if(Serial.available())
  {
    char temp = Serial.read();
    if(temp == '+' || temp == 'a')
      calibration_factor += 10;
    else if(temp == '-' || temp == 'z')
      calibration_factor -= 10;
  }
  delay(1000);
}
```

- Conclusion
  Three load cells were ordered, same testing and calibration was used for all of them. Two load cells were fully functional and worked as expected, but one load cell was faulty and was not used in the system.

Connections with Arduino Uno

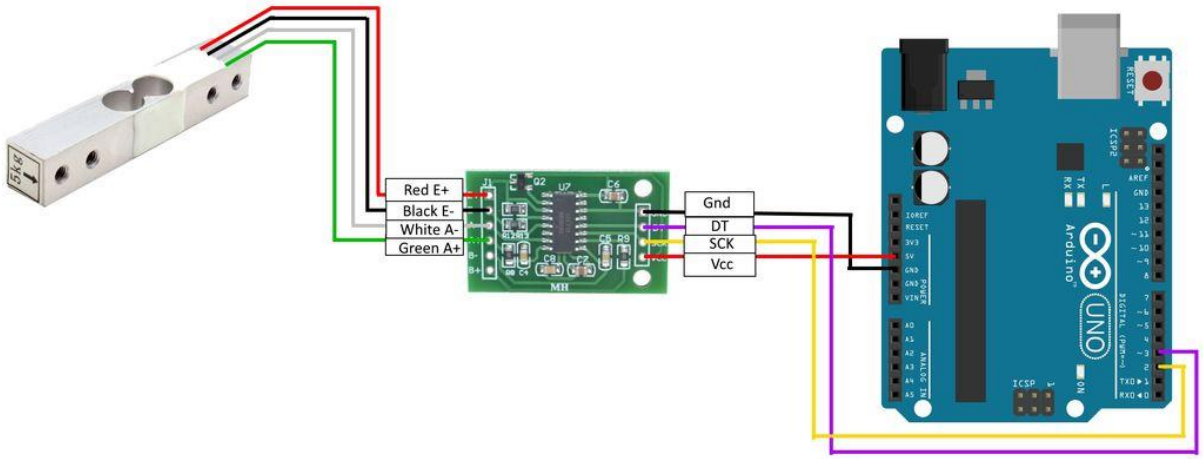


*Figure 9: Connection of Load cell with Arduino Uno*

Load Cell to HX711:

- E+ : RED
- E- : BLACK
- A- : WHITE
- A+: GREEN

HX711 to Arduino Uno:

- VCC to 5V
- GND TO GND
- SCK to D5
- DT TO D6

### iii. IR Sensor (discarded)

Reason for choice:
This digital infrared sensor is compatible with the chosen microcontroller (Arduino Uno), easy to connect and is cheap. It has been chosen because it fully satisfies the requirements for the prototype.

Important parts of an IR sensor

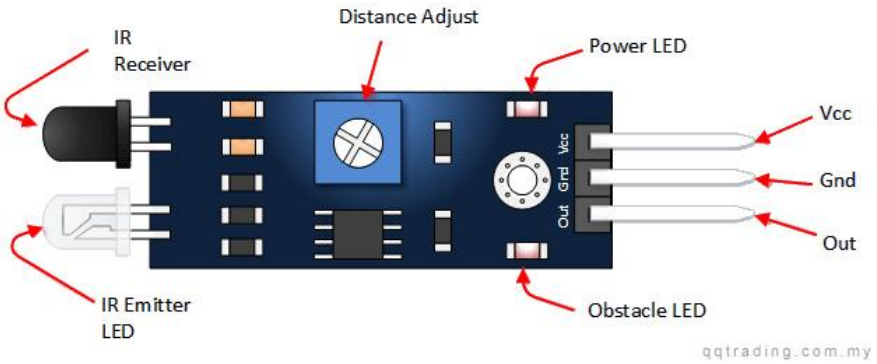


*Figure 10: IR sensor and its parts*

- Power LED: always on once connected
- Obstacle LED: turns on whenever an object is detected
- Distance Adjust: "potentiometer" the distance at which an object has to be detected can be adjusted by rotating the knob;
    - clockwise (detection distance increase)
    - counterclockwise (detection distance decrease).

Functionality:
It is used to control (open or close) the valve. When a glass is placed in front of the IR sensor, it detects it and opens the valve, and when nothing is detected, the valve is closed.

Testing of IR sensor

- Testing Connections:
  The output pin of the IR sensor, Vcc pin and ground pin were connected to digital pin 8, VCC and GND of the Arduino Uno board, respectively.

  This was achieved by using a breadboard where the IR sensor was inserted and small wires were taken and used to connect it to the Arduino Uno, which was powered by the USB Cable of 5Volts.

The following figure is of the connections made.

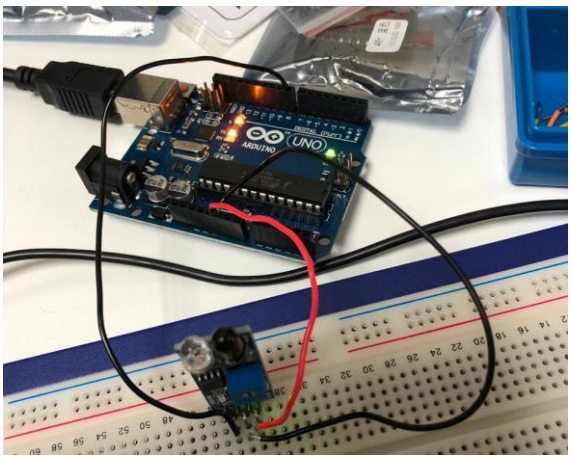

*Figure 11: Connections while testing IR*

- Testing Code Snippet:
  The following code was uploaded to the Arduino Uno for the IR Sensor

```
void setup() {
 Serial.begin(9600);
 pinMode(8, INPUT);// set pin as input

}

void loop() {
  //Written for Robojax on Dec 28, 2017
  int detect = digitalRead(8);// read obstacle status and store it
into "detect"
  if(detect == LOW){

   Serial.println("Obastacle on the way");
  }else{

   Serial.println("All clear");
  }
  delay(300);
}
```

- Testing Procedure:
  After the making the connections and uploading the code, the POWER LED was expected to be ON the entire time and the OBSTACLE LED was expected to be OFF the entire time unless an object was detected which would make it turn ON.
  The following figure shows that only the POWER LED (on the right) is ON because no object is detected.

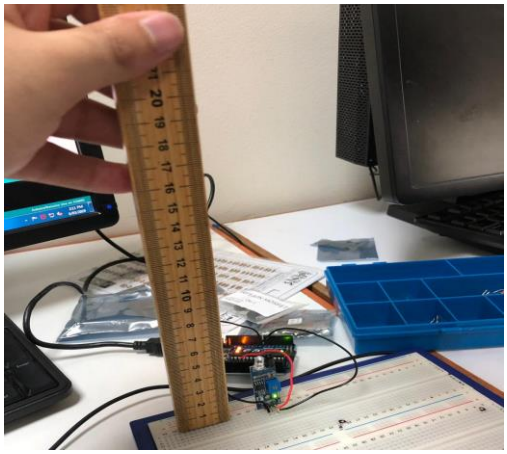

*Figure 12: IR not detecting an object*

The following figure shows that both leds; POWER LED (on the right) and OBSTACLE LED (on the left) both are ON when an object (in this case a hand) is detected.

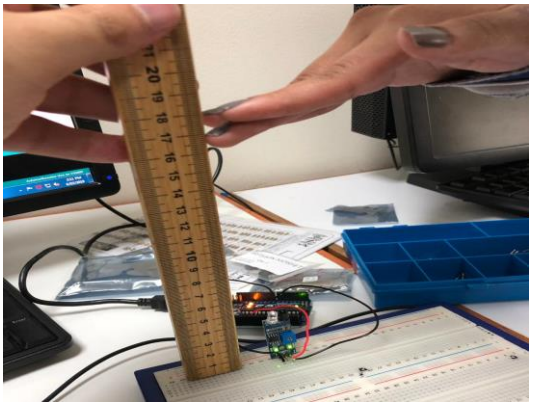

*Figure 13: IR detecting an object*

- Conclusion:
  With the help of a ruler and adjusting the distance potentiometer at different lengths, the experiment was repeated, and every time the IR sensor detected the presence of an object, therefore, it is fully functional.

Connection with Arduino

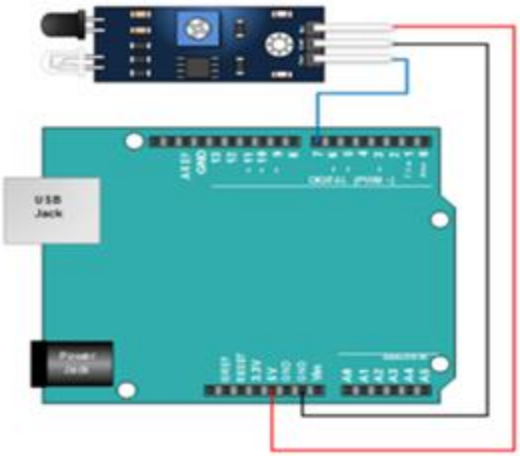

*Figure 14: Connection of IR sensor with Arduino Uno*

IR sensor to Arduino Uno

- The output pin of the IR sensor is connected to digital pin 8 of the
- The Vcc pin of the sensor to the Vcc
- The ground pin of the sensor to the ground

Problem encountered while using IR sensor in the project
The component got misplaced because of which 'Ultrasonic Sensor' was used in the project instead of the IR sensor.

### iv. Ultrasonic Sensor

Reason for choice:
This sensor is used as a substitute for the IR sensor due its availability and compatibility with this project. It is more accurate than the IR sensor and has the ability to not only detect an object but also to measure the distance between the sensor and the object.

Functionality:
It is used in combination with two LEDs; red and green. When a cup is detected by the ultrasonic sensor, the green LED would turn ON, and the red LED is ON during the absence of a cup.

Testing of the Ultrasonic Sensor

- Testing connections:
  The Vcc pin, ground pin, trig pin and echo pin were connected to Vcc, GND, digital pin 9 and digital pin 10 of the Arduino Uno board, respectively.

- Testing code snippet:

```
/*
* Ultrasonic Sensor HC-SR04 and Arduino Tutorial
*
*/

// defines pins numbers
const int trigPin = 9;
const int echoPin = 10;

// defines variables
long duration;
int distance;

void setup() {
pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
pinMode(echoPin, INPUT); // Sets the echoPin as an Input
Serial.begin(9600); // Starts the serial communication
}

void loop() {
// Clears the trigPin
digitalWrite(trigPin, LOW);
delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 microseconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds
duration = pulseIn(echoPin, HIGH);

// Calculating the distance
distance= duration*0.034/2;

// Prints the distance on the Serial Monitor
Serial.print("Distance: ");
Serial.println(distance);
}
```

- Testing procedure:
  After the connections were made, the code was uploaded on the Arduino Uno. Objects were placed in front of the ultrasonic sensor.

- Conclusion:
  At different distances, the object was detected by the ultrasonic sensor. The corresponding value of the distance between the object and the ultrasonic sensor was displayed on the serial monitor.

```
Distance: 9
Distance: 9
Distance: 9

Distance: 6
Distance: 6
Distance: 6
Distance: 6
```

*Figure: Snapshot of the testing result for Ultrasonic Sensor*

Connections with Arduino Uno for the Project

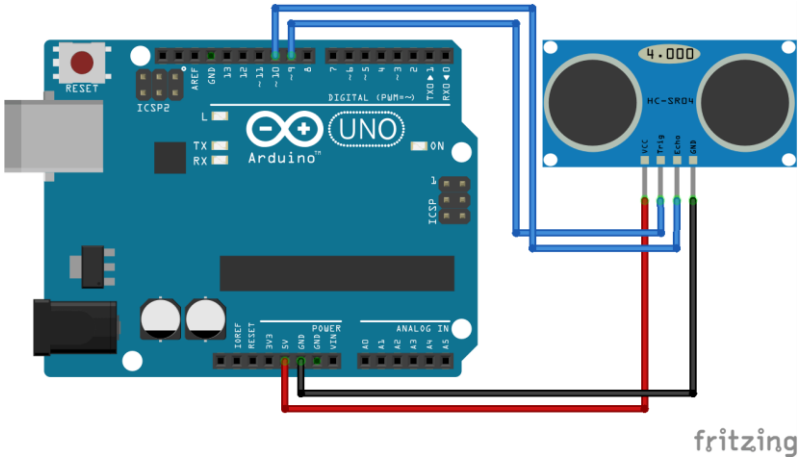


*Figure 15: Connection of Ultrasonic Sensor with Arduino Uno*

Ultrasonic Sensor to Arduino Uno

- VCC to 5V
- GND to GND
- TRIG pin to digital pin 9
- ECHO pin to digital pin 10

## v. Bluetooth Module

Reason for choice

Another microcontroller could have been chosen which had a built-in Bluetooth facility. Therefore, using this Bluetooth module with the selected microcontroller was suitable.

Functionality:

Bluetooth HC-05 module was so chosen to establish a connection between the Arduino Uno and the mobile phone, which would aid in controlling the product through an app.

Testing of the Bluetooth Module

- Testing connections:
  The VCC pin, GND pin, RXD and TXD of the Bluetooth module are connected to 5V, ground TXD and RXD slots of the Arduino Uno, respectively. An LED's positive is connected to pin 13 of the Arduino through a resistance of value 220 and its negative is connected to GND. Figure of connections is shown below.

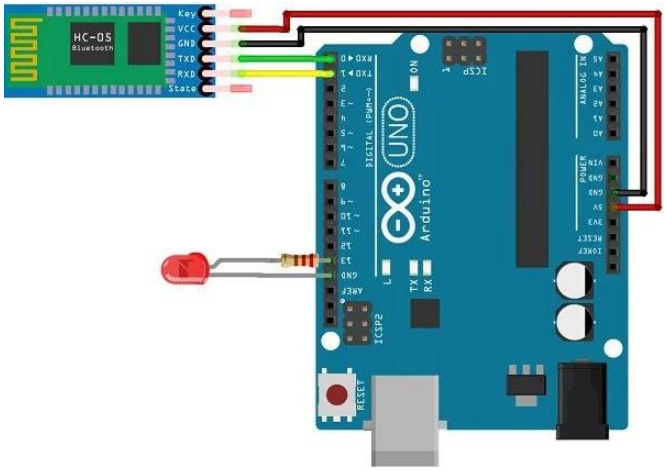


*Figure 16: Testing connection for Bluetooth module*

- Testing code snippet:

```
/*
 *  Bluetooh Basic: LED ON OFF
 *  This program lets you to control a LED on pin 13 of
arduino using a bluetooth module
 */
char Incoming_value = 0;                //Variable for
storing Incoming_value
void setup()
{
  Serial.begin(9600);          //Sets the data rate in bits
per second (baud) for serial data transmission
  pinMode(13, OUTPUT);         //Sets digital pin 13 as
output pin
}
void loop()
{
  if(Serial.available() > 0)
  {
    Incoming_value = Serial.read();       //Read the incoming
data and store it into variable Incoming_value
    Serial.print(Incoming_value);         //Print Value of
Incoming_value in Serial monitor
    Serial.print("\n");          //New line
    if(Incoming_value == '1')             //Checks whether
value of Incoming_value is equal to 1
      digitalWrite(13, HIGH);  //If value is 1 then LED
turns ON
    else if(Incoming_value == '0')        //Checks whether
value of Incoming_value is equal to 0
      digitalWrite(13, LOW);   //If value is 0 then LED
turns OFF
  }

}
```

- Testing Procedure:
  This test ensures the working of the Bluetooth module HC-05, with the use of an app to control the LED. Once the above-mentioned connections are made, the code needs to be uploaded to the Arduino Uno but while uploading the code, the RXD and TXD wires are disconnected as they interrupt the transmission of data from the PC to the Arduino. Once the code is uploaded, the RXD and TXD wires are connected again, and the LED app (which can be downloaded from here) is installed in a mobile phone. Pairing the app with the Bluetooth module is required by entering the password 1234 or 0000. After the

after the Bluetooth connection is made, the status of the LED can be controlled by clicking on the ON and OFF on the app.

- Conclusion:
  The figure below shows the result of the testing, in which turning the LED on and off was possible by clicking the buttons on the app.

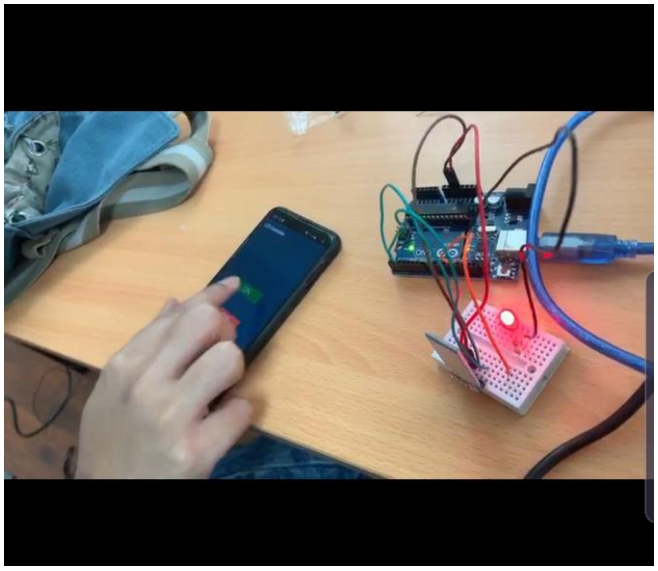

*Figure 17: Testing output of the Bluetooth module*

Connections with Arduino Uno for the Project

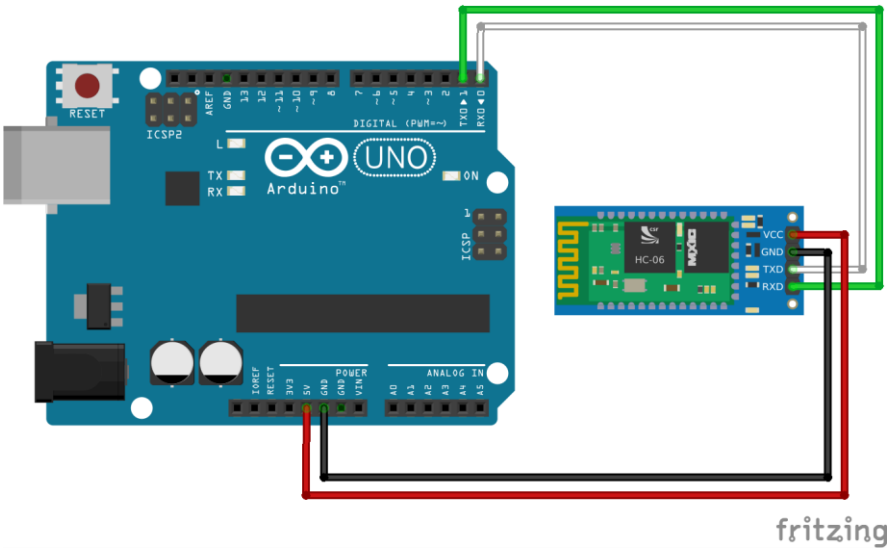


*Figure 18: Connection of Bluetooth module with Arduino Uno*

Bluetooth Module to Arduino Uno

- VCC to 5V

- GND to GND
- RXD to TXD (pin1)
- TXD to RXD (pin0)

### vi. Solenoid Valve

Reason for choice

This component has been selected for our project due to its compatibility with the microcontroller, which very easy to control as well as not expensive to purchase and setup.

Functionality

Solenoid Valve is a component that is used to control the flow of water. This component is used in combination with an IR sensor; when IR sensor detects a cup, valve is opened and allows the flow of water, and it is closed otherwise.

Connections with Arduino Uno

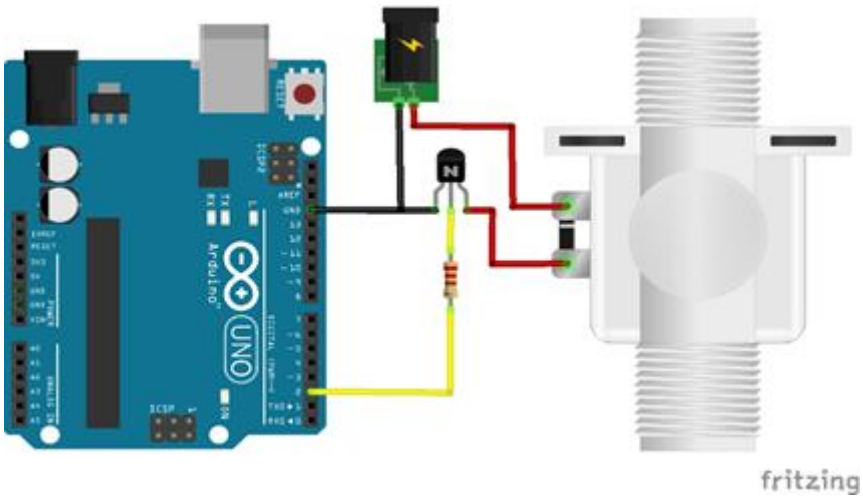


*Figure 19: Connection of Solenoid Valve with Arduino Uno*

Problem encountered while using Solenoid Valve in the project

The solenoid valve delivered had a different rating (12V) which wasn't compatible with the microcontroller, therefore the microcontroller couldn't supply enough power for the valve to operate. A 5V relay switch was ordered to achieve the working of the valve but 12V relay was delivered, which again did not contribute in integrating Arduino Uno and the valve. Other solutions like buying a compatible valve were not considered because of lack of time.

Substitute

Two LEDs, red and green, were used to represent the working of valve in combination with the ultrasonic sensor. When a cup was detected by the ultrasonic sensor, the green LED would turn ON representing the valve to be open. The red LED would be ON in absence of the glass representing the valve to be closed. *The guideline for the substitute was given by the professor.*

## c. Software Design

### i. Data collection, processing and output

**Data collection**

The main input of the system is from load cells which is connected to the digital pin of Arduino, and it reads the value from each load cell.

The other input is from the ultrasonic sensor to determine the presence of a cup in order to control the valve's status

**Data Processing and Output**

The nutritional details like calories, sodium, potassium, protein and carbohydrates of each fruit are stored as constants, and the stock amount (weight) is variable, as it is from the weight sensor. An example of the fixed details stored in system, is shown in the table below:

*Table 2: Nutritional Details of Fruits*

| Nutrients per 100g/ Fruits | Apple | Banana |
|---|---|---|
| Calories | 52 | 89 |
| Carbohydrates | 14g | 23g |
| Sugar | 10g | 12g |

Whenever a new juice is made, the required fruit is selected and removed from its stock weight sensor. The weight removed is deducted from the stock level, and the nutritional details of the removed weight are displayed on the screen. If more than one fruit is used, then the nutritional details of all the removed weights are added together, section by section, e.g.

Total Calories = calories from apple + calories from bananas
Total Carbohydrates = carbohydrates from apple + carbohydrates from bananas
Total Sugars = sugars from apple + sugars from bananas

The computed details of all nutrients are outputted by system, are to be displayed on the app.

Once a fruit is removed, the stock level is compared with minimum stock level which is fixed to be 200g for all fruits. If the stock level of the fruit removed is less than the minimum stock level, then a message is sent to the user, notifying him that reordering is required for that specific fruit. For the communication between the Arduino Uno and the app, Bluetooth module is used.

When a cup is detected by the ultrasonic sensor, it turns the green LED on, representing the valve to be open. When a cup is not detected by the ultrasonic sensor, it turns the red LED on, representing the valve to be closed.

## ii. System Flowchart

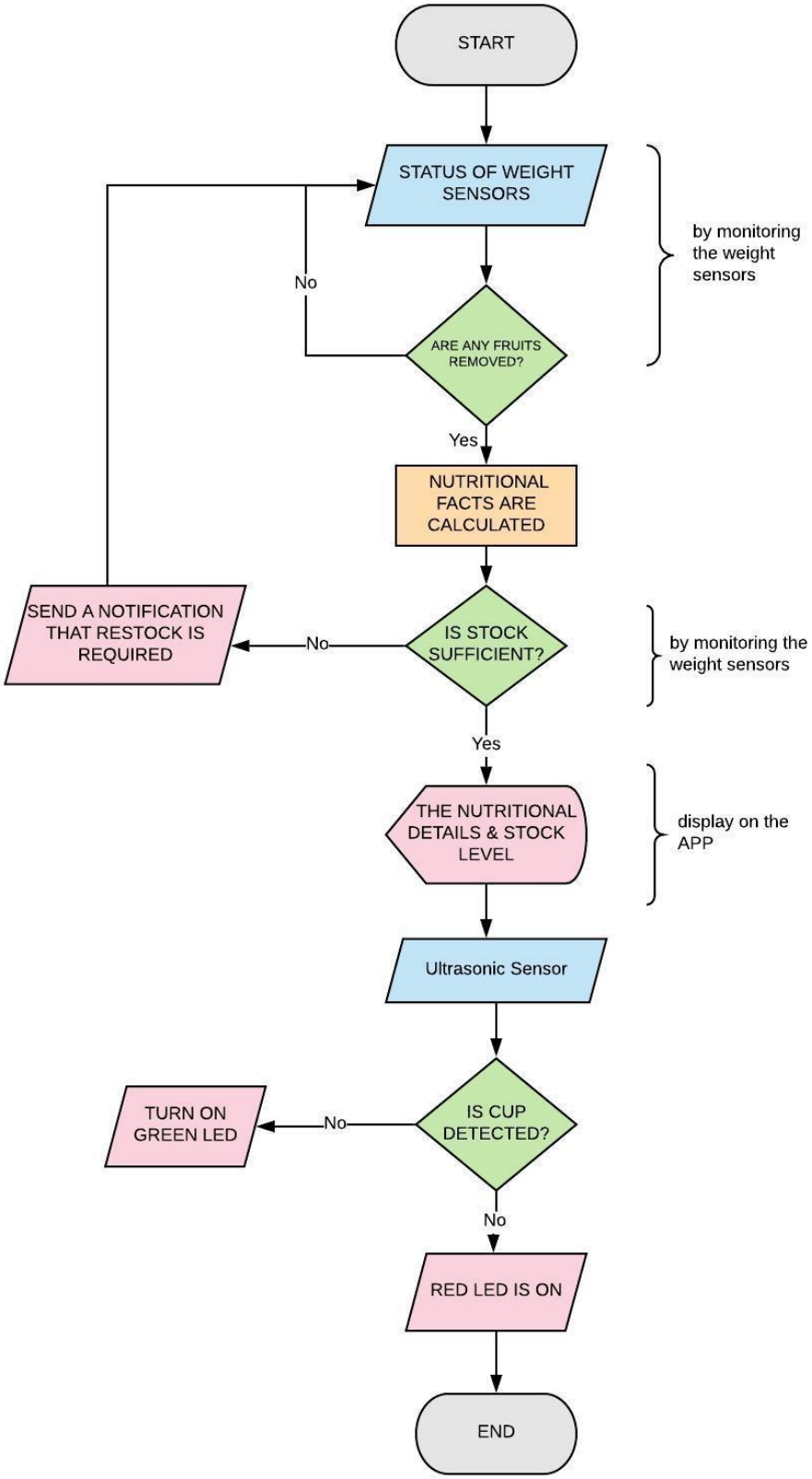


*Figure 20: Flowchart of the System*

# 5. Product Testing Procedures and Results

The sensor circuits and output components were combined to work as complete system.

The Arduino Uno received all the data from the sensors (load cells, ultrasonic sensor),

The load cells measured the weight and sent it to the microcontroller, whenever the fruits were removed. All the changes in weight were recorded and displayed on the serial monitor

The nutritional facts were calculated from the information collected due to the change of weight of fruits from the load cells.

When a cup was detected by the ultrasonic sensor, the green LED was turned ON and during the absence was the cup, the red LED was on.

The Bluetooth module made a connection between the Arduino Uno and the app, the output of the system; the nutritional facts and the stock level was displayed on the mobile phone's app.

*(Note: Testing for individual circuits has been done along with the description of each component in the hardware section)*

- The following are screenshots from the app which were taken at different instances when testing was performed

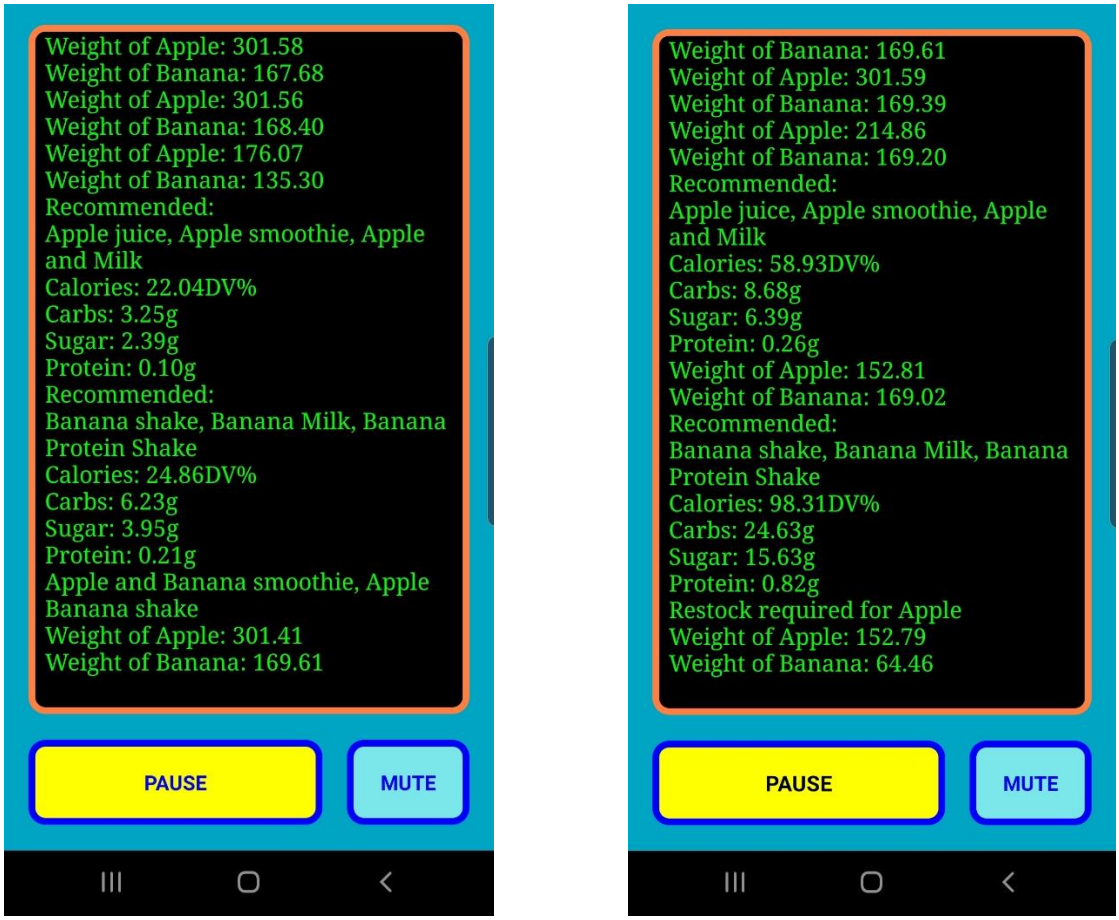


*Figure 21: Screenshots of the system output on the App*

# 6. Financials

The predicted tables for the entire system production **previously** were as follows:

Predicted cost for labor is shown in the following table:

*Table: Labor Cost*

| Type | Hours per week | Total number of hours(5 people) | Cost (dhs) |
|---|---|---|---|
| Feasibility Report | 10 | 50 | 2400 |
| Project Plan | 11 | 55 | 2640 |
| Design chapter | 50 | 250 | 12000 |
| Final poster | 10 | 50 | 2400 |
| Project report | 15 | 75 | 3600 |
| Project technical presentation | 8 | 40 | 1920 |
| Technical demonstration | 3 | 15 | 720 |
| Innovation fair | 20 | 100 | 4800 |
| | | **Total** | **30480** |

Predicted cost for components is shown in the following table:

*Table: Components Cost*

| Type | Quantity | Cost(dhs) |
|---|---|---|
| Micro controller | 1 | 160 |
| Weight sensor | 5 | 250 |
| Ethernet cable | 1 | 10 |
| LCD | 1 | 50 |
| Valve | 2 | 80 |
| LED | 6 | 2 |
| Ethernet shield | 1 | 75 |
| Plate | 1 | 5 |
| | **Total** | **632** |

The actual cost during the implementation of the entire system is as follows:

*Table 3: Actual Labor Cost*

| Type | Hours per week (per person) | Total number of hours (5 people) | Cost (AED) |
|---|---|---|---|
| **Feasibility Report** | 12 | 60 | 9000 |
| **Project Plan** | 13 | 65 | 9750 |
| **Design chapter** | 32 | 160 | 24000 |
| **Final poster** | 5 | 25 | 3750 |
| **Project report** | 20 | 100 | 15000 |
| **Project technical presentation** | 17 | 85 | 12750 |
| **Technical demonstration** | 40 | 225 | 33750 |

| | | | |
|---|---|---|---|
| **Innovation fair** | 7 | 35 | 5250 |
| | | **Total Amount** | **113250** |

**The labor cost is 150AED per hour for every individual*

*Table 4: Actual Components Cost*

| Component Name | Quantity | Total Cost (AED) |
|---|---|---|
| **Load Cell and HX711** | 3 | 72 |
| **LCD** | 1 | 13.90 |
| **Solenoid Valve** | 1 | 28.62 |
| **Arduino Uno Rev3** | 1 | 105 |
| **Wi-Fi Module** | 1 | 58.28 |
| **Transistor IRF540N MOSFET** | 5 | 21.19 |
| **IR sensor** | 1 | 17 |
| **Bluetooth Module** | 1 | 40 |
| **Containers** | 1 | 35 |
| **Relay** | 1 | 40 |
| **Ultrasonic Sensor** | 1 | 27 |
| | **Total Amount** | **458** |

The components cost is less than the one which is predicted, though more components were ordered than planned due to unexpected errors which came during the implementation but still not only staying in the budget but also decreasing the predicted amount is a success for the team.

The labor cost is much more than predicted because initially one third of the actual cost per hour for each engineer was used (AED48), whereas, in the actual implementation AED150 was per hour for every individual.

## 7. Discussion of Design and Product Feasibility

The overall functionality of the system was completely met except for the working of the valve. The ultrasound sensor was able to detect the cup, and from that the line of code which should open the valve works as well but because of the voltage rating on the valve was a little too high (220V), therefore the valve was not able to operate with the system. Using a relay was considered but due to receiving a relay with wrong voltage, 12V instead of 5V, it was not a success as well.

The product had two load sensors which were calibrated accordingly, to calculate the weight taken out of the stocks and provide nutritional facts for the user who is looking after their healthy lifestyle, with that they can manage to follow their macros and be able to tell how many calories they had per day for the daily caloric intake. This was all shown on an application with the use of a simple GUI to guide the user on how to set some fruits for some scales and display the amount of stock left in kgs, in case the stock was running low on fruits; then the system would notify the user/fruit company through a message on the application itself to tell them to restock which is done automatically every time it runs down below a said threshold. Once the restock is done, the system is then ready for use again. All the functionality is achieved with the use the microcontroller, Arduino Uno alone.

To show the possible working of a valve, two different LEDs were used as an output in which indicate whether there is a cup or not. When the ultrasound sensor detects that there is a cup, green LED would go on, but if the cup was to be taken away from the designated spot, the red LED would go on and the green LED would switch off. If the green LED is on; the valve (which was not present or even usable in this case) would open and start pouring said needed liquid. The container which would contain the fluid were too small and fragile to hold liquids (water/milk) in them and because of how frail the wood is, it could barely hold it together without collapsing, that is multiple pieces of wood were used to support the posture of the FitSmart juicer.

Overall, on a larger scale with sufficient time and effort put for the project, the functioning of the entire system for calculating nutritional details along with the only non-working component, valve is possible. The product is easy to use, user-friendly and could be used in different environments like in fruit juice companies for business, at a gym to attract more people or personal use at home, as it requires no training to operate it.

## 8. Statement on Achievements

The project was considered a success overall since most of the goals have been met and done. The list is as the following:

- *Tray for each fruit with load sensor attached to it:* The containers have load cells attached to them which measures the weight of the container with the fruits inside of them which also leads to the next goal.

- *Determining nutritional facts:* sugars, carbohydrates, vitamins etc. are displayed on the application to help the user identify the caloric intake he/she has per day on certain times of the day with certain types of drinks (rather recipes for drink) that can be made for said fruits taken out of the containers, depending on the weight, the system calculates the nutritional facts per gram of fruit, accuracy is achieved.

- *Application with user-friendly GUI:* Simple android app which helps read the nutritional facts, weight and stock needs if it ever needed a refill automatically by notifying the user through message on the application which sends the fruit company to the user and restock their juicer.

- *Use of solenoid valve*: Valve did not work at all due to its rating (as mentioned section 7). However, a funnel was used instead of it with two LEDs which represent opening and closing the valve where it starts to pour when the cup is detected (green LED on) and stops when the cup is away (red LED on).

- *Model representing the original system*: Model has changed significantly from what was started from. A lot of sketches took places but the one the stuck was the big container for water, milk and solenoid valve connected to it to allow liquids to flow through.

- *Sensor used with solenoid valve:* During the work on the model the IR sensor got misplaced due to which instead of the IR sensor, an ultrasonic sensor was used to detect the cup and help the valve open when the cup is detected.

- *Waste management:* This goal is achievable when the system is practically in use so that the users only buy fruit when the stock of a specific fruit run low and in order to eliminate unnecessary buying of fruits.

Overall, most goals were met, but those which weren't met were because of lack of time and lack of effort and work put in for the project.

## 9. Detailed Design Recommendations

The following are a few possible design improvements and advancements that could be made in future.

- Trays for stocking can be added and removed according to the buyer's specification.
- Vegetables can also be stocked.
- As the system calculates the nutritional facts, it can be integrated with an actual juicer.
- Along with notifying the whenever the stock runs low, a link can be added in the app itself for the user to order the required fruits from their mobile phone.
- If the system is used by fruit juicing companies, the app can be made available to customers for them to place an order through the app.

## 10. References

https://www.instructables.com/id/Infra-Red-Obstacle-Detection/

https://www.instructables.com/id/How-to-Interface-HX711-Balance-Module-With-Load-Ce/

https://maker.pro/arduino/tutorial/bluetooth-basics-how-to-control-led-using-smartphone-arduino

https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/

# 11. Appendices

## Code of the Entire System

```
#include "HX711.h"

// CIRCUIT:
// HX711 DOUT   TO PIN 2
// HX711 PD_SCK  TO PIN 3
//
// Pin definitions D2 and D3 are provided by "nodemcu" board,
// see `nodemcu/pins_arduino.h`.
// HX711 circuit wiring
const int LOADCELL_DOUT_PIN = 2 ;
const int LOADCELL_SCK_PIN = 3;
float calibration_factor1 =  1355;
float calibration_factor2 =  9000  ;
float reading;
float reading2;
float newreading;
float newreading2;

 long duration, distance;
#define trigPin 13

#define echoPin 12

#define led 11

#define led2 6

HX711 scale1;
HX711 scale2;

void setup() {
  Serial.begin(9600);
  pinMode(8, INPUT);
  //scale.begin(D2, D3);
  scale1.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
  scale2.begin(5,4 );
   scale1.set_scale(calibration_factor1);
   scale1.tare();
   scale2.set_scale(calibration_factor2);
   scale2.tare();

   pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

pinMode(led, OUTPUT);

pinMode(led2, OUTPUT);
```

```
 // scale.begin(A3,A4);
}

void loop() {


  //if (scale1.is_ready()| scale2.is_ready() ) {

      reading = (scale1.get_units(20)/0.3461);
    reading2 = (scale2.get_units(20)/0.04105);

   if (reading < 0)
  {
    reading = 0.00;
  }

  if (reading2 < 0)
  {
    reading2 = 0.00;
  }

    Serial.print("Weight of Apple: ");
    Serial.println(reading   );

    Serial.print("Weight of Banana: ");
    Serial.println(reading2  );


   delay(3000);

 newreading = (scale1.get_units(20)/0.3461);
 newreading2 = (scale2.get_units(20)/0.04105);

if (newreading < 0)
  {
    newreading = 0.00;
  }

  if (newreading2 < 0)
  {
    newreading2 = 0.00;
  }

float a;
float b;


  if (reading > newreading+20)
  {
     a = reading - newreading;
     Serial.println("Recommended:");
   Serial.println("Apple juice, Apple smoothie, Apple and Milk");
```

```
     Serial.print("Calories: ");
Serial.print(a*0.95);  Serial.println("DV%");

        Serial.print("Carbs: ");  Serial.print(a*0.14);
Serial.println("g");

        Serial.print("Sugar: "); Serial.print(a*0.103);
Serial.println("g");

        Serial.print("Protein: ");   Serial.print(a*0.00413);
Serial.println("g");


  }


 if (reading2 > newreading2+20){

   b = reading2 - newreading2  ;
     Serial.println("Recommended:");
   Serial.println("Banana shake, Banana Milk, Banana Protein
Shake");
    Serial.print("Calories: "); Serial.print(b*0.95);
Serial.println("DV%");

            Serial.print("Carbs: "); Serial.print(b*0.238);
Serial.println("g");

        Serial.print("Sugar: "); Serial.print(b*0.151);
Serial.println("g");

        Serial.print("Protein: ");   Serial.print(b*0.0079);
Serial.println("g");


  }
   else{
    ;
  }

   if ( reading > newreading +20 && reading2 > newreading2+20){
     Serial.println("Apple and Banana smoothie, Apple Banana
shake");

   }

   else{
    ;
  }
```

```
   if(reading  <160 ){

Serial.println("Restock required for Apple");

   }
    if (reading2  <50){
    Serial.println("Restock required for Banana");


   }
//ir sensor


/*digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);

distance = (duration/2) / 29.1;

if (distance < 10)

{ digitalWrite(led,HIGH);
digitalWrite(led2,LOW);

}

else {
digitalWrite(led2,HIGH);
digitalWrite(led,LOW);

}*/


  //} //else {
    //Serial.println("HX711 was not found or not ready.");
  //Serial.println("HX711 not found.");
  //}

  delay(1000);
  }
```

## 12. User Manual


# User Manual

**By: The Firm Logistics**

## Contents in User Manual

Description of the product

Product and its components

How to use the product?

Why choose this product?

Recommended buyers

Handling the product and safety measures

Customizations available

## Description of the product

FitSmart Fruit Juicer is a product which recommends different juices and calculates their nutritional facts. In addition, it monitors the stock by notifying the user when the stock of a fruit runs low. All these features of this product can be used with the aid of our user-friendly app. The functionality is achieved by using trays for fruits on which the fruits' stock is placed. The trays help in determining the amount of the fruit used to prepare the juice, and calculate its carbohydrates, calories, etc. These details and the remaining stock on are displayed on the app.

## Product and its components

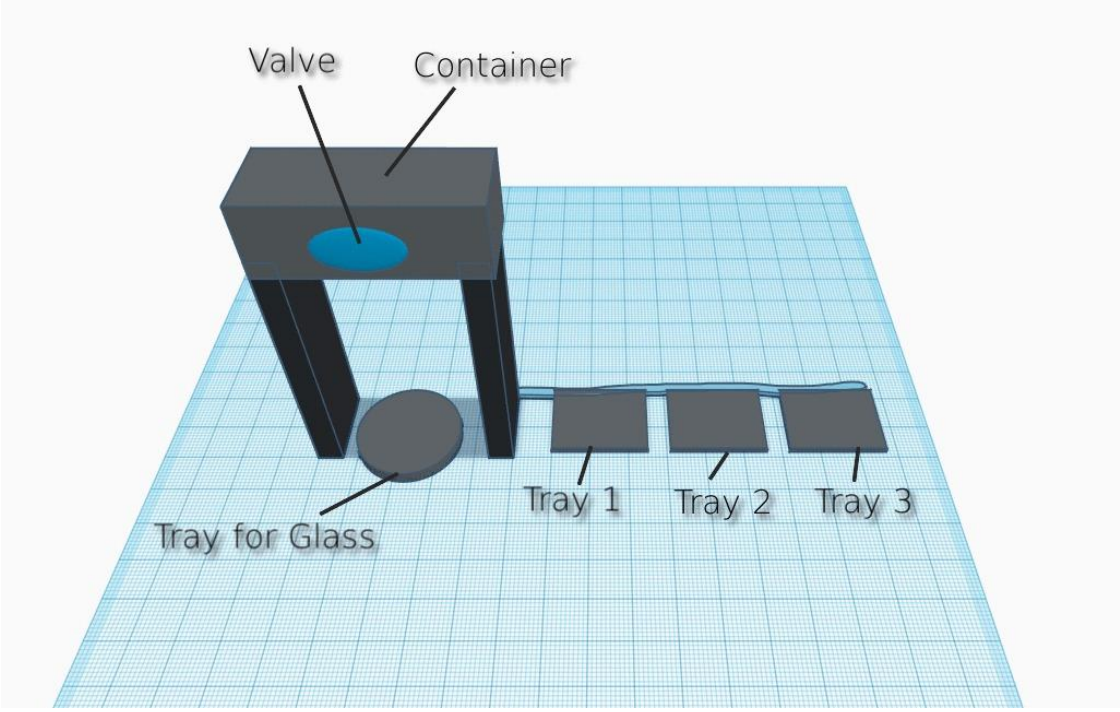


*Figure: Model of the Product*

- Container: Filled with water
- Valve: Automatic Valve which is used for letting the water flow
- Tray for Glass: Used for placing for glass to be placed so that it is filled with the container's fluid.
- Tray 1: Used for placing the stock of fruits
- Tray 2: Used for placing the stock of fruits
- Tray 3: Used for placing the stock of fruits
- App: Used for displaying the stock and nutritional details of the juice made.

## How to use the product?

- Step 1: Install the app and connect it to the product
- Step 2: Assign the trays (tray 1, tray 2 and tray 3) with a specific fruit of your choice
- Step 3: Choose a drink from the recommended drinks available

- Step 4: Wait for the choice to be processed
- Step 5: Remove the fruits from the stock according to the choice and use them for the making juice
- Step 6: The nutritional details of your serving are displayed along with the available stock

## Why choose this product?

The companies can benefit from the machine by providing the customer with details on the shake or drink they make for them, based on how many fruits have been blended together. A notification is sent to the user to restock their specific fruit when it reaches a low stock. The automated restocking feature would save companies a lot of wasted resources, and boost efficiency, the system also calculates an accurate measurement of calories for the juice per serving, allowing the juice companies to be more appealing to people seeking a healthy lifestyle, hence boosting sales. There are other juicers which calculate the nutrition per serving and other restocking systems, but this product is a perfect blend of both, making it easier for companies to install one system that fulfils both the requirements.

## Recommended buyers

Companies that need a device which calculates caloric in-take and make it look healthier to attract people that are on a diet. It is also great for personal use at work or even at home, for families who enjoy a trendy machine which looks after their health. Moreover, it promotes the importance of healthy lifestyle amongst people in an attractive manner.

## Handling the product and safety measures

The device is automatic, easy to operate and does not require any special training for its users. Also, there are almost no potential risks (e.g. injuries) for people operating it. It is safe to use at home and for business purposes.

## Customizations available

- Trays for stocking can be added and removed according to the buyer's specification
- Vegetables can also be stocked