Team HUMMUS

Laura Schep, Mats van Braam, Julia van der Geest, Maud van der Hall,
Ruben Koole, Bart Blom en Famke van den Boom

1. Short description of your challenge/problem (500 words)

Meat waste is a big problem. The production of meat is a high cost for the environment, a lot of greenhouse gasses are emitted by the industry and animals. Meat decomposes because of fungi and bacteria, which are borne by their implements, the people handling the meat production and by the animal it comes from. If the meat is not treated, it will be infectious, unappetizing or poisonous within hours or days. This is something people should be careful with, but the problem is that we often throw away meat that is still edible.

2. Short description of your solution (500 words)

We want to make a device that detects if the meat has gone bad. We
will design an Arduino based container with a sensor that detects certain
gases. These gases are detected by the sensor before a human is able to sense
them. and based on that the device will give an indication on whether the meat is still safe to eat.
This way the amount of meat that gets thrown away, before the
actual expiration date, will be reduced. It also has a healthcare factor that we really liked. Expired food is bad for people, even dangerous in some cases.

3. Short description of method and tools used (software, hardware) (1,000 words)

For this project a requirement was the use of sensors. The TGS 822 and DHT11 sensors were used, they are gas and humidity/ temperature sensors. The reason these sensors were chosen is because the TGS 822 measures the gasses released when meat spoils. The pace of the spoiling process is dependent on its surroundings this is why the DHT11 was chosen to measure the humidity and the temperature.

<u>Temperature and Humidity sensor (DHT11)</u>
The DHT11 measures relative humidity. Relative humidity is the amount of water vapor in air vs. the saturation point of water vapor in air. At the saturation point, water vapor starts to condense and accumulate on surfaces forming dew. The saturation point changes when the air temperature changes. Cold air can hold less water vapor before it becomes saturated, and hot air can hold more water vapor before it becomes saturated. Relative humidity is expressed as a percentage. At 100% RH, condensation occurs, and at 0% RH, the air is completely dry.

<u>How the DHT11 measures</u>
The humidity is determined by measuring the electrical resistance between the two electrodes, it detects the water vapor. Inside the humidity sensor there is a substrate with the two electrodes attached to the surface. When the substrate absorbs the water vapor it releases ions. This increases the conductivity between the two electrodes. Because of this the resistance is comparable to the relative humidity. Higher relative humidity decreases the resistance between the two electrodes, while the lower relative humidity increases the resistance between the two electrodes.

The DHT11 uses an NTC temperature sensor (thermistor), Their resistance responds to temperature changes. Here the resistance decreases when the temperature increases.
The ranges and accuracy of the DHT11:
- Humidity Range: 20-90% RH
- Humidity Accuracy: ±5% RH
- Temperature Range: 0-50 °C
- Temperature Accuracy: ±2% °C

Gas sensor (TGS 822)
The TGS 822 measures the concentration of gasses. The gasses that it can measure are ethanol, carbon monoxide, ammonia, sulphur dioxide, alcohol, gasoline, methane, acetone.
The sensing element of the TGS 822 gas sensor is a tin dioxide ($SnO2$) semiconductor that has low conductivity in clean air. In the presence of a detectable gas, the sensor's conductivity increases depending on the gas concentration in the air. An electrical circuit converts the change in conductivity to an output signal that corresponds to the gas concentration. Due to the change in conductivity the resistance of the gas sensor also changes.

Temperature sensor
We used a temperature sensor to measure the temperature. Temperature is one of the biggest factors in how fast your food will expire. If you leave food, especially meat, out to long in room temperature, it can cause the growth of bacteria to levels which are dangerous. Temperatures between 4 *C and 60 *C are called the Danger Zone. Between these temperatures' bacteria grows the fastest. It is not safe anymore to eat your food if you have left it 2 hours outside the fridge. If your fridge temperature isn't exactly right, your food will go faster to waste. We build in our temperature sensor to be aware of your temperature all the time. You will be able to check if you left your meat outside of the fridge or if your fridge isn't exactly working right.[1]

Humidity sensor
We also added a humidity sensor to our project. The amount of humidity your food is exposed to causes it to stay fresh or how fast it expires. This mostly influences vegetables. Tomatoes lose their taste if the keep them in the refrigerator with a too low humidity level.[2] A low humidity level causes the meat to dry out and the structure to become stiffer and fixed and it won't be tender anymore. It will loses his flavors and taste. A high humidity level creates a better environment for bacteria to grow. Controlling the humidity levels in your refrigerator will result in longer lasting and better tasting food.

The basic Measuring Circuit.

[1] https://www.fsis.usda.gov/wps/portal/fsis/topics/food-safety-education/get-answers/food-safety-fact-sheets/safe-food-handling/how-temperatures-affect-food/ct_index

[2] https://www.canr.msu.edu/news/refrigerator_humidity_effects_on_produce_quality

The difference in sensor resistance is measured as a change in the voltage across the load resistor. A Measured output voltage can be converted into sensor resistance with the nest formula.

$R_S=(V_c/V_{RL} -1)\times R_L$

R_S = Sensor resistance
R_L = Load resistance
V_C = Circuit voltage
V_{RL} = Output voltage/signal

The previous formula can be rewritten to the following equation:
$V_{RL}= V_c/(R_S/R_L +1)$

So, if the concentration of the vapor increases, RS decreases. And in the previous formula can be concluded that if Rs decreases, VRL increases.

The ranges of the TGS 822:
- Temperature Range: -10 - 40 °C
- Detection Range 50 – 5000 ppm

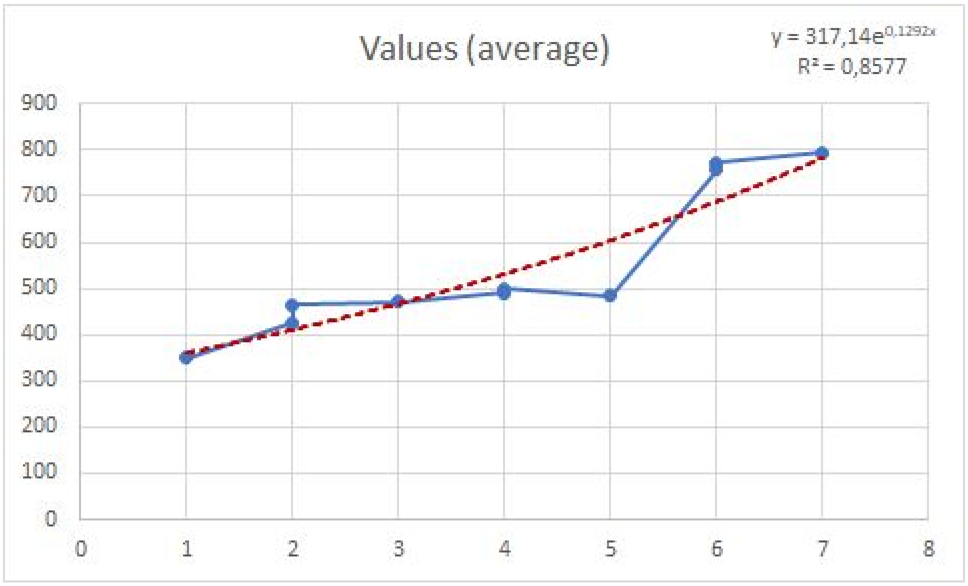
<u>Prototype</u>
For making the prototype of the MeatMe box a Tupperware box was used as a basis. We made a hole in the center of the lid. With wires through the hole the sensors were placed on the inside of the lid. To make it airtight, the wires were split in sections and the hole was filled with epoxy glue. On top of the lid the Arduino and the breadboard were placed. In the breadboard three LEDs were installed as indication. Green meaning ‘okay to eat’, orange meaning ‘eat it soon’ and read meaning ‘spoiled, throw it out’.

Laura Schep, Mats van Braam, Julia van der Geest, Maud van der Hall,
Ruben Koole, Bart Blom en Famke van den Boom

4. Short description of results/observations/data gathered/graphs (1,000 words and/or 3-4 figures/graphs).

Measurements with sensor

We had a certain method of working to take measurements with this sensor. First, we took 100 grams of ground meat in the container and started with measuring with the same measuring distance between the meat and sensor.



Graph 1: *Measurements with sensor*

In the graph above you can see the results from our measurements using the TSG 800 sensor. Every day the concentration of gasses was measured at least once. Using the measurements, an average per measurement per day was calculated. This is why on some days there are two dots. These values were plotted in a graph, with a trend line to show the average concentration of these gases.

Measurements without sensor

We bought ground beef with an expiration date 7 days after purchasing. This date is usually used as an indication for when the meat is no longer safe to eat. After the sell by date the meat started to smell, 2 days later the meat started to turn grey on the outside. As previously stated, when the meat fully turns gray, it may be inedible.

A second box was put close to a heater to speed up the process. However, because we only had one gas sensor, it was not possible to measure these values with the sensors. After the 3rd day the color already started to change and on the 4th day, the meat was spoiled. In order to get conclusions out of this test, we should have measured with a sensor as well.

Conclusion

In conclusion, on day 5-6 after purchasing the meat from the first box, the color turned from red to brown, the smell became worse and the humidity increased a lot. This aligns with the findings from the measurements with the sensor. At day 5-6 you see a big increase in graph 1.

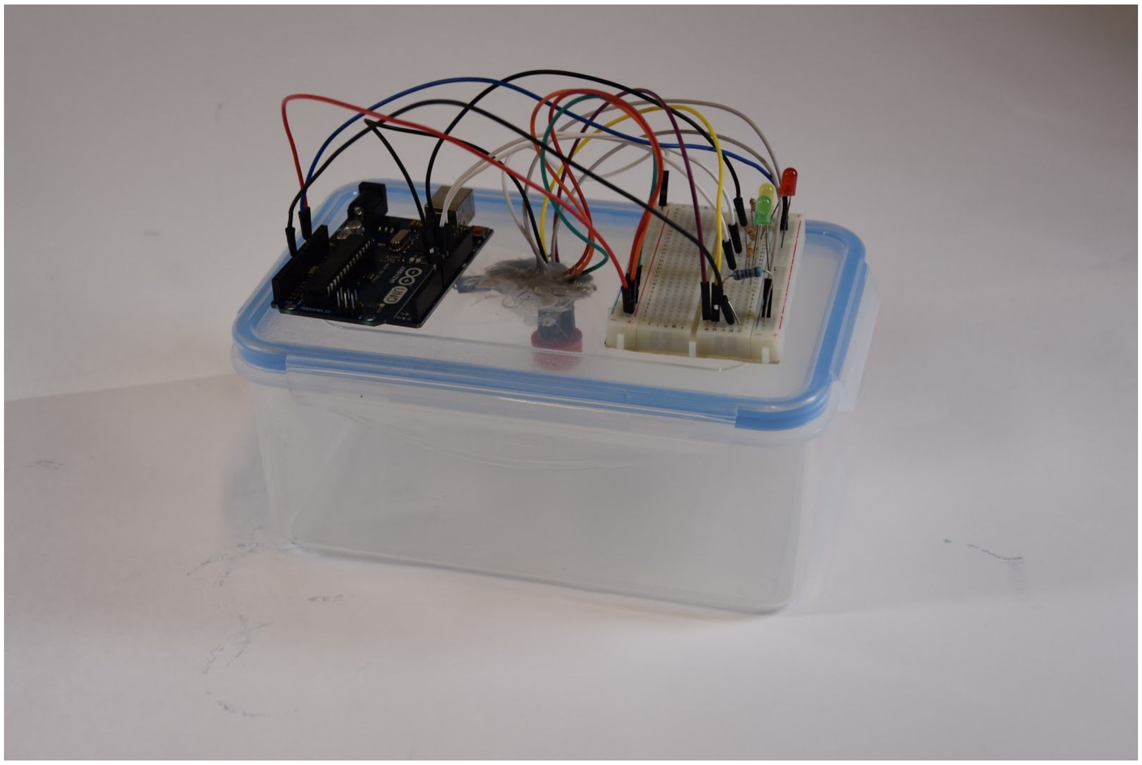
Using the results from the sensor and the results from the measurements without the sensor we decided on the following values for fresh, eat quickly or spoiled meat:

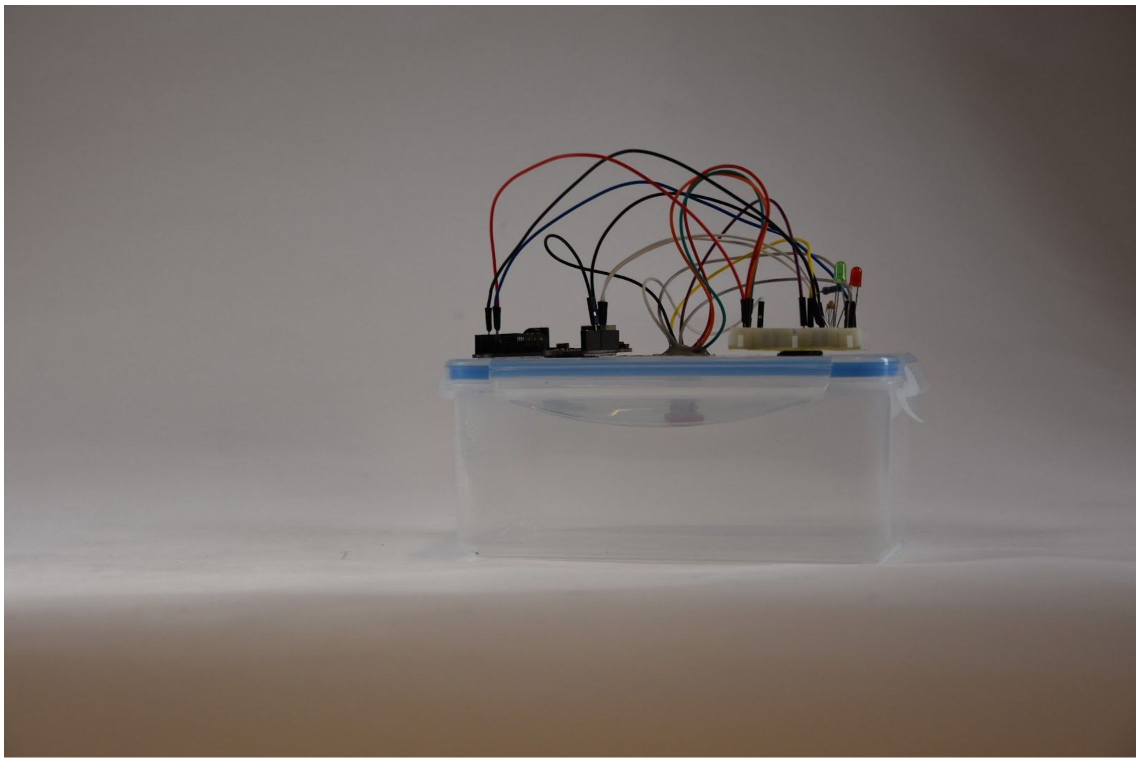
- <475 fresh, so green light
- 475-550 eat quickly, use your own commonsense to decide, so yellow/orange light
- >550 spoiled, so red light

Team HUMMUS

Laura Schep, Mats van Braam, Julia van der Geest, Maud van der Hall,
Ruben Koole, Bart Blom en Famke van den Boom

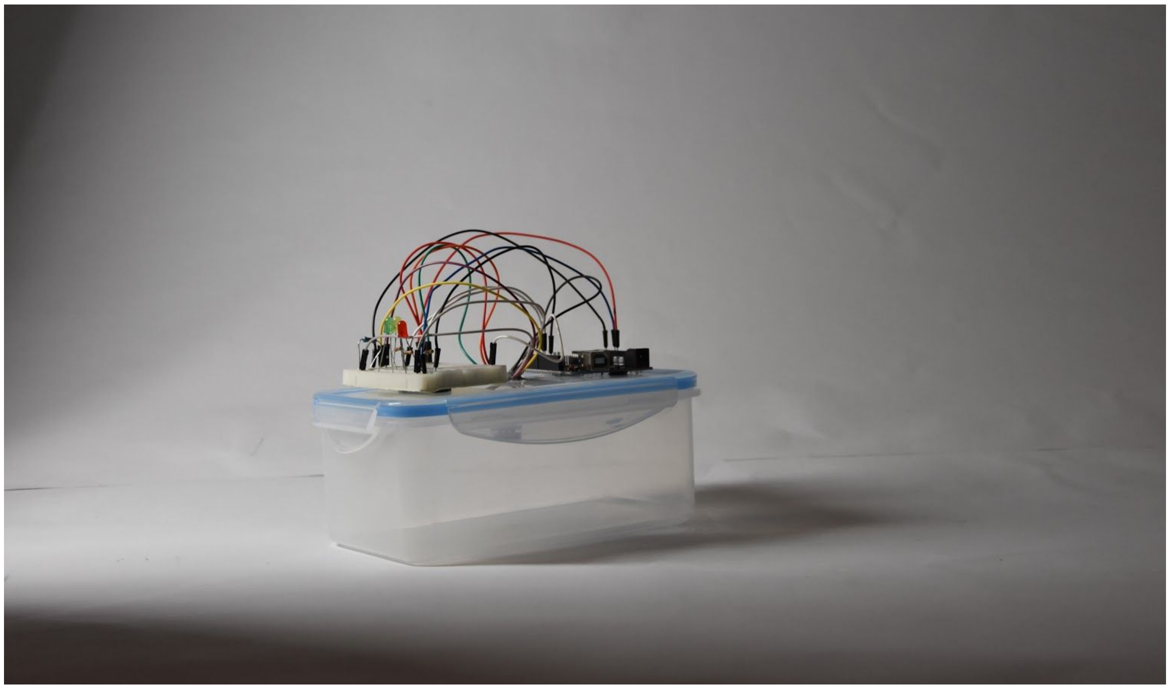
5. Up to 5 photos of your prototype





Team HUMMUS

Laura Schep, Mats van Braam, Julia van der Geest, Maud van der Hall,
Ruben Koole, Bart Blom en Famke van den Boom



6. Up to 2-minute video explaining what the prototype does.

https://youtu.be/PyYGaQBZVmg

7. Full working code of the project.

```
//--- Smart Environment Project ---
//------ Creative Technology ------
//------------- 2020 --------------
// ----------- MeatMe -------------

// Special thanks to Circuit Basics and the ElectronFun for helping getting along
with the code for the sensors
// Circuit Basics:
http://www.circuitbasics.com/how-to-set-up-the-dht11-humidity-sensor-on-an-arduino/
// Electronfun.com: http://electronfun.com/project_2.php

// Including the libary for the DHT11 Sensor and defining at on pin 7
#include <dht.h>
dht DHT;
#define DHT11_PIN 7

// Digital pin 8 will be called 'Red'
int Red = 8;
// Digital pin 9 will be called 'Yellow'
int Yellow = 9;
// Digital pin 10 will be called 'Green'
int Green = 10;
// Analog pin 0 will be called 'Gassensor'
int Gassensor = A0;
// Set the GassensorValue to 0
int GassensorValue = 0;
// interger for a time loop
int lastMillis = 0;


// The setup routine runs once when you press reset
void setup() {
  // Initialize the digital pins as an output
  pinMode(Red, OUTPUT);
  pinMode(Yellow, OUTPUT);
  pinMode(Green, OUTPUT);
  // Initialize serial communication at 9600 bits per second
  Serial.begin(9600);
}
```

```
// The loop routine runs over and over again forever
void loop() {
  if (millis() > lastMillis + 1000) {
    lastMillis = millis();
    // Read and print out Tempsensor
    int chk = DHT.read11(DHT11_PIN);
    Serial.print('T');
    Serial.println((int)DHT.temperature);
    Serial.print('H');
    Serial.println((int)DHT.humidity);

    // Read the input on analog pin 0 (named 'Gassensor')
    GassensorValue = analogRead(Gassensor);
    // Print out the value you read
    Serial.print('S');
    Serial.println(GassensorValue, DEC);
  }


  // If sensorValue is lower than 475 the green led will turn on
  if (GassensorValue < 475) {
    digitalWrite(Green, HIGH);
  }
  else {
    // Deactivate digital output- the LED will not light up
    digitalWrite(Green, LOW);
  }

  // If sensorValue is between 475 and 550 the orange led will turn on
  if (GassensorValue >= 475 && GassensorValue <=550) {
    digitalWrite(Yellow, HIGH);
  }
  else {
    // Deactivate digital output - the LED will not light up
    digitalWrite(Yellow, LOW);
  }

  // If sensorValue is greater than 550 the red led will turn on
  if (GassensorValue > 550) {
    digitalWrite(Red, HIGH);
  }
  else {
    // Deactivate digital output - the LEDs will not light up
    digitalWrite(Red, LOW);
  }
}
```

Procesing:

```
/*
   This program gives an idea of how an app, that supports the MeatMe, could look.
In the App we display multiple things: the temperature, humidity and value measured
by the Figaro TGS 822. The temperature and the humidity are displayed next to icons
to make the app more clear for the user. The value measured by the sensor is
projected using a meatbar. This meatBar will fill itself according to the
"condition" of the meat; the higher the value measured by the sensor, the bigger
part of the meatbar will be coloured. Next to that, the colour of the meatbar will
also change according to the value measured be the sensor. If the meat is edible,
the bar will be green, if the meat is close to spoil, the bar will be orange, and
last, if the meat is spoiled, the bar will be red.
   Made by: Team Hummus
 */

import processing.serial.*;

Serial port;
App app;

int AMOUNT = 3; // The amount of values (sensors) the Arduino sends.
String buff = "";
char header[] = {'T', 'H', 'S'};
int value[] = new int[AMOUNT];
int diffValue[] = new int[AMOUNT];
int NEWLINE = 10;

void setup() {
  size(1500, 900);
  app = new App();
  for (int i = 0; i<Serial.list ().length; i++) {
      print("[" + i + "] ");
      println(Serial.list()[i]);
  }
  port = new Serial(this, Serial.list()[0], 9600);
}

void draw() {
  while (port.available() > 0) {
      serialEvent(port.read()); // read data
  }
  app.play(); //display the app.
}

// This function collects the input from the Arduino.
void serialEvent(int serial)
{
  try {      // try-catch because of transmission errors
      if (serial != NEWLINE) {
      buff += char(serial);
      } else {
      // The first character tells us which axis this value is for
      char c = buff.charAt(0);
      // Remove it from the string
      buff = buff.substring(1);
      // Discard the carriage return at the end of the buffer
      buff = buff.substring(0, buff.length()-1);
      // Parse the String into an integer
      for (int z=0; z<AMOUNT; z++) {
      if (c == header[z]) {
      value[z] = Integer.parseInt(buff);
      }
      }
      buff = "";          // Clear the value of "buff"
      }
  }
  catch(Exception e) {
      println("No valid data");
  }
}
```

```
// This Class handles the app, so makes sure the pictures and values are displayed.
class App {
  color red = color(255, 73, 69);
  color orange = color(246, 163, 65);
  color green = color(111, 187, 124);
  color black = color(0);
  color white = color (255);
  PImage background; //The background of the app.
  PImage thermometer; //Picture of a thermometer to indicate the temperature.
  PImage water; //Picture of an waterdrup to indicate the humidity.
  PImage meatLevel; //Picture of the text "(M)eat Level"
  PImage meatMe; //Picture of the text "MeatMe"
  PImage logo; //Picture of our logo.
  int temperature = 0; //The temperature measured in our prototype.
  int humidity = 0;  //The humidity measured in our prototype.
  int sensor = 0; //The value received from the sensor.
  int heightBox = 76; //Height of the meatLevel bar.
  int widthBox = 876; //Width of the meatLevel bar.
  int topCornerX = 312; //X coordinate of the left top corner of the meatLevel bar.
  int topCornerY = 287; //Y coordinate of the left top corner of the meatLevel bar.

  App() {
       addImages();
       textSize(220);
       textAlign(CENTER);
       rectMode(CENTER);
       noStroke();
  }

  // This functions draws all the elements that are displayed in the app.
  void play() {
       try {
       temperature = value[0];
       humidity = value[1];
       sensor = value[2];
       }
       catch(Exception e) {
       println("No valid data");
       }
       fill(0);
       image(background, 0, 0);
       image(thermometer, 0, 0);
       image(water, 0, 0);
       image(meatLevel, 0, 0);
       image(meatMe, 0, 0);
       image(logo, 0, 0);
       text(humidity, 1180, 730);
       meatLevel(); //Displays the meatLevel bar
       displayTemperature(); //Displays the temperature.
  }

  // This function loads all the images we use in our program. Next to that all the
images are resized to the right width and height.
  void addImages() {
       background = loadImage("background.jpg");
       background.resize(1500, 900);
       thermometer = loadImage("thermometer1.png");
       thermometer.resize(1500, 900);
       water = loadImage("water1.png");
```

```
      water.resize(1500, 900);
      meatLevel = loadImage("meatlevel.png");
      meatLevel.resize(1500, 900);
      meatMe = loadImage("meatMe.png");
      meatMe.resize(1500, 900);
      logo = loadImage("logo.png");
      logo.resize(1500, 900);
  }

  /* This function fills the meatLevel according to the value measured by the
sensor. If the value is lower than 475, the meatLevel will be colored green. If the
   value is bigger the 550, the meatLevel will be colored red. If the value is in
between, then the meatLevel will be orange. */
  void meatLevel() {
      rectMode(CENTER);
      fill(black);
      rect(750, 325, 900, 101); // Draws the black outline of the meatlevel.
      fill(white);
      rectMode(CORNER);
      rect(topCornerX, topCornerY, widthBox, heightBox); // Draws the white inside
of the meatlevel.
      float widthBar = map(sensor, 0, 1000, 0, widthBox); //Calculate width of the
meatLevel bar.
      if (sensor < 475) {
      fill(green); //The meatLevel will be green.
      } else {
      if (sensor > 550) {
      fill(red); //The meatLevel will be red.
      } else {
      fill(orange); //The meatLevel will be orange.
      }
      }
      rect(topCornerX, topCornerY, widthBar, heightBox); // Draws the indication
part of the meatLevel
  }

  /* This function displays the temperature. The temperature will be displayed
black if temperature is okay (below 7 degrees), otherwise the temperature
  will be displayed in red. */
  void displayTemperature() {
      fill(black);
      if (temperature > 7) { // Checks if the temperature is bigger than 7
degrees.
      fill(red); //The temperature will be red.
      }
      text(temperature, 380, 730); //Display the temperature
```

```
    }
}
```