K.J. Somaiya College of Engineering, Vidyavihar, Mumbai-77.
(Autonomous College affiliated to University of Mumbai)

IoT Based Smart Pet Food Dispenser

by

Pranav Kalambe - 1913023 (A2)

Mrunmayee More - 1913030 (A2)

Navya Jain- 1913031 (A2)

Amogh Pai - 1913034 (A2)

Internal Assessment-II

Departmental Elective- Internet of Things (2UTE616)

TY B.Tech (Semester-VI)

January to May 2022 Term

K.J. Somaiya College of Engineering, Vidyavihar, Mumbai-77.
(Autonomous College affiliated to University of Mumbai)

Certificate

This is to certify that the report on "**IoT Based Smart Pet Food Dispenser** " is bona fide record of the work done by

1. Pranav Kalambe 1913023

2. Mrunmayee More 1913030

3.Navya Jain 1913031

4. Amogh Pai 1913034

in the academic year **2021-22**, **Department of Electronics and Telecommunication Engineering** for the course "**Internet of Things (2UTE616)"**

_____

Signature of Faculty

Date:

Place: Mumbai-77

❖ Title of application

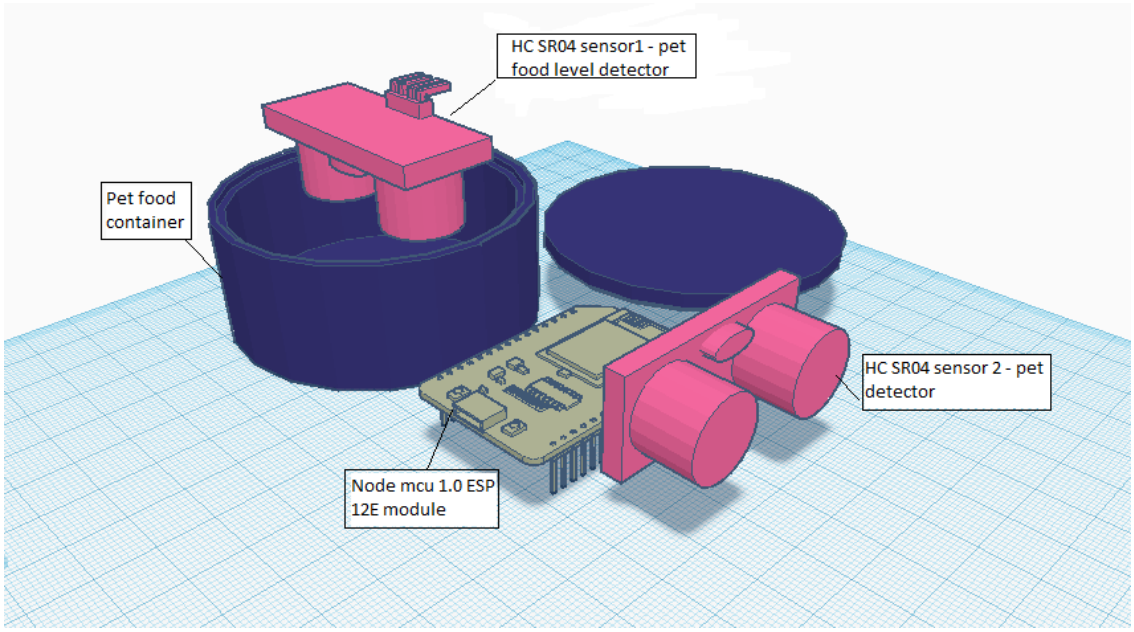
'IoT Based Smart Pet Food Dispenser'

❖ Brief introduction of the application

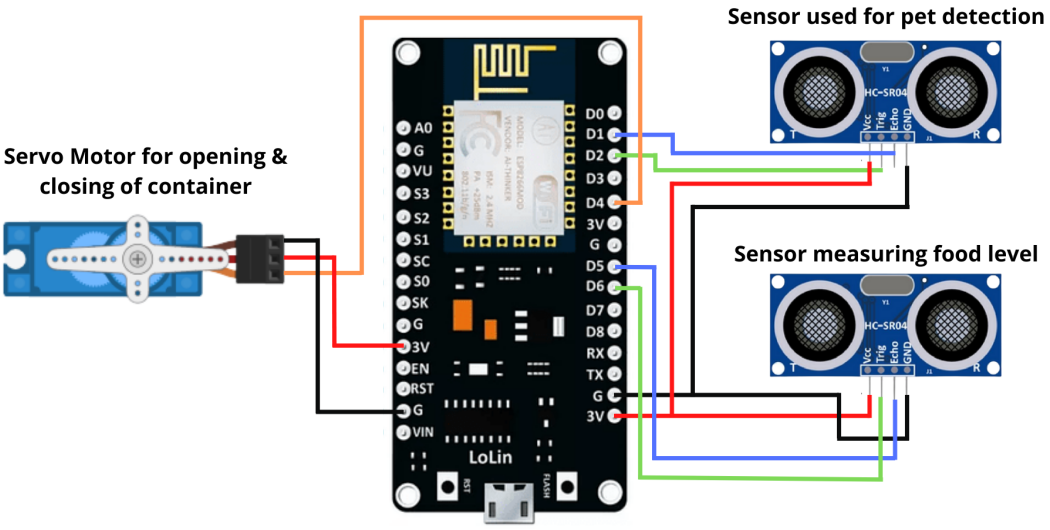
Pets such as cats and dogs are indeed an integral part of our families. But there is a big issue of feeding these pets whenever the owner needs to be away from the house and cannot take his/her pet. The objective of this project is to develop automatic pet feeding with Internet of Things (IoT). It will be very useful whenever a pet owner is outside the residence and/or unable to feed his/her pets normally.

This project is a simple microcontroller-centric one with 2 ultrasonic sensors(HC-SR04) ; one for detecting the presence of the pet in front of the feeder and the second one for measuring the quantity of food left in the feeder container, post every feed. This information will be regularly supplied to the owner by integrating a IOT based cloud platform, UbiDots, which will be responsible for updating the feed-time to the pets and the quantity of food left in the container. The platform will also be sending out regular emails to the owner, alerting him in cases of deficiency of pet food in the container.

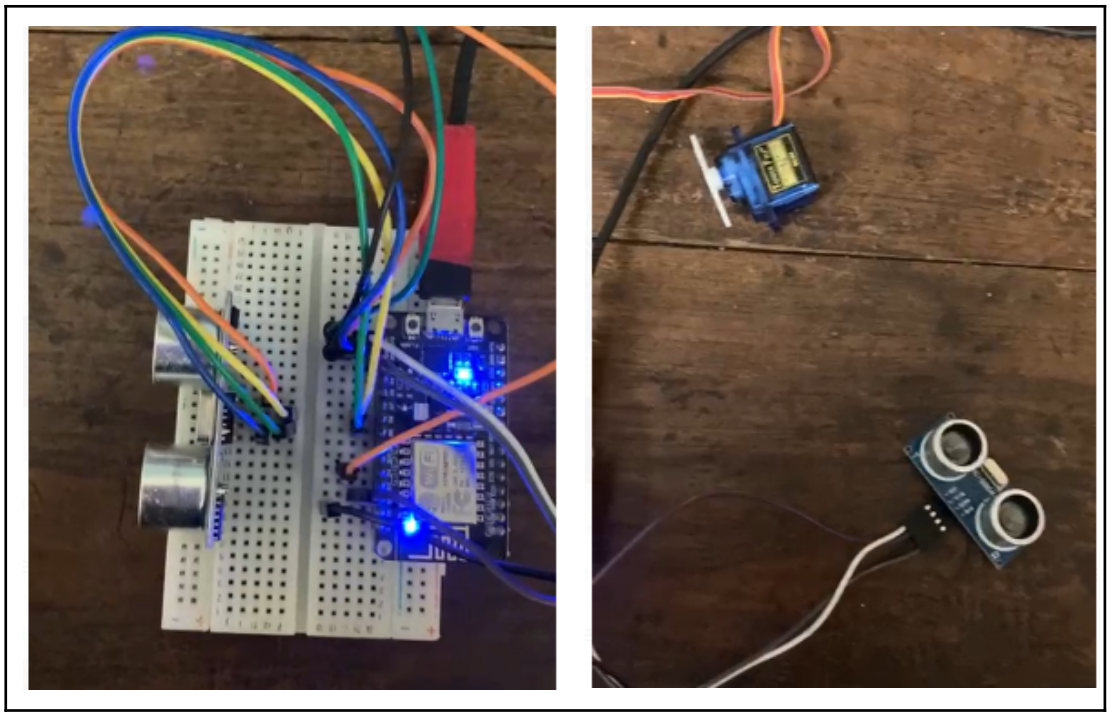
❖ Circuit diagram/block diagram along with detailed working of the circuit



Circuit diagram:



Photos of the implemented circuit:



Pin description of circuit and connections:

Vcc - is the power supply for HC-SR04 Ultrasonic distance sensor which we connect to the 5V pin on the Arduino.	**Trigger - pin is used to trigger the ultrasonic sound pulses.**
Gnd - connected to mode-mcu ground	**Echo - pin produces a pulse when the reflected signal is received. The length of the pulse is proportional to the time it took for the transmitted signal to be detected**

HC SR 04 - Sensor 1 - Pet food level detector	**Node mcu 1.0 on-board pins**
Vcc	**node mcu Vcc**
Gnd	**node mcu gnd**
Trigger	**D6**
Echo	**D5**
HC SR 04 - Sensor 2 - Pet presence detector	**Node mcu 1.0 on-board pins**
Vcc	**node mcu Vcc**
Gnd	**node mcu gnd**
Trigger	**D2**
Echo	**D1**

Code used on Arduino IDE:

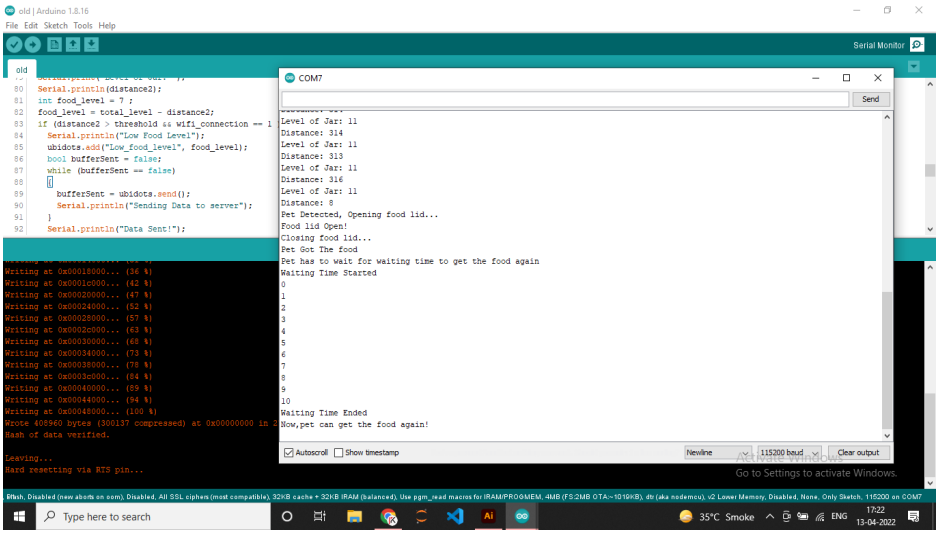
```
const int trigPin1 = 4; // d2
const int echoPin1 = 5; // d1 - for the pet detection
// defines variables
long duration1;
int distance1;
int i;
int total_level = 100;
int threshold = 50;

#include "Ubidots.h"

const char* UBIDOTS_TOKEN = "BBFF-OZAd01PVVJswycalTXy6Qp4oBDoIbB";   // Put
here your Ubidots TOKEN
//BBFF-OZAd01PVVJswycalTXy6Qp4oBDoIbB
const char* WIFI_SSID = "pk";      // Put here your Wi-Fi SSID
const char* WIFI_PASS = "pkpkpkpkpk";      // Put here your Wi-Fi password
Ubidots ubidots(UBIDOTS_TOKEN, UBI_HTTP);
int wifi_connection = 0;


const int trigPin2 = 12; //d6
const int echoPin2 = 14; //d5 - for level detection

long duration2;
int distance2;

//int min_food_level = 50;
int openTime = 3; // input in seconds
int waitingTime = 10; // input in seconds
int pet_detected_distance = 20; // input in centimeter
int open_angle = 150; // input in degree

#include <Servo.h>

Servo myservo;  // create servo object to control a servo
int pos = 0;    // variable to store the servo position


void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(trigPin1, OUTPUT); // Sets the trigPin222 as an Output
  pinMode(echoPin1, INPUT); // Sets the echoPin as an Input

  pinMode(trigPin2, OUTPUT); // Sets the trigPin222 as an Output
  pinMode(echoPin2, INPUT); // Sets the echoPin as an Input

  Serial.begin(115200);
  Serial.println("Initializing Serial Communication with baud rate - 115200");
  Serial.println("Trying to connect to wifi");

  for (int a = 0 ; a < 5 ; a++) {
    wifi_connection = ubidots.wifiConnect(WIFI_SSID, WIFI_PASS);
    if (wifi_connection == 1) {
      Serial.println("Wifi Connected Successfully");
      Serial.println(wifi_connection);
```

```
      a = 6;
    }
    else {
          Serial.println("Wifi not connected try reseting / turn on wifi
hotspot");
    }
  }
  //Serial.begin(9600); // Starts the serial communication
  myservo.attach(2); //D4
}
void loop() {

  // Clears the trigPin
  digitalWrite(trigPin2, LOW);
  delayMicroseconds(2);
  // Sets the trigPin1 on HIGH state for 10 micro seconds
  digitalWrite(trigPin2, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin2, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration2 = pulseIn(echoPin2, HIGH);
  // Calculating the distance
  distance2 = duration2 * 0.034 / 2;
  // Prints the distance on the Serial Monitor
  Serial.print("Level of Jar: ");
  Serial.println(distance2);
  int food_level = 7 ;
  food_level = total_level - distance2;
  if (distance2 > threshold && wifi_connection == 1 ) {
    Serial.println("Low Food Level");
    ubidots.add("Low_food_level", food_level);
    bool bufferSent = false;
    while (bufferSent == false)
    {
      bufferSent = ubidots.send();
      Serial.println("Sending Data to server");
    }
    Serial.println("Data Sent!");
    Serial.println("Server will send a mail to user");
    }

    // Clears the trigPin
    digitalWrite(trigPin1, LOW);
    delayMicroseconds(2);
    // Sets the trigPin1 on HIGH state for 10 micro seconds
    digitalWrite(trigPin1, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin1, LOW);
    // Reads the echoPin, returns the sound wave travel time in microseconds
    duration1 = pulseIn(echoPin1, HIGH);
    // Calculating the distance
    distance1 = duration1 * 0.034 / 2;
    // Prints the distance on the Serial Monitor
    Serial.print("Distance: ");
    Serial.println(distance1);
    digitalWrite(LED_BUILTIN, HIGH);
```

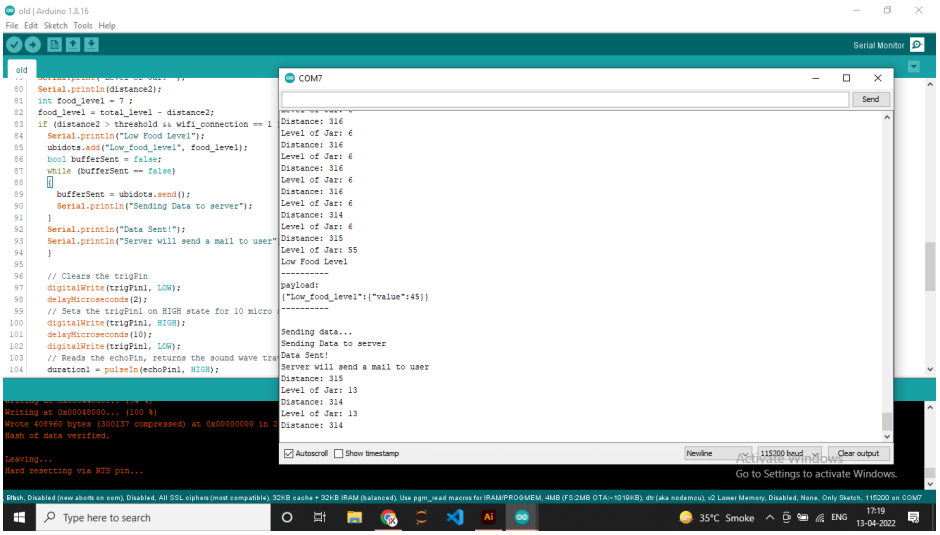
```
    if (distance1 < pet_detected_distance) {
      digitalWrite(LED_BUILTIN, LOW);
      Serial.println("Pet Detected, Opening food lid...");
       for (pos = 0; pos <= open_angle; pos += 1) { // goes from 0 degrees to
180 degrees
        // in steps of 1 degree
         myservo.write(pos);                // tell servo to go to position in
variable 'pos'
         delay(5);                       // waits 15 ms for the servo to reach
the position
      }
      Serial.println("Food lid Open!");
      delay(openTime * 1000);
      Serial.println("Closing food lid...");
      for (pos = open_angle; pos >= 0; pos -= 1) { // goes from 180 degrees to
0 degrees
         myservo.write(pos);                // tell servo to go to position in
variable 'pos'
        delay(15);                       // waits 15 ms for the servo to reach
the position
      }
      Serial.println("Pet Got The food");
      delay(1000);
         Serial.println("Pet has to wait for waiting time to get the food
again");
      delay(1000);
      Serial.println("Waiting Time Started");
      for (i = 0 ; i <= waitingTime; i += 1) {
        Serial.println(i);
        delay(1000);
      }
      Serial.println("Waiting Time Ended");
      Serial.println("Now,pet can get the food again!");
      delay(2000);
    }
    delay(1000);
  }
```

Output obtained on serial monitor, ubidots:

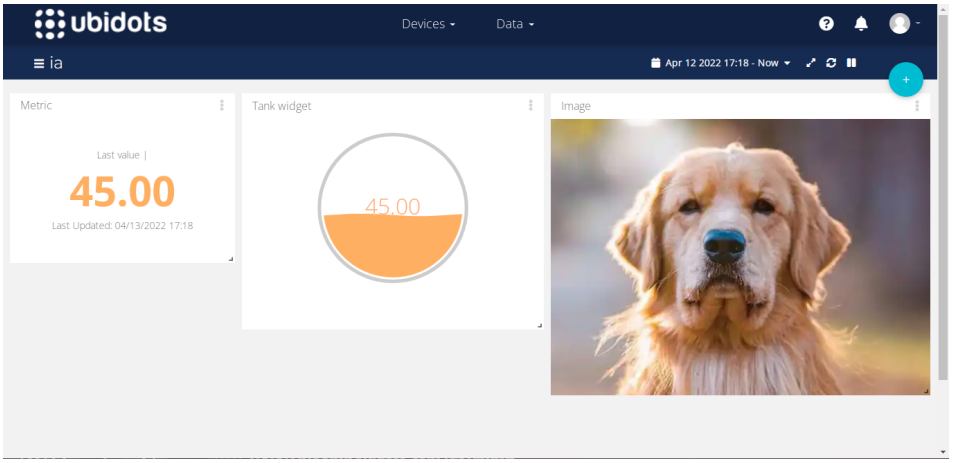
1) Detection for pet presence:

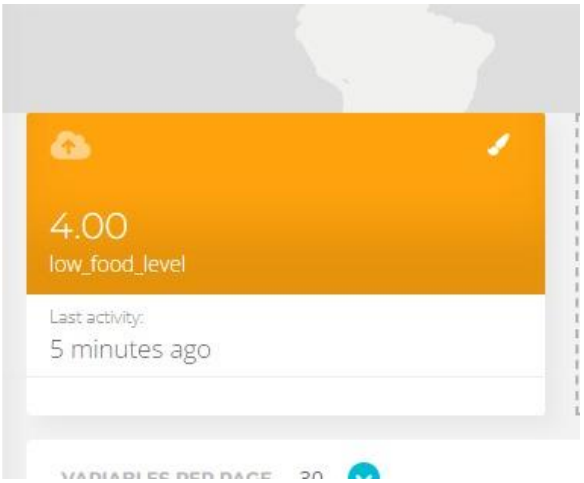


2) Measuring level of remaining pet food by calculating the distance:



3) Detection of low pet food level accompanied by cloud updation and sent mail to user:

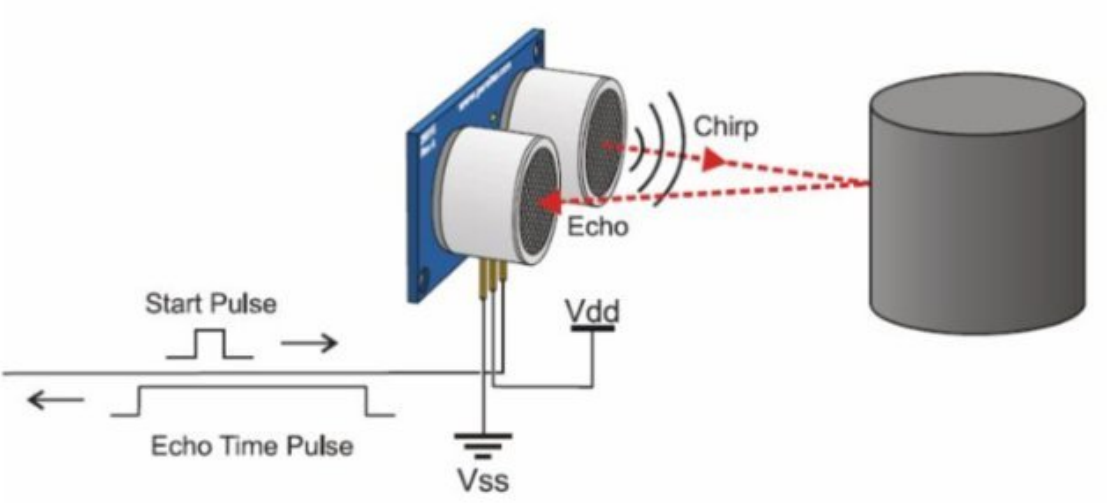




Working of HC SR04 sensor:

HC-SR04 Ultrasonic distance sensor consists of two ultrasonic transducers.

- **The trigger pin of the sensor acts as a transmitter which converts electrical signals into 40 KHz ultrasonic sound pulses.**
- **The receiver listens for the transmitted pulses, which will be reflected back to the sensor upon collision.**
- **If it receives them, the Echo pin produces an output pulse whose width can be used to determine the distance the pulse travelled.**
- **In our project, the detection limit for the presence of the pet is set as 20cm**



- **Details of parameters/notations/equations with minimum and maximum values for each parameter (wherever possible)**

Suppose we have an object in front of the sensor at an unknown distance and we receive a pulse of width 500 µS on the Echo pin. Since the speed of sound is 0.034 cm/sec,
Distance = Speed x Time = 0.034 cm/µs x 500 µs. but since the time traveled includes the transmitted and received pulse,
Distance = (0.034 cm/µs x 500 µs) / 2 = 8.5 cm

Operating parameters for the UV sensor:

Operating Voltage DC - 5V	**Operating Current - 15mA**
Operating Frequency - 40KHz	**Max Range - 4m**
Min Range - 2cm	**Ranging accuracy - 3mm**
Measuring angle - 15 degree	**trigger input signal - 10µS TTL pulse**

- **Details of the software/version used for simulation and simulated circuit picture and steps for simulation.**

1) **Arduino IDE**

It is an open-source arduino software for developing IoT projects because they contain a large set of common input and output functions. The program is written in C and C++ language. It is used to write and upload programs to Arduino compatible boards and other vendor development boards such as ESP32. This software is very widely used among programmers because of its open source.

2) **UbiBot**

A privatized loT platform which is based on the UbiBot's mature public lot data platform. It supports enterprise intranet management with high performance, high stability and high scalability software systems. The whole series of UbiBot hardware products can be privatized to access with UbiBot OPP.

- **Description of parameters which can be changed with circuit components values (e.g. certain components affect the output or any other parameter of circuit then give relation and explain the relation)**

  Pet food nutritional value and other food parameters can be incorporated to provide diet monitoring. Temperature and bio-sensors to check the freshness of the pet food and alert system to alert the owner if the food gets stale.

- **Record the video of IA-II application explaining complete working and upload on the following link also upload report on the same link with your group number.**

LINK FOR THE VIDEO

- **Application of the project/circuit**

This project can be used on domestic levels for regular servings of pet food. This prototype idea can be introduced into commercial granaries to supply quantities of food grains and keep regular track of the available quantities in storage units. Codes and circuit diagrams for this project can be used from the toll tax Arduino project or from Arduino piggy bank.

- **If given a chance what modifications/variations/improvement will you do to a given task? Suggest**

Improvement in the amount of information provided to the owner on the cloud platform, backed with mobile application alerts and notifications. Feed-capping system to avoid overfeeding to the pets. Mobile applications that would provide analysis to the owner in terms of the pet's feeding patterns and nutritional intake, to align with its diet chart. Introduction to medicine dispenser along with food dispenser, to ensure the pet's overall care.

Contribution of each member in group

Roll Number	Name	Work Done for project
1913023	Pranav Kalambe	Technical design, implementation and testing of the project
1913030	Mrunmayee More	
1913031	Navya Jain	
1913034	Amogh Pai	Project Report