IoT Based Automatic Pet Feeder

Submitted in partial fulfillment of the requirements

of the Mini-Project for Second Year of

Bachelors of Engineering

by

Siddharth Pallar (211P016)

Ameen Khan (211P025)

Aditya Vishwakarma (211P035)

Guide:

PROF. Manila Gupta

Department of Computer Engineering

Rizvi College of Engineering

University of Mumbai

2022 - 2023

CERTIFICATE

This is to certify that the mini-project entitled **"IoT Based Automatic Pet Feeder"** is a bona-fide work of **"Siddharth Pallar (211P016), Ameen Khan (211P025) & Aditya Vishwakarma (211P035) "** submitted to the University of Mumbai in partial fulfillment of the requirement for the Mini-Project for Second of the Bachelor of Engineering in **"Computer Engineering"**.

PROF. MANILA GUPTA

Guide

Prof. Shiburaj Pappu	Dr. Varsha Shah
Head of Department	**Principal**

Declaration

We declare that this written submission represents our ideas in our own words and where others ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Siddharth Pallar (211P016)

Ameen Khan (211P011)

Aditya Vishwakarma (211P033)

Date:

ABSTRACT

An IoT-based automatic pet feeder is a system that allows pet owners to automate their pets' feeding schedules using internet-connected devices. The system is designed to provide convenience for pet owners who may have unpredictable schedules or who are frequently away from home. The system consists of a feeding device, a microcontroller and sensors. The microcontroller controls the feeding mechanism and ensures that the right amount of food is dispensed at the right time. The sensors monitor the pet's feeding behaviour, and the remote control provides real-time information on the pet's feeding habits and allows remote control of the feeding process. The automatic pet feeder is an efficient way to ensure that pets are fed regularly and that they receive the correct amount of food, promoting their health and well-being.

Keywords: "Automatic", "Pet Feeder", "Feeding Mechanism-Remote Control", "Pet Health", "Convenient".

Index

Chapter I

Introduction

Pets are beloved members of many households, providing companionship and joy to their owners. However, pet owners often face challenges in ensuring their pets receive proper care and attention, especially when they are away from home for extended periods or have busy schedules.

Feeding pets at regular intervals is a crucial aspect of pet care that requires consistency and accuracy. Traditional feeding methods can be unreliable, especially when owners are away from home. To address this issue, an IoT-based automatic pet feeder has been developed that provides pet owners with an efficient and convenient way to feed their pets remotely using an internet-connected device.

This system consists of a pet feeder, a microcontroller and sensors. The microcontroller controls the feeding mechanism, and the sensors monitor the pet's feeding behavior. The remote control provides real-time information on the pet's feeding habits and enables remote control of the feeding process.

This technology promotes pet health by ensuring pets are fed on time and receiving the correct amount of food, regardless of the owner's location. In this article, we will explore the features and benefits of an IoT-based automatic pet feeder and its potential impact on pet care.

Chapter II

Review Literature

There are several existing IoT automatic pet feeders in the market, and many researchers and developers have explored this area. Here are some examples of other people's work in the field of IoT automatic pet feeders which we have referred to :

Paper 1: "Smart Pet Feeder with Facial Recognition": In 2017, a team of researchers from the University of Oviedo in Spain developed a smart pet feeder with facial recognition capabilities. The feeder uses computer vision algorithms to recognize the pet's face and dispense food only when the correct pet is in front of the feeder. The feeder also tracks the pet's eating habits and sends notifications to the owner's smartphone.

Paper 2: "Automatic Pet Feeder with Voice Recognition": In 2018, a team of researchers from the University of Tokyo developed an automatic pet feeder with voice recognition capabilities. The feeder uses a microphone to recognize the owner's voice and dispense food accordingly. The feeder can also be controlled remotely via a smartphone app.

Paper 3: IoT Automatic Pet Feeder with Machine Learning: In 2019, a team of researchers from the University of Portsmouth in the UK developed an IoT automatic pet feeder with machine learning capabilities. The feeder uses machine learning algorithms to learn the pet's eating habits and dispense food accordingly. The feeder can also be controlled remotely via a smartphone app.

Paper 4: Smart Pet Feeder with Health Monitoring: In 2020, a company called Petkit launched a smart pet feeder with health monitoring capabilities. The feeder has a built-in scale that measures the pet's weight, and it can also track the pet's eating habits and send notifications to the owner's smartphone if there are any abnormalities.

These examples show that IoT automatic pet feeders are constantly evolving, and new features and capabilities are being added to improve the feeding experience for pets and their owners.

Chapter III

Report on the Present Investigation

3.1 Proposed plan of work

3.1.1 Define project requirements and goals: The first step is to identify the project requirements and goals. This includes determining the type of pet feeder to develop, the feeding mechanism, connectivity requirements and monitoring sensors.

3.1.2 Design the system architecture: Based on the project requirements, design the system architecture, including the microcontroller, sensors, and connectivity components.

3.1.3 Develop the hardware: Develop the hardware components, including the feeding mechanism and sensors. Test and refine the hardware until it meets the project requirements.

3.1.4 Develop the software: Develop the software components, including the microcontroller firmware, and communication protocols. Test and refine the software until it meets the project requirements.

3.1.5 Integrate the hardware and software: Integrate the hardware and software components and test the system as a whole to ensure it meets the project requirements.

3.1.6 Conduct testing and debugging: Conduct extensive testing to ensure the system operates as intended. Debug any issues and refine the system until it meets the project requirements.

3.1.7 Produce a prototype: Once the system is fully tested and refined, produce a prototype to test in real-world conditions.

3.1.8 Conduct field testing: Conduct extensive field testing to evaluate the system's performance and identify any issues that need to be addressed.

3.1.9 Finalize the design: Based on the field testing results, refine the system design as necessary and finalize the system architecture, hardware, and software components.

3.1.10 Launch the product: Launch the IoT-based automatic pet feeder to the market and continue to gather feedback from users to refine and improve the product over time.

Overall, the project goals and requirements of an IoT-based automatic pet feeder are to provide pet owners with an efficient, convenient, and reliable way to feed their pets remotely while ensuring the pet's health and well-being.

3.2 Methodology

3.2.1 Requirement gathering: The first step is to identify the needs and requirements of the pet feeder. This includes understanding the pet's feeding behavior, feeding schedule, types of food, and the pet owner's lifestyle and preferences.

3.2.2 System architecture design: Based on the requirements, design the system architecture, including the hardware and software components. This may include selecting the microcontroller, sensors and communication protocols.

3.2.3 Hardware development: Develop the hardware components, including the feeding mechanism and sensors. Test and refine the hardware until it meets the project requirements.

3.2.4 Software development: Develop the software components, including the microcontroller firmware, and communication protocols. Test and refine the software until it meets the project requirements.

3.2.5 Integration and testing: Integrate the hardware and software components and test the system as a whole to ensure it meets the project requirements.

3.2.6 Prototype production: Once the system is fully tested and refined, produce a prototype to test in real-world conditions.

3.2.7 Field testing: Conduct extensive field testing to evaluate the system's performance and identify any issues that need to be addressed.

3.2.8 System optimization: Based on the field testing results, refine the system design as necessary and optimize the system architecture, hardware, and software components.

3.2.9 User feedback and improvement: Gather feedback from users and continue to improve the system over time to meet evolving user needs and preferences.

3.2.10 Commercialization: Once the system is fully optimized and refined, launch the product to the market and promote it to potential users.

Overall, the methodology for developing an IoT-based automatic pet feeder involves understanding the user's needs and preferences, designing a robust system architecture, developing the hardware and software components, integrating and testing the system, refining and optimizing the system based on user feedback, and commercializing the product.

3.3 Implementation Plan

3.3.1 Define the requirements: Identify the specific needs of your pet and the necessary functionalities of the feeder, such as portion size, feeding schedule, food type, and water supply.

3.3.2 Choose the hardware components: Select the appropriate hardware components, including a microcontroller or single-board computer (e.g., Arduino, Raspberry Pi), sensors (e.g., weight sensor, moisture sensor), actuators (e.g., servo motor), power supply, and communication modules (e.g., Wi-Fi, Bluetooth).

3.3.3 Design the circuit: Create a circuit diagram for the feeder that includes all the hardware components and their connections.

3.3.4 Develop the software: Write the software code for the microcontroller or single-board computer to control the feeder's operation. This includes configuring the sensors and actuators, setting the feeding schedule, and establishing communication with the user's smartphone app.

3.3.5 Build the feeder: Assemble the hardware components according to the circuit diagram and the software code.

3.3.6 Test and debug: Test the feeder and the software to ensure that it is functioning as intended. Troubleshoot any issues that arise during testing.

3.3.7 Integrate with IoT platform: Connect the feeder to an IoT platform (e.g., AWS IoT, Azure IoT, Google Cloud IoT) to enable remote monitoring and control.

3.3.8 Maintain and update: Regularly maintain and update the system to ensure its reliability, security, and compatibility with the latest technologies and standards.

Overall, the implementation of an IoT-based automatic pet feeder involves designing and developing the hardware and firmware, testing and debugging, conducting field testing, optimizing the system based on user feedback, and commercializing the product.

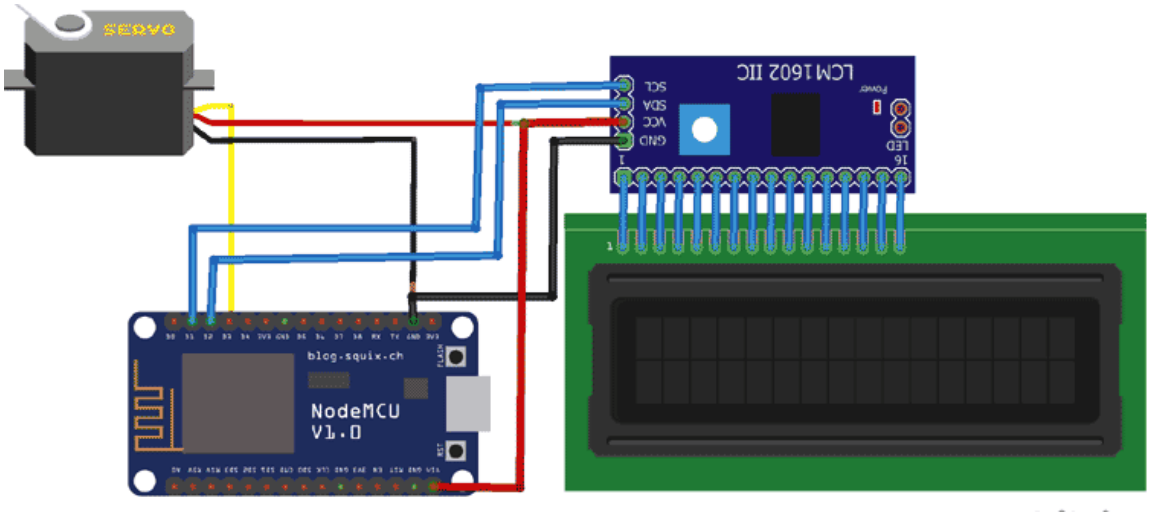


Fig 3.1 Final circuit diagram of the planned project.

Program Code:

```
#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"
#include<Servo.h>
#include <NTPClient.h>
#include <WiFiUdp.h>
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
Servo servo;
LiquidCrystal_I2C lcd(0x27, 16, 2);

#define WIFI_SSID "S22 Ultra"
#define WIFI_PASS "sp312124"
#define MQTT_SERV "io.adafruit.com"
#define MQTT_PORT 1883
int SERVO_PIN = D3;                 // The pin which the servo is attached to
int CLOSE_ANGLE = 0;                // The closing angle of the servo motor arm
int OPEN_ANGLE = 60;                // The opening angle of the servo motor arm
int  hh, mm, ss;
int feed_hour = 0;
int feed_minute = 0;
void setup() {
  Serial.begin(9600);
  timeClient.begin();
```

```
 Wire.begin(D2, D1);
  lcd.begin();                                  //Connect to WiFi
  Serial.print("\n\nConnecting Wifi... ");
  WiFi.begin(WIFI_SSID, WIFI_PASS);
  while (WiFi.status() != WL_CONNECTED)  {
    delay(500); }}

void loop() {
   MQTT_connect();
   timeClient.update();
   hh = timeClient.getHours();
   mm = timeClient.getMinutes();
   ss = timeClient.getSeconds();
   lcd.setCursor(0,0);
   lcd.print("Time:");
  if(hh>12)  {
    hh=hh-12;
    lcd.print(hh);
    lcd.print(":");
    lcd.print(mm);
    lcd.print(":");
    lcd.print(ss);
    lcd.println(" PM  ");  }
  else{
    lcd.print(hh);
    lcd.print(":");
    lcd.print(mm);
    lcd.print(":");
    lcd.print(ss);
    lcd.println(" AM  ");    }

void MQTT_connect() {
  int8_t ret;                                   // Stop if already connected.
  if (mqtt.connected()) {
    return;  }
 uint8_t retries = 3;
  while ((ret = mqtt.connect()) != 0)          // connect will return 0 for connected     {
     mqtt.disconnect();
       delay(5000);  // wait 5 seconds
       retries--;
       if (retries == 0)  {                    // basically die and wait for WDT to reset me
         while (1);  } } }
void open_door() {
   servo.write(OPEN_ANGLE);  // Send the command to the servo motor to open the trap door}
void close_door()  {
  servo.write(CLOSE_ANGLE);   // Send te command to the servo motor to close the trap door
}
```

Chapter IV

Results and Output

4.1 Result.

4.1.1 Accuracy: The pet feeder should accurately dispense the correct amount of food at the scheduled time.

4.1.2 Reliability: The pet feeder should operate reliably over an extended period without malfunctioning or breaking down.

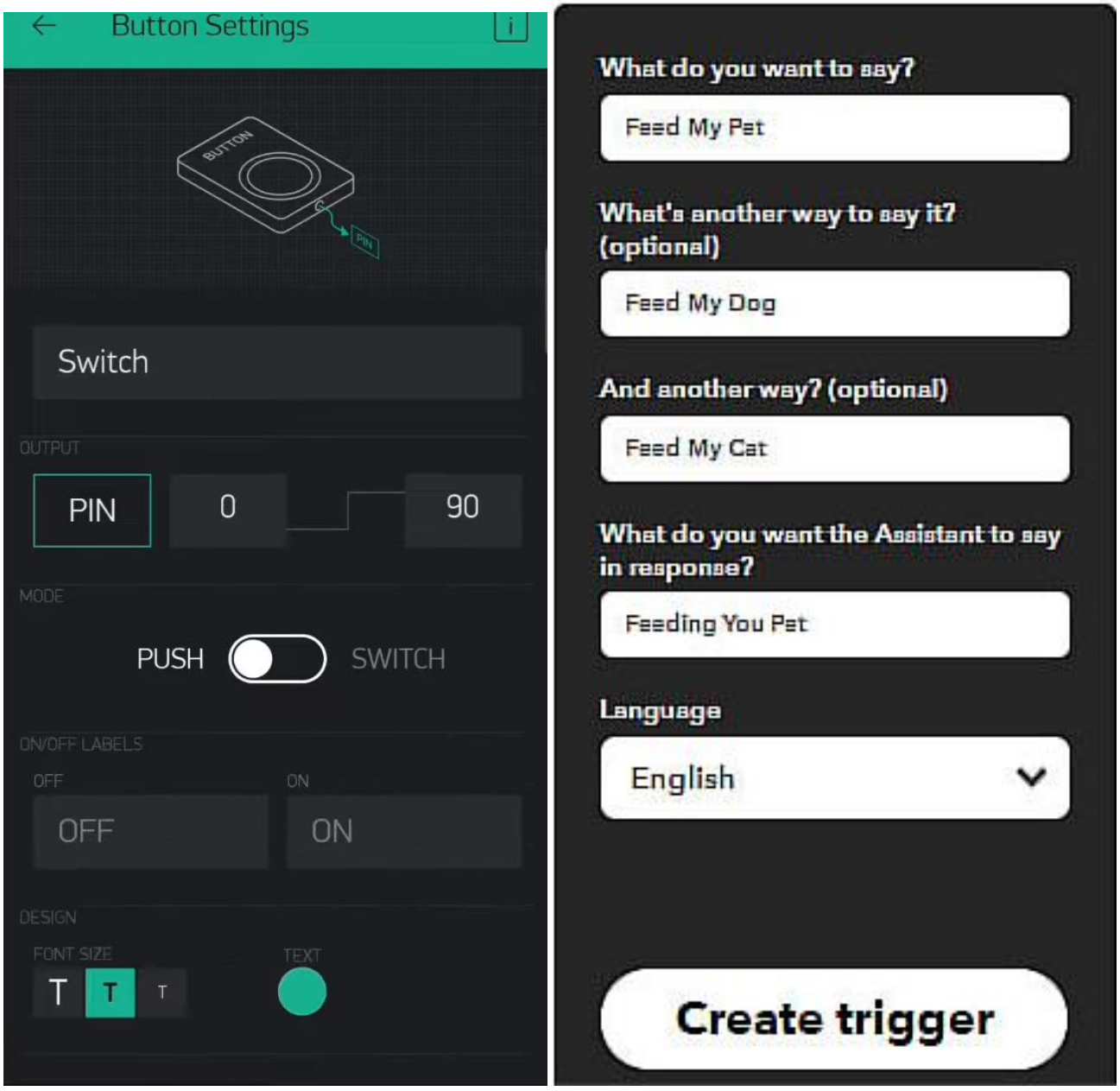
4.1.3 User interface: The mobile application should provide an intuitive user interface that allows pet owners to monitor and control the pet feeder remotely.

4.1.4 Connectivity: The pet feeder should connect to the internet reliably and securely.

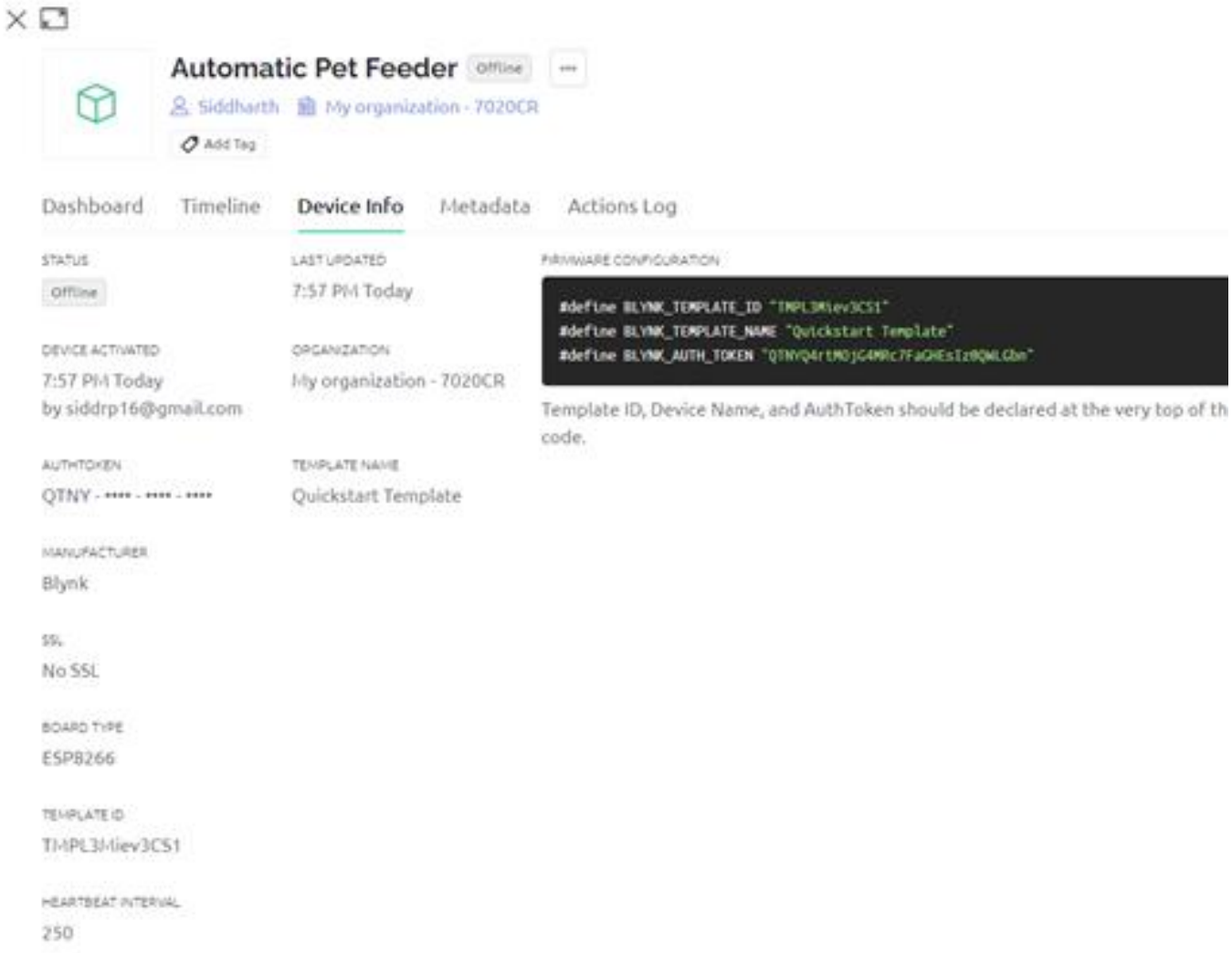
4.1.5 Battery life: The pet feeder should have a long battery life, minimizing the need for frequent charging or replacement of batteries.

4.1.6 Feedback and reporting: The pet feeder should provide real-time feedback and reporting on the pet's feeding habits, enabling the pet owner to monitor their pet's health and wellbeing.

4.2 Output



UI of the application



UI of the Web Portal

Chapter V

Conclusion

In conclusion, an IoT-based automatic pet feeder can be a convenient and helpful solution for pet owners who want to ensure their pets are fed on time, even when they're away from home. With the help of various sensors, such as weight sensors and motion sensors, the feeder can dispense the right amount of food and even alert the pet owner if the food level is running low.

Moreover, by integrating the feeder with a mobile app or a web-based interface, pet owners can monitor their pet's feeding habits, adjust the feeding schedule, and even remotely dispense food. This can provide peace of mind to pet owners and help them manage their pets' diet and health.

However, it's important to note that an IoT-based automatic pet feeder should not replace human interaction with pets. Pets still need socialization and exercise, and pet owners should not rely solely on technology to care for their pets. Overall, an IoT-based automatic pet feeder can be a valuable tool for pet owners, but it should be used in conjunction with proper pet care practices.

Chapter VI

References

Links

- https://forum.arduino.cc/
- https://community.blynk.cc/
- https://docs.blynk.io/
- https://forum.allaboutcircuits.com/
- https://www.esp8266.com/viewtopic.php?p=70305
- https://www.esp8266.com/
- https://arduino-esp8266.readthedocs.io/
- https://www.arduino.cc/reference/en/libraries

Acknowledgements

We are profoundly grateful to Prof. Manila Gupta for her expert guidance and continuous encouragement throughout to see that this project rights its target.

We would like to express deepest appreciation towards Dr. Varsha Shah, Principal RCOE, Mumbai and Prof. Shiburaj Lappu HOD Computer Department whose invaluable guidance supported us in this project.

At last, we must express my sincere heartfelt gratitude to all the staff members of Computer Engineering Department who helped us directly or indirectly during this course of work.

Siddharth Pallar (211P016)
Ameen Khan (211P025)
Aditya Vishwakarma (211P035)

Publications

[Add you published research paper on this topic in any Conference / Journal.]