A Project report on

INDOOR PLANT MONITORING SYSTEM USING IOT

Submitted in partial fulfillment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

By

A. LIKHITHA	**(204G1A0548)**
R. LALITHA	**(204G1A0547)**
K. ASHA NIDHI	**(204G1A0516)**
K. HEMA LATHA	**(204G1A0539)**

Under the Guidance of

Dr. B. Harichandana M.Tech., Ph.D

Associate Professor

Department of Computer Science & Engineering

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(AUTONOMOUS)

Rotarypuram Village, B K Samudram Mandal, Ananthapuramu - 515701

2023-2024

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(AUTONOMOUS)

(Affiliated to JNTUA, Accredited by NAAC with 'A' Grade, Approved by AICTE, New Delhi & Accredited by NBA (EEE, ECE & CSE)
Rotarypuram Village, BK Samudram Mandal, Ananthapuramu-515701

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Certificate

This is to certify that the project report entitled INDOOR PLANT MONITORING SYSTEM USING IOT is the bonafide work carried out by **A. Likhitha, R. Lalitha, K. Asha Nidhi, K. Hema Latha** bearing Roll Number **204G1A0548, 204G1A0547, 204G1A0516, 204G1A0539** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering** during the academic year 2023-2024.

Project Guide

Dr. B. Harichandana M.Tech., Ph.D
Associate Professor

Head of the Department

Mr. P. Veera Prakash M.Tech., (Ph.D)
Assistant Professor

Date:

Place: Rotarypuram

External Examiner

ACKNOWLEDGEMENTS

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that I have now the opportunity to express my gratitude for all of them.

It is with immense pleasure that we would like to express our indebted gratitude to my Guide **Dr. B. Harichandana, Associate Professor, Computer Science & Engineering**, who has guided me a lot and encouraged me in every step of the project work. We thank her for the stimulating guidance, constant encouragement and constructive criticism which have made possible to bring out this project work.

We express our deep felt gratitude to **Mr. C Lakshminatha Reddy, Assistant Professor and Mr. M. Narasimhulu, Assistant Professor, Project Coordinators** for their valuable guidance and unstinting encouragement enabled us to accomplish our projectsuccessfully in time.

We are very much thankful to **Mr. P. Veera Prakash, Assistant Professor & Head of the Department, Computer Science & Engineering,** for his kind support and for providing necessary facilities to carry out the work.

We wish to convey our special thanks to **Dr. G. Bala Krishna, Principal** of **Srinivasa Ramanujan Institute of Technology (Autonomous)** for giving the required information in doing our project work. Not to forget, we thank all other faculty and non- teaching staff, and our friends who had directly or indirectly helped and supported us in completing our project in time.

We also express our sincere thanks to the Management for providing excellent facilities**.**

Finally, We wish to convey our gratitude to our families who fostered all the requirements and facilities that we need.

Project Associates

204G1A0548

204G1A0547

204G1A0516

204G1A0539

DECLARATIONS

We Ms. A. Likhitha bearing reg no: 204G1A0548, Ms. R. Lalitha bearing reg no: 204G1A0547, Ms. K. Asha Nidhi bearing reg no: 204G1A0539, Ms. K. Hema Latha bearing reg no: 204G1A0539, students of **SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY**, Rotarypuram, hereby declare that the dissertation entitled "**INDOOR PLANT MONITORING SYSTEM USING IOT**" embodies the report of our project work carried out by us during IV Year Bachelor of Technology under the guidance of **Dr. B. Harichandana**, Associate Professor, Department of CSE and this work has been submitted for the partial fulfillment of the requirements for the award of Bachelor of Technology degree.

The results embodied in this project report have not been submitted to any other Universities of Institute for the award of Degree.

A. LIKHITHA	Reg no: 204G1A0548
R. LALITHA	Reg no: 204G1A0547
K. ASHA NIDHI	Reg no: 204G1A0516
K. HEMA LATHA	Reg no: 204G1A0539

CONTENTS

List of Figures

List of Tables

Abbreviations

GSM	Global system for Mobile Communication
DHT	Digital Temperature and Humidity Sensor
IoT	Internet of Things
ARM	Advanced Rise Machine
CPU	Central processing unit
RAM	Random-access memory
USB	Universal Serial Bus
GPIO	General Purpose Input / Output
VNC	Virtual Network Computing
IDE	Integrated Development Environment
GUI	Graphical user interface
LPDPR	Low-Power double Data Rate
SDLC	Software Development Life Cycle

ABSTRACT

Nowadays, a lot of individuals are quite interested in doing their own gardening. Indoor gardening has a variety of advantages, including the ability to produce organic vegetables, use of the plants in home design, and air purification. People's busy schedule is the major obstacle to indoor gardening, because plants require more attention for their growth and health, also the plants need to be protected from the damage caused by some animals and birds, necessitating the hiring of a "plant sitter" if they leave on vacation. This problem can be solved by automating the plant monitoring using the "Internet of Things". Automated gardening systems have revolutionized the way we grow plants indoors, making it easier than ever to cultivate a thriving garden. These systems ensure optimal conditions for the plants to flourish. Utilising Raspberry pi and variety of environmental sensors to track the temperature, moisture and light. Installing a camera to monitor the actions of animalia in case of detection, a buzzer sound is made and servo motors to provide shade for plants and a water pump to provide water which is controlled using relay. Raspberry pi acts as the main controlling unit that can control the operation of various sensors and camera and providing data on each of these elements via Push bullet app.

Keywords: Indoor Planting, Plant Sitter, Raspberry Pi, Camera, Relay, Animalia, Push bullet App.

.

1. INTRODUCTION

1.1 Internet of Things

Internet of Things (IoT) revolutionizes connectivity by linking physical devices to the digital world. IoT encompasses interconnected sensors, devices, and systems that gather, exchange, and act upon data autonomously. It enables real-time monitoring, automation, and data-driven decision-making across various domains, from smart homes to industries and healthcare. With its ability to enhance efficiency, optimize processes, and improve user experiences, IoT is reshaping industries and society, promising a future of unprecedented innovation and connectivity.

The Internet of Things (IoT) is revolutionizing how we interact with the physical world by connecting devices, sensors, and objects through internet connectivity. This interconnected ecosystem enables real-time data collection, exchange, and analysis across various domains like healthcare, transportation, agriculture, and environmental monitoring.

An emerging application of IoT is indoor plant monitoring systems, which use sensors and actuators to track essential environmental conditions for plant growth, such as temperature, humidity, soil moisture, and nutrient levels. By harnessing IoT capabilities, these systems provide real-time data and analytics to optimize plant growth conditions, detect anomalies, and automate tasks like watering.

The work looks at the design, development, and implementation of an IoT-based indoor plant monitoring system. We look at the underlying technologies, protocols, and architectures involved, as well as their practical uses and advantages. Furthermore, we undertake trials and evaluations to determine the system's efficacy, efficiency, and scalability in enhancing indoor plant cultivation and management techniques.

We envision that the findings will contribute to the growing field of IoT-enabled applications while also providing insights into indoor plant monitoring system design and deployment. These findings have the potential to influence sustainable agriculture, urban farming, and environmental conservation efforts.

1.2 Objective

The proposed system aims to safeguard plants from various threats by employing an integrated approach. By utilizing sensors to monitor temperature and soil moisture levels, the system ensures that plants are protected from excess heat while conserving water resources through efficient watering mechanisms. Additionally, the system incorporates overseeing mechanisms to deter bird activity and safeguard plant survival. Through real-time monitoring and proactive interventions, our system aims to create a conducive environment for plant growth while mitigating potential risks posed by environmental factors and external threats.

1.3 Problem Definition

Due to the fast-paced nature of people's lives, indoor plants often lack the necessary attention for optimal growth and health. To ensure the continuous growth and robustness of plants, it is essential to monitor and manage predetermined levels of temperature, moisture, light, and humidity, as well as to detect any bird activity that may pose a threat.

2. LITERATURE SURVEY

In emphasizing the crucial role of Raspberry Pi integration, Zuraida Muhammad, Muhammad Azri and Asyraf Mohd Hafez, position it as the unifying core for the sensor array [1]. This integration, vital for seamless communication and control, aligns precisely with their technical emphasis on efficient data processing. Additionally, the authors highlight the significance of soil moisture sensors in maintaining optimal moisture levels for nutrient absorption and preventing water stress. The concise focus on Raspberry Pi integration resonates with their insights on elevating overall system robustness.

Highlighting a sensor-driven system perfected for precise detection of humidity, temperature, and soil moisture, this research, led by Yogeshwari Barhate and Mr. Rupesh Borse, integrates IoT for automated irrigation and real-time plant monitoring [2]. Offering insights into optimal sustainability parameters, the system seamlessly delivers data to users through smartphones or laptops, eliminating the need for manual intervention and reducing the risk of errors.

Structured into three integral components, the system encompasses animal detection and identification (for discerning animal presence), an alarm system (activating a buzzer for animal deterrence), and a GSM module (for notifying authorized personnel) [3]. Upon animal detection, the sensor triggers distance calculation, prompting the camera to capture images. These images, supervised by Selvamuthukumaran and Evangeline Sneha, undergo comparison with a stored dataset for precise animal identification. Upon surpassing a predefined distance threshold, the buzzer activates, and an alert message is transmitted to designated recipients.

Illuminating the application of PushBullet in plant monitoring systems, the study showcases its widespread adoption as a convenient approach for remotely overseeing and managing plant conditions [4]. The discussion underscores PushBullet's prevalence as a user-friendly and effective IoT platform, empowering users to craft personalized mobile applications tailored to their IoT ventures.

In the work on the Utilization and Applications of IoT (Internet of Things), Prof. Ms. P. V. Dudhe, Prof. Ms. N. V. Kadam, Prof. R. M. Hushangabade, and Prof. M. S. Deshmukh [6] offer profound technical insights. The authors intricately navigate the expansive realm of IoT applications, shedding light on its transformative impact across various sectors. Their analysis spans smart homes, industrial automation, and beyond, presenting a comprehensive understanding of the technical nuances that define the landscape of IoT deployment.

3. PLANNING

3.1 Existing System

The existing system monitors the environmental conditions using various sensors.

The system uses NodeMCU as microcontroller. NodeMCU comes with the inbuilt ESP8266 WiFi-module which connects our system to blynk app using WiFi. The program which controls the functioning of the whole system is fed into the microcontroller using Arduino IDE which is an environment which integrates code with the hardware. Soil moisture sensor continuously detects the level of moisture in the soil and displays it on the Virtual LCD widget on the Blynk app. If the water content in the soil is less than what is required by the plant, a notification is sent to the user‟s smartphone and he/she can switch ON the button widget in Blynkapp which will turn ON the water supply. Real time values from the DHT11 temperature sensor are also displayed on the virtual LCD.

3.1.1 System Architecture

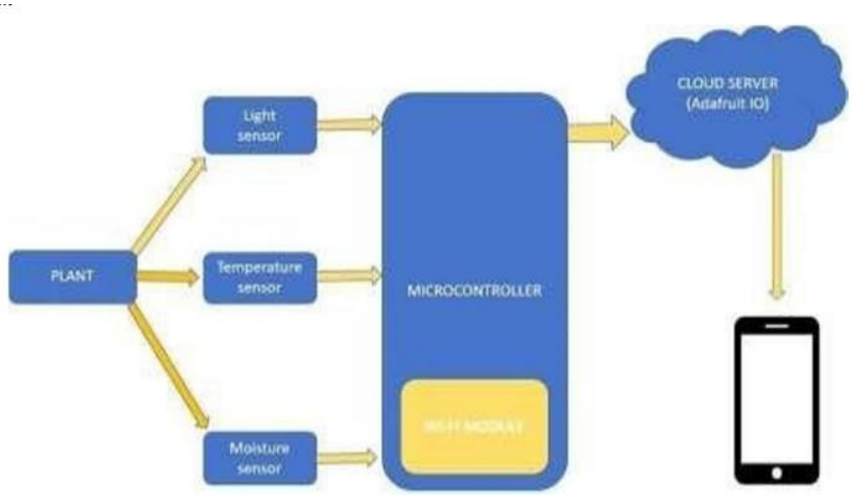


Fig 3.1: Architecture of the Existing System

The user is notified about each and every step through the notification feature of the Blynkapp. Hence, this system monitors and controls the plants requirements remotely.

3.1.2 Software

Arduino IDE:

- Arduino IDE is an open-source software which makes easy to write code for microcontroller and allows to upload on board. The environment is written in java and based on processing and other source software.
- Arduino IDE can be used on Windows, Linux (Both 32 and 64 bits), and Mac OS. Current versions are Arduino 1.0x or Arduino 1.5x Beta Version.

3.1.3 Hardware

- Microcontroller - NodeMCU
- Light Sensor: Passive sensor
- Moisture sensors
- DHT11 sensor
- Power supply: +5V power supply

3.1.4 Disadvantages

- In this system ultrasonic sensors are used which are temperature sensitive.
- This system doesn't provide the accurate alerts during the cool weather conditions
- This system is used to alert only one side of the curvy road, thus the other side is left in blind.

3.2 Proposed System

The system, driven by Raspberry Pi, employs environmental sensors for real-time monitoring of humidity, temperature, and moisture levels. A camera module enhances the system by detecting bird presence and activating a buzzer as a deterrent. This integrated approach enables precise and efficient plant condition monitoring.

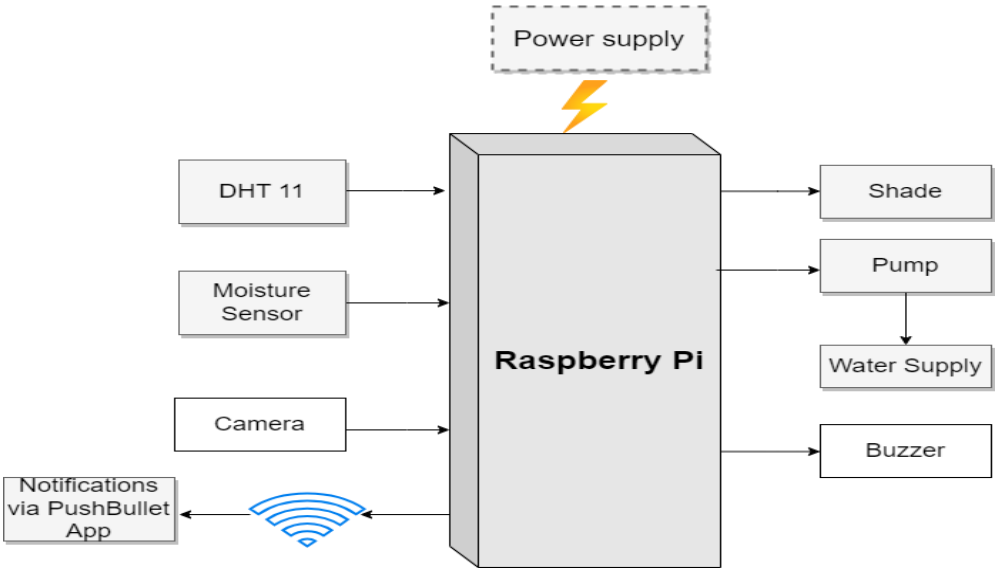


Fig 3.2: Architecture of the Proposed System

3.2.1 Methodology

The model proposes indoor plant monitoring system which will notify the people about the various conditions of plant. The system is divided into two modules; they are checking environmental conditions and bird detection. The first module is implemented on indoor plants and second module is implemented on birds coming near to the plants. Each module provides notifications and alerts to the mobile via PushBullet.

The first module is to detect the various conditions of the plant and send notification through PushBullet. The DHT11 sensor is used to detect the humidity and temperature of the surroundings and sends information to the raspberry pi and notifies if the humidity and temperature are greater than threshold. The soil moisture sensor detects the moisture content in the soil and this information is sent to raspberry pi then it instructs the water pump to automatically pump the water to the plants if the moisture is low.

The second module is to detect the bird activity. This is done using the camera

module. The camera detects the bird coming near to the plant and sends information to the raspberry pi then it instructs the buzzer to make sound which results in flying away. This prevents the plant from being affected from bird activity.

Fig 3.2.1: Prototype of Proposed System

3.3 Hardware

- Microcontroller (Raspberry Pi 4 Model B)
- DHT 11 Sensor
- Soil Moisture Sensor
- Camera Module
- Servo Motor
- Water Pump
- Relay Module
- Buzzer
- Power supply (+5V)

3.3.1 Raspberry Pi 4

The Raspberry Pi 4, developed by the Raspberry Pi Foundation, serves as the cornerstone of our indoor plant monitoring system. Featuring dual-band Wi-Fi, USB interfaces, and a quad-core ARM Cortex-A72 CPU with 2GB, 4GB, or 8GB RAM options, it excels in real-time data processing. The VideoCore VI graphics engine enables two 4K screens, enhancing multimedia capabilities. The GPIO ports facilitate seamless interfacing with sensors for light intensity and soil conditions tracking. Its compatibility with various operating systems, including Raspberry Pi OS, ensures

adaptability for our specific IoT application

Fig 3.3: Raspberry Pi 4

3.3.2 DHT11 Sensor

A prominent digital temperature and humidity sensor in indoor plant monitoring systems is the DHT11. It tracks humidity from 20% to 90% with ±5% precision and operates in the temperature range of 0 to 50 degrees Celsius with an accuracy of ±2°C. The sensor usually runs at 3.3V or 5V and communicates via a single-wire digital interface. It provides consistent temperature and humidity readings with a fast reaction time.

Fig 3.4: DHT11 Sensor

3.3.3 Soil Moisture Sensor

A key element in our indoor plant management system is precise soil moisture monitoring using advanced sensors. Placed strategically in the root zone, these sensors utilize cutting-edge technology to provide real-time soil moisture data. Integrated with General Purpose Input/Output (GPIO) pins, they establish a direct link to the CPU, facilitating seamless data collection and transfer. This ensures timely and accurate information for informed irrigation decisions, optimizing water use, and promoting overall plant health and growth.

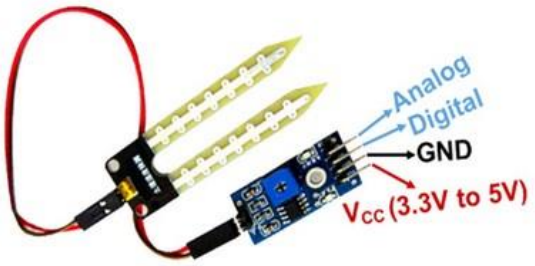


Fig 3.5: Soil Moisture Sensor

3.3.4 Camera

The USB camera integrated into our indoor plant monitoring system is a sophisticated imaging device chosen for its capability to capture high-quality visual data essential for detecting bird activity within the monitored environment. The selected camera model features a high-resolution sensor with 25 megapixels ensuring detailed and sharp imagery. With a frame rate of 30 fps, the camera captures smooth and continuous footage, allowing for precise monitoring of bird interactions.

Fig 3.6: USB Camera Module

3.3.5 Servo Motor

Servo motors are vital for precise angular control in indoor plant monitoring. Operating at low voltages (4.8V to 6V), they integrate seamlessly with microcontrollers and come in various sizes with distinct torque and speed ratings. Limited to a rotation range of 180 degrees, they adjust sensor positions for targeted data collection. Consideration of torque, speed, and power is crucial when selecting servo motors for optimal performance in monitoring systems.

Fig 3.7: Servo Motor

3.3.6 Water Pump

A reliable water pump, selected for its responsiveness to real-time soil moisture data, plays a crucial role in automated watering within our indoor plant monitoring system. Interfaced seamlessly with the central processing unit, this pump allows dynamic adjustments to watering schedules based on the current soil moisture levels. Engineered for simplicity and smooth integration, the pump ensures uniform and precise watering, a fundamental factor for optimal hydration and health of indoor plants.

Fig 3.8: Water Pump

3.3.7 Relay Module

The relay module is a critical component in indoor plant monitoring, acting as an electronic switch for high-power devices. Operating on 5V or 12V with single or multiple channels, it interfaces seamlessly with microcontrollers for integration into plant monitoring projects. Prioritizing safety through isolation between control circuits and loads, relay modules automate device control based on sensor data, enhancing indoor environmental conditions for plants. Their versatility makes them essential for managing various aspects of indoor plant care.

Fig 3.9: Relay Module

3.3.8 Buzzer

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers and beepers include alarm devices, timers, train and confirmation of user input such as a mouse click or keystroke.

Fig 3.10: Buzzer

3.4 Software

- Raspbian OS
- VNC Viewer
- Thonny IDE
- Push Bullet

3.4.1 Raspbian OS

Raspbian, a Debian-based Linux distribution, serves as the official operating system for Raspberry Pi single-board computers, tailored to their unique architecture and functionalities. The default and recommended OS for applications like our indoor plant monitoring system, Raspbian provides a user-friendly environment. Renowned for its reliability and pre-loaded applications, it accommodates users of all skill levels. With extensive support for programming languages, developers can easily create and deploy applications. Raspbian's package management simplifies software updates and installations, ensuring compatibility with diverse projects. Its versatility and regular updates from the Raspberry Pi Foundation make Raspbian a robust choice for our system.

3.4.2 VNC Viewer

The VNC (Virtual Network Computing) Viewer program is pivotal in our indoor plant monitoring system, enabling remote desktop access and control to the Raspberry Pi. It grants users graphical user interface (GUI) access to the Raspberry Pi's desktop interface from remote devices like computers or mobile phones. This access facilitates seamless management, configuration, and troubleshooting of the Raspberry Pi desktop environment, regardless of physical proximity. Operating over secure connections, VNC Viewer ensures data integrity and confidentiality during transmission. Its intuitive interface and cross-platform compatibility enhance accessibility and usability, contributing significantly to the system's smooth operation.

3.4.3 Thony IDE

An integrated development environment (IDE) facilitates computer programmers by integrating fundamental tools (e.g., code editor, compiler, and

debugger) into a single software package. Users do not need to install the language's compiler/interpreter on their machines; an IDE provides the environment itself.

Thonny is a free, dedicated IDE for Python designed for beginners.

The following are some of the primary features of Thonny:

- It auto-completes code.
- It inspects code to provide bracket matching and highlight errors.
- It is easy to start with as its installer also installs Python 3.7.
- It enables users to step into a function call by providing details about local variables and displaying the code pointer.
- Its debugger is simple to use as no knowledge of breakpoints is required.

Pros of Thonny IDE:

- It has an easy-to-use user interface.
- The user interface does not contain any distractions for beginners.

Cons of Thonny IDE:

- It offers basic functionality as opposed to other advanced IDEs (i.e., PyCharm).
- Users may encounter some issues for which a quick fix is not available.

3.4.4 Push Bullet

Push bullet, a configurable cross-platform application, facilitates seamless data exchange and communication across diverse devices. Serving as a conduit between desktops, tablets, smartphones, and interconnected devices, push bullet enables effortless sharing of files, links, and notifications. In our indoor plant monitoring system, push bullet operates as a notification system, promptly alerting users to significant changes or events, such as fluctuations in soil moisture levels, system alarms, or notable plant conditions. Through instantaneous notifications delivered to smartphones or linked devices, users remain informed and can swiftly respond to critical information. Push bullet's extensive interoperability and streamlined interface enhance the responsiveness and communication capabilities of our indoor plant monitoring system, fostering accessibility and efficiency.

3.5 Functional Requirements

- To continuously measuring the humidity, temperature and moisture content by tracking the soil moisture and temperature levels.
- To provide water when moisture level is low by using pump and adjusting to temperature level.
- When bird is detected, it identifies with previous dataset and buzzer sounds an alarm.
- To send notifications through push bullet app when all these activities are notified.

3.6 Non-Functional Requirements

3.6.1 Usability

- This Project enables users to remotely control certain aspects of the system through the app, such as watering plants if necessary.

3.6.2 Portability

- Push bullet app is accessible and functional across various mobile devices and platforms for remote monitoring from anywhere.

3.6.3 Speed

- This project is speed in sending alert messages to the user for the respective bird activities and condition of the plant.

3.7 Scope

- Alert the person by providing Buzzer sound when a bird comes.
- Display a notification message on a mobile device of moisture and temperature based on a certain threshold.
- Providing shade and water to plants automatically.

3.8 Performance

The Performance of the project is efficient by the following:

- The Raspberry Pi 4 Model B features a powerful quad-core Cortex-A72 CPU running at 1.5GHz and is available with 1GB, 2GB, or 4GB LPDDR4 RAM options. It offers dual-band Wi-Fi, Gigabit Ethernet, Bluetooth 5.0, 2x USB 3.0 ports, and 2x USB 2.0 ports for versatile connectivity. With Video Core VI graphics supporting OpenGL ES 3.x, it ensures smooth graphical performance for various applications.
- Quantum web cameras are designed for high-quality imaging in various lighting conditions and may offer features such as 4K resolution, high frame rates, wide dynamic range, and advanced image processing algorithms.

3.9 Methodology

To implement this project Iterative Model is used. It involves continuous cycle of Planning, Analysis, Implementation and Evaluation.

The Iterative Model allows the accessing earlier phases, in which the variations made respectively. The final output of the project renewed at the end of the Software Development Life Cycle (SDLC) process.

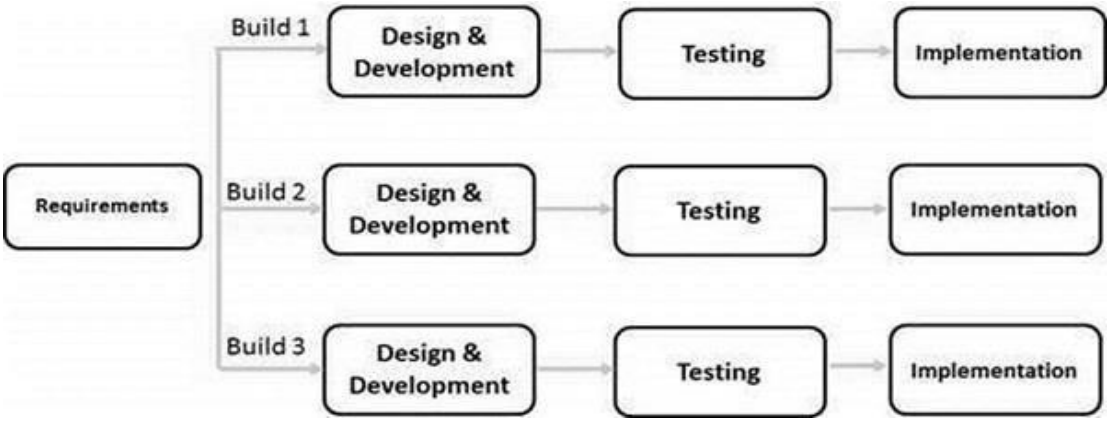


Fig 3.11: Iterative Model

3.9.1 Advantages

- It is easily acceptable to ever-changing needs of the project.
- Testing and debugging during smaller iteration is easy.
- A parallel development can plan.

3.10 Cost Estimation

Table 3.1: Estimated cost of the equipment

S.No	Component	Cost (Rs)	Quantity	Total Cost (Rs)
1	Raspberry Pi 4	5000	1	5000
2	DHT11 Sensor	150	1	150
3	Soil Moisture Sensor	108	1	108
4	Buzzer	20	1	20
5	Water Pump	250	1	250
6	Camera Module	1250	1	1250
7	Servo Motor	200	1	200
8	Jumper Wires	10	1	90
9	Soldering	10	40	500
10	Shipping	400	-	400
	Total Cost			**7968**

3.11 Time Estimation

Basically, our project is divided into two phases.

- Phase – 1: Pre-requisites, Planning and Designing

Table 3.2: Activities performed in Phase – 1

S.No	Activity	Duration
1	Domain Selection	1 Week
2	Literature Survey and Problem Definition	2 Weeks
3	Planning and Designing	2 Weeks

- Phase – 2: Developing the model.

Table 3.3: Activities performed in Phase - 2

S.No	Activity	Duration
1	Gathering all the Components	1 Week
2	Developing the Environmental checking system (Module 1)	2 Weeks
3	Developing the Bird detection system (Module 2)	3 Weeks
4	Implementing the System	1 Week
5	Testing and Finalizing the System	1 Week

4. DESIGN

4.1 Architecture of monitoring moisture and temperature

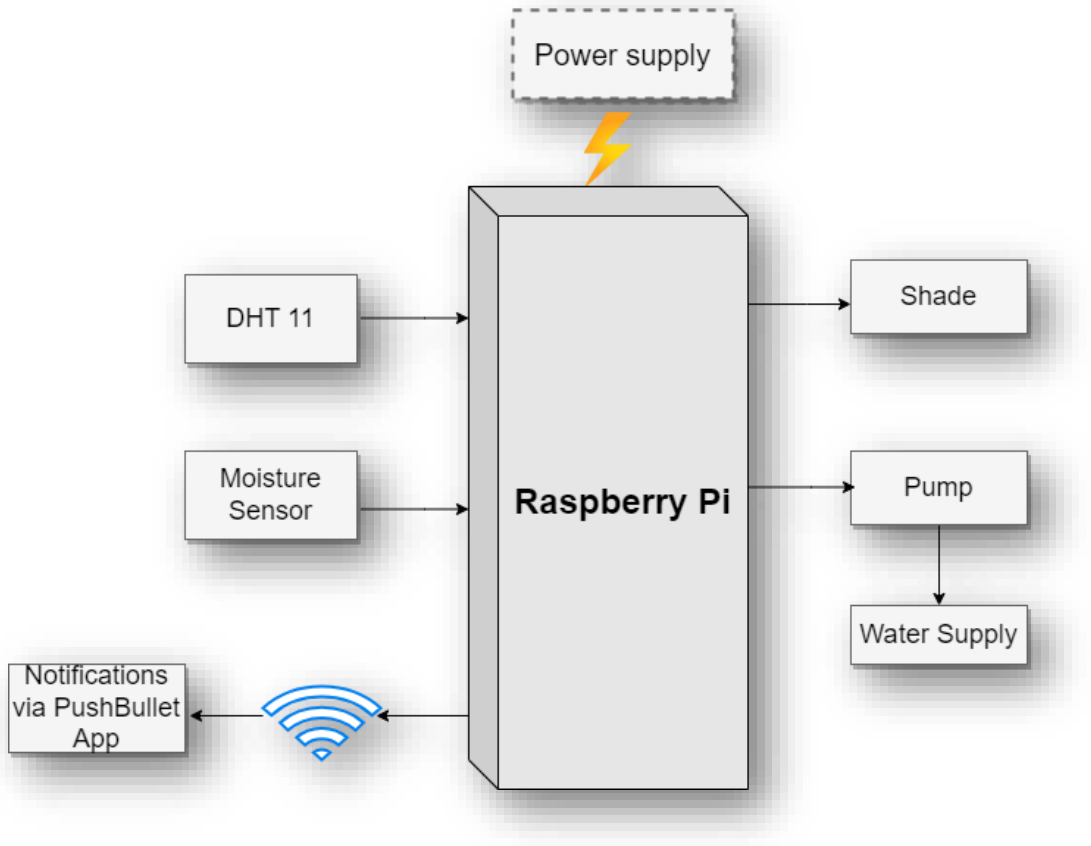


Fig 4.1: Monitoring Moisture and Temperature Architecture

The monitoring environmental conditions architecture shows that DHT 11 and Moisture sensor areconnected to the Raspberry Pi 4. The sensors send the information to the micro controller and then the micro controller will provide the shade and water to the plants depending upon the information given by the sensors.

4.2 Flow Chart of monitoring moisture and temperature

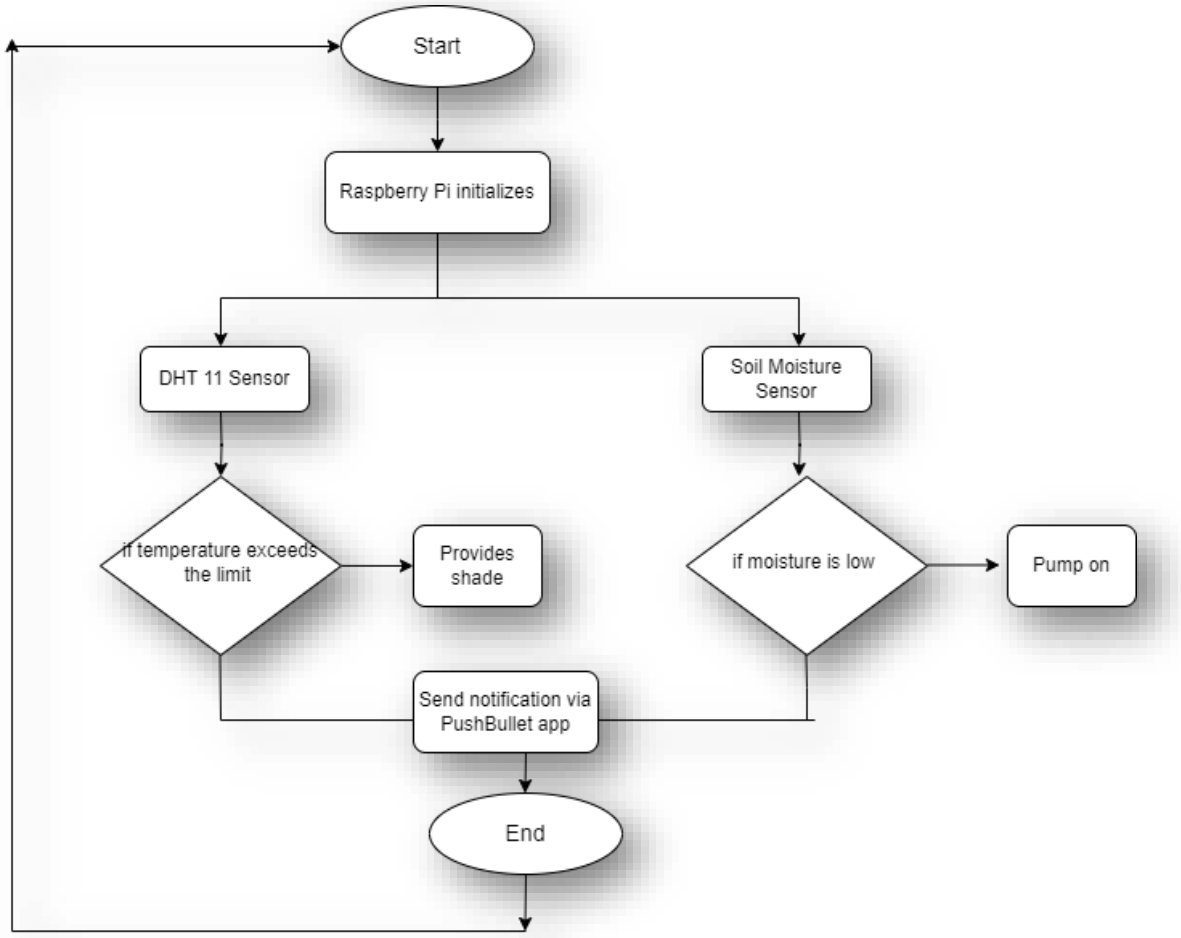


Fig 4.2: Flow chart of Monitoring Moisture and Temperature Architecture

The above flow chart describes how the monitoring environmental system works. Initially shade and water pump will be in off condition. Whenever the temperature exceeds and moisture level decreases, the DHT 11 sensor and moisture sensor detects the conditions and will provide the shade and water automatically also notify the user by giving the notifications on push bullet.

4.3 Bird Detection Architecture

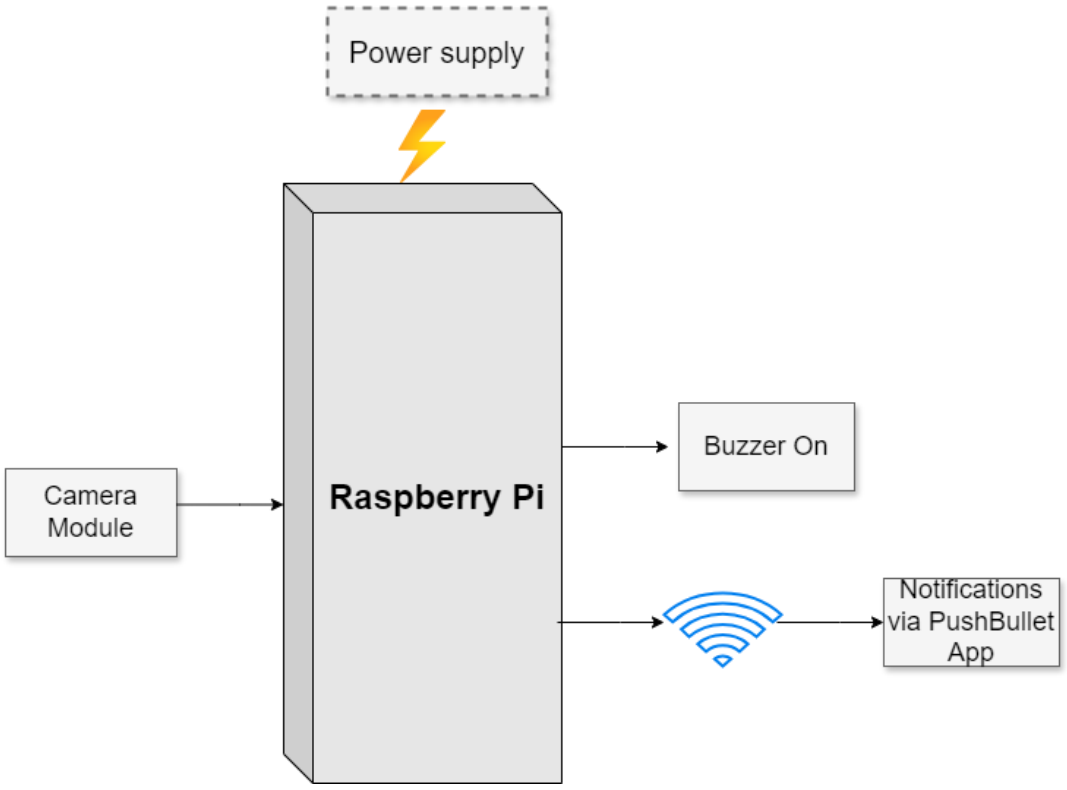


Fig 4.3: Bird Detection Architecture

The bird detection system architecture shows that camera is connected to the Raspberry Pi 4 micro controller. Based on the information passed by the camera, the micro controller will make the buzzer on and display the notification message on the push bullet.

4.4 Flow Chart of Bird Detection

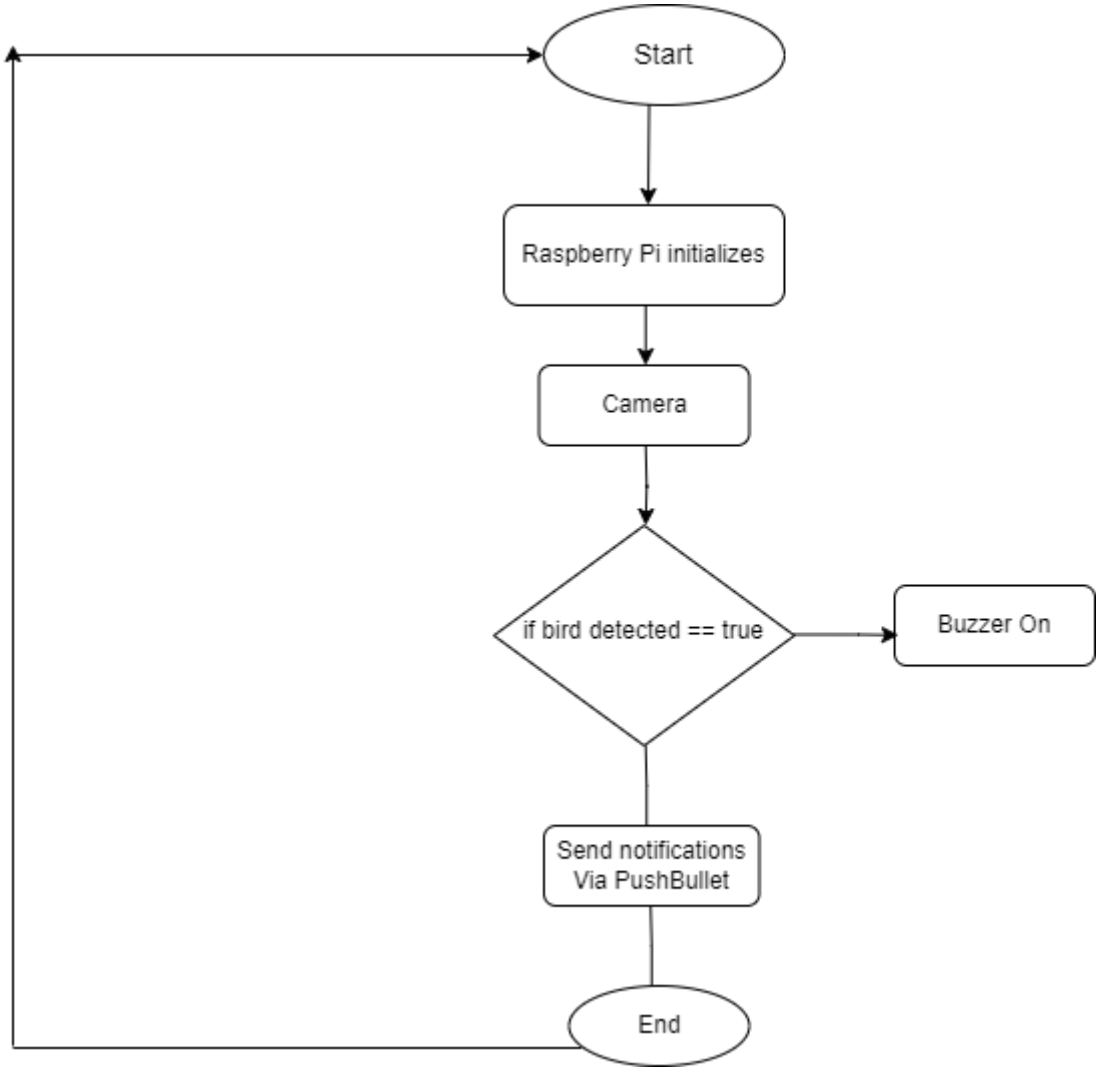


Fig 4.4: Flow chart of Bird Detection

The bird detection system flow chart describes how the user will get the respective notification message if any bird comes near to the plant. Firstly, the camera is connected to the Raspberry Pi 4. The camera will capture the image and checks for the in the data provided and sends information to the micro controller then the micro controller will turn on the buzzer and display a notification message on the users mobile through push bullet.

5. IMPLEMENTATION

5.1 Hardware Implementation

This model proposes indoor plant monitoring system which will notify the people about the various conditions of plant. This system is divided into two modules; they are checking environmental conditions and bird detection. The first module is implemented on indoor plants and second module is implemented on birds coming near to the plants. Each module provides notifications and alerts to the mobile via PushBullet.

5.1.1 Monitoring temperature and Mositure

This module is to detect the various conditions of the plant and send notification through push bullet. The DHT11 sensor is used to detect the humidity and temperature of the surroundings and sends information to the raspberry pi and notifies if the humidity and temperature are greater than threshold and provides automatic shade to the plant.

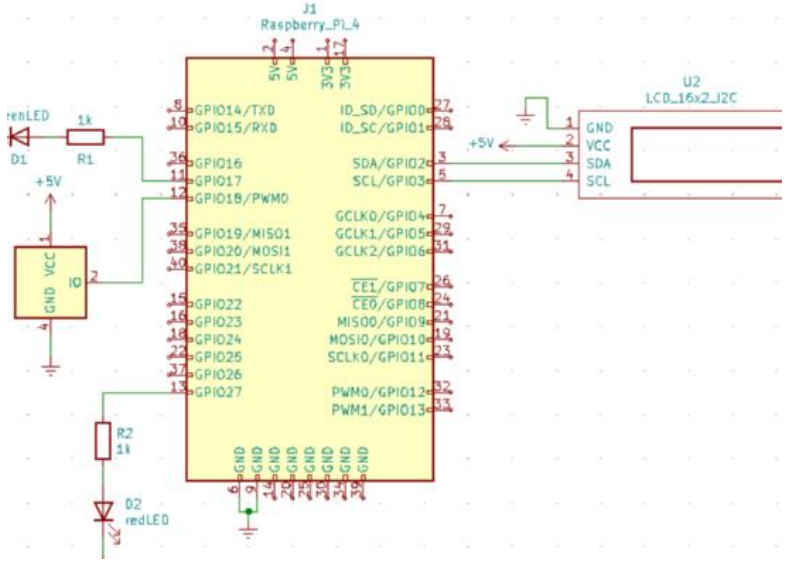


Fig 5.1: Circuit Diagram of Monitoring Temperature

The soil moisture sensor detects the moisture content in the soil and this information is sent to raspberry pi then it instructs the water pump to automatically pump the water to the plants if the moisture is low.

Pump Activation= 1 if moisture<threshold

0 Otherwise

Pump Activation Signal=PumpActivation×Relay Signal

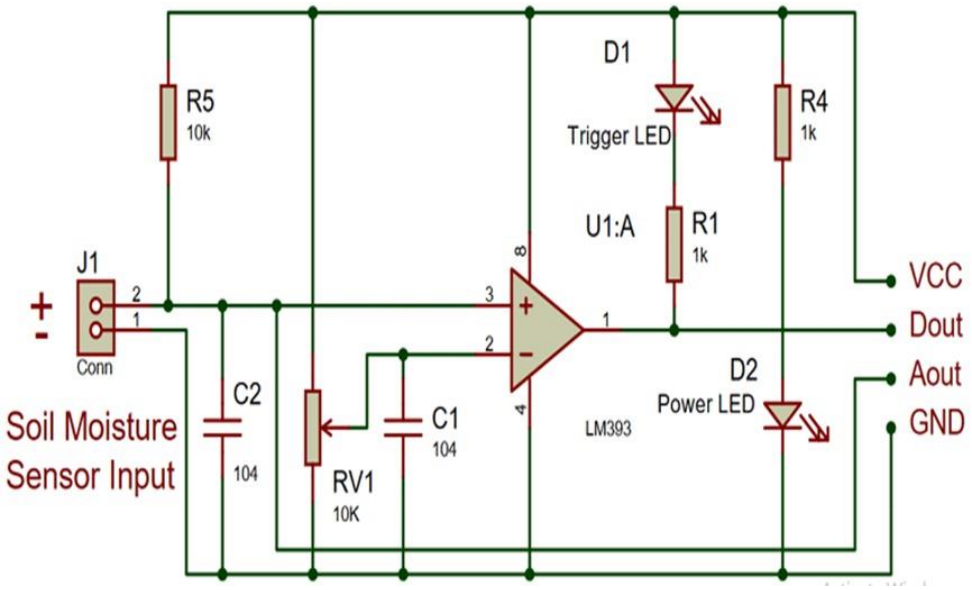


Fig 5.2: Circuit Diagram of Monitoring Moisture

5.1.2 Bird Detection System

The second module is to detect the bird activity. This is done using the camera module. The camera detects the bird coming near to the plant and sends information to the raspberry pi then it instructs the buzzer to make sound which results in flying away. This prevents the plant from being affected from bird activity.

if detection confidence>=threshold
Bird Identified=1
0 otherwise
model = MobileNetV2(weights='imagenet')

This multi-faceted technical implementation significantly elevates the system's capabilities, providing a robust defense mechanism for indoor plants through advanced object detection functionalities and comprehensive environmental monitoring.

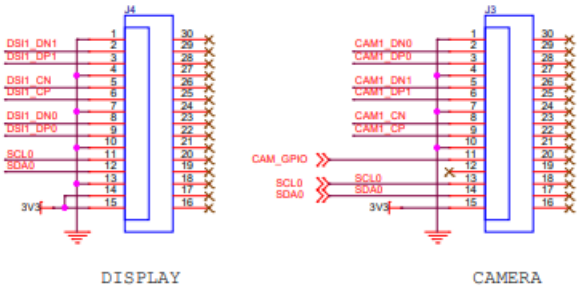


Fig 5.3: Circuit Diagram of the Bird Detection System

Real-time Object Detection

predictions =predict_frame(frame)

Object-specific Notifications and Buzzer Activation

notification =send_notification (title, body)

Pushbullet Notifications push = pb.push_note (title, body)

Send Notification=1 if conditions met, 0 otherwise

Data flow

- Sensors collect data -> Raspberry Pi processes data -> PushBullet for remote monitoring.

- Moisture level triggers relay -> Activates mini-DC pump if watering is needed.

- Camera captures images -> MobileNetV2 identifies birds -> Buzzer activates to deter birds.

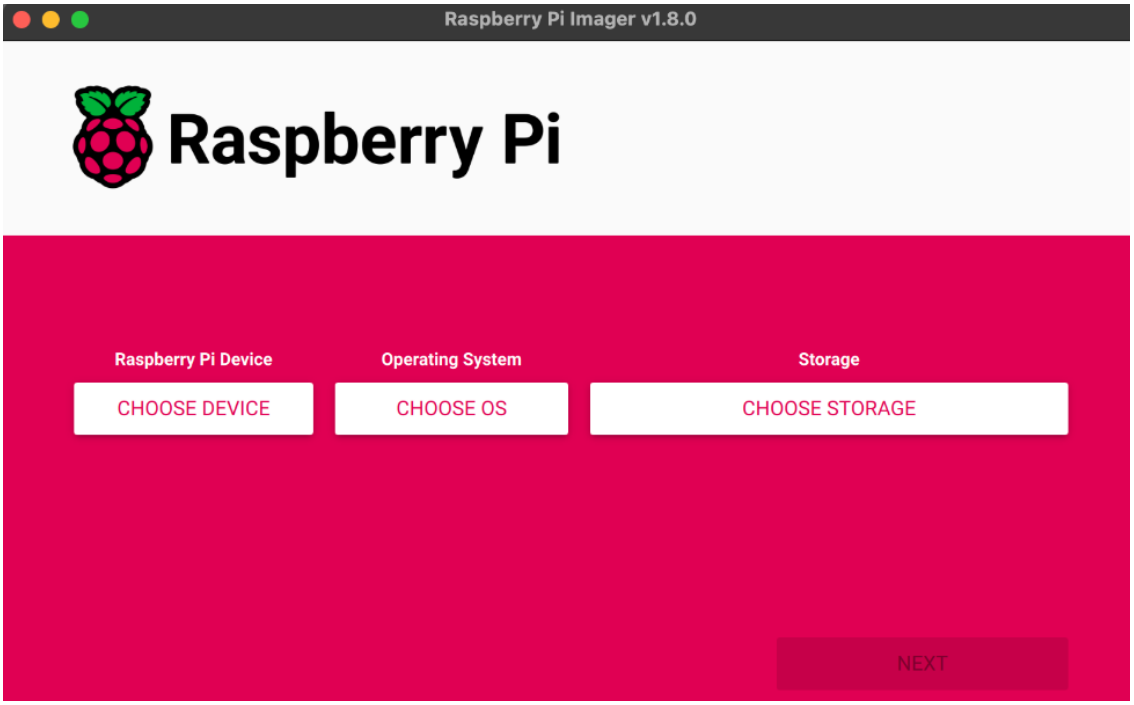
5.2 Software Implementation

5.2.1 Installation of Raspbian OS

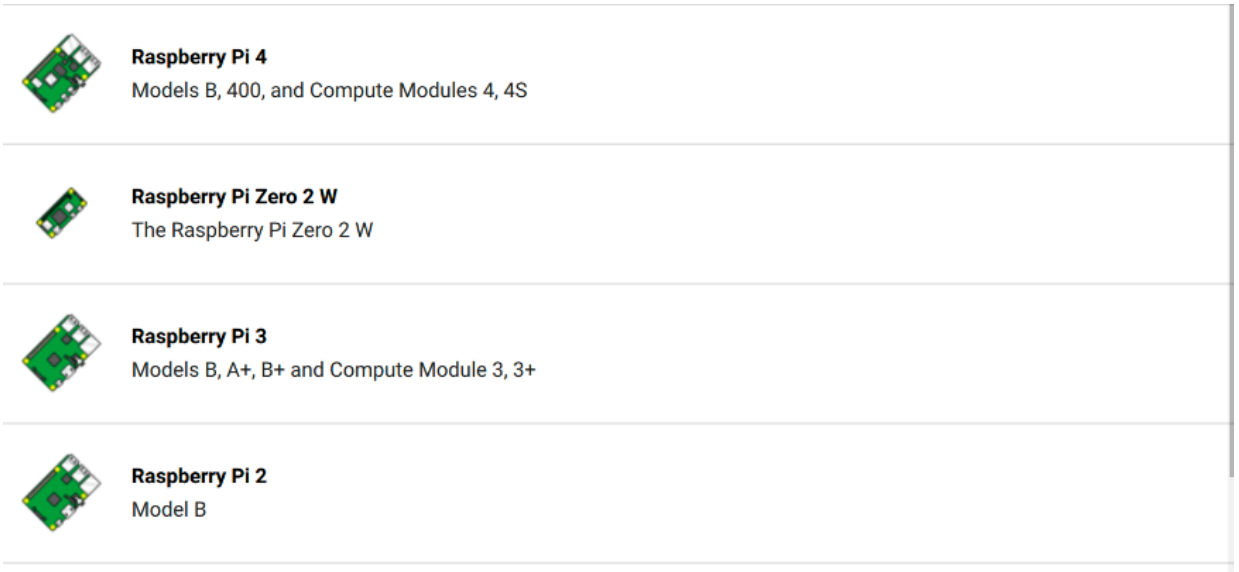
We can install Raspbian os using imager:

1. Download the latest version from raspberrypi.com/software and run the installer.

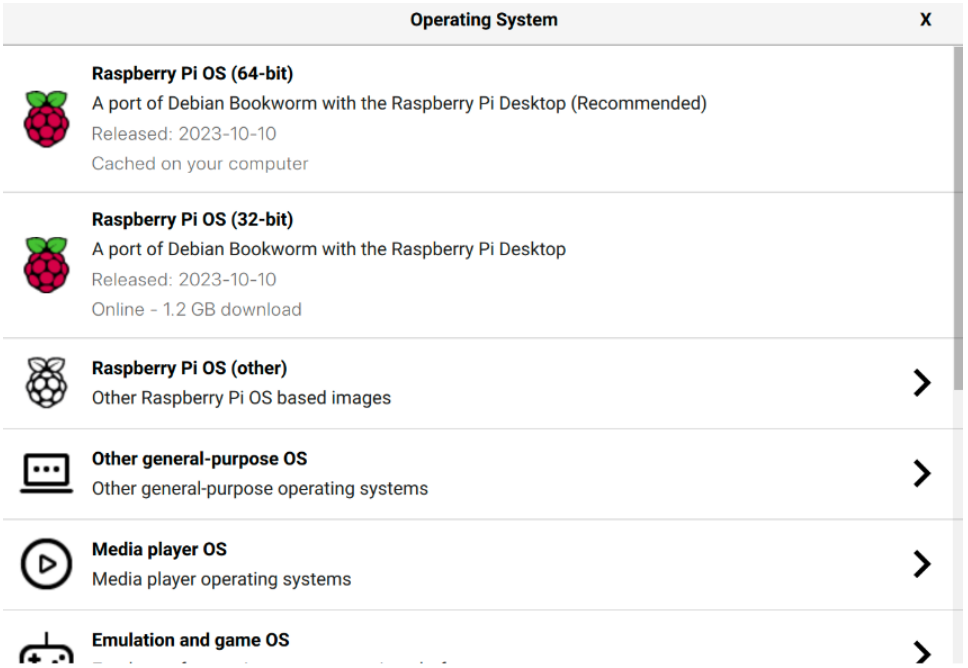
2. Install it from a terminal using your package manager, e.g. sudo apt install rpi-imager.



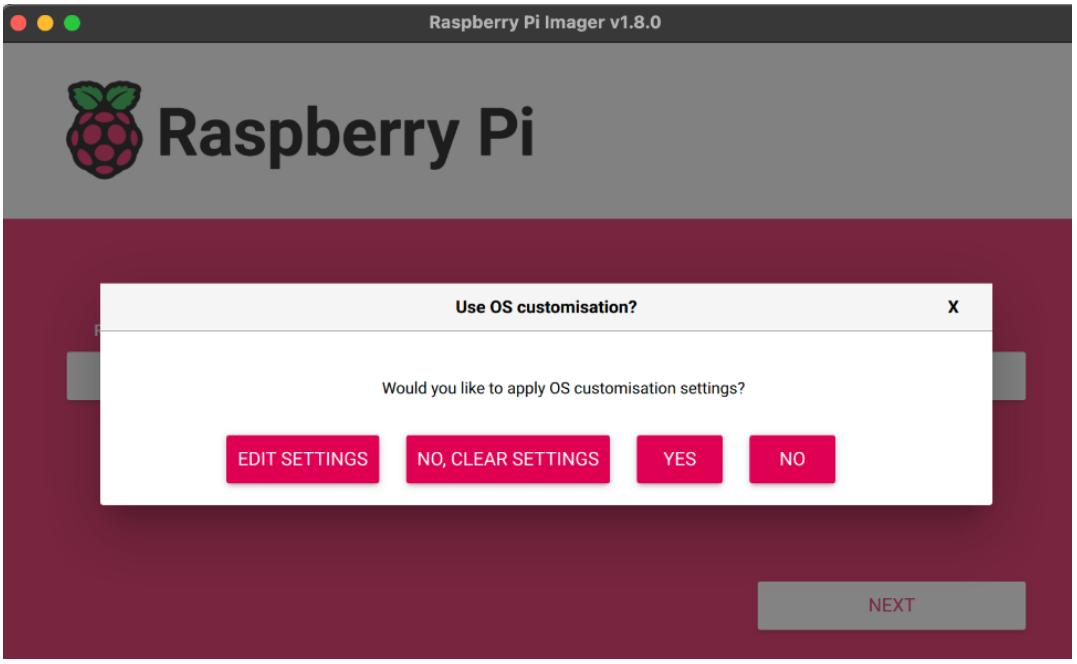
3. Click Choose device and select your Raspberry Pi model from the list.



4. Next, click Choose OS and select an operating system to install. Imager always shows recommended version of Raspberry Pi OS for your model at the top of the list.



5. Next, click Next.



5.2.2 Python code

A. Source code for temperature and moisture monitoring System

```
import Adafruit_ADS1x15
import Adafruit_DHT
import requests
import time
import RPi.GPIO as GPIO
from pushbullet import
Pushbullet
# Replace
'YOUR_PUSHBULLET_
API_KEY' with your
actual API key
API_KEY =
'o.AEKiQmSN5q1Maxajy
Xas0LdWX39iO2f2'
# Create an ADS1115
instance
adc =
Adafruit_ADS1x15.ADS1
115()
# Choose a gain (1 for +/-
4.096V, 2 for +/-2.048V,
etc.)
GAIN = 1
# Define the ADC channel
```

```
(0-3)
ADC_CHANNEL = 0
# ThingSpeak parameters
channel_id = "2392052"  #
Replace with your
Channel ID
write_api_key =
"BQIWVTISH96T7S7F"
# Replace with your Write
API Key
# Configure DHT sensor
DHT_SENSOR =
Adafruit_DHT.DHT11
DHT_PIN = 24  # GPIO
pin connected to the DHT
sensor
# Servo and buzzer setup
SERVO_PIN = 8  # GPIO
pin connected to the servo
BUZZER_PIN = 26  #
GPIO pin connected to the
buzzer
# Set GPIO mode
GPIO.setmode(GPIO.BC
M)
GPIO.setwarnings(False)
```

```
GPIO.setup(SERVO_PIN,
GPIO.OUT)
GPIO.setup(BUZZER_PI
N, GPIO.OUT)
servo =
GPIO.PWM(SERVO_PIN
, 50)  # PWM frequency
servo.start(0)
dry_adc_value = 27887
wet_adc_value = 60
def
read_temperature(percenta
ge):
    humidity, temperature =
Adafruit_DHT.read(DHT_
SENSOR, DHT_PIN)
    if humidity is not None
and temperature is not
None:
        print(f'Humidity
:{int(humidity)}')
        print(f'Temperature
:{int(temperature)}')
        # Create the
ThingSpeak update URL
        url =
```

```
f"https://api.thingspeak.co
m/update?api_key={write
_api_key}&field1={tempe
rature}&field2={humidity
}&field3={percentage}"
        # Send data to
ThingSpeak
        response =
requests.get(url)
        if
response.status_code ==
200:
print("Data sent to
Pushbutton")
        else:
            print(f"Failed to
send data. Status code:
{response.status_code}")
        time.sleep(10)  #
Delay for 20 seconds
before sending next data
    else:
        print("Failed to read
sensor data")
    return temperature
def
calculate_moisture_percen
```

```
tage(raw_value,
dry_value, wet_value):
    # Calculate the range
between dry and wet
values
    value_range =
dry_value - wet_value
    # Calculate the
percentage
    moisture_percentage =
100 - ((raw_value -
wet_value) / value_range)
* 100
    # Ensure the percentage
is within 0-100% bounds
    moisture_percentage =
max(0, min(100,
moisture_percentage))
    return
moisture_percentage
def read_channel_0():
    # Read the analog
sensor value from channel
0
    channel_0_value =
adc.read_adc(ADC_CHA
NNEL, gain=GAIN)
```

```
    return channel_0_value
def
activate_servo_and_buzze
r():
    servo.ChangeDutyCycle
(7.5)  # Rotate servo to
180 degrees
    #GPIO.output(BUZZE
R_PIN, GPIO.HIGH)  #
Turn on the buzzer
def
deactivate_servo_and_buz
zer():
    servo.ChangeDutyCycle
(0)  # Set servo back to 0
degrees
    #GPIO.output(BUZZE
R_PIN, GPIO.LOW)  #
Turn off the buzzer
def send_notification(title,
body):
    try:
        pb =
Pushbullet(API_KEY)
        push =
pb.push_note(title, body)
        print("Notification
```

```
sent successfully!")
    except Exception as e:
        print(f"Error sending
notification: {e}")
motor_on_flag = True
motor_off_flag = True
try:
    while True:
        raw_adc_value =
adc.read_adc(ADC_CHA
NNEL, gain=GAIN)
        moisture_percentage
=
int(calculate_moisture_per
centage(raw_adc_value,
dry_adc_value,
wet_adc_value))
        print(f"Moisture
percentage:
{moisture_percentage}%"
)       # Read from DHT11
sensor
        temperature =
read_temperature(moistur
e_percentage)
        if temperature is not
None and temperature >=
```

```
35:  print(f"Temperature:
{temperature}°C -
Threshold Reached")
            activate_servo_and
_buzzer()  # Activate
servo and buzzer
            notification_title =
"Temperature is high!!"
            notification_body
= "Temperature is high!!"
            send_notification(n
otification_title,
notification_body)
        else:
            deactivate_servo_a
nd_buzzer()  # Deactivate
servo and buzzer
        if
moisture_percentage >=
95  and motor_on_flag ==
True:    GPIO.output(BUZ
ZER_PIN, GPIO.LOW)
        motor_on_flag=False
         motor_off_flag=True
#print(f"T:{temperature}°
C - Threshold Reached")
#activate_servo_and_buzz
```

```
er()  # Activate servo and
buzzer notification_title =
"Motor is off!!"
notification_body =
"Motor is
off!!"send_notification(not
ification_title,
notification_body)
 elif moisture_percentage
<= 20 and motor_off_flag
== True:
GPIO.output(BUZZER_PI
N, GPIO.HIGH)
 motor_on_flag=True
    motor_off_flag=False
notification_title = "Motor
is on!!"
 notification_body =
"Motor is on!!"
send_notification(notificat
ion_title,
notification_body)
        time.sleep(1)
except KeyboardInterrupt:
    servo.stop()  # Stop
PWM    GPIO.cleanup()  #
Clean up GPIO pins
```

B. Source code for Bird Detection System

```
import cv2
import numpy as np
import RPi.GPIO as IO
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.applications.imagenet_utils import decode_predictions
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from pushbullet import Pushbulletmachine import Pin, I2C
# Load the pre-trained MobileNetV2 model
model = MobileNetV2(weights='imagenet')
# Replace 'YOUR_PUSHBULLET_API_KEY' with your actual API key
API_KEY = 'o.AEKiQmSN5q1MaxajyXas0LdWX39iO2f2'
BUZZER_PIN_1 = 6
IO.setmode(IO.BCM)
IO.setwarnings(False)
#GPIO.setup(SERVO_PIN, GPIO.OUT)
IO.setup(BUZZER_PIN_1, IO.OUT)
def preprocess_image(image):
    image = cv2.resize(image, (224, 224))
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    image = preprocess_input(image)
    image = np.expand_dims(image, axis=0)
    return image
def predict_frame(frame):
    processed_frame = preprocess_image(frame)
    predictions = model.predict(processed_frame)
    decoded_predictions = decode_predictions(predictions, top=1)
    return decoded_predictions[0]
def send_notification(title, body):
    try:
        pb = Pushbullet(API_KEY)
        push = pb.push_note(title, body)
        print("Notification sent successfully!")
    except Exception as e:
```

```
        print(f"Error sending notification: {e}")
def real_time_recognition():
    cap = cv2.VideoCapture(0)
    if not cap.isOpened():
        print("Cannot open USB camera")
        return
    object_detected = {}  # Dictionary to track detected objects and notifications
    try:
        while True:
            ret, frame = cap.read()
            if not ret:
                break
            predictions = predict_frame(frame)
            label = predictions[0][1]
            print(label)
            # Check for specific objects and send notification (once per object)
            specific_objects = ['dog', 'cat', 'bird', 'elephant', 'german_shepherd',
'golden_retriewer']
            for obj in specific_objects:
                IO.output(BUZZER_PIN_1, IO.HIGH)
                if obj in label.lower() and obj not in object_detected:
                    notification_title = "Object Detected"
                    notification_body = f"{obj.capitalize()} is in view!"
                    send_notification(notification_title, notification_body)
                    object_detected[obj] = True  # Mark object as detected
            else:
                 IO.output(BUZZER_PIN_1, IO.LOW)
            #cv2.imshow('USB Camera', frame)
            if cv2.waitKey(1) & 0xFF == ord('q'):
                break
    finally:
        cap.release()
        cv2.destroyAllWindows()
# Start real-time recognition
real_time_recognition()
```

6. TESTING

6.1 Testing approach

We will test the project in two stages: software and hardware. The software part is to be tested via the Thonny IDE, whereas the hardware part has to be tested physically. It is necessary to check whether the system is working properly or not.

6.2 Features to be tested

After building the whole circuit we test it. This project should satisfy somefeatures. Features to be tested as follows:

- The DHT11 and Moisture sensor should give proper output when it measures the temperature and moisture level.
- The camera should detect the particular bird when it comes near to the plant and should make the buzzer sound automatically.
- The push bullet should send the alert message whenever the actual values exceed the threshold values and the watering and shade to the plants must be provided automatically.

6.3 Testing tools and environment

For testing of the project we require some tools, like to test program we require software called Thonny IDE in the Raspbian os. Using this we can check the program if it is working properly or not. For hardware checking we require power supply and threshold values which are fixed manually.

6.4 Test cases

In this section we discuss about the inputs, expected output, testing procedure.

6.4.1 Inputs

This project requires two inputs:

1. Power supply is the basic need of any electronic circuit. Here we use 5v power.
2. The names of animals and birds should be given, so that the camera detects.

6.4.2 Expected Output

The expected output of this project should be automatically provided shade and water to the plants and alert message through push bullet app with a buzzer sound. The output should also be seen on the serial monitor of the Thonny IDE.

6.4.3 Testing Procedure

For testing, first connect the circuit to the power supply is given to the Raspberry Pi 4 using 5v adapter. In this way the whole testing circuit is built. Summary of the testing procedure -

- Connect the circuit according to the diagram
- Give power to the system.
- Vary humidity and temperature values to fix the threshold values.
- Get the output from the camera module.
- Display the alert message on push bullet with a buzzer sound.

7. RESULTS & ANALYSIS

The proposal suggests an indoor plant monitoring system that will alert individuals to the different plant conditions. There are two components in this system: one for detecting birds and the other for monitoring the surroundings. The first module is used with indoor plants, while the second one is used with birds that approach the plants. Using push bullet, each module sends alarms and notifications to the mobile device.

A. Monitoring temperature and soil moisture

Focusing on plant condition monitoring, the system employs a DHT11 sensor to gauge ambient humidity and temperature. The gathered data is then transmitted to the Raspberry Pi, which issues notifications if the humidity and temperature exceed predetermined thresholds. Simultaneously, a soil moisture sensor assesses the moisture content in the soil, relaying this information to the Raspberry Pi. If the soil moisture is below a set threshold, the Raspberry Pi directs the water pump to automatically irrigate the plants.

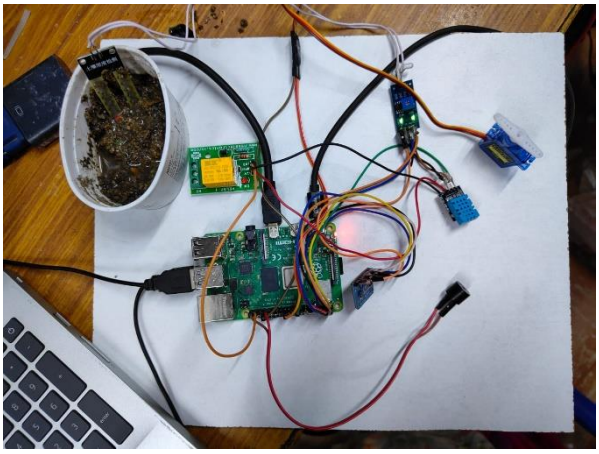


Fig 7.1: Checking for the moisture and temperature

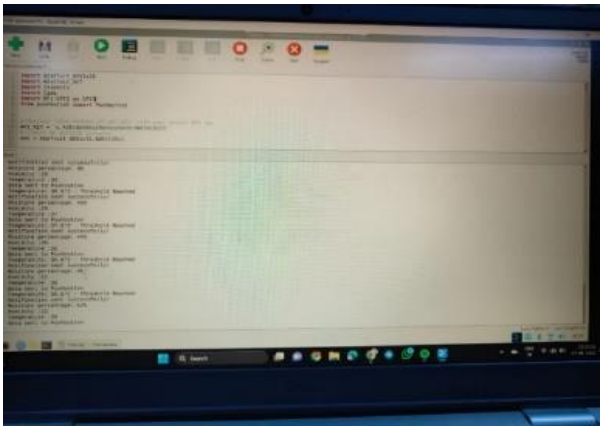


Fig 7.2: Results

B. Bird Detection System

The system employs a camera module to detect bird activity near the plant. It works by continuously monitoring the area surrounding the plant and analyzing the incoming visual data to identify the presence of birds. When a bird is detected approaching the plant, the camera captures this information and sends it to the Raspberry Pi.

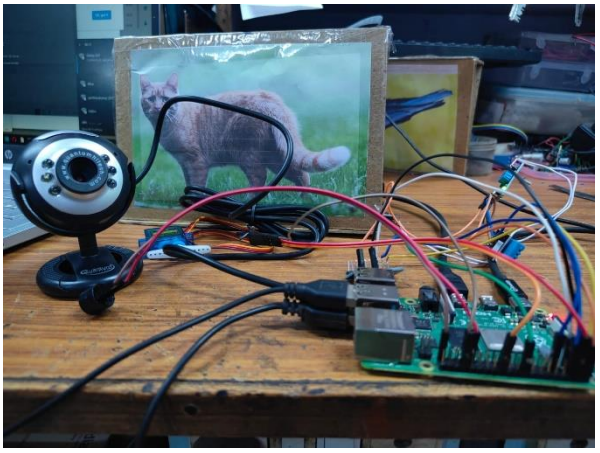


Fig 7.3: Bird detection System

Upon receiving the data from the camera, the Raspberry Pi processes it to confirm bird activity. Once confirmed, the Raspberry Pi activates the buzzer component of the system. The buzzer emits a sound signal designed to deter birds and discourage them from approaching the plant further.

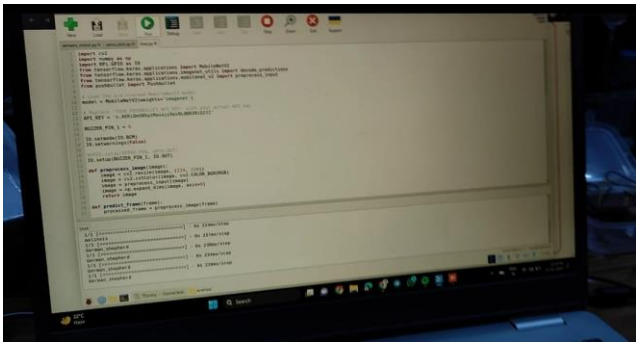


Fig 7.4: Detections of avians

This proactive approach helps protect the plant from potential damage caused by birds, such as pecking at leaves or disturbing the soil around the plant's roots. By promptly detecting bird activity and triggering the buzzer, the system effectively safeguards the plant and helps maintain its health and growth.

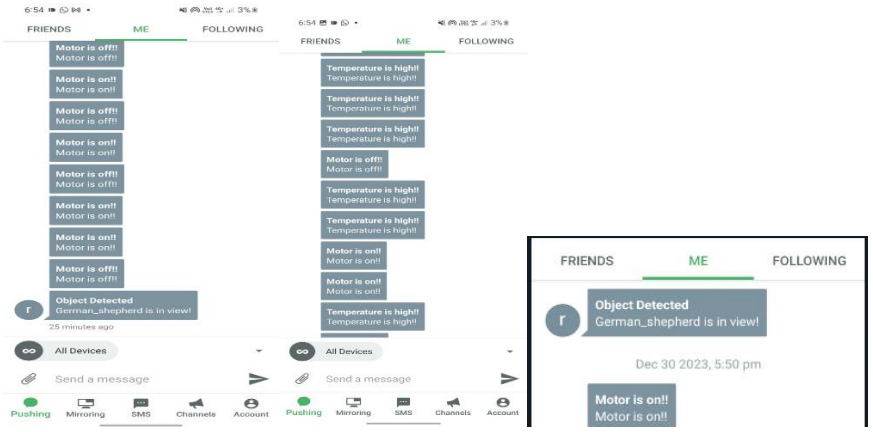


Fig 7.4: Sending notifications

The implementation of the Raspberry Pi and IoT-powered automated indoor gardening system yielded successful results in plant care management. Utilizing moisture and DHT11 sensors along with a camera, the system effectively transmitted real-time data to PushBullet for remote monitoring. The user-friendly interface enhanced accessibility, and automated responses, including watering, proved effective in safeguarding plants from animal and bird damage. The bird detection system, triggering a buzzer sound upon detecting birds, provided immediate alerts, significantly enhancing plant protection. This implementation showcases a streamlined and technologically advanced solution for indoor plant cultivation, ensuring precise and timely intervention.

CONCLUSION

The Raspberry Pi and IoT-driven automated indoor plant monitoring system efficiently tackles challenges posed by busy schedules in plant care. Through the automation of monitoring via moisture and DHT11 sensors, alongside a camera, the system enables remote observation and data transmission using PushBullet. This technological advancement not only supports organic vegetable cultivation and improves home aesthetics but also eliminates the necessity for 'plant sitters' during absences. The work highlights the harmonious integration of technology and gardening, providing a practical and efficient solution for modern lifestyles, with potential for further technical enhancements in the future.

REFERENCES

[1] Zuraida Zuraida Muhammad, Muhammad Azri Asyraf Mohd Hafez, Nor Adni Mat Leh, Zakiah Mohd Yusoff and Shabinar Abd Hamid, Smart Agriculture Using Internet Of Things with Raspberry Pi 10th IEEE International Conference on Control System, Computing and Engineering (ICCSCE2020) pp. 85-90, August. 2020.

[2] Ms. Yogeshwari Barhate, Mr. Rupesh Borse, Ms. Neha Adkar and Mr. Gaurav Bagul, Plant Watering and Monitoring System Using IOT and Cloud Computing International Journal of Scientific Development and Research vol. 5, pp. 157-162, April. 2020.

[3] Selvamuthukumaran N, Evangeline Sneha J, Thriambika K B and Vishali Babu B, Intelligent Animal Detection System Using IOT and Deep Learning International Research Journal of Modernization in Engineering Technology and Science, vol. 3, pp. 2433-2438, April. 2021.

[4] Yogesh kumar Jayam, Venkatesh Tunuguntla, Sreehari J B, S Harinarayanan, Smart Plant Monitoring System Using IOT Fourth International Conference on Trends in Electronics and Informatics (ICOEI 2020), pp. 271-277, July. 2020.

[5] Gaurav Patil, Akash Patil, Shashank Pathmudi, Plant Monitoring System International Journal of Engineering Research & Technology (IJERT), vol.10, pp. 1-4, September 2021.

[6] Prof.Ms.P.V.Dudhe, Prof.Ms.N.V.Kadam, Prof. R. M. Hushangabade, Prof. M. S. Deshmukh, Internet of Things (IOT): An Overview and it's Applications International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS-2017), pp. 2650-2653, 2017.

[7] R Lathesparan, A Shranjah, R Thushanth, S Kenurshan, MNM Nifras and WU Wickramaarachi, Real-time Animal Detection and Prevention System for Crop Fields, 13th Internationl Research Conference General Sir John Kotelawala Defence University, pp. 70-78, June 2021.

[8] Y. Manoj, T. Sai Naveen Kumar, Sk Adeeb Jainab, T. Sailaja and A. Anasuyamma, Greenhouse Monitoring Using Raspberry Pi, International Journal of Creative Research Thoughts, vol. 11, pp. 429-433, May 2023.

[9] Sachin Kumar, Pryag Tiwari, Mikhail Zymbler, Internet of Things is a revolutionary approach for future technology enhancement: a review, Springer Open, pp. 1-21, 2019.

[10] Zainab H. Ali, Hesham A. Ali, Mahmoud M. Badawy, Internet of Things (IoT): Definitions, Challenges and Recent Research Directions, International Journal of Computer Applications, vol. 128-No 1, pp. 37-47, October 2021.

[11] Radouan Ait Mouha, Internet of Things (IoT), Journal of Data Analysis and Information Processing, pp. 77-101, April 2021.

[12] Muthumanickam Dhanaraju, Poongodi Chenniappan, Kumaraperumal Ramalingam, Sellaperumal Pazhanivelan and Raghunath kaliaperumal, Smart Farming: Internet of Things (IoT)-Based Sustainable Agriculture, MPDI, October 2022.

[13] https://projects.raspberrypi.org/en/projects/raspberry-pi-getting-started

[14] https://components101.com/sensors/dht11-temperature-sensor

[15] https://medium.com/technology-hits/simplified-raspberry-pi-plant-watering-system-942099e4e2cd

[16] https://www.elprocus.com/soil-moisture-sensor-working-and-applications/

[17] https://www.instructables.com/Object-Detection-on-Raspberry-Pi/

[18] https://robu.in/how-to-connect-relay-to-raspberry-pi-3/

[19] https://www.digikey.com/en/maker/tutorials/2021/how-to-control-servo-motors-with-a-raspberry-pi

[20] https://raspberrypi-guide.github.io/electronics/using-usb-webcams