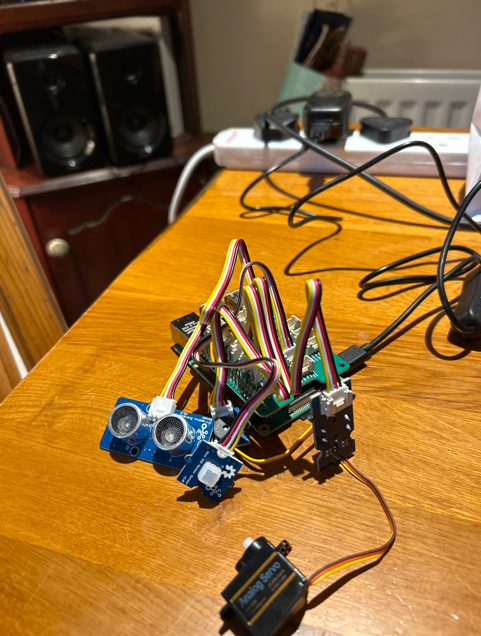
Smart IoT Pet Feeder – Final Report

# 1. Introduction

This project presents a Smart IoT Pet Feeder system designed to detect pet presence, automatically dispense food, and report telemetry using MQTT and Blynk. It supports remote interaction and real-time monitoring through mobile and cloud-based systems(mqtt telemetry to mqtt explorer).

# 2. Hardware and Electronics

The system uses the following Grove components:  
- PIR Motion Sensor  
- Ultrasonic Ranger  
- Servo Motor  
- LED  
- Buzzer  
- LCD Display (optional)  
- Fan (powered directly by 5V/GND)

GPIO Mapping:  
- PIR Sensor: GPIO 16 (D16) – input  
- Ultrasonic Sensor: GPIO 18 – distance measurement  
- Servo Motor: GPIO 24 – PWM for flap control  
- LED: GPIO 5 – output indicator  
- Buzzer: GPIO 22 – feedback during feeding  
- LCD: I2C (SDA/SCL)  
- Fan: 5V (Pin 2) and GND (Pin 6)

# 3. Software and Code

The pet feeder system relies on two key Python scripts: one for the IoT device (Raspberry Pi), and one for the remote server (PC/laptop) that simulates cloud-side logic. These scripts interact through **MQTT.**

p**et\_feeder.py – Main IoT Script (Raspberry Pi)**

This script runs directly on the Raspberry Pi and performs all real-world interactions. Its responsibilities include:

* Initializing **GPIO pins** for all connected components:
  + **PIR motion sensor** to detect presence
  + **Ultrasonic ranger** to measure distance to the pet
  + **LED** and **buzzer** for feedback
  + **Servo motor** for dispensing food
  + **LCD display** for system messages
* Setting up a **PWM signal** to control the servo motor’s angle (open/close flap).
* Interfacing with **Blynk**, a mobile IoT dashboard, using virtual pins:
  + **V0** = Enable/Disable feeder system
  + **V1** = Motion status
  + **V2** = Distance display
* Reading sensor values and publishing **JSON telemetry** to test.mosquitto.org on topic oisin123/telemetry. Each message contains:
  + "motion" (boolean)
  + "distance\_cm" (float)
  + "timestamp" (string)
* Displaying live values on the **LCD** (distance, system status).
* Running logic to prevent false triggers:
  + Only triggers feeding when motion is detected and the distance is below 40 cm **twice in a row**.
* Choosing between:
  + big\_feed() (initial food drop) and
  + small\_feed() (shorter repeat feed after cooldown).

**pet\_feeder\_server.py – Remote Listener Script (PC/Simulated Cloud)**

This script runs on a laptop or PC and represents the cloud or backend logic. It performs the following:

* Subscribes to oisin123/telemetry and listens for incoming sensor data from the Pi.
* Parses incoming **JSON messages** and checks:
  + Is "motion" set to true?
  + Is "distance\_cm" less than 40?
* If both conditions are met, it sends a feed command:
  + Publishes {"feed\_now": true} to the oisin123/commands topic.
* The Raspberry Pi receives this message and triggers either a **big** or **small** feed depending on cooldown state.

**Technologies Used**

* RPi.GPIO – for direct hardware control
* grove.gpio – for Grove-compatible sensors (e.g., ultrasonic ranger)
* paho-mqtt – for MQTT publish/subscribe
* BlynkLib – for mobile app control and feedback
* json, time – for data formatting and delays
* JHD1802 – for controlling the I2C 16x2 LCD screen

**A computer screen shot of a program code

AI-generated content may be incorrect.📸 Screenshot 1: MQTT & Blynk Setup  
Caption:  
*Initialises MQTT connection to test.mosquitto.org and links Blynk virtual pins for remote control (V0 = enable, V1 = motion, V2 = distance).***

# *📸 Screenshot 2: GPIO & Hardware Setup Caption: Maps each hardware component (PIR, LED, buzzer, servo, ultrasonic) to a specific GPIO pin. Sets up servo PWM and sensor initialization.*

# *Screenshot 3: Feeding Functions Caption: Defines big\_feed() and small\_feed() functions for different food portions. Activates servo, LED, buzzer, and updates LCD screen.*

# A screen shot of a computer program AI-generated content may be incorrect.

# A screen shot of a computer*📸 Screenshot 4: Main Loop & Feeding Trigger Caption: Main decision loop: checks motion + distance, publishes telemetry to MQTT, updates Blynk and LCD, and triggers feeding after 2 detections.*

# A screen shot of a computer program AI-generated content may be incorrect.*Screenshot 5: Server-Side MQTT Trigger Caption: Listens to telemetry from the Pi. If motion is true and distance is below 40 cm, sends a command (feed\_now: true) back to the Pi via MQTT.*

# A bag of dog food AI-generated content may be incorrect.4. Challenges and Solutions

- Servo jitter resolved by resetting PWM with ChangeDutyCycle(0)  
- Blynk unresponsiveness reduced by limiting time.sleep() usage and using updated import   
- Telemetry delay fixed by improving message structure and MQTT timing

# 5. Demo Video

A short demo video has been recorded and uploaded to YouTube. [https://youtu.be/l2gy-SSyuCk]