**Part-A**

**Step- 1: Understanding and Defining the Problem Analysis:**

The problem aim is to design a low-cost, programmable automated pet feeder that can reliably serve both cats and dogs. The automatic pet feeder must detect whether the food has been eaten, dispense the correct amount of food for dogs and cats at set times, and give an alert to any issues, for example, no food dispensed, leftover food, or mechanical problems. The system should be affordable, programmable, and easy to implement using basic sensors and servo motors.

**Key features the feeder must have:**

**Scheduled feeding**: Each feeder can dispense different amounts of food at set times throughout the day.

**Food consumption monitoring:** The system determines whether the bowl weight decreases sufficiently to determine whether the meal is consumed within a predetermined period of time, for example, 30 minutes.

**Actuator control**: A motor (servo ) dispenses the correct amount of food by rotating at a set angle or operating for a specified duration.

**Alert light status:**

**Green LED:** When feeding is successful or a meal is delivered and consumed on schedule, the green LED status light goes on.

**Red LED:** If there is an issue, for instance, no food provided, jam, an empty storage, or food not consumed, the red LED goes on.

**Buzzer alert and system log:** Give an alert with beeps and keep records of the incident for future analysis.

**Log system**: Time, food quantity, outcome, and any issues for every feeding are all saved with a log system.

**Manual feeding control:** Instantly feed the pet using the manual feed button, but first must need to check for any issues.

**Food dispensing verification:** Following feeding, the system uses an infrared sensor that detects the presence of food or a weight sensor that makes the bowl heavier to determine whether food is in the bowl.

**Assumptions and limitations**:

**Assumptions:** Using a servo motor with open-loop timing, the feeder will only dispense dry kibble in a set size. It will make use of a hopper level sensor, a load cell or infrared sensor at the bowl, and an optional jam detection switch. The feeding schedule will be stored locally on a 24-hour clock with LEDs and an alert buzzer by a low-power microcontroller with little memory. When the bowl weight is decreased by ≥ M grams or the IR sensor clears in a predetermined amount of time, for example, 30 minutes, food is deemed eaten.

**Limitations:** The feeder can only feed one animal at a time and is intended for indoor use only. It depends on timed operation because of its open-loop motor control, which makes it impossible to measure the dispenser's precise movement. The quantity of feeding events or logs that can be stored may be limited by memory, and in the absence of cloud or app connectivity, all alerts and logs will only be accessible via local indicators, such as buzzers, LEDs, or on-device storage.

**Block diagram for an automated pet feeder:**

**A diagram of a system

AI-generated content may be incorrect.**

**Step-2: Organize and describe the data**

**Data table for inputs:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input name** | **Type/unit** | **Data range** | **Data source** | **Data function** |
|  |
| Time reading | Time | 00:00-23:59 | Real-time clock | To start feeding at the appropriate time, compare with the timetable. |  |
| scheduled feeding time | Portion size and time(g) | 8.30,18.30,40g | Local storage | stores portion quantities and feeding schedules. |  |
| Manual feeding control | Boolean(0/1) | 0=off,1=on | Push button | manually feed outside of the scheduled time. |  |
| Food storage level | Boolean(0/1) | 0=low,1=ok | Storage level sensor | makes sure the hopper has enough food. |  |
| Jam detected | Boolean(0/1) | 0=no jam,1=jam | Jam detection sensor | checks for blockages in the dispenser. |  |
| Food bowl weight | Numeric | 0 to 4000g | Bowl weight sensor | determines the amount of food in the dish. |  |
| Food detected | Boolean(0/1) | 0=no,1=yes | Infrared bowl sensor | determines whether food is present. |  |

**Data table for outputs:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Output name** | **Unit/type** | **Data range** | **Data source** | **Data function** |
|  |
| Dispense control system | Run time(ms) | 1200ms | Control unit output | powers the dispensing part with a motor. |  |
| Buzzer output | Boolean(0/1) | 0=off,1=on | Control unit output | sounds that are on the lookout for errors. |  |
| Operation ok(Green-LED) | Boolean(0/1) | 0=off,1=on | Control unit output | Displays a successful feeding cycle. |  |
| Error status(Red-LED) | Boolean(0/1) | 0=off,1=on | Control unit output | indicates a problem (not eaten, jam, or empty hopper). |  |
| System log entry | Log data and time | 8.30, 40g serve,30g eaten-ok | Stored in log system | keeps the feeding history for further use. |  |

**Data table for operational parameters and constraints:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Parameter** | **Unit** | **Sample value** | **Data source** | **Data limit/Constraint** |
|  |
| Minimum consumption level | gram | cat-18g,dog-50g | User-define config | The minimum quantity consumed to be considered successful |  |
| Consumption time limit | minutes | 30min | User-define config | Maximum amount of time to eat before being alert |  |
| Feeding entry allowance | count | 1 | User-define config | Avoids motor wear and overfeeding |  |
| Maximum daily feedings | Portion | up to 6 times daily | User-define config | Prevents overfeeding |  |
| Log storage capacity | events | 600log | Hardware specification | In full logs, the oldest logs are overwritten. |  |
| sensor accuracy | gram | up to 3–5 grams higher or lower | sensor technical data | Noise reduction through averaging |  |
| Motor activation duration | ms | 1200ms | Calibration data | 3000 ms maximum to prevent motor overheating |  |
| Dispense amount | gram | 40g | User-define config | Depending on the species and dietary pattern |  |

**Step- 3: Solution plan and design the algorithm:**

**A diagram of a flowchart

AI-generated content may be incorrect.**

**Step-4: Implement the solution:**

Step 1: Wait for Trigger  
The system starts and waits for a feeding trigger. A trigger happens when the current time matches a scheduled feeding time (e.g., 8:00 AM or 6:00 PM) or when the manual feed button is pressed. If no trigger is detected, the system keeps checking until one occurs.

Step 2 — Check Hopper Level  
Once triggered, the system checks the hopper’s food level using a sensor. If the hopper is low, it turns on the red LED, sounds the buzzer, logs the event as “Hopper Low,” and ends the feeding cycle. If there is enough food, it moves to the next step.

Step 3 — Check for Jam  
The system checks for a jam in the dispenser. If a jam is detected, the red LED turns on, the buzzer sounds, “Jam Detected” is logged, and the feeding cycle ends. If no jam is detected, the system records the bowl’s current weight before dispensing.

Step 4 — Dispense Food  
The system activates the motor for a calibrated time (e.g., 1200 milliseconds) to release the set portion of food into the bowl. This timing is chosen to deliver the correct weight of food.

Step 5 — Verify Food Dispensed  
After dispensing, the system checks if food is actually in the bowl. This is done using the infrared food sensor or by checking if the bowl’s weight has increased. If no food is detected, the red LED turns on, the buzzer sounds, “No Food Dispensed” is logged, and the cycle ends. If food is present, it moves to the next step.

Step 6 — Start Eating Timer  
The system starts a 30-minute timer to allow the pet time to eat. During this period, it will monitor the food consumption.

Step 7 — Check Food Consumption  
While the timer is running, the system checks if the pet has eaten enough food. This is confirmed if the bowl’s weight decreases by at least the minimum threshold or if the infrared sensor shows the food is gone. If the food is consumed, the green LED turns on, “Feeding Successful” is logged, and the cycle ends. If the timer runs out and food is not eaten, the red LED turns on, the buzzer sounds, “Food Not Eaten” is logged, and the cycle ends.

End of Cycle  
After either a success or an alert, the system returns to waiting for the next scheduled feeding time or manual trigger.

**Step 4: Implement the Solution (Word Coding)**

1. Once upon a time… the feeder powers on and gets ready.
2. It waits for the right moment: either the clock matches a feeding time **or** the manual feed button is pressed.
3. If nothing happens, it keeps waiting and checking again.
4. When triggered, it first checks the hopper to see if there’s enough food.
5. If the hopper is low, it turns on the red LED, beeps the buzzer, logs **“Hopper Low”**, and stops this cycle.
6. If the hopper is okay, it checks the dispenser for a jam.
7. If a jam is detected, it turns on the red LED, beeps the buzzer, logs **“Jam Detected”**, and stops this cycle.
8. If there is no jam, it runs the motor for the calibrated time to dispense a portion.
9. It then verifies that food actually reached the bowl (weight increased or the IR sensor sees food).
10. If no food is detected, it turns on the red LED, beeps, logs **“No Food Dispensed”**, and stops this cycle.
11. If food is present, it starts a 30-minute eating timer.
12. While the timer runs, it checks whether the pet has eaten enough (weight dropped by the minimum amount or the IR shows the bowl is clear).
13. If the food is eaten within the time, it turns on the green LED, logs **“Feeding Successful”**, and ends the cycle.
14. If the time runs out and the food is not eaten, it turns on the red LED, beeps, logs **“Food Not Eaten”**, and ends the cycle.
15. Then it goes back to waiting for the next feeding time or button press.

**Step 5: Test and Refine the Solution (Debug & Verify)**

The system was tested with several scenarios:

1. **Pet eats as expected** – trigger at feeding time, hopper and jam OK, motor dispenses, food detected, and consumption ≥25 g within 30 minutes.

Result: green LED, log “Feeding Successful” – matched expectations.

(2) **Pet does not eat** – feeding occurs but weight change is <25 g after 30 minutes, red LED, buzzer, and log “Food Not Eaten” – matched expectations.

(3) **Hopper empty** – hopper sensor low at trigger, red LED, buzzer, and log “Hopper Low” – matched expectations.

(4) **Jam detected** – jam sensor active before dispensing, red LED, buzzer, log “Jam Detected” – matched expectations.

(5) **Dispense fails** – motor runs but no food detected and weight unchanged, red LED, buzzer, log “No Food Dispensed” – matched expectations. In all cases, outputs matched the flowchart logic.

Refinements suggested include:

adding a single automatic re-try if dispensing fails, debouncing the manual button to prevent accidental feeds, ending the timer early once enough food is eaten, adding a brief grace period for slow eaters, calibrating motor time and thresholds per pet type, improving log detail for troubleshooting, and setting a daily feed limit for safety. These changes would make the system more reliable and user-friendly while keeping it low-cost.

START

1. Wait for Trigger:

IF (Current Time matches Scheduled Feeding Time) OR (Manual Feed Button pressed)

→ Proceed to Step 2

ELSE

→ Loop back and wait.

2. Check Hopper Level:

IF (Hopper Low)

→ Turn ON Red LED, Sound Buzzer, Log "Hopper Low", END.

ELSE

→ Proceed to Step 3.

3. Check for Jam:

IF (Jam Detected)

→ Turn ON Red LED, Sound Buzzer, Log "Jam Detected", END.

ELSE

→ Proceed to Step 4.

4. Dispense Food:

Activate Motor for calibrated duration (e.g., 1200ms).

5. Verify Food Dispensed:

IF (Food Detected in Bowl OR Bowl Weight increased)

→ Proceed to Step 6.

ELSE

→ Turn ON Red LED, Sound Buzzer, Log "No Food Dispensed", END.

6. Start Eating Timer:

Begin countdown (e.g., 30 minutes).

7. Check Food Consumption:

IF (Bowl Weight decreased ≥ Minimum Threshold OR IR Sensor shows food cleared)

→ Turn ON Green LED, Log "Feeding Successful", END.

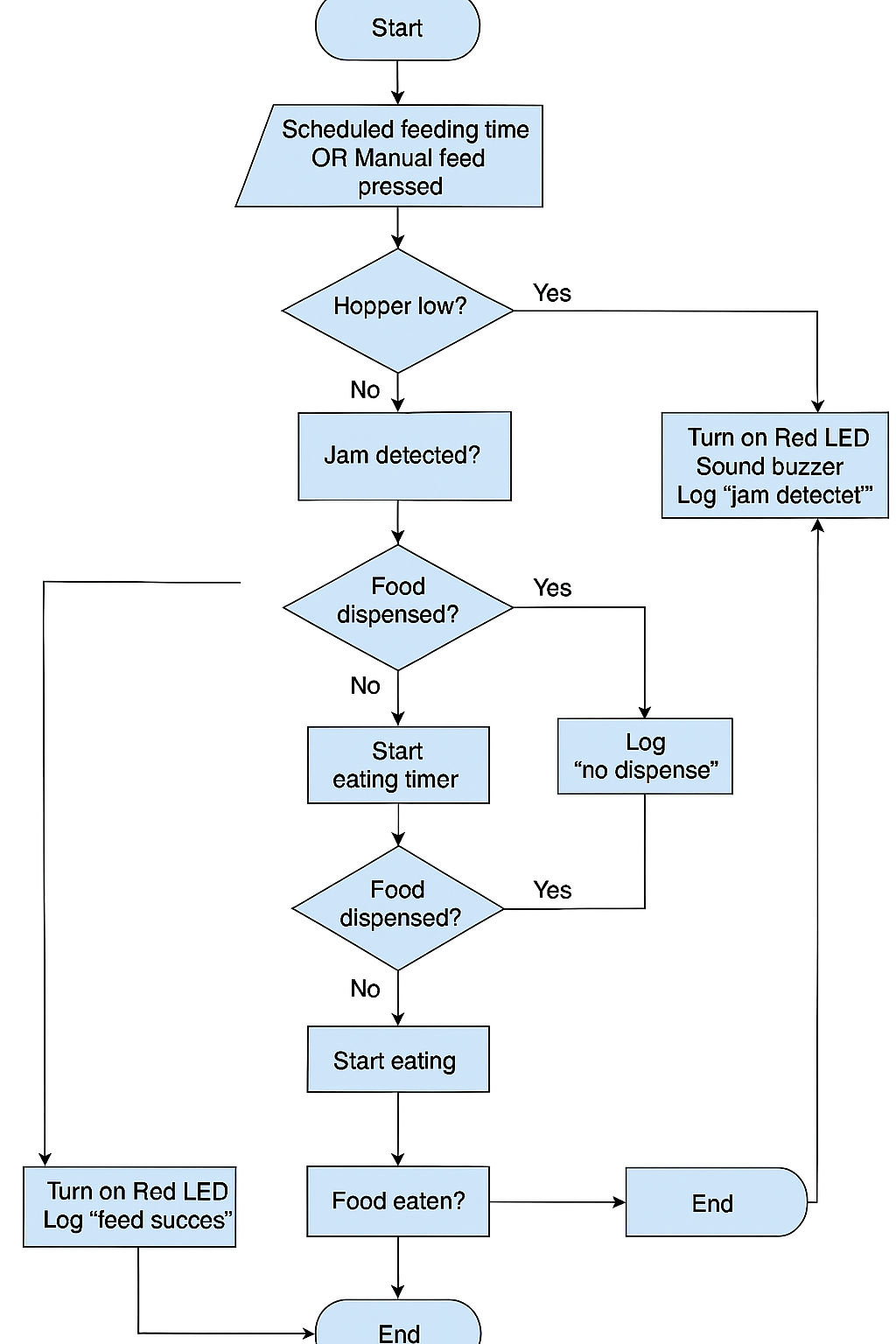
ELSE

→ Turn ON Red LED, Sound Buzzer, Log "Food Not Eaten", END.

END

**Step-wise Pseudocode (exact match to flowchart)**

1. Start
2. Trigger check  
   Wait until Scheduled feeding time OR Manual feed is pressed.  
   (If not triggered, stay here.)
3. Hopper low?
   * Yes → Turn Red LED ON, sound buzzer, log "hopper low" → End.
   * No → go to Step 4.
4. Jam detected?
   * Yes → Turn Red LED ON, sound buzzer, log "jam detected" → End.
   * No → go to Step 5.
5. Food dispensed? (first check)
   * Yes → go to Step 6.
   * No → log "no dispense" → End.
6. Start eating timer → go to Step 7.
7. Food dispensed? (second check during timer)
   * Yes → Turn Red LED ON, log "feed success" → End.
   * No → go to Step 8.
8. Start eating → go to Step 9.
9. Food eaten?
   * Yes → End.
   * No → Turn Red LED ON, log "feed success" → End.
10. End



[Pritihossain/u3221003\_Assignment\_1: Introduction to information technology](https://github.com/Pritihossain/u3221003_Assignment_1)