**PART 1: On the Solving Problem Process**

**IMPLEMENTING THE SOLUTION (WORD CODING)**

Variables which can be included in the logic are:

|  |  |  |
| --- | --- | --- |
| **Variables** | **Sample Values / Examples** | **Comments** |
| currentTime | systemClock() | Real–Time Clock |
| feedingSchedule[] | [“08:00”, “18:00”] | Pre-set Feeding Time |
| portionSize[] | [30,50,80] | Grams in the bowl and motor runs accordingly |
| bowlEmpty | checkBowlSensor() | Boolean: TRUE = Empty, FALSE = Full |
| motorRunTime | 0 | Seconds (depends on portion size) |
| foodDispensed | FALSE(default value) | As no food is dispensed yet in this cycle |
| foodEaten | FALSE(default value) | Food cannot be eaten until it is dispensed |

* **currentTime, bowlEmpty, motorRunTime** these three variables are very meaningful as it convicts to check the basic modular task that is check time then check bowl then run motor then monitor food consumption then prepare for next feed.
* And **bowlEmpty, foodEaten, foodDispensed** these handles the error in system.

WHILE systemPower = ON:

**# 1. Check Current Time**

currentTime: systemClock()

IF currentTime = feedingSchedule THEN

**# 2. Check Bowl Status**

bowlEmpty = checkBowlSensor()

IF bowlEmpty = TRUE THEN

alertStaff("**Fill Bowl**") # Bowl has no food, needs to be filled

logEvent("**Alert is Bowl Empty at " + currentTime**)

CONTINUE # Skip to next cycle

ENDIF

**# 3. Dispense Food**

motorRunTime = getMotorTimeForPortion(currentTime)

runMotor(motorRunTime)

logEvent("**Food is Dispensed at " + currentTime**)

foodDispensed = TRUE

**# 4. Post-Dispense Monitoring**

WAIT 10 minutes

foodEaten = checkBowlSensor() # Check if bowl became empty

IF foodEaten = FALSE THEN

triggerBuzzer()

logEvent(**"Food Not Eaten, Alert Staff"**)

ELSE

logEvent("**Feeding Successful, Food Eaten**")

ENDIF

**# 5. Prepare for Next Feeding**

updateDisplay("**Next Feed Scheduled**")

setSystemIdle()

ELSE

# No feeding time matched – continue monitoring

CONTINUE

ENDIF

END WHILE