WEEKEND ASSIGNMENT 1

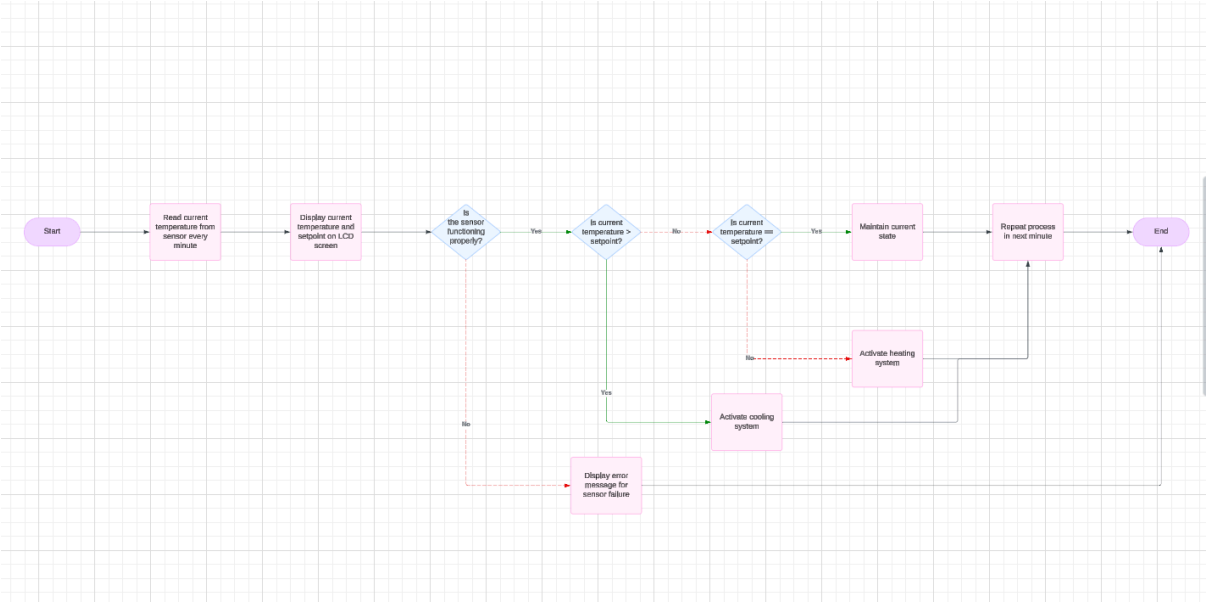
Write the Pseudocode and Flowchart for the problem statements mentioned below:

---

1. Smart Home Temperature Control Problem Statement: Design a temperature control system for a smart home. The system should read the current temperature from a sensor every minute and compare it to a user-defined setpoint. Requirements: • If the current temperature is above the setpoint, activate the cooling system. • If the current temperature is below the setpoint, activate the heating system. • Display the current temperature and setpoint on an LCD screen. • Include error handling for sensor failures.

1)define setpoint
2)define temperaturesensor
2)initialize LCD screen for displaying temperature
3)loop()
4)read temperature from sensor
5)if(currentTemperature > setpoint) then
ACTIVATE cooling system
DEACTIVATE heating system
else if(currentTemperature < setpoint) then
ACTIVATE heating system
DEACTIVATE cooling system

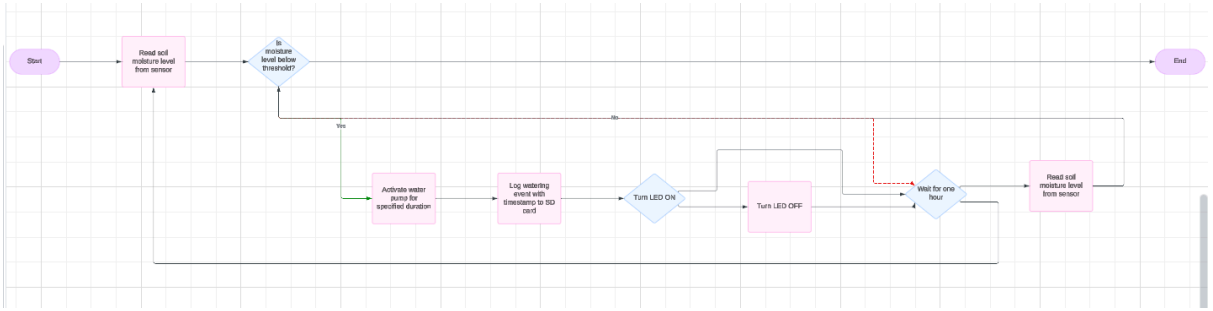
else if(currentTemperature == setpoint) then
DEACTIVATE heating and cooling system
6)if there is a sensor failure
display error on LCD screen



2. Automated Plant Watering System Problem Statement: Create an automated watering system for plants that checks soil moisture levels and waters the plants accordingly.

Requirements: • Read soil moisture level from a sensor every hour. • If moisture level is below a defined threshold, activate the water pump for a specified duration. • Log the watering events with timestamps to an SD card. • Provide feedback through an LED indicator (e.g., LED ON when watering).

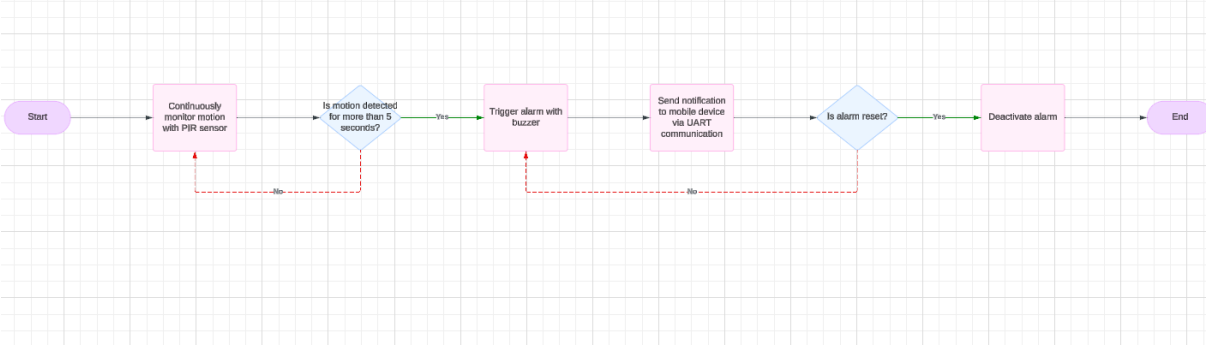
#define Threshold 40
#define Watering_Duration 5
initialize soilmoisture_sensor
initialize water pump
initialize LED
every hour
Try
read soilmoisture_senor
IF (soilmoistureLevel < Threshold) THEN
ACTIVATE water pump
Turn on LED
Log "Watering event" with timestamp to SD card
Wait watering_Duration
DEACTIVATE water pump
Turn off LED
else
Log "Moisture level adequate" with timestamp to SD card
Catch sensor failure:
Log "Sensor failure" with timestamp to SD card
DEACTIVATE water pumb



3. Motion Detection Alarm System Problem Statement: Develop a security alarm system that detects motion using a PIR sensor. Requirements: • Continuously monitor motion detection status. • If motion is detected for more than 5 seconds, trigger an alarm (buzzer). • Send a notification to a mobile device via UART communication. • Include a reset mechanism to deactivate the alarm.

#define motionduration = 5 seconds
Initialize PIR_sensor
initialize alarmstate = off
Initialize buzzer
Initialize UART
Initialize reset_button

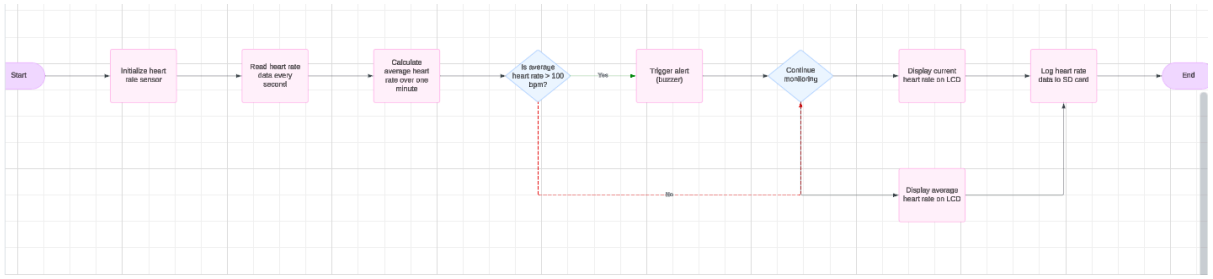
Initialize MotionCount = 0
Loop()
if PIR_sensor detects motion
MotionCount += 1
if(MotionCount >= motionduration) then
Activate Alarm
send "motion detected" notification through UART
else reset MotionCount
if reset_button pressed
Deactivate alarm
reset MotionCount
end loop



4. Heart Rate Monitor Problem Statement: Implement a heart rate monitoring application that reads data from a heart rate sensor. Requirements: • Sample heart rate data every second and calculate the average heart rate over one minute. • If the heart rate exceeds 100 beats per minute, trigger an alert (buzzer). • Display current heart rate and average heart rate on an LCD screen. • Log heart rate data to an SD card for later analysis.

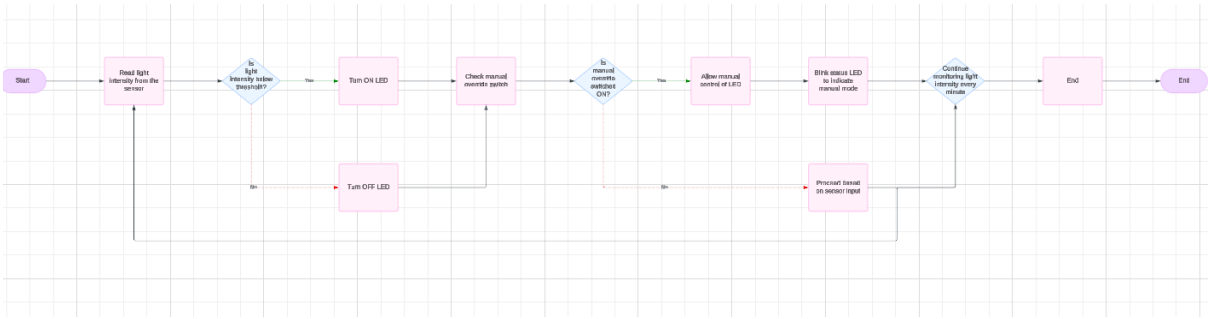
#define HeartRateThreshold = 100
#define Avg_Calc = 60
Set HeartRateSensor
initialize HeartRateReadings = 0
Set buzzer
initialize LCD
initialize SD card
Set CurrentHeartRate = 0
initialize Time = 1
HeartRateData[]
loop()
Set CurrentHeartRate = Read HeartRateSensor
Append CurrentHeartRate with HeartRateData
HeartRateReadings += CurrentHeartRate
Time += 1
End loop()
Avg_HeartRate = HeartRateReadings / Avg_Calc
if (Avg_HeartRate > HeartRateThreshold) Then

Activate Buzzer
Display alert
Display "Current Heartrate: " + CurrentHeartRate on LCD
Display "Average Heartrate: " + Avg_HeartRate on LCD
Log HeartRateData to an SD card for later analysis



5. LED Control Based on Light Sensor Problem Statement: Create an embedded application that controls an LED based on ambient light levels detected by a light sensor. Requirements: • Read light intensity from the sensor every minute. • If light intensity is below a certain threshold, turn ON the LED; otherwise, turn it OFF. • Include a manual override switch that allows users to control the LED regardless of sensor input. • Provide status feedback through another LED (e.g., blinking when in manual mode).

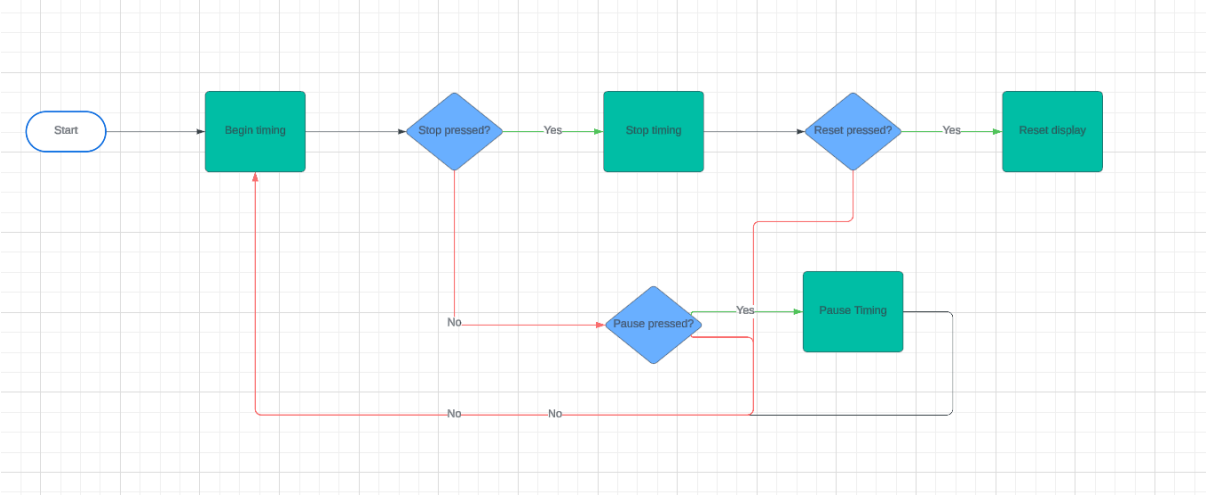
define Threshold
initialize light sensor
initialize LED
initialize override switch
for every minute
if(manual override switch == on)
turn on LED feedback status
else
read light intensity
if (light intensity < Threshold)
turn on LED
else
Turn off LED



6. Digital Stopwatch Problem Statement: Design a digital stopwatch application that can start, stop, and reset using button inputs. Requirements: • Use buttons for Start, Stop, and Reset functionalities. • Display elapsed time on an LCD screen in hours, minutes, and

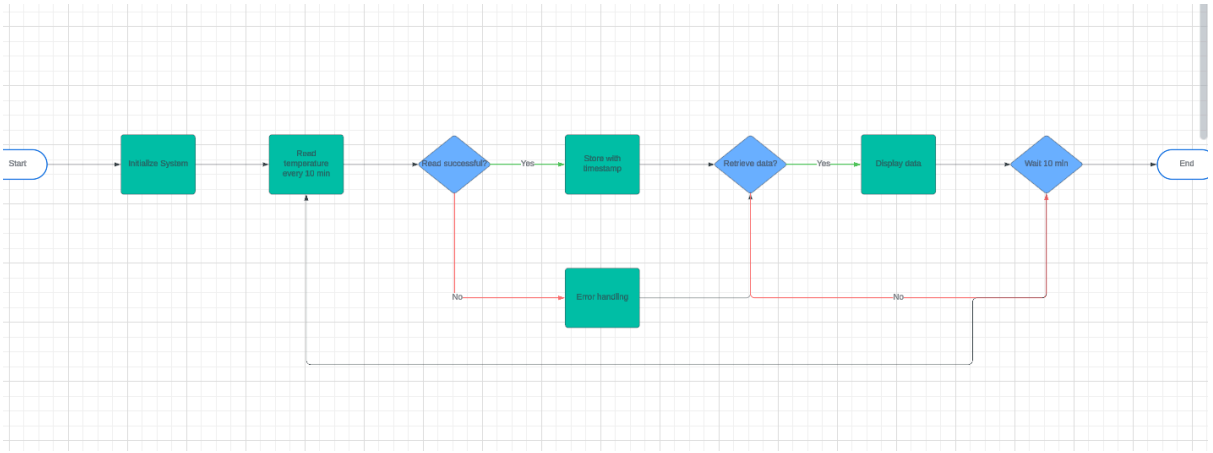
seconds format. • Include functionality to pause and resume timing without resetting. • Log start and stop times to an SD card when stopped.

initialize start, stop and reset button
initialize LCD screen
initialize SD card for logging
if start button is pressed then
Log starttime to SD card
if stop button is pressed then
Log stoptime to SD card
if reset button is pressed then
Display time on LCD



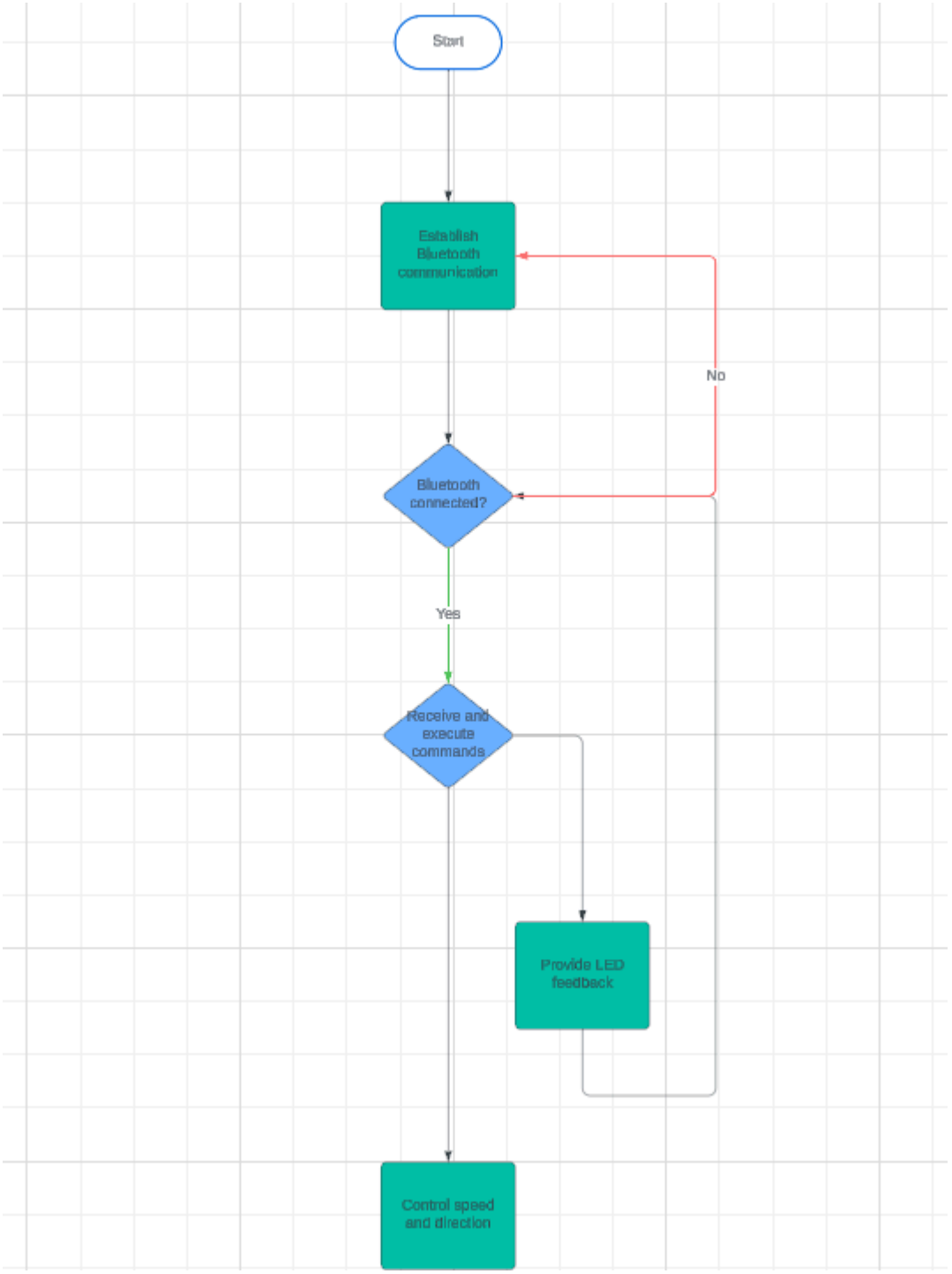
7. Temperature Logging System Problem Statement: Implement a temperature logging system that records temperature data at regular intervals. Requirements: • Read temperature from a sensor every 10 minutes. • Store each reading along with its timestamp in an array or log file. • Provide functionality to retrieve and display historical data upon request. • Include error handling for sensor read failures.

initialize TemperatureSensor
initialize currentTemperature and currentTimestamp
initialize errorFlag
for every 10 minutes
SET currentTemperature = READ TemperatureSensor
IF currentTemperature is invalid Then
SET errorFlag = true
HANDLE error
else SET errorFlag = false
SET currentTimestamp = GET current time from RTC
STORE currentTemperature and currentTimestamp in storage
IF user requests data THEN
RETRIEVE data from storage DISPLAY



8. Bluetooth Controlled Robot Problem Statement: Create an embedded application for controlling a robot via Bluetooth commands. Requirements: • Establish Bluetooth communication with a mobile device. • Implement commands for moving forward, backward, left, and right. • Include speed control functionality based on received commands. • Provide feedback through LEDs indicating the current state (e.g., moving or stopped).

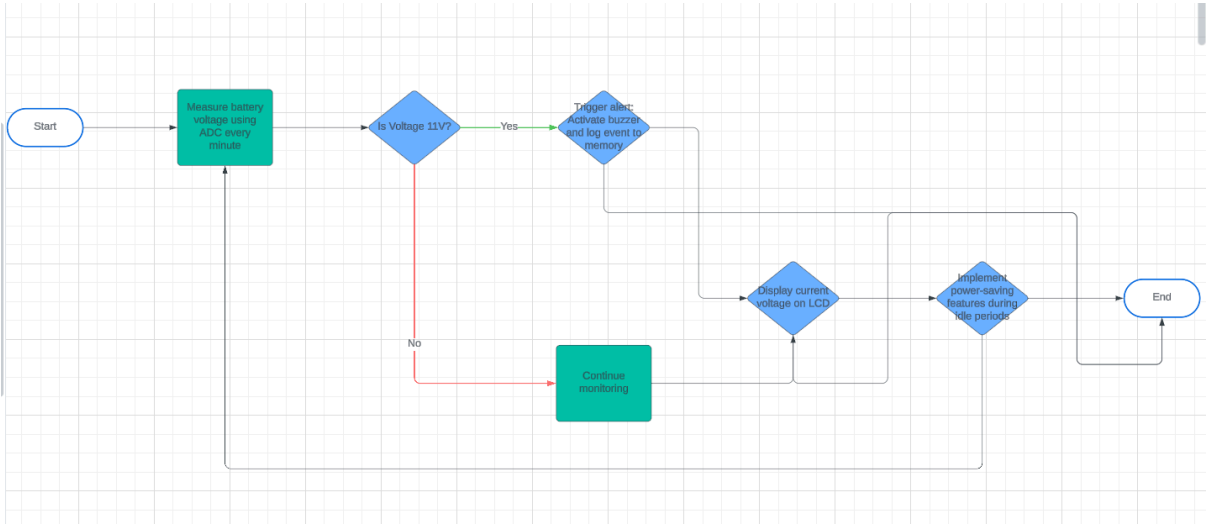
initialize Bluetooth
initialize LED for feedback
read data from Bluetooth
process commands
if command is "forward"
  set device to move forward
  provide feedback through LED
else if command is "backward"
  set device to move backward
  provide feedback through LED
else if command is "right"
  set device to move right
  provide feedback through LED
else if command is "left"
  set device to move left
  provide feedback through LED
else if command is "speed"
  adjust device speed
else if command is "stop"
stop device



9. Battery Monitoring System Problem Statement: Develop a battery monitoring system that checks battery voltage levels periodically and alerts if voltage drops below a safe threshold. Requirements: • Measure battery voltage every minute using an ADC (Analog-to-Digital Converter). • If voltage falls below 11V, trigger an alert (buzzer) and log the event to memory. • Display current voltage on an LCD screen continuously. • Implement power-saving features to reduce energy consumption during idle periods.

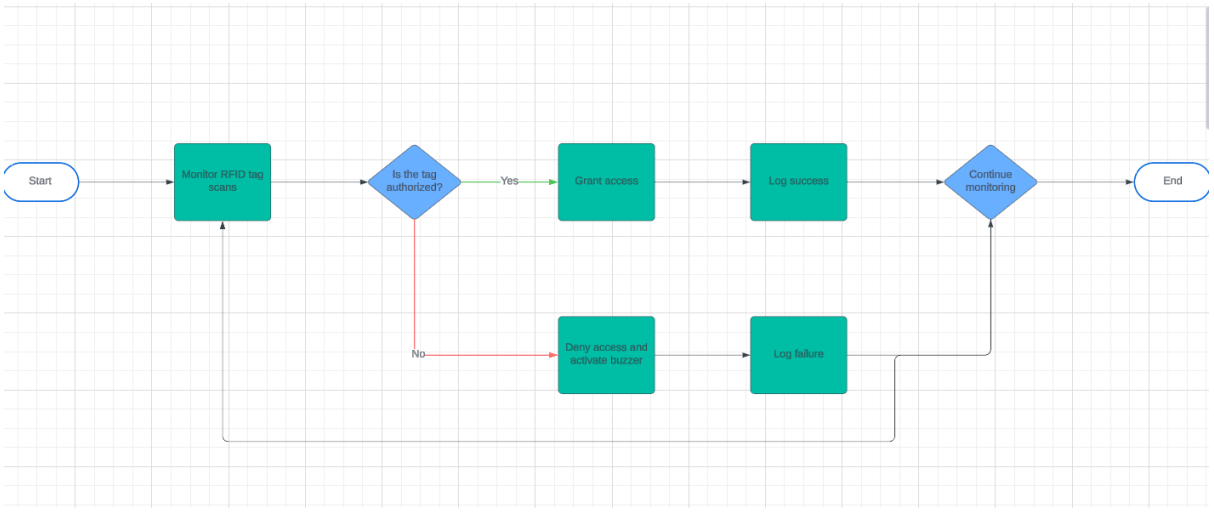
define Threshold 11V
initialize buzzer

initialize ADC
initialize LCD screen
for every minute
 evaluate voltage using ADC
 Display current voltage on LCD
 if voltage < Threshold
    Enable buzzer
    Display current voltage on LCD
 else
    Disable buzzer
    Display current voltage on LCD



10. RFID-Based Access Control System Problem Statement: Design an access control system using RFID technology to grant or deny access based on scanned RFID tags. Requirements: • Continuously monitor for RFID tag scans using an RFID reader. • Compare scanned tags against an authorized list stored in memory. • Grant access by activating a relay if the tag is authorized; otherwise, deny access with an alert (buzzer). • Log access attempts (successful and unsuccessful) with timestamps to an SD card

.initialize RFID reader
initialize buzzer
initialize relay
initialize SD
loop()
read scanned tag from RFID reader
if(scanned tag is in authorized list)
   Activate relay to grant access
   Log "Access granted" to SD card
else
    Activate buzzer
    Log "Access denied" to SD card
end


Start
Monitor RFID tag scans
Is the tag authorized?
Yes
Grant access
Log success
Continue monitoring
End
No
Deny access and activate buzzer
Log failure