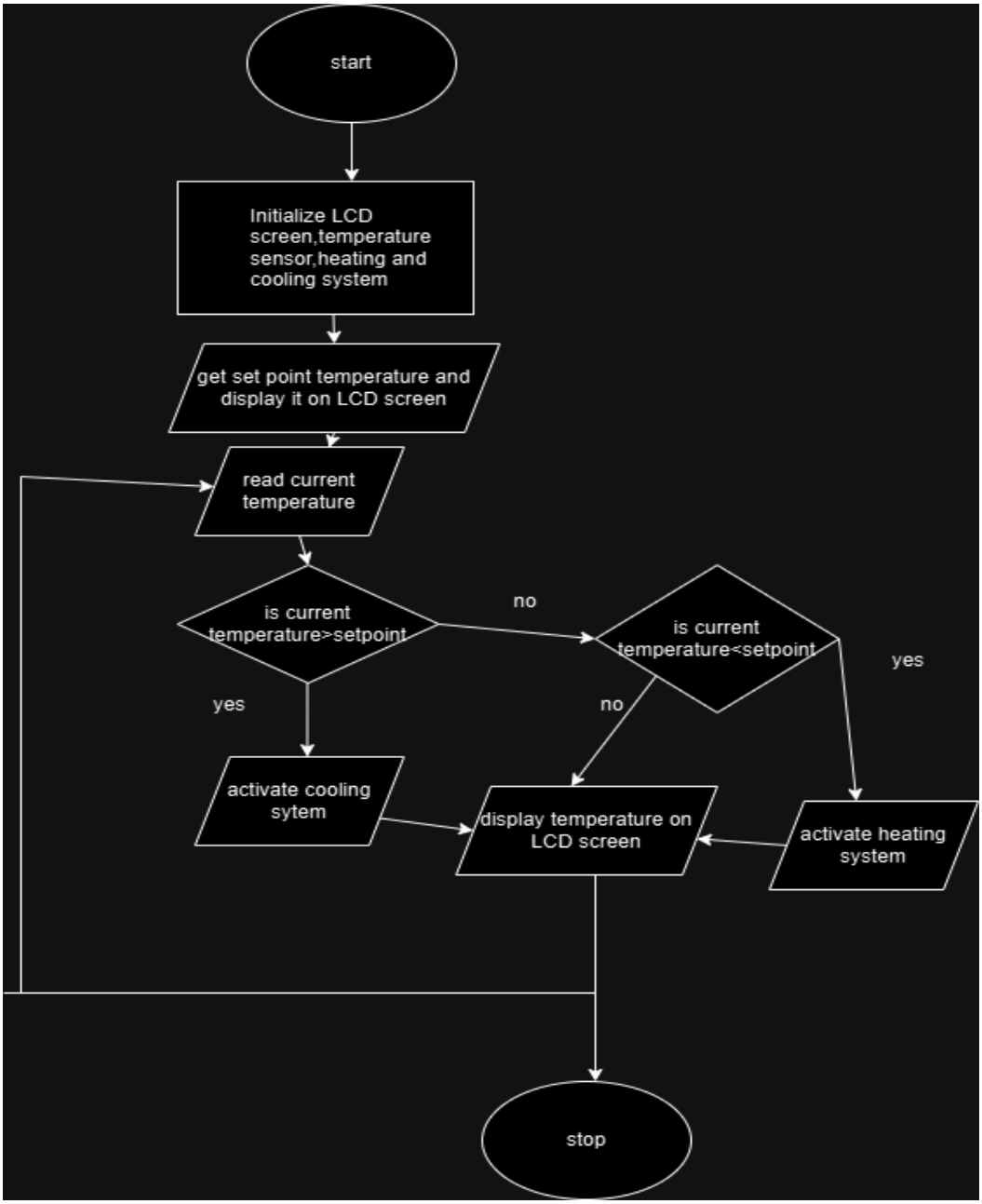
WEEKEND ASSIGNMENT

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.

   PSEUDOCODE:
   1.Initialize LCD screen
   2.Intialize Temperature sensor
   3.Initialize cooling and heating system
   4. Set the setpoint(user defined)
   Display the setpoint on the LCD screen
   5.Every minute:
   Try:
   Read current temperature
   If current temperature>setpoint
   Display the current temperature on LCD screen
   Activate cooling system
   Else if current temperature<setpoint
   Display the current temperature on LCD screen
   Activate heating system
   Else:
   Display stable temperature on LCD screen

   Catch sensor failure:
   Display error message on LCD screen
   Deactivate heating and cooling system



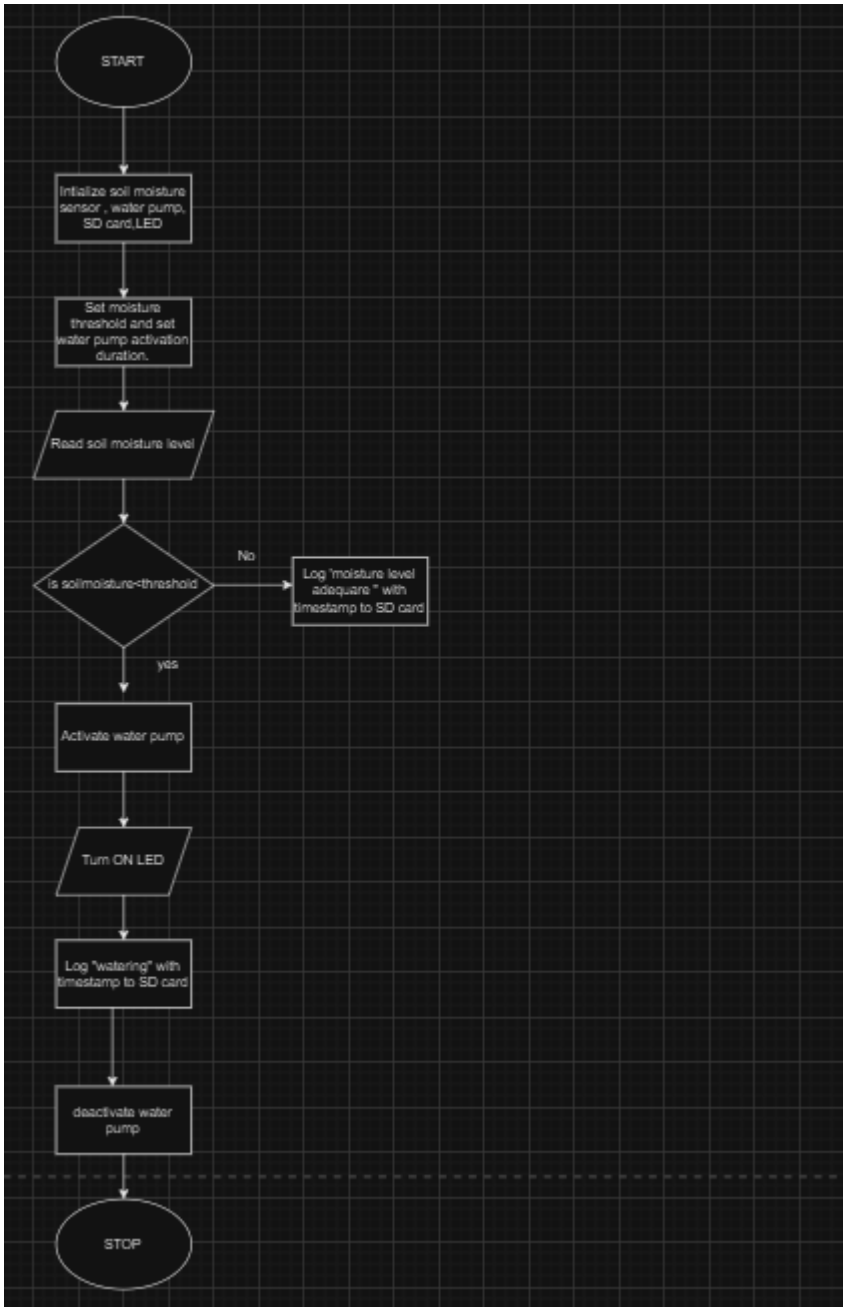
2.**Automated Plant Watering System**

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)

PSEUDOCODE

```
START
Initialize soil moisture sensor
Initialize water pump
Initialize SD card for logging watering events
Initialize LED indicator
Set moisture threshold
Set water pump activation duration
every hour:
Try:
Read soil moisture level
  if soil moisture level < threshold
        Turn ON LED indicator
        Activate water pump for specified duration
        Log "Watering event" with timestamp to SD card
        Turn OFF water pump(after specified duration)
        Turn OFF LED indicator
    Else
        Log "Moisture level adequate" with timestamp to SD card
Catch sensor failure:
        Log "Sensor failure" with timestamp to SD card
        Deactivate water pump
```



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.

PSEUDOCODE

START

Initialize PIR sensor

Intialize buzzer

Intialize UART communication

Check /Read PIR sensor status

If motion detected

If time>5seconds:

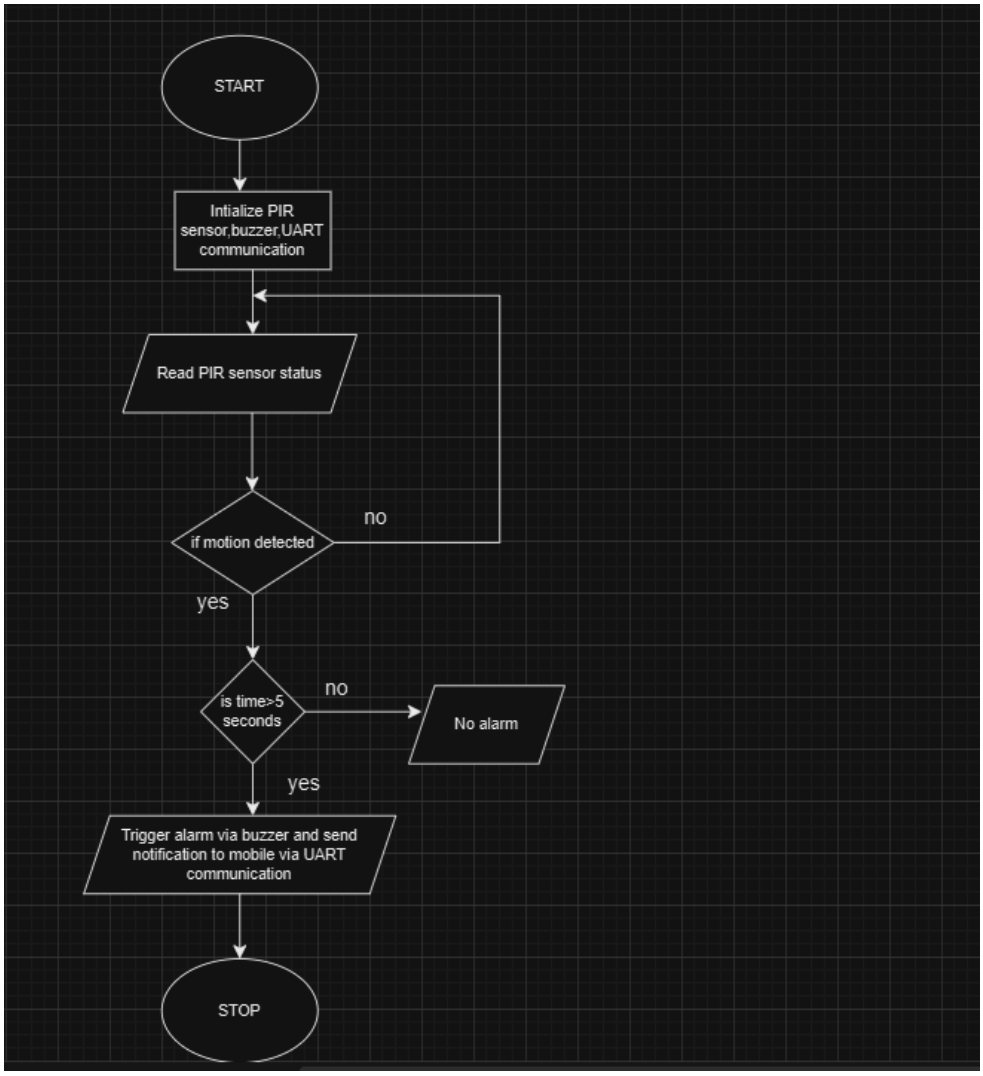
Trigger alarm via buzzer

Send notification to mobile via UART communication

Else:

No alarm

Reset the system



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.

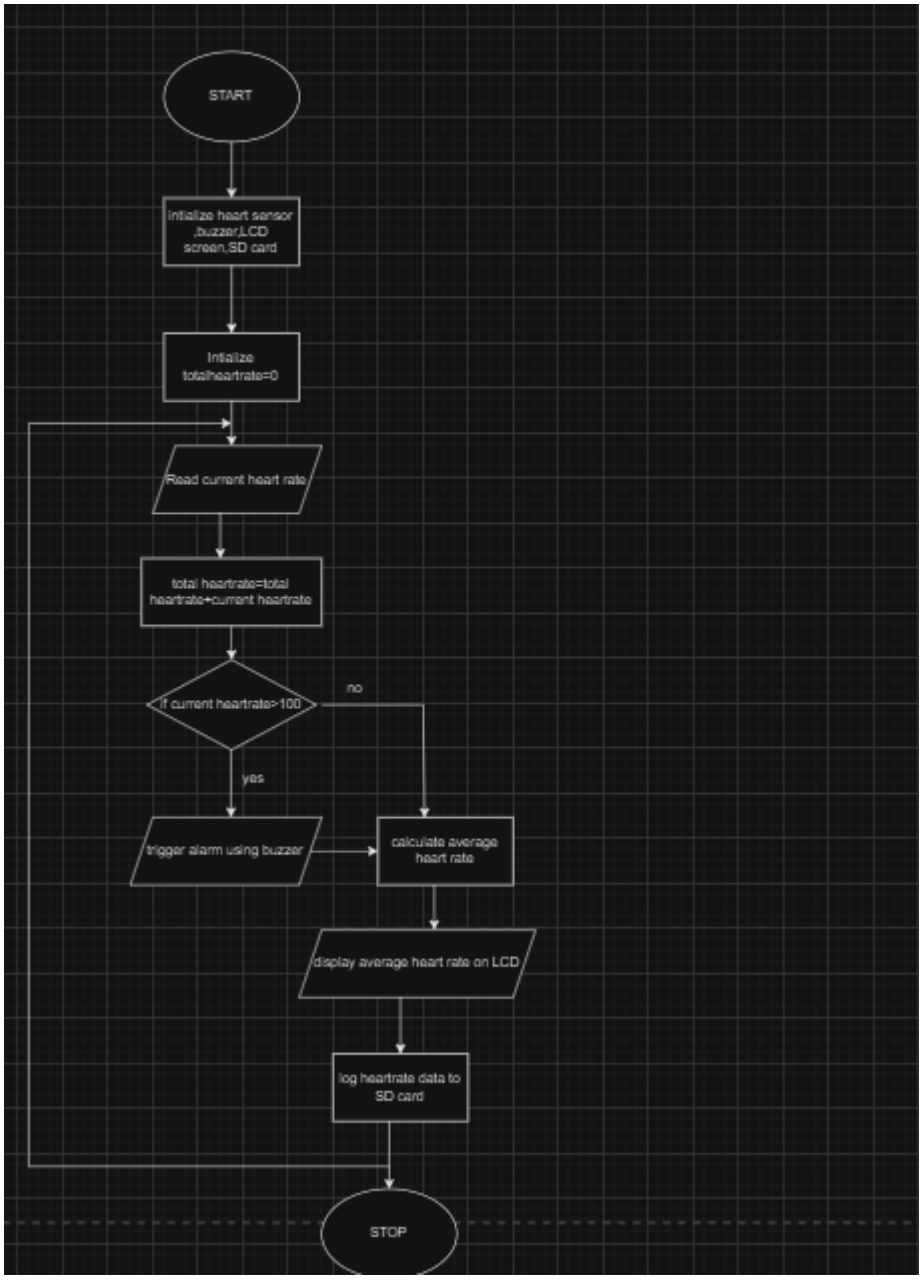
• If the heart rate exceeds 100 beats per minute, trigger an alert (buzzer).

• Display current heart rate and average heart rate on an LCD screen.

• Log heart rate data to an SD card for later analysis.

PSEUDOCODE

```
START
Intialize heart rate sensor
Intialize buzzer
Intialize LCD screen
Intialize SD card
Initialize total heartrate=0
For each second
    Try:
        Read current heart rate
        Total heartrate+= current heartrate
        Display current heartrate on LCD screen
        If current heart rate>100
                Trigger alarm using buzzer
        End if
    Catch sensor error:
            Display sensor error on LCD
Calculate average heart rate=total heartrate/60
Display average heart rate on LCD screen
Log heart rate data 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).

PSEUDOCODE

START

Intialize light sensor

Intialize manual override switch

Intialize LED

Set the threshold

For every minute

If manual override switch is On

    Turn on status feedback LED

Else

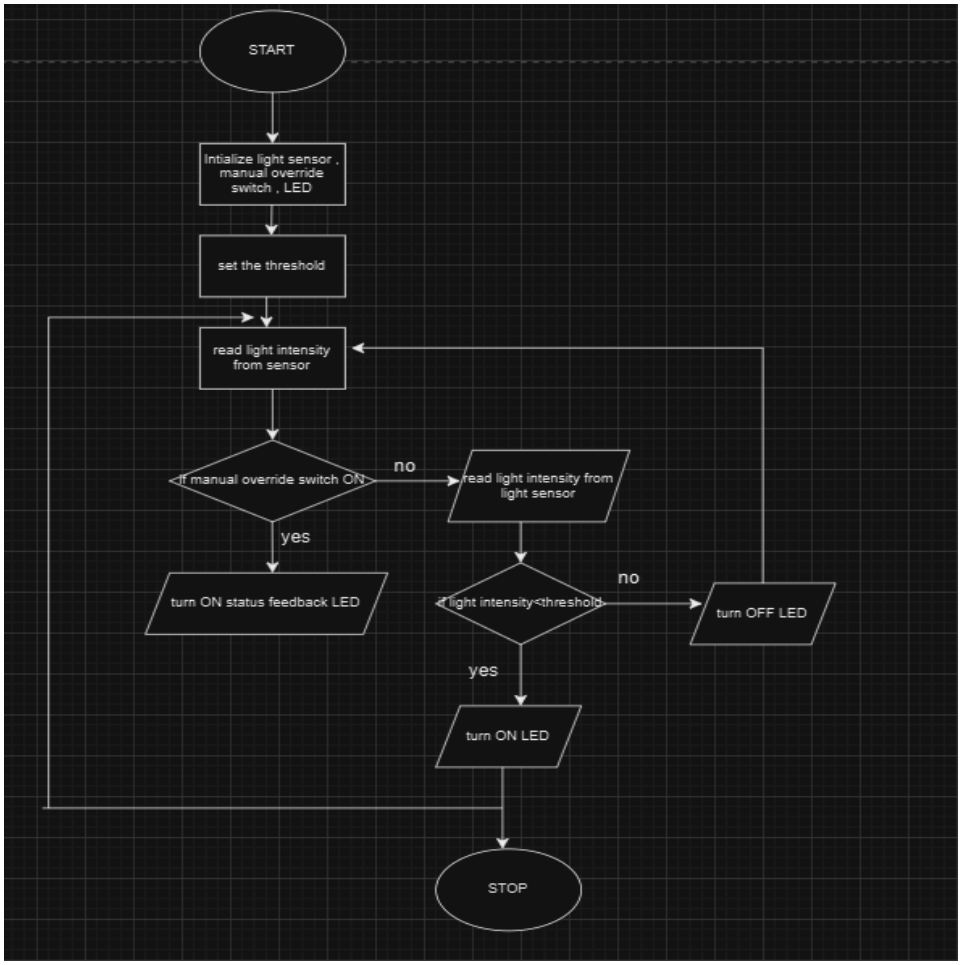
  Read light intensity from light sensor

  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.

PSEUDOCODE

START

Initialize start button,stop button,reset button

Initialize LCD

Initialize SD card

Check start button

If start button is pressed then

Log start time to SD card

Check stop button

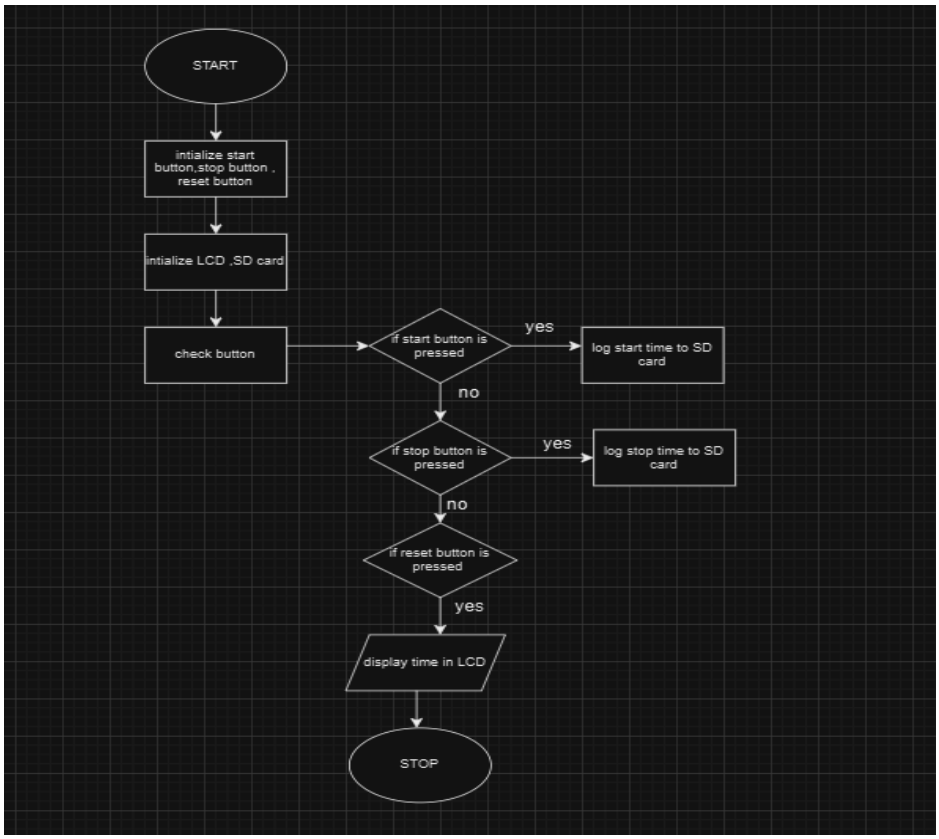
if stop button is pressed then

Log stop time to SD card

Check reset button

If reset button is pressed then

Display time in 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.

PSEUDOCODE

START

Initialize sensor

Initialize array to store temperature and timestamp

For every 10 minute

Read temperature from sensor

Store the temperature and timestamp in an array

Check for data retrieval request

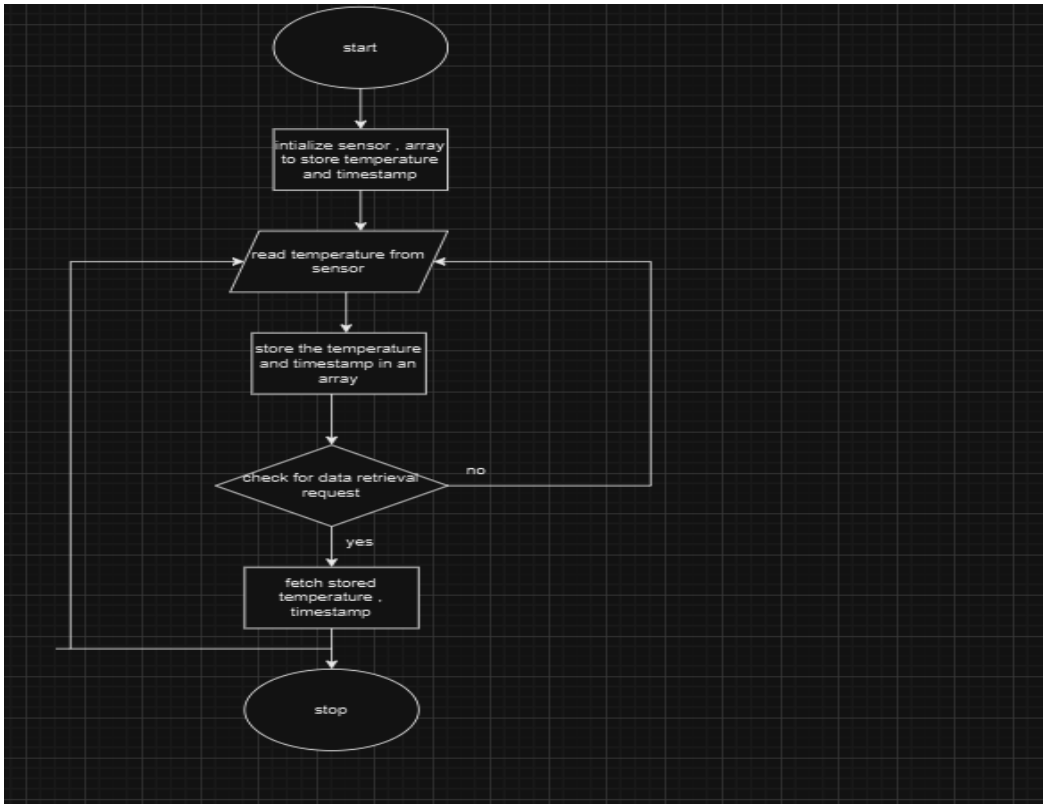
If yes

Fetch stored temperature , timestamp

Display the historical data

Else

Continue reading temperature



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.

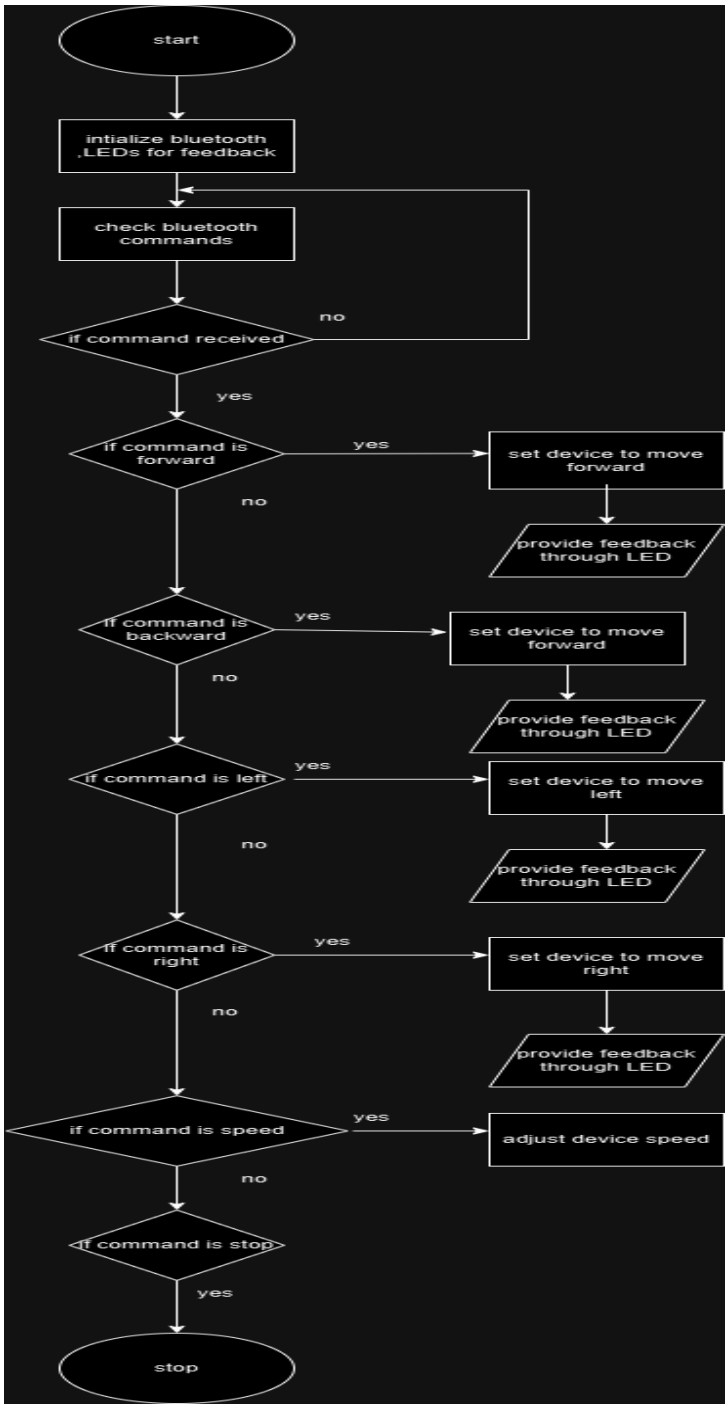
• Provide feedback through LEDs indicating the current state (e.g., moving or stopped).

PSEUDOCODE

```
START
Initialize Bluetooth module
Initialize LEDs for feedback
check Bluetooth commands
    If command received
        If command is "forward"
            Set the 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 "left"
            Set device to turn left
            Provide feedback through LED
        Else if command is "right"
            Set device to turn right
            Provide feedback through LED
        Else if command is "speed"
            Adjust the device speed
```

Else if command is "stop"

Stop the 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.

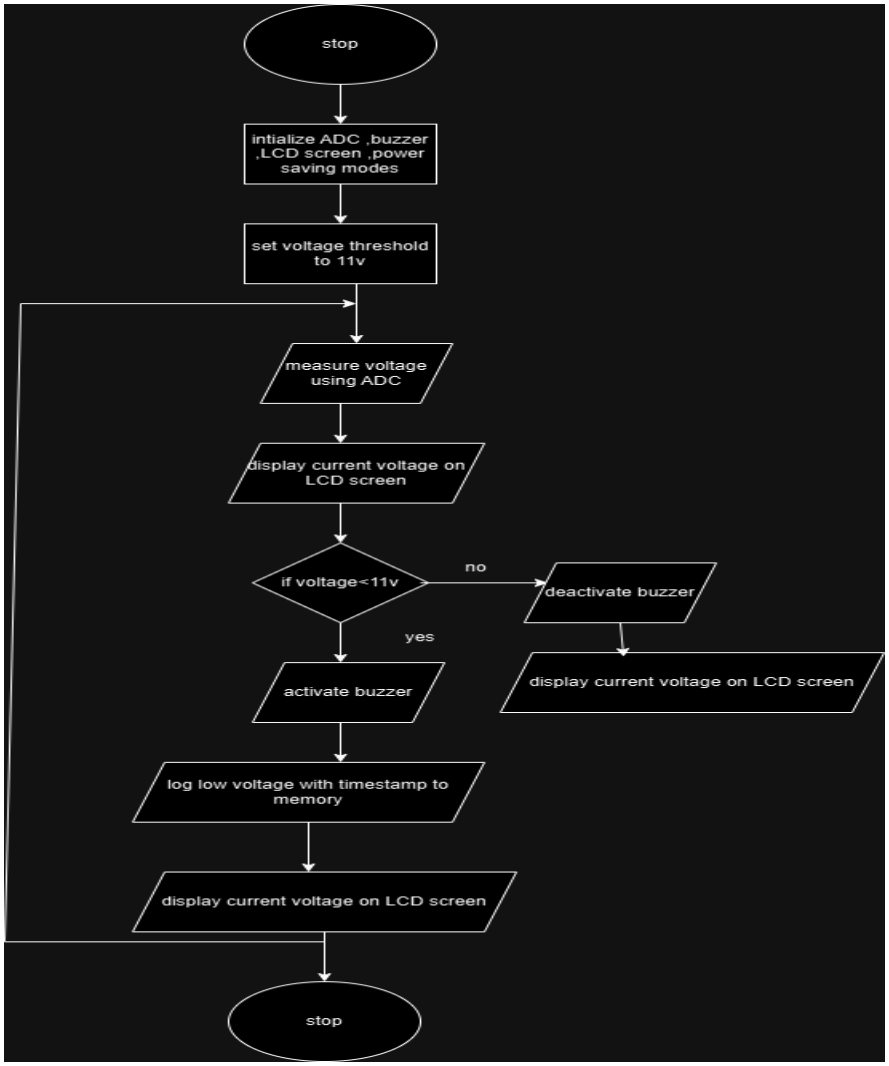
• Implement power-saving features to reduce energy consumption during idle periods.

PSEUDOCODE

```
START
Initialize ADC
Initialize buzzer
Initialize LCD screen
Initialize power-saving modes

Set voltage threshold to 11V
For every minute
    Measure voltage using ADC
     Display current voltage on LCD screen

    IF voltage < 11V THEN
       Activate buzzer
       Log "Low voltage alert" with timestamp to memory
       Display current voltage on LCD screen
    ELSE
       Deactivate buzzer
      Display current voltage on LCD screen
```



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.

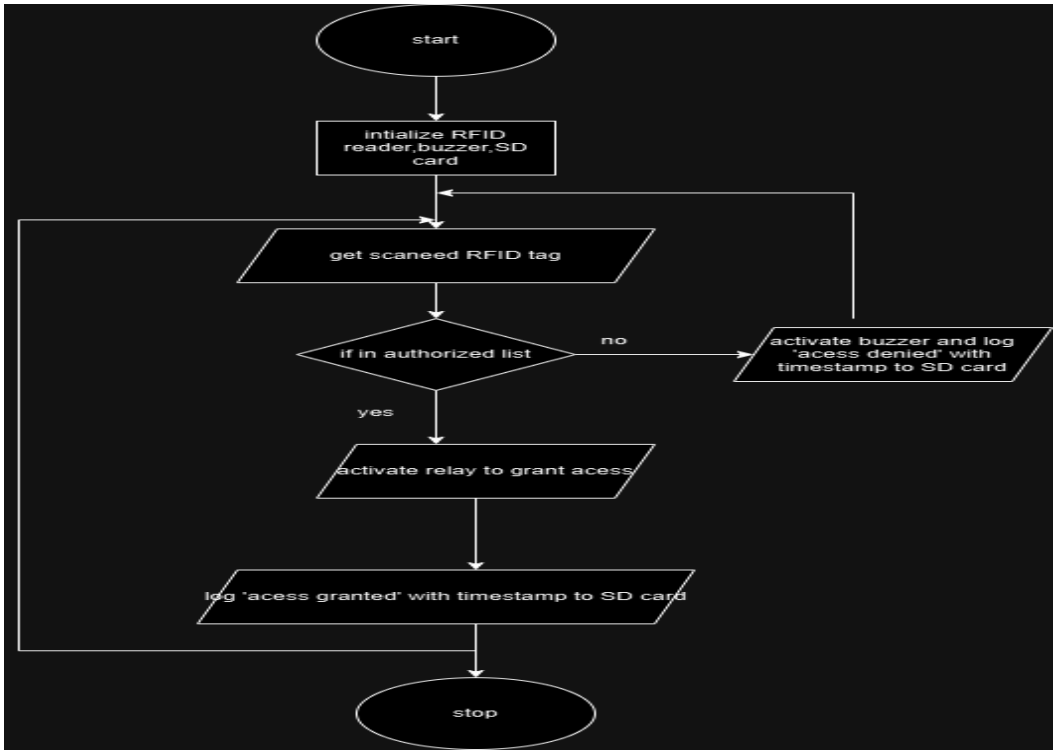
• Grant access by activating a relay if the tag is authorized; otherwise, deny access with an alert (buzzer).

• Log access attempts (successful and unsuccessful) with timestamps to an SD card.

PSEUDOCODE

```
START
Initialize RFID reader
Initialize buzzer
Initialize SD card for logging
 WAIT for RFID tag scan
   IF RFID tag scanned THEN
      GET scanned RFID tag

      IF scanned RFID tag is in authorized list THEN
         Activate relay to grant access
         Log "Access granted" with timestamp to SD card
      ELSE
         Activate buzzer to deny access
         Log "Access denied" with timestamp to SD card
```


start
intialize RFID reader,buzzer,SD card
get scaneed RFID tag
if in authorized list
no
activate buzzer and log 'acess denied' with timestamp to SD card
yes
activate relay to grant acess
log 'acess granted' with timestamp to SD card
stop