3.4 - Light and Comfort System

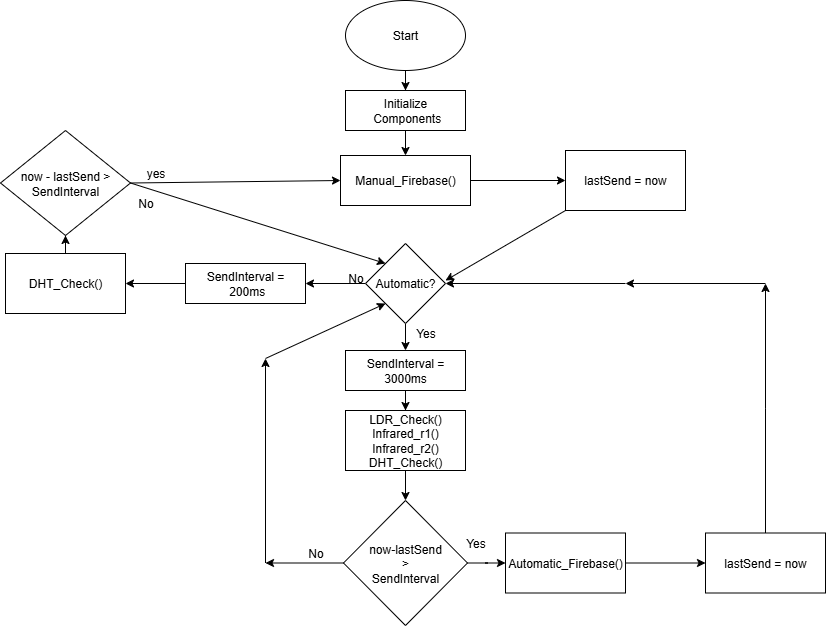
The System is a smart home subsystem designed to improve energy efficiency and user comfort by automating the control of lighting and air conditioning.

The system consists of :

1. Outing lights
2. Rooms lights
3. Ventilation system

It works in two modes :

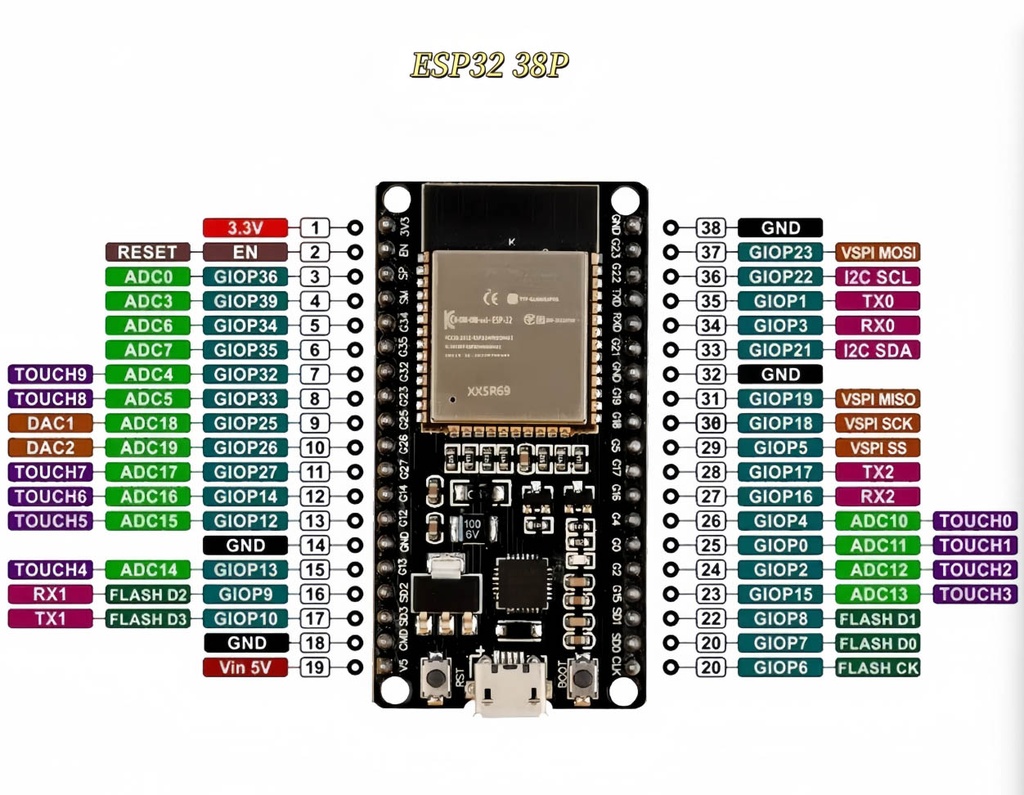
1. Automatic mode : depends on reading sensors to control the leds and the air conditioning
2. Manual mode : user control the lighting and the conditioner manually using his mobile phone



This diagram shows us how the main loop function run to achieve our targets

3.4.1 – Esp32 wroom 38 pin

We decided to use ESP32 microcontroller for this purpose , but why?

The **ESP32** microcontroller was chosen as the main controller for the Light & Climate System due to its advanced capabilities, which are well-suited for smart home applications. It offers a powerful combination of performance, connectivity, and low power consumption, making it ideal for systems that require wireless communication and real-time sensor integration.

**Key Features of the ESP32:**

1. **Built-in Wi-Fi and Bluetooth:**  
   Enables seamless wireless communication with the mobile application through Firebase Realtime Database.
2. **Dual-core processor up to 240 MHz:**  
   Provides high performance for handling multiple sensors and processing data in real-time.
3. **Multiple GPIO pins:**  
   It has 38 pins most of them can be used for I/O and in Analog/Digital which Allows connection of various components such as motion sensors, temperature/humidity sensors, LEDs, and relays
4. **Support for multiple protocols (I2C, SPI, UART):**  
   Makes it easy to interface with a wide range of sensors and modules.
5. **Low power consumption:**  
   Ideal for always-on systems and smart devices that need to remain efficient over time.
6. **Easy programming via Arduino IDE:**  
   The system was developed using the Arduino IDE, which simplifies coding, testing, and debugging.

All these features could integrate together to achieve high efficiency for our project

3.4.2 – Automatic mode

The Automatic Mode enables the system to operate without manual intervention, relying entirely on sensor inputs to make decisions. When this mode is activated, the light is automatically turned on when motion is detected and people counter has positive value . Similarly, the Ventilation system is controlled based on temperature and humidity readings; it turns on when the environment becomes uncomfortable and turns off when optimal conditions are restored. It also Automatically controls outing lights depends on whether we are in morning , evening or night . This mode is designed to optimize energy consumption and enhance user comfort by responding dynamically to changes in the environment .

Automatic mode sends all data about the lighting , Temperature and humidity to the mobile application via firebase realtime but users doesn't have any control on them .

Automatic mode is more suitable for children or elders who have some difficulty in treating with mobile application so all lights will be turned on and off automatically depends on their presence in the room and their motion .

3.4.2.1 – Outing lights in Automatic mode

Outing lights are the external lights of the house, such as those placed in the garden, garage, or entrance area. In automatic mode, these lights are controlled based on the time of day. A light-dependent mechanism determines whether it is morning, evening, or night, based on the brightness of natural sunlight or a real-time clock module.

* In morning time: the outing lights remain off, since there is sufficient natural light.
* In evening time: the outing lights turn onautomatically to ensure proper visibility and safety, but it doesn't work with all its power as there's some lights from the sun.
* In night time: the outing lights turn onautomatically with its highest power to ensure proper visibility and safety.

This automation helps reduce unnecessary power consumption during the day, while enhancing outdoor visibility and security during darker hours.

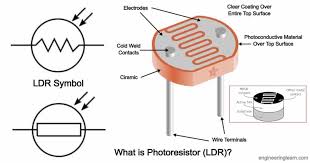
The system can also be configured to gradually increase or decrease brightness depending on the ambient light levels if desired.

We used LDR to know the amount of light from the sun

**How LDR Works**

An LDR (Light Dependent Resistor) is a type of resistor whose resistance varies depending on the amount of light falling on its surface. It is also known as a photoresistor.

* In bright light conditions: the resistance of the LDR decreasessignificantly, allowing more current to pass through.
* In darkness or low light: the resistance of the LDR increases, reducing the current flow.

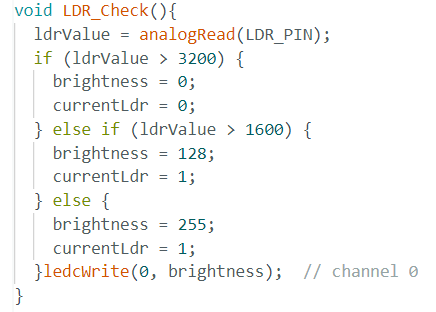
****

**How this was implemented**

To implement the outing lights control logic, we used a Light Dependent Resistor (LDR) sensor to detect the ambient light level. The LDR was connected to one of the analog GPIO pins of the ESP32. The system reads the light intensity and compares it to predefined thresholds to determine whether it is morning, evening, or night.

* If the light level is above the threshold → it is considered morning, and the outing lights remain off.
* If the light level is medium → it is considered evening and the outing lights are automaticallyturned on with moderate value
* If the light level is low → it is considered night and the outing lights are automaticallyturned on with high value

This approach ensures a real-time response to natural lighting conditions without requiring a fixed schedule. The threshold value is calibrated experimentally to match the actual lighting conditions in the environment



3.4.2.2 – Rooms lighting in Automatic mode

The automatic room lighting system is designed to improve convenience and save energy by intelligently turning lights on or off based on occupancy and motion detection.

The system relies on two main inputs:

1. People Counter**:** Keeps track of the number of people currently present in the room.

we implemented this using IR sensors

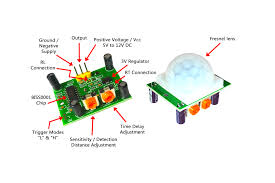


IR module consists of two limbs . first one the emitter of the infrared light , second one is the receiver to detect the reflection .

In the people counting mechanism, we use two active IR sensors placed at the entrance of the room. The system tracks the sequence in which the IR beams are interrupted:

* If Sensor A is triggered before Sensor B, a person is entering → counter is incremented.
* If Sensor B is triggered before Sensor A, a person is leaving → counter is decremented.

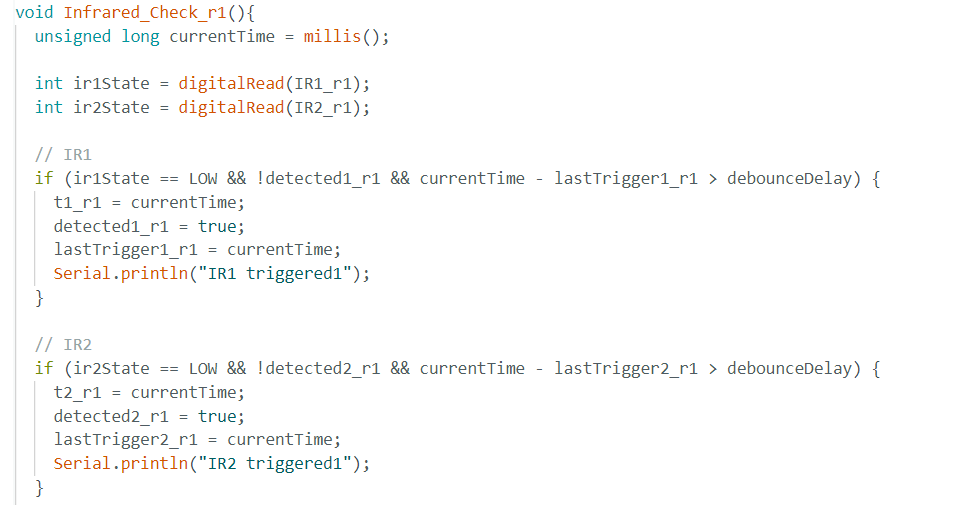
1. Motion Sensor (PIR Sensor): Detects movement within the room.

****

**How It Works:**

* When motion is detected and the people counter has a positive value, the room lights automatically turn on.
* If nomotion is detected for a specific period the lights will turn off .
* If nomotioncontinues and the people counter becomes zero, the lights will be automaticallyturned off.
* If there's a motion but peopleCount = 0 it will stay turned off , maybe it's just a pet.

This system ensures that lighting is only active when needed, reducing electricity usage and enhancing user comfort—especially for children, the elderly, or people who may forget to manually control the lights .



In this part of code we check each sensor to know if any of them has detected an obstacle . and if that happened we will store the timming

How to prevent debouncing ?

We store the trigger time at "lastTriggern\_rx" to make sure that the current trigger has happened after a small period of last trigger to ensure that it's another person

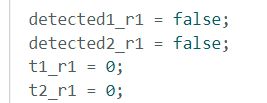
But what is the functionality of "detectedn\_rx" ?

It is the variable we used to check that r1&r2 have been triggered and there's no large delay between them.

Cases :

* 1. detected1 then detected2 && delay between them < 1500ms -> person entered
  2. detected2 then detected1 && delay between them < 1500ms -> person exited
  3. detected1 && 1500ms has passed without triggring ir2 -> nothing
  4. detected2 && 1500ms has passed without triggring ir1 -> nothing

And after each case of these all variables retuen to zero again



While peopleCounter has positive value it will check the PIR sensor to make sure that the motion is still there and the person in the room hasn't go sleep or stopped doing what he was doing . If the motion is stopped for about 4-6 minutes it will turn the lights off until he makes a motion again

3.4.2.3 - Ventilation system in Automatic mode

The ventilation system, represented by the air conditioning unit in our smart home, operates automatically based on real-time environmental readings to maintain a comfortable indoor climate and optimize energy usage.

The system relies on data from a temperature and humidity sensor DHT11 connected to the ESP32. These readings are continuously monitored and compared against predefined thresholds to decide when the air conditioner or fans should be turned on or off.

How It Works:

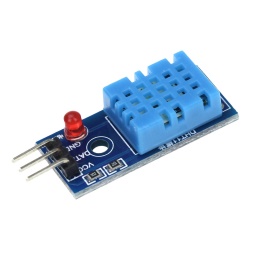
* If temperature or humidity levels exceed comfort thresholds, the system automatically turns on the air conditioning/fans to cool and dehumidify the space.
* Once the environmental conditions return to the acceptable range, the air conditioning/fans is turned off to conserve energy.

For example:

* If temperature > 27°C or humidity > 60% → Turn ON air conditioner.
* If temperature ≤ 25°C and humidity ≤ 50% → Turn OFF air conditioner.

This automation ensures that the indoor environment remains comfortable without requiring manual intervention. The system is especially beneficial for:

* Users who may forget to adjust the air conditioner.
* Homes with elderly residents or children who require stable environmental conditions



3.4.3 - Manual mode

The Manual Mode allows users to take full control of the Light & Ventilation System through a mobile application. In this mode, automation is disabled, and users can directly manage the system’s components such as room lighting, outing lights, and the air conditioning unit.

How It Works:

* The user opens the mobile app and connects to the system via Firebase Realtime Database.
* From the app interface, the user can:
  + Turn ON/OFF room lights.
  + Control the brightness of outing lights.
  + Switch the air conditioner ON or OFF.

Use Cases:

Manual mode is especially useful in the following situations:

* When the user wants to override automatic decisions.
* During specific events or occasions where custom lighting or cooling settings are preferred.
* When remote control is needed (e.g., turning on the air conditioner before arriving home).

Implementation Details:

* The app communicates with Firebase, which updates specific control values in the cloud.
* The ESP32 continuously listens for changes in these values and acts accordingly.

Manual mode provides flexibility and user freedom, allowing the system to adapt to personal preferences while maintaining the benefits of centralized control.

SO, in manual mode all senseors are neglected except DHT11 which still send the temperatue and humidity to the mobile application but there's no automatic actions depends on that .

3.4.4 - Firebase

Firebase Realtime Database was chosen as the cloud-based backend for our Light & Climate System to enable seamless communication between the mobile application and the ESP32 microcontroller.

Firebase provides a reliable and fast way to store and sync data across all connected clients in real time. This allows users to monitor and control their home environment remotely through their mobile phones, while the ESP32 can instantly reflect any changes made.

We used firebase in both Automatic mode and Manual mode but with different functionality.

1. In Automatic mode we send the decisions that have been taken depends on sensors
2. In Manual mode we allow users to control lights and air conditioning by himself and still read temperature sent by DHT11 sensor