Smart Stairwell Lightning Control System with Motion Detection and Integrated Security Camera

A Project Report Submitted
In Partial Fulfilment of the Requirement for the Degree of

BACHELOR OF TECHNOLOGY
in
ELECTRONICS AND COMMUNICATION ENGINEERING

by

Urjal Shrivastava (2020041149)
Himanshu Kushwaha (2020041072)
Krishan Kant Singh (2020041081)
Javed Ansari (2021042009)

Under the Supervision of
Dr. Dharmendra Kumar
Assistant Professor, ECED

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Madan Mohan Malaviya University of Technology, Gorakhpur (U.P.)- India
May,2023

© M. M. M. University of Technology, Gorakhpur, (U.P.) – 273010, INDIA

ALL RIGHTS RESERVED

CERTIFICATE

Certified that ***Urjal Shrivastava (2020041149), Himanshu Kushwaha (2020041072), Krishan Kant Singh (2020041081), Javed Ansari (2021042009)*** has carried out the research work presented in this thesis entitled **"*Smart Stairwell Lightning Control System with Motion Detection and Integrated Security Camera*"** for the award of **Bachelor of Technology** in Electronics and Communication Engineering from **Madan Mohan Malaviya University of Technology, Gorakhpur** under my supervision and guidance. The report embodies result of original work and studies carried out by Student themselves and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody.

Dr. Dharmendra Kumar
Assistant Professor
Department of ECE
M.M.M.U.T Gorakhpur

Date: May 2024

CANDIDATE’S DECLARATION

I declare that this written submission represents my work and ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Urjal Shrivastava
Roll No. 2020041149

Himanshu Kushwaha
Roll No. 2020041072

Krishan Kant Singh
Roll No. 2020041081

Javed Ansari
Roll No. 2021042009

B.Tech (ECE)
Department of Electronics and Communication Engineering

APPROVAL SHEET

This project report entitled "***Smart Stairwell Lightning Control System with Motion Detection and Integrated Security Camera***" by ***Urjal Shrivastava, Himanshu Kushwaha, Krishan Kant Singh, Javed Ansari*** is approved for the degree of **Bachelor of Technology** in Electronics and Communication Engineering.

Examiner

____

Supervisor

Dr. Dharmendra Kumar

____

Head of Department
Prof. S K Soni

____

Date: May 2024

Place: Gorakhpur

ACKNOWLEDGEMENT

It is a matter of great pleasure and satisfaction for me to present this dissertation work entitled " **Smart Stairwell Lightning Control System with Motion Detection and Integrated Security Camera** ", as a part of curriculum for award of "Bachelor of Technology" from Madan Mohan Malaviya University of Technology. Gorakhpur (U.P.) India.

I am very grateful to my Head of the Department, **Prof S.K. Soni**. It has been truly reassuring to know that he is always willing to share his quest for new problem new solutions form a very challenging and rewarding environment with us. He provides all kind of academic as well as administrative support for smooth completion of me dissertation work. Without his valuable guidance, this work would never have been a successful one.

I am very much thankful to my supervisor, **Dr. Dharmendra Kumar** also encouraged me to perform work in emerging-area of research i.e., Lightning Control System with Motion Detection and Integrated Security Camera as well as their continuous guidance and support throughout my work. I would also like to thank all my classmates for their valuable suggestions and helpful discussions at last, I am grateful to my family members, especially my beloved parents, for their encouragement and tender. Without them, I was unable to have enough strength to finish this dissertation.

Urjal Shrivastava (2020041149)
Himanshu Kushwaha (2020041072)
Krishan Kant Singh (2020041081)
Javed Ansari (2021042009)

Date: May 2024

LIST OF FIGURES

Figure No.	Description of Figures	Page No.
Fig 3.1	Arduino Uno	5
Fig 3.2	ESP32 Cam-Wi-Fi Module	5
Fig 3.3	PIR Sensor	6
Fig 3.4	Relay	6
Fig 3.5	Bulb	6
Fig 3.6	Jumping Wires	6
Fig 3.7	USB cable	6
Fig 3.1.1	Arduino Uno Pin Diagram	7
Fig 3.2.1	PIR Sensor Diagram	9
Fig 3.3.1	Relay Pin Diagram	10
Fig 3.4.1	ESP-32 Diagram	11
Fig 4.6.1	Block diagram of proposed model	23
Fig 4.7.1	Circuit diagram of proposed model	23
Fig 4.8.1	Final model	24
Fig 4.8.2	Circuit diagram of Final model	24

ABSTRACT

Smart lighting in stairwells with security cameras is a growing trend in home automation and security systems. This technology offers several benefits, including:

• **Improved safety**:

Smart stairwell lighting can help to prevent accidents by providing illumination when someone is present, especially at night or in low-light conditions. This is especially important for the elderly and people with disabilities.

• **Reduced energy costs**:

Smart stairwell lighting can be programmed to turn on and off only when needed, which can save energy and reduce utility bills.

• **Enhanced security**:

Smart stairwell cameras can deter crime and help to identify intruders. If motion is detected, the camera can automatically capture a picture of the person and send it to the homeowner's smartphone or computer. This technology is relatively inexpensive and easy to install, making it a good option for homeowners of all budgets.

TABLE OF CONTENTS

Certificate ………………………………………………………………………… iii

Candidate's Declaration ……………………………………………………….. iv

Approval Sheet ……………………………………………………………….. v

Acknowledgement ……………………………………………………………... .vi

List of Figures ……………………………………………………………….. vii

List of Tables …………………………………………………………….... ix

Abstract ……………………………………………………………………….. x

Table of Contents ……………………………………………………………….. xi

CHAPTER 1 INTRODUCTION ……………………………………………………. 1

CHAPTER 2 LITERATURE REVIEW ………………………………...……….. 3

2.1 Motivation ……………………………………………………….. 4

CHAPTER 3 HARDWARE DESCRIPTION……………………………………….. 5

3.1 Arduino UNO R3…..……………………………………………....... 7

3.2 PIR Sensor……….……………………………………….…………. 9

3.3 Relay ……………………………………………………………........10

3.4 ESP -32………………………………..…………………………..… 11

3.5 Jumper Wires ……..…………………………..……………………… 12

3.6 Micro USB Cable……………………………………….………....... 12

CHAPTER 4 IMPLEMENTATION DETAILS AND RESULT …………………... 13

4.1 PIR Sensor detects motion ……………………………………….…........ 13

4.2 Arduino is triggered ……………………….…………………..….… 13

4.3 Integration of ESP-32 for Image Uploading to Google Drive ………….. 16

4.4 Automated Deactivation of Lights and Camera…….……………..…… 22

4.5 Block Diagram of proposed model……………………………………….. 23

4.6 Circuit Diagram of Proposed Model……………….……………………23

4.7 Final Implementation……………………………………………………...24

CHAPTER 5 CONCLUSION & FUTURE SCOPE………………………………… 25

REFERENCES…………………………………………………………………… 27

CHAPTER 1

INTRODUCTION

In the rapidly evolving landscape of smart home technology, the intersection of safety and convenience has given rise to innovative solutions that redefine our living spaces. The Smart Staircase Lighting Control System with Motion Detection and Integrated Security Camera stands as a testament to the transformative power of technology in addressing contemporary challenges. Staircases, ubiquitous in homes and public spaces, represent more than mere transitions between levels; they are integral components of our daily routines. However, the inherent risks associated with poorly illuminated staircases and the increasing concerns for security in our homes have spurred the development of advanced systems that go beyond traditional lighting solutions.

At its core, this Smart Staircase Lighting Control System is designed to provide a comprehensive solution to the multifaceted challenges posed by staircases. Safety, a fundamental aspect of any living environment, takes centre stage with the integration of motion detection technology. Traditional lighting systems, often static and reliant on manual switches, can be inefficient and fail to address the dynamic nature of human movement. The motion detection feature ensures that illumination is activated precisely when needed, enhancing safety by preventing accidents and ensuring a well-lit pathway. Beyond its fundamental safety features, the system incorporates an Integrated Security Camera, elevating it to a multifunctional tool for both safety and surveillance. In an era where security is paramount, the ability to monitor and record activities on staircases provides an additional layer of protection. Whether deterring potential intruders or simply keeping a watchful eye on everyday activities, the integrated security camera contributes to a sense of security and peace of mind for homeowners. One of the defining aspects of this system is its emphasis on user-centric design and technological integration. The accompanying mobile application serves as the central hub for controlling and monitoring the entire system remotely.

Users can effortlessly adjust lighting settings, monitor the security camera feed, and receive real-time alerts—all from the convenience of their smartphones. This level of control not only enhances convenience but also aligns with the growing demand for interconnected smart home ecosystems.

The customization options embedded in the system add another layer of sophistication. Users can tailor the behaviour of the Smart Staircase Lighting Control System to suit their preferences and lifestyle. Whether adjusting the brightness levels of the lights, setting specific time-based routines, or configuring motion sensitivity, the system adapts to the unique needs of each user, providing a personalized and intelligent user experience.

In a world increasingly focused on sustainability and energy efficiency, the Smart Staircase Lighting Control System takes a step towards a greener future. The integration of smart lighting controls ensures that energy is used judiciously, with lights activating only when necessary. This not only reduces electricity consumption but also aligns with the broader goals of creating environmentally conscious and responsible technological solutions.

As we embark on this journey towards a smarter and safer living environment, the Smart Staircase Lighting Control System with Motion Detection and Integrated Security Camera emerges as a beacon of innovation. It is more than a mere technological advancement; it is a testament to our commitment to creating living spaces that are not only technologically advanced but also safe, convenient, and attuned to the evolving needs of modern life.

CHAPTER 2

LITERATURE REVIEW

The integration of motion detection systems with surveillance technology has become increasingly prevalent in both residential and commercial settings due to its effectiveness in enhancing security measures. This literature review aims to explore the various components and methodologies employed in such systems, focusing on a specific project that integrates a Passive Infrared (PIR) sensor, Arduino microcontroller, camera, and cloud-based storage solution.

Motion detection systems serve as the foundational element in many security setups. The PIR sensor, a common choice for motion detection, operates by detecting changes in infrared radiation within its field of view. This technology has been extensively studied and utilized for its simplicity, reliability, and cost-effectiveness.

The Arduino microcontroller serves as the central processing unit in our project, facilitating the interaction between the PIR sensor, camera, and other peripheral devices. Arduino's open-source platform, coupled with its versatility and ease of use, has made it a popular choice for DIY projects and prototyping in the field of electronics and automation.

Incorporating a camera into the system enables visual verification of detected motion. Continuous monitoring and recording of the surrounding environment provide valuable data for security analysis and incident response. Various camera modules and surveillance techniques have been explored in the literature, ranging from basic image capture to advanced video analytics.

The integration of cloud-based storage solutions, such as Google Drive, adds a layer of accessibility and scalability to the system. By uploading captured images or video footage to the cloud, users can access the data remotely from any internet-enabled device. This approach has gained traction in recent years due to the increasing availability of high-speed internet and the growing demand for remote monitoring solutions.

Efforts to improve the efficiency and sustainability of motion detection systems have led to the implementation of automated control mechanisms. In our project, we propose a method to deactivate the lights and camera after a predefined period of inactivity, thereby conserving energy and prolonging the lifespan of the components. This approach aligns with the broader trend towards energy-efficient and eco-friendly technologies.

Several challenges and considerations must be addressed in the design and implementation of motion detection systems. These include minimizing false alarms, optimizing detection accuracy, ensuring compatibility between different components, and addressing privacy concerns related to data collection and storage. Additionally, factors such as cost, scalability, and ease of deployment play a significant role in determining the suitability of a particular solution for a given application.

The integration of motion detection systems with surveillance technology offers a powerful means of enhancing security and situational awareness in various environments. By leveraging components such as PIR sensors, Arduino microcontrollers, cameras, and cloud-based storage solutions, we can develop efficient and scalable systems capable of detecting and responding to potential threats in real time. However, ongoing research and development efforts are needed to address existing challenges and further improve the performance and usability of these systems in diverse applications.

2.1 MOTIVATION

The motivation for developing the "Smart Stairwell Lighting Control System with Motion Detection and Integrated Security Camera" stems from a growing need to enhance energy efficiency and security in modern multistory buildings. Stairwells, often overlooked, consume significant energy due to lights being left on unnecessarily for prolonged periods. This not only leads to increased energy costs but also contributes to environmental degradation through higher carbon footprints. Addressing this issue with an automated solution aligns with global sustainability goals, promoting responsible energy usage.

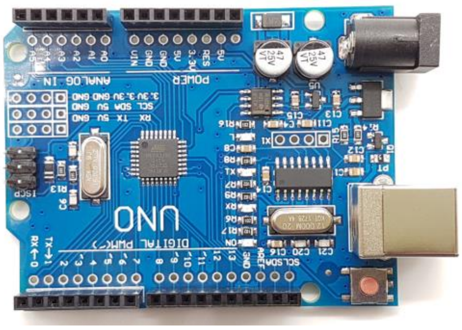
Additionally, the security of building occupants is a paramount concern. Stairwells can be vulnerable points for unauthorized access or criminal activities, necessitating improved surveillance. Integrating motion detection with a security camera offers a proactive approach to monitoring these areas, ensuring a safer environment for residents and visitors.

This project also leverages the advancements in IoT and smart technology, showcasing the potential to create more intelligent and responsive building systems. By combining energy management and security, this system provides a dual benefit, making it an attractive solution for building managers and owners looking to modernize their facilities. The pursuit of this project is driven by the desire to create safer, more sustainable, and technologically advanced living and working spaces.

CHAPTER 3

HARDWARE DESCRIPTION

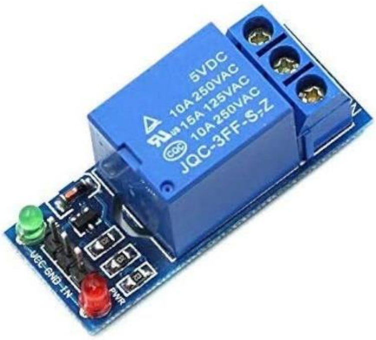
- Arduino UNO R3
- PIR Sensor
- Relay
- ESP32 Cam-Wi-Fi Module
- USB Cable (for power supply)
- Jumper Wires
- Bulb



Arduino UNO (3.1)

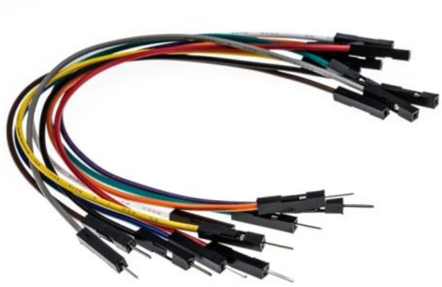
ESP32 Cam-Wifi Module (3.2)

PIR Sensor (3.3)



5V Relay (3.4)

Bulbs (3.5)



Jumping Wires (3.6)

USB Cable (3.7)

3.1 Arduino UNO R3

Arduino UNO is a microcontroller board based on the **ATmega328P**. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header and a reset button.

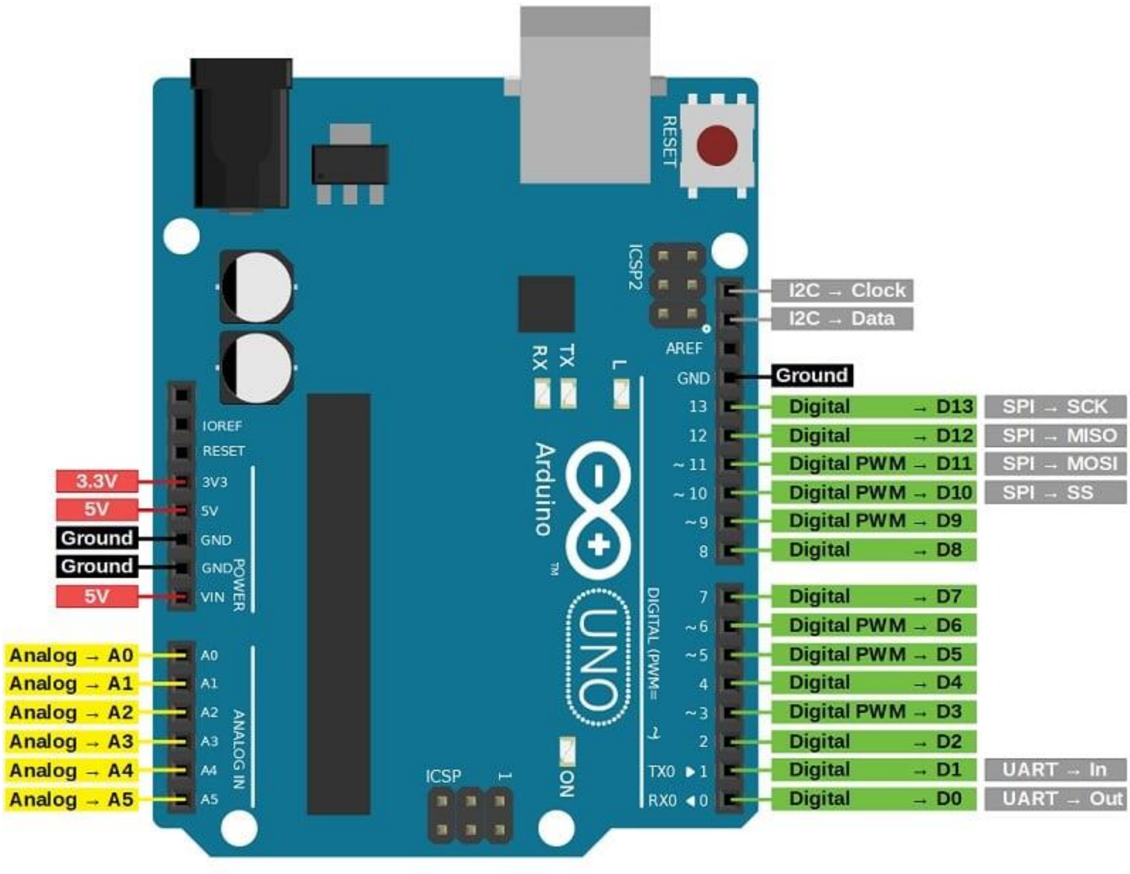


Fig. 3.1.1

Vin: This is the input voltage pin of the Arduino board used to provide input supply from an external power source.

5V: This pin of the Arduino board is used as a regulated power supply voltage and it is used to give supply to the board as well as onboard components.

3.3V: This pin of the board is used to provide a supply of 3.3V which is generated from a voltage regulator on the board

GND: This pin of the board is used to ground the Arduino board.

Reset: This pin of the board is used to reset the microcontroller. It is used to Resets the microcontroller.

Analog Pins: The pins A0 to A5 are used as an analog input and it is in the range of 0-5V.

Digital Pins: The pins 0 to 13 are used as a digital input or output for the Arduino board.

Serial Pins: These pins are also known as a UART pin. It is used for communication between the Arduino board and a computer or other devices. The transmitter pin number 1 and receiver pin number 0 is used to transmit and receive the data resp.

External Interrupt Pins: This pin of the Arduino board is used to produce the External interrupt and it is done by pin numbers 2 and 3.

PWM Pins: This pin of the board is used to convert the digital signal into an analog by varying the width of the Pulse. The pin numbers 3,5,6,9,10 and 11 are used as a PWM pin.

SPI Pins: This is the Serial Peripheral Interface pin, it is used to maintain SPI communication with the help of the SPI library. SPI pins include:

1. SS: Pin number 10 is used as a Slave Select
2. MOSI: Pin number 11 is used as a Master Out Slave In
3. MISO: Pin number 12 is used as a Master In Slave Out
4. SCK: Pin number 13 is used as a Serial Clock

LED Pin: The board has an inbuilt LED using digital pin-13. The LED glows only when the digital pin becomes high.

AREF Pin: This is an analog reference pin of the Arduino board. It is used to provide a reference voltage from an external power supply.

3.2 PIR Sensor

A passive infrared (PIR) sensor (Fig. 5) is an electronic sensor that measures infrared (IR) light radiating from objects in its field of view. They're most frequently utilized in PIR-based motion detectors. PIR sensors are commonly utilized in security alarms and automatic lighting applications.

Fig. 3.2.1

PIR sensors detect general movement, but don't give information as to who or what moved. For that purpose, an imaging IR sensor is required. PIR sensors are commonly called simply "PIR" or sometimes "PID" for "passive infrared detector." The term passive refers to the fact that PIR devices don't radiate energy for detection purposes. They work entirely by detecting infrared (radiant heat) emitted by or reflected from objects.

Specifications:

- Input voltage: DC 4.5-20V
- Static Current: <50uA
- Level output: High 3.3 V / Low 0V
- Trigger: L cannot be repeated trigger / H can be repeated trigger
- Delay time: 5-200s
- Block time: 2.5s

3.3 Relay

Relays are electromechanical switches that can be controlled by the Arduino UNO R3. In this project, the relays are used as intermediary components between the Arduino and the lights. They allow the Arduino, upon receiving a signal from the PIR sensor indicating motion detection, to either activate or deactivate the lights.

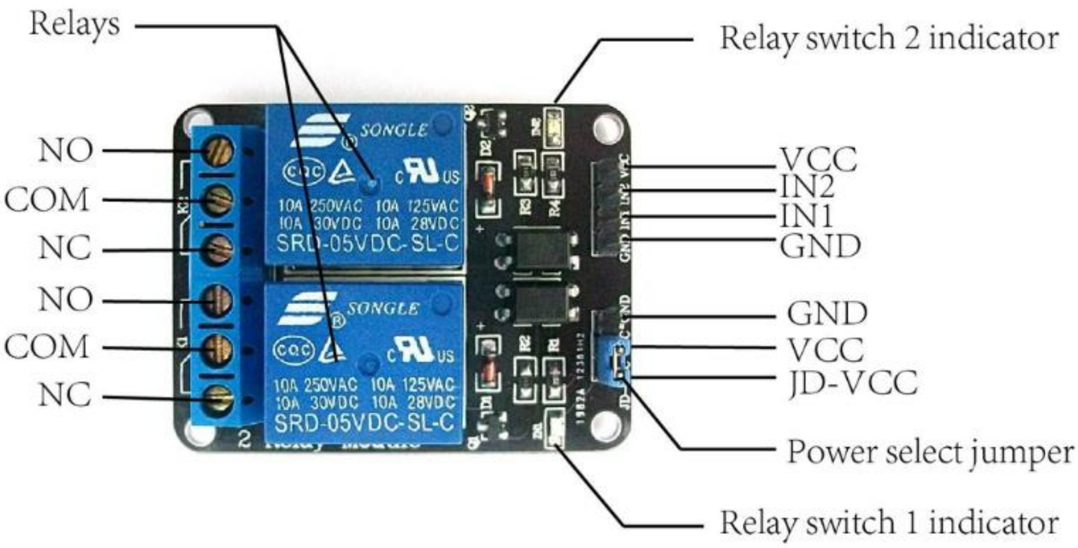


Fig. 3.3.1

<u>Specifications:</u>

- Supply voltage – 3.75V to 6V
- Quiescent current: 2mA
- Current when the relay is active: ~70mA
- Relay maximum contact voltage – 250VAC or 30VDC
- Relay maximum current – 10A

3.4 ESP32 cam Wi-Fi Module

The ESP32-CAM is a small size, low power consumption camera module based on ESP32. It comes with an OV2640 camera and provides onboard TF card slot.

The ESP32-CAM can be widely used in intelligent IoT applications such as wireless video monitoring, WiFi image upload, QR identification, and so on. **DHT 11**

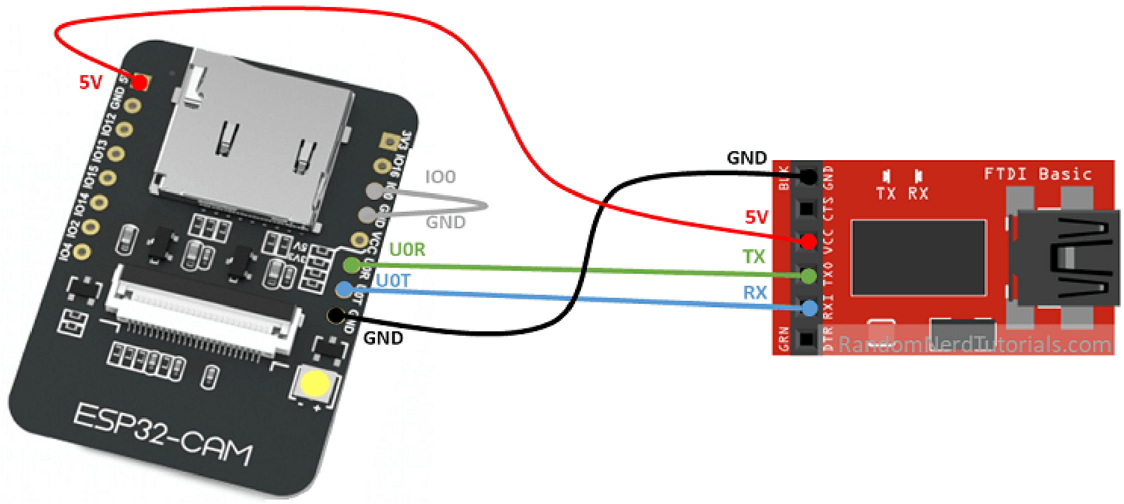


Fig. 3.4.1

<u>Specifications:</u>

- WIFI module: ESP-32S
- Processor: ESP32-D0WD
- Built-in Flash: 32Mbit
- RAM: Internal 512KB + External 4M PSRAM
- Antenna: Onboard PCB antenna
- WiFi protocol: IEEE 802.11 b/g/n/e/i
- Bluetooth: Bluetooth 4.2 BR/EDR and BLE
- WIFI mode: Station / SoftAP / SoftAP+Station
- Supported TF card: up to 4G
- Peripheral interface: UART/SPI/I2C/PWM
- IO port: 9

3.5 Jumper Wires

Jumper wires are simply wires that have connector pins at each end, allowing them to be used to connect two points to each other without soldering.

Jumper wires are typically used with breadboards and other prototyping tools to make it easy to change a circuit as needed. Fairly simple. In fact, it doesn't get much more basic than jumper wires.

Though jumper wires come in a variety of colours, the colours don't actually mean anything. This means that a red jumper wire is technically the same as a black one. But the colours can be used to your advantage to differentiate between types of connections, such as ground or power.

3.6 Micro USB Cable

This is Multipurpose Micro USB Cable mainly use connect Arduino boards and flight controller board as well as can be used as mobile data transfer cable. With its best quality connectors and built, it can provide a good lifecycle.

It can also be used to program the Arduino Pro Mini, Node MCU boards and power the boards through USB.

CHAPTER 4

IMPLEMENTATION DETAILS AND RESULTS

In this project, we integrated a Passive Infrared (PIR) sensor with an Arduino microcontroller to detect motion. When motion is detected, the Arduino sends a high signal to activate both a light and a camera. The light illuminates the area, while the camera starts taking pictures at 5-second intervals. These images are then uploaded to Google Drive using an ESP-32 microcontroller equipped with Wi-Fi capabilities. To conserve energy and resources, if no motion is detected for 20 seconds, the Arduino turns off the lights and stops the camera from capturing pictures. This automated system ensures efficient monitoring and documentation of detected motion while minimizing energy consumption. The result is a reliable and sustainable security solution that integrates motion detection, lighting, camera surveillance, and cloud-based storage for comprehensive real-time monitoring and analysis.

4.1 PIR Sensor detects motion:

In the first step, the PIR (Passive Infrared) sensor plays a crucial role in motion detection. When the PIR sensor detects motion, it generates a high signal (typically 3.3V or 5V) which is then sent to the connected Arduino. The PIR sensor is designed to sense the infrared radiation emitted by moving objects, usually humans or animals, within its detection range. The Arduino, a microcontroller board, receives this high signal through one of its digital input pins. Upon receiving the signal, the Arduino can be programmed to perform various actions, such as turning on a light, sounding an alarm, or sending a notification. This setup allows for automated responses based on detected movement, making it a fundamental step in projects like home security systems, automated lighting, or interactive installations. The simplicity of connecting a PIR sensor to an Arduino makes it an accessible and powerful tool for various DIY and professional applications.

4.2 Arduino is triggered:

In the next step, the Arduino responds to the high signal from the PIR sensor by sending high signals to both a light and a camera. When the Arduino receives the motion detection signal, it activates its digital output pins connected to the light and camera. This means it sends a high voltage signal (typically 5V) to these devices. As a result, the light turns on, illuminating the area, and the camera starts recording or taking pictures. This automated response ensures that when motion is detected, the area is immediately lit, and any activity is captured on camera. This setup is essential for security systems, enabling real-time monitoring and recording of any detected movement.

4.2.1 Light Activation:

When the Arduino sends a high signal to the light, it triggers the light to turn on, illuminating the area. This can be achieved by connecting the light to a relay module controlled by the Arduino, allowing it to handle higher voltage devices safely. The light remains on as long as the Arduino maintains the high signal. This immediate response ensures that any motion detected by the PIR sensor is met with adequate lighting, deterring potential intruders and providing visibility for cameras and occupants. This step is critical in security and automation systems, as proper illumination enhances safety and ensures that the camera can capture clear images. By integrating the light control with the motion detection system, the project ensures a well-coordinated and efficient response to any detected movement.

4.2.2 Camera Activation:

When the Arduino sends a high signal to the camera, it activates the camera to start taking pictures at intervals of 5 seconds. This is achieved by programming the camera module connected to the Arduino to capture images repeatedly with a 5-second delay between each shot. The Arduino can control the timing using its internal clock to ensure precise intervals. This step is crucial for continuous monitoring and documentation of any activity in the area. By capturing images every 5 seconds, the camera provides a series of snapshots that can be used for security purposes, helping to identify any intruders or unusual activities. This interval-based capture ensures that no significant movement goes undocumented, making it an effective way to monitor a space over time

4.2.3 Programming for Arduino:

```
const int pirPins[3] = {2, 3, 4}; // Replace with your actual PIR
sensor pins
const int ledPins[3] = {8, 9, 10}; // Replace with your actual
LED pins
const int timeout = 5000; // Timeout in milliseconds (30 seconds)

unsigned long lastActiveTime[4]; // Declare array for last motion
detection times

void setup() {
  for (int i = 0; i < 3; i++) {
    pinMode(pirPins[i], INPUT); // Set PIR pins as input
    pinMode(ledPins[i], OUTPUT); // Set LED pins as output
    lastActiveTime[i] = 0; // Initialize lastActiveTime to 0 for
each sensor
  }
  pinMode(7,OUTPUT); //Set Camera pin as output
  lastActiveTime[3] = 0;
}

void loop() {
 if(digitalRead(2) == HIGH)
 {
   lastActiveTime[3] = millis(); // Update last motion detection
time
   digitalWrite(7, HIGH);
 }
 if (millis() - lastActiveTime[3] >= timeout)
      digitalWrite(7, LOW);


  for (int i = 0; i < 3; i++) {
    int currentPirState = digitalRead(pirPins[i]);

    // Check for motion detection
    if (currentPirState == HIGH) {
      lastActiveTime[i] = millis(); // Update last motion
detection time
      digitalWrite(ledPins[i], HIGH); // Turn on LED
    }

    // Turn off LED after timeout
    if (millis() - lastActiveTime[i] >= timeout) {
      digitalWrite(ledPins[i], LOW); // Turn off LED even if
motion is still detected
    }
  }

  delay(10); // Short delay to avoid overwhelming loop
}
```

4.3 Integration of ESP-32 for Image Uploading to Google Drive:

In the final step of the project, the captured images are uploaded to Google Drive using the ESP-32 microcontroller. The ESP-32, a powerful Wi-Fi and Bluetooth-enabled microcontroller, is instrumental in adding wireless communication capabilities to the system. This module features a dual-core processor with a rich set of I/O interfaces, making it suitable for IoT applications and advanced projects like ours.

Upon capturing the images at 5-second intervals, the Arduino communicates with the ESP-32 to transfer the image data. This is typically achieved through a serial communication protocol where the Arduino sends the captured image data to the ESP-32. The ESP-32, equipped with Wi-Fi connectivity, connects to a predefined wireless network. Once connected, it uses HTTP or HTTPS protocols to interact with Google Drive’s API.

To facilitate this process, the ESP-32 runs a custom firmware that includes the necessary libraries for Wi-Fi connectivity and HTTP communication. Libraries such as WiFi.h and HTTPClient.h are commonly used in the Arduino IDE environment to handle these tasks. The firmware is programmed to authenticate with Google Drive using OAuth 2.0, a standard authorization framework. This ensures secure access to the Google Drive account.

After establishing a connection and authentication, the ESP-32 uploads the image files to a specific folder in Google Drive. The upload process involves creating an HTTP POST request containing the image data and appropriate headers that indicate the file type and authorization token. Once the server processes the request, the images are stored in the designated Google Drive folder, accessible from anywhere with internet access.

This step not only automates the process of storing and backing up the captured images but also enables remote monitoring and analysis. By leveraging the ESP-32’s wireless capabilities, the system becomes more flexible and scalable, illustrating the practical application of IoT in enhancing security and surveillance systems. This integration underscores the efficiency and modernity of utilizing cloud storage solutions in real-time monitoring projects.

4.3.1 Programming for ESP-32:

```
//======================================== Including the
libraries.
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"
#include "Base64.h"
#include "esp_camera.h"
//========================================

//========================================
CAMERA_MODEL_AI_THINKER GPIO.
#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27

#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22
//========================================

// LED Flash PIN (GPIO 4)
#define FLASH_LED_PIN 4

//======================================== Enter your WiFi ssid
and password.
const char* ssid = "Wifi";
const char* password = "76587658";
//========================================

//======================================== Replace with your
"Deployment ID" and Folder Name.
String myDeploymentID =
"AKfycbw9UcD1uo4syedY_aR0FTCoU62cC_BfmSJOtN62XK8vNdsV9tUeDeHwuUvV
xJ1ja4Oc";
String myMainFolderName = "ESP32-CAM";
//========================================

//======================================== Variables for
Timer/Millis.
unsigned long previousMillis = 0;
const int Interval = 5000; //--> Capture and Send a photo every
20 seconds.
//========================================

// Variable to set capture photo with LED Flash.
// Set to "false", then the Flash LED will not light up when
capturing a photo.
// Set to "true", then the Flash LED lights up when capturing a
photo.
bool LED_Flash_ON = true;

// Initialize WiFiClientSecure.
WiFiClientSecure client;

//
________________________________________________________________
```

```
________________________________________________________________
_______________ Test_Con()
// This subroutine is to test the connection to
"script.google.com".
void Test_Con() {
  const char* host = "script.google.com";
  while(1) {
    Serial.println("-----------");
    Serial.println("Connection Test...");
    Serial.println("Connect to " + String(host));

    client.setInsecure();

    if (client.connect(host, 443)) {
      Serial.println("Connection successful.");
      Serial.println("-----------");
      client.stop();
      break;
    } else {
      Serial.println("Connected to " + String(host) + "
failed.");
      Serial.println("Wait a moment for reconnecting.");
      Serial.println("-----------");
      client.stop();
    }

    delay(1000);
  }
}
//
________________________________________________________________
_______________

//
________________________________________________________________
_______________ SendCapturedPhotos()
// Subroutine for capturing and sending photos to Google Drive.
void SendCapturedPhotos() {
  const char* host = "script.google.com";
  Serial.println();
  Serial.println("-----------");
  Serial.println("Connect to " + String(host));

  client.setInsecure();

  //---------------------------------------- The Flash LED blinks
once to indicate connection start.
  digitalWrite(FLASH_LED_PIN, HIGH);
  delay(100);
  digitalWrite(FLASH_LED_PIN, LOW);
  delay(100);
  //----------------------------------------

  //---------------------------------------- The process of
connecting, capturing and sending photos to Google Drive.
  if (client.connect(host, 443)) {
    Serial.println("Connection successful.");

    if (LED_Flash_ON == true) {
      digitalWrite(FLASH_LED_PIN, HIGH);
      delay(100);
    }

    //............................ Taking a photo.
    Serial.println();
    Serial.println("Taking a photo...");

    for (int i = 0; i <= 3; i++) {
      camera_fb_t * fb = NULL;
```

4.3.2 Programming for ESP-32 BASE 64:

```
#ifndef _BASE64_H
#define _BASE64_H

/* b64_alphabet:
 *      Description: Base64 alphabet table, a mapping between
integers
 *                   and base64 digits
 *      Notes: This is an extern here but is defined in Base64.c
 */
extern const char b64_alphabet[];

/* base64_encode:
 *      Description:
 *          Encode a string of characters as base64
 *      Parameters:
 *          output: the output buffer for the encoding, stores
the encoded string
 *          input: the input buffer for the encoding, stores the
binary to be encoded
 *          inputLen: the length of the input buffer, in bytes
 *      Return value:
 *          Returns the length of the encoded string
 *      Requirements:
 *          1. output must not be null or empty
 *          2. input must not be null
 *          3. inputLen must be greater than or equal to 0
 */
int base64_encode(char *output, char *input, int inputLen);

/* base64_decode:
 *      Description:
 *          Decode a base64 encoded string into bytes
 *      Parameters:
 *          output: the output buffer for the decoding,
 *                  stores the decoded binary
 *          input: the input buffer for the decoding,
 *                 stores the base64 string to be decoded
 *          inputLen: the length of the input buffer, in bytes
 *      Return value:
 *          Returns the length of the decoded string
 *      Requirements:
 *          1. output must not be null or empty
 *          2. input must not be null
 *          3. inputLen must be greater than or equal to 0
 */
int base64_decode(char *output, char *input, int inputLen);

/* base64_enc_len:
 *      Description:
 *          Returns the length of a base64 encoded string whose
decoded
 *          form is inputLen bytes long
 *      Parameters:
 *          inputLen: the length of the decoded string
 *      Return value:
 *          The length of a base64 encoded string whose decoded
form
 *          is inputLen bytes long
 *      Requirements:
 *          None
 */
int base64_enc_len(int inputLen);

/* base64_dec_len:
 *      Description:
 *          Returns the length of the decoded form of a
 *          base64 encoded string
 *      Parameters:
 *          input: the base64 encoded string to be measured
 *          inputLen: the length of the base64 encoded string
 *      Return value:
 *          Returns the length of the decoded form of a
 *          base64 encoded string
 *      Requirements:
 *          1. input must not be null
 *          2. input must be greater than or equal to zero
 */
int base64_dec_len(char *input, int inputLen);

#endif // _BASE64_H
```

4.3.3 Programming for ESP-32 BASE 64:

```
// Base64.cpp

#if (defined(__AVR__))
#include <avr\pgmspace.h>
#else
#include <pgmspace.h>
#endif

const char PROGMEM b64_alphabet[] = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
        "abcdefghijklmnopqrstuvwxyz"
        "0123456789+/";

/* 'Private' declarations */
inline void a3_to_a4(unsigned char * a4, unsigned char * a3);
inline void a4_to_a3(unsigned char * a3, unsigned char * a4);
inline unsigned char b64_lookup(char c);

int base64_encode(char *output, char *input, int inputLen) {
    int i = 0, j = 0;
    int encLen = 0;
    unsigned char a3[3];
    unsigned char a4[4];

    while(inputLen--) {
        a3[i++] = *(input++);
        if(i == 3) {
            a3_to_a4(a4, a3);

            for(i = 0; i < 4; i++) {
                output[encLen++] = pgm_read_byte(&b64
_alphabet[a4[i]]);
            }

            i = 0;
        }
    }

    if(i) {
        for(j = i; j < 3; j++) {
            a3[j] = '\0';
        }

        a3_to_a4(a4, a3);

        for(j = 0; j < i + 1; j++) {
            output[encLen++] = pgm_read_byte(&b64
_alphabet[a4[j]]);
        }

        while((i++ < 3)) {
            output[encLen++] = '=';
        }
    }
    output[encLen] = '\0';
    return encLen;
}

int base64_decode(char * output, char * input, int inputLen) {
    int i = 0, j = 0;
    int decLen = 0;
    unsigned char a3[3];
    unsigned char a4[4];


    while (inputLen--) {
        if(*input == '=') {
            break;
        }

        a4[i++] = *(input++);
        if (i == 4) {
            for (i = 0; i <4; i++) {
                a4[i] = b64_lookup(a4[i]);
            }

            a4_to_a3(a3,a4);

            for (i = 0; i < 3; i++) {
                output[decLen++] = a3[i];
            }
            i = 0;
        }
    }
```

```
    if (i) {
        for (j = i; j < 4; j++) {
            a4[j] = '\0';
        }

        for (j = 0; j <4; j++) {
            a4[j] = b64_lookup(a4[j]);
        }

        a4_to_a3(a3,a4);

        for (j = 0; j < i - 1; j++) {
            output[decLen++] = a3[j];
        }
    }
    output[decLen] = '\0';
    return decLen;
}

int base64_enc_len(int plainLen) {
    int n = plainLen;
    return (n + 2 - ((n + 2) % 3)) / 3 * 4;
}

int base64_dec_len(char * input, int inputLen) {
    int i = 0;
    int numEq = 0;
    for(i = inputLen - 1; input[i] == '='; i--) {
        numEq++;
    }

    return ((6 * inputLen) / 8) - numEq;
}

inline void a3_to_a4(unsigned char * a4, unsigned char * a3) {
    a4[0] = (a3[0] & 0xfc) >> 2;
    a4[1] = ((a3[0] & 0x03) << 4) + ((a3[1] & 0xf0) >> 4);
    a4[2] = ((a3[1] & 0x0f) << 2) + ((a3[2] & 0xc0) >> 6);
    a4[3] = (a3[2] & 0x3f);
}

inline void a4_to_a3(unsigned char * a3, unsigned char * a4) {
    a3[0] = (a4[0] << 2) + ((a4[1] & 0x30) >> 4);
    a3[1] = ((a4[1] & 0xf) << 4) + ((a4[2] & 0x3c) >> 2);
    a3[2] = ((a4[2] & 0x3) << 6) + a4[3];
}

inline unsigned char b64_lookup(char c) {
    if(c >='A' && c <='Z') return c - 'A';
    if(c >='a' && c <='z') return c - 71;
    if(c >='0' && c <='9') return c + 4;
    if(c == '+') return 62;
```

4.4 Automated Deactivation of Lights and Camera:

In this step of the project, we implement a system to deactivate the lights and camera after a 20-second period of no motion detection. This functionality is crucial for energy efficiency and resource management. The Arduino continuously monitors the signal from the PIR sensor. If no high signal, indicating motion, is detected for 20 seconds, the Arduino initiates a shutdown sequence.

The Arduino employs a timer that resets each time a high signal is detected. If the timer reaches 20 seconds without resetting, the Arduino sends a low signal to the digital output pins controlling the light and camera. This low signal triggers the relay to turn off the light and sends a command to the camera module to cease capturing images.

This automated deactivation not only conserves power by turning off the lights when not needed but also extends the lifespan of the camera by preventing unnecessary operation. This step exemplifies the integration of efficient resource management into the system, enhancing its practicality and sustainability. It ensures that the security system is both effective and energy-efficient, demonstrating the thoughtful design considerations essential for modern automated systems.

4.5 Block diagram of proposed model:

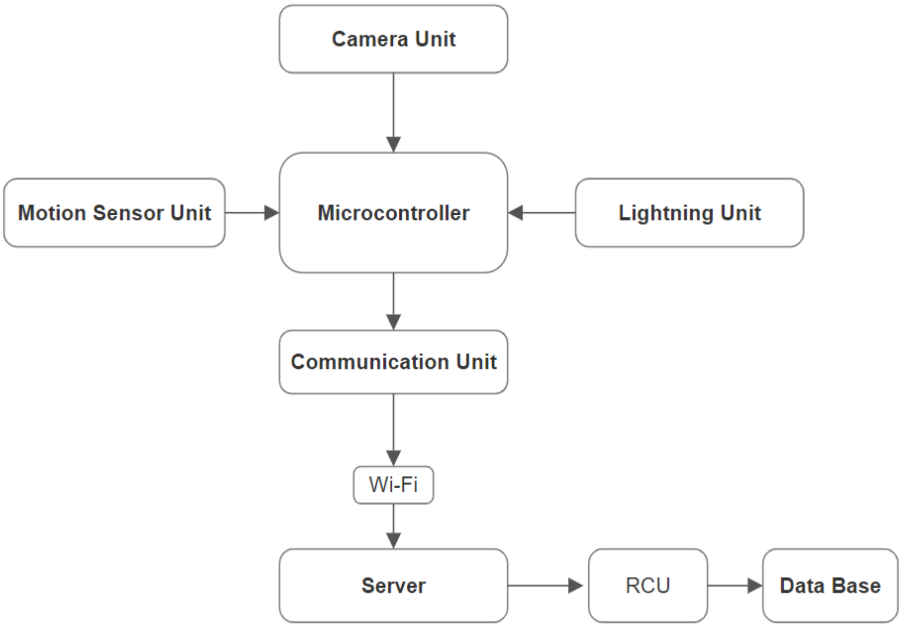


Fig 4.5.1

4.6 Circuit Diagram of proposed model:

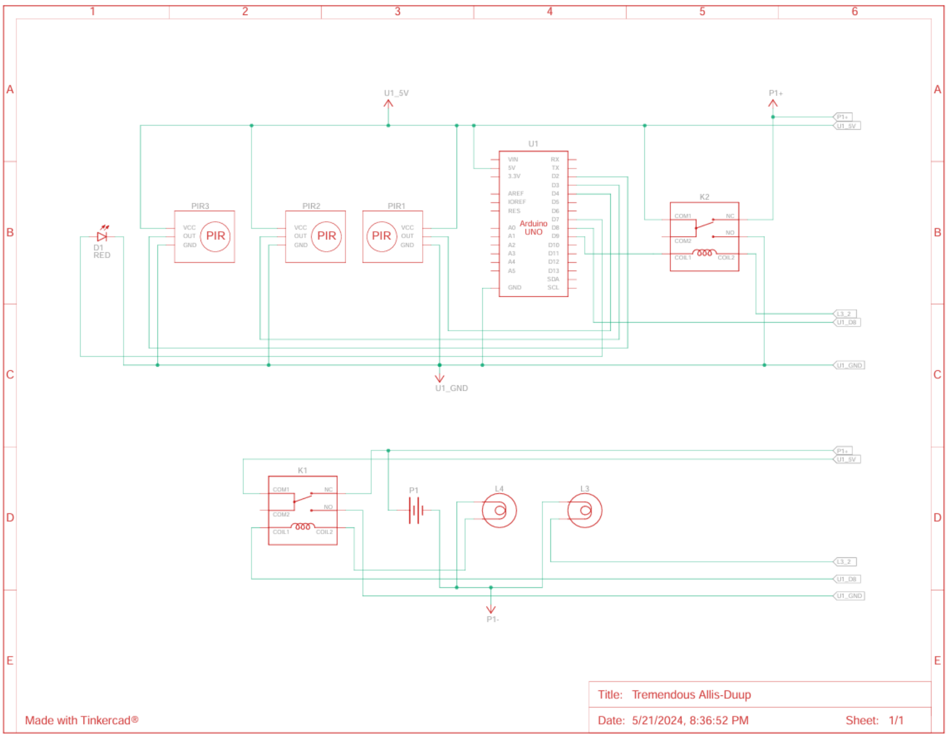


Fig 4.6.1

4.7 Final implementation:

Fig. 4.8.1

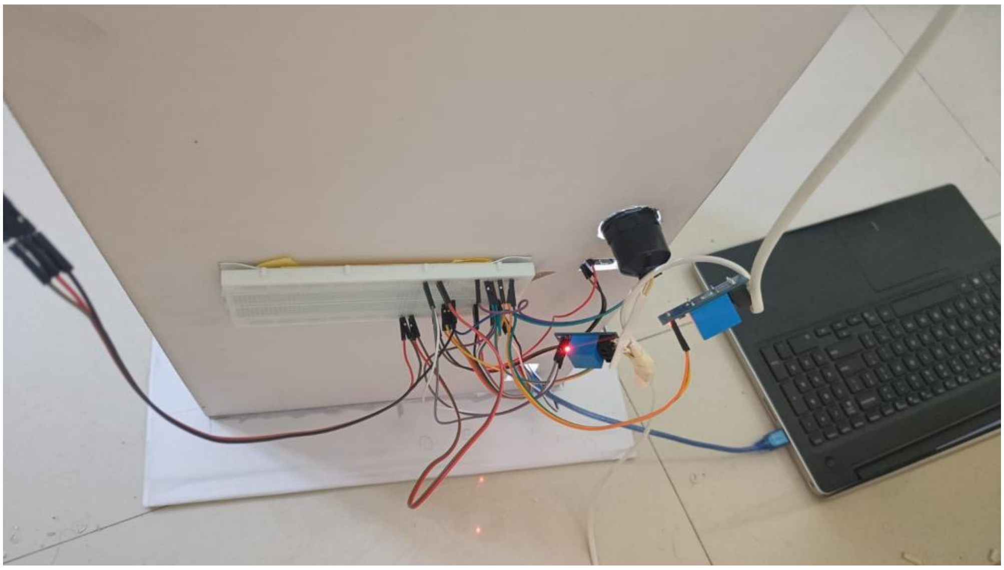


Fig 4.8.2

Chapter-5

CONCLUSION AND FUTURE SCOPE

5.1 Impact and Feasibility:

Incorporating motion detection and an integrated security camera greatly enhances safety and convenience for staircases. This system uses existing technologies and meets the growing demand for smart home solutions.

It can adapt to future innovations like AI and IoT, making it a forward-thinking choice. Energy-efficient lighting controls contribute to sustainability, and the system can be customized to fit various user needs.

Overall, this project is cost-effective, technically viable, and aligns well with current market trends, making it a practical and impactful solution.

5.2 Conclusion:

The "Smart Stairwell Lighting Control System with Motion Detection and Integrated Security Camera" project represents a significant advancement in the integration of automation, energy efficiency, and security within multistorey buildings. By employing modern sensor technology and intelligent control systems, this project addresses the dual needs of reducing energy consumption and enhancing building security.

The core component of this system, the motion sensor, ensures that lights in the stairwell are only active when needed, thereby significantly cutting down on unnecessary energy usage. This not only contributes to cost savings but also aligns with global efforts to promote sustainable energy practices. The integration of a smart lighting control system ensures seamless operation, switching lights on upon detecting motion and turning them off after a predefined period of inactivity. This automated approach eliminates the dependence on manual switches, providing convenience and reliability.

Moreover, the inclusion of a security camera adds an essential layer of safety by capturing images of individuals when motion is detected. This feature can be particularly useful for monitoring and security purposes, aiding in the identification of unauthorized access or suspicious activities. The ability to store and potentially transmit these images enhances the building’s overall security infrastructure.

Utilizing a microcontroller or a single-board computer like the Raspberry Pi to interface with the motion sensor, lighting system, and camera, this project demonstrates the effective use of available technology to create a cohesive and functional system. The programming and networking capabilities of these devices allow for the integration of additional features such as remote monitoring and data storage, further expanding the system’s utility.

In conclusion, this project not only provides practical solutions to everyday challenges faced in building management but also underscores the potential of smart technology in enhancing living and working environments. By focusing on energy efficiency and security, the Smart Stairwell Lighting Control System embodies a forward-thinking approach that can be adapted and scaled to various settings, paving the way for smarter, safer buildings in the future.

5.3 Future Scope:

Looking forward, the Smart Staircase Lighting Control System envisions advancements in motion sensors, artificial intelligence, and seamless integration with smart home ecosystems. Future developments may include sophisticated facial recognition, voice-controlled interfaces, and enhanced cybersecurity measures. Exploring energy harvesting technologies and cloud-based analytics for security footage are on our roadmap. We aim to establish global compatibility standards and extend the system's application to commercial spaces. Continuous refinement based on user feedback and behaviour analysis ensures a forward-looking solution, committed to safety, sustainability, and convenience in the evolving landscape of smart home technology.

Future Enhancements:

• Collaborating with smart grid technologies to align staircase lighting with broader energy management strategies, contributing to a more efficient and sustainable grid.

• Collaborating with broader smart building management systems to contribute to overall building efficiency, safety, and security.

• Researching and implementing energy harvesting technologies, such as piezoelectric sensors on stairs, to generate power from footsteps, contributing to sustainable energy practices.

REFERENCES:

[1] Yuvaraj Prof. and Ramesh S., "Improved Response Time on Safety Mechanism Based on PIR sensor," International Journal of Emerging Technology and Advanced Engineering, vol. 2, no. 4, April 2012

[2] Designing of Automatic Corridor Lighting System Using PIR Motion Sensor Pratibha; Ritik Rajput; Anita Yadav; Akhtar Saleem Ansari; Mohammed Aslam Husain; S P Singh Publication Year: 2021

[3] Riyanto I, Margatama L, Hakim H, Martini, Hindarto DE. Motion Sensor Application on Building Lighting Installation for Energy Saving and Carbon Reduction Joint Crediting Mechanism. Applied System Innovation. 2018; 1(3):23. https://doi.org/10.3390/asi1030023

[4] Yilmaz, A., M. Shah, "Contour Based Object Tracking with Occlusion Handling in Video Acquired Using Mobile Cameras", Proc. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015.

[5] Kamal Sehairi, Fatima Chouireb, and Jean Meunier "Comparative study of motion detection methods for video surveillance systems," *Journal of Electronic Imaging* 26(2), 023025 (25 April 2017). https://doi.org/10.1117/1.JEI.26.2.023025

[6] X. Lu and R. Manduchi, "Fast image motion segmentation for surveillance applications," Image Vision Comput., 29 (2), 104 –116 (2011). http://dx.doi.org/10.1016/j.imavis.2010.08.001 G

[7] K. Karmann, A. Brandt, "Moving object recognition using and adaptive background memory," Time-Varying Image Processing and Moving Object Recognition, 2 289 –307 1990

[8] P.-M. Jodoin et al., "Overview and benchmarking of motion detection methods," in Background Modeling and Foreground Detection for Video Surveillance, (2014). http://dx.doi.org/10.1201/b17223-30

[9] A. Yilmaz and M. Shah, Object tracking: A survey, Journal ACM Computing Surveys 38 (2006) article 13

[10] X. Wang, Intelligent multi-camera video surveillance: A review, Pattern Recognition Letters, 34 (2013) 3-19