Indian Institute of Technology,

(Indian School of Mines),

Dhanbad

Department of Computer Science &

Engineering

CSD513

Minor Project on Internet of Things

Academic Year 2022-2023

Motion Detection and Capture with PIR sensor

S. No.	Student Name	Admission No.
18	Chirag Gupta	20JE0297
20	Gulshan Anand	20JE0382
25	Lakshya Gupta	20JE0514

Acknowledgement

We would like to extend our heartfelt thanks to everyone who contributed to the success of this project. Firstly, we want to express our gratitude to our IoT course instructor, Prof. Tarachand Amgoth for giving us the opportunity to work on a real life project like this.

We also want to thank NVCTI IIT ISM for providing us with the necessary resources and facilities to complete our project. Without their support this project would not have been completed.

We would like to express our appreciation to the Telegram community for providing their API and unlimited free storage space for chats and media to the users. This feature drastically reduced the cost of our project.

Lastly, we would like to acknowledge the open-source community for providing us with access to the wide range of open-source technologies that we leveraged in our project.

We are grateful for the support and encouragement of everyone involved in this project.

Thank you!

Abstract

This IoT project involves the use of a Passive Infrared (PIR) sensor for motion detection and capture. The project aims to design a smart security system that can detect any motion and capture images of the intruder. The PIR sensor detects the infrared radiation emitted by a moving object, which triggers the capture of the image using a connected camera.

The project involves the development of a system that processes the data received from the PIR sensor and controls the connected camera to capture the images. The captured images are then sent to the owner's telegram chat, which can be accessed in the app itself. Also Telegram provides unlimited chat and media storage for free, so there won't be any issue for storing the captured images.

This system can be used for home security, surveillance systems, or even in industrial environments to detect and monitor movements. This project provides a cost-effective and efficient solution for motion detection and capture, making it ideal for various applications.

Contents

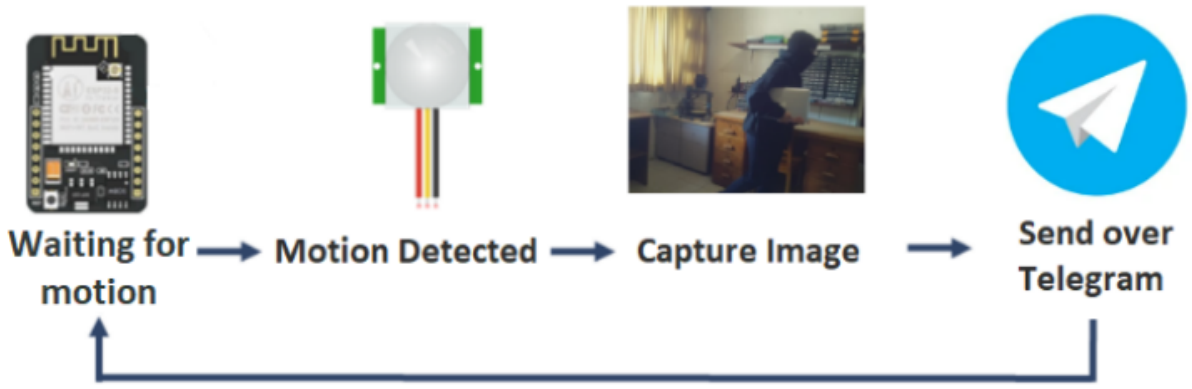
Introduction

This project is an innovative Internet of Things (IoT) application that focuses on motion detection and image capture using a PIR (Passive Infrared) sensor. The PIR sensor detects motion by sensing the infrared radiation emitted by a warm object, and triggers the capture of an image through a connected camera. Once the image is captured, it is instantly sent to the owner's Telegram chat, making it easy for users to monitor and stay up-to-date with the current status of their space.

With this technology, users can remotely detect and capture images of any unexpected activity in their home or office, providing a simple yet effective solution for home security and surveillance.

This project demonstrates the power of IoT in creating innovative solutions to real-world problems, and highlights the potential of combining different technologies to create more advanced and efficient systems.

Here is a quick overview of how the project works.



- The ESP32-CAM will be waiting for motion, until a motion is detected.

- Whenever a motion is detected, the ESP32-CAM gets a signal from the PIR sensor.
- The ESP32-CAM takes a photo and sends it to the user over the Telegram App.
- It again goes back to its initial state.

This project has numerous applications, including home security, monitoring pets, and tracking the presence of people in public spaces. It can be further customized to meet specific needs, such as adjusting the sensitivity of the PIR sensor or modifying the code to integrate other features such as facial recognition.

Overall, this IoT project on motion detection and capture with PIR sensor is an excellent example of how simple components can be combined to create innovative solutions to real-world problems. By integrating different technologies, we can build more efficient and effective systems that make our lives easier and more secure.

Problem Statement

With the increasing need for home security and surveillance, there is a demand for an affordable and effective solution that allows homeowners to remotely monitor their homes. Traditional security systems can be expensive and complicated to install, and may not provide the necessary level of customization and real-time alerts. This can lead to false alarms, missed alerts, or inadequate coverage.

Moreover, remote monitoring is becoming increasingly important for homeowners who may be away from their homes for extended periods of time, such as during vacations or business trips.

Therefore, there is a need for a system that can detect motion and capture images using readily available components like PIR sensor and camera modules, and transmit the images to the user in real-time through a simple and accessible platform like Telegram.

The goal of the project is to create a reliable and effective system that can enhance home security and surveillance, while also providing the user with greater flexibility and control over their monitoring needs.

Proposed Work

Our IoT project uses a PIR (Passive Infrared) sensor to detect motion in its environment. When motion is detected, the sensor triggers a camera to take a picture, which is then sent to a Telegram chat using an internet connection.

To implement this solution, we used the following hardware components:

- PIR sensor
- ESP32 Camera module
- TTL Programmer
- Power supply

The PIR sensor is connected to the microcontroller, which is programmed to wait for a signal from the sensor. When the PIR sensor detects motion, the microcontroller activates the camera module to capture an image. The image is then processed and sent to the Telegram chat through internet connection.

To make this project work, we need to write a program that runs on the microcontroller. The program should include the following steps:

- Connect to Wi-Fi
- Initialize the PIR sensor and camera module.
- Wait for a signal from the PIR sensor.
- Activate the camera module to take a picture.
- Process the image.
- Send the image to the Telegram chat using the Telegram API.

We also need to configure the Telegram chat to receive the images. This is done by creating a Telegram bot and adding it to the chat. The bot should have the necessary permissions to access and send messages to the chat.

Overall, our IoT project provides a simple yet effective way to detect motion and capture images. It can be used for various applications, such as security monitoring, wildlife observation, and home automation.

Experiment Results

Hardware Taken:

- **EPS32-CAM** - ESP32-CAM is a small-sized camera module that integrates an ESP32-S chip and an OV2640 camera. It allows capturing images, streaming video, and running applications on the ESP32 microcontroller, making it a powerful platform for various IoT applications that require camera functionality.
  The ESP32-CAM module also includes support for Wi-Fi and Bluetooth connectivity, making it possible to send data and images wirelessly to other devices. The module has a small form factor and low power consumption, making it suitable for our project.

- **TTL Programmer** - TTL (Transistor-Transistor Logic) programmer is a device that is used to program and communicate with digital integrated circuits that use TTL logic. TTL logic uses bipolar junction transistors to perform digital operations, and it is a popular logic standard for digital circuits. systems, robotics, and more.

- **PIR Motion Sensor** - A PIR (Passive Infrared) motion sensor is a type of electronic sensor that detects motion by sensing changes in infrared radiation emitted by objects in its field of view.
  The sensor works by detecting the infrared energy emitted by objects in the form of heat. When an object moves within the sensor's range, the infrared energy emitted by the object changes, which is detected by the sensor. The PIR sensor then sends a signal to EPS32-CAM to capture an image of the object.

- **Other basic accessories** - Other accessories like power bank was taken to power up the microcontroller, jumper wires for connections and breadboard for mounting the devices.

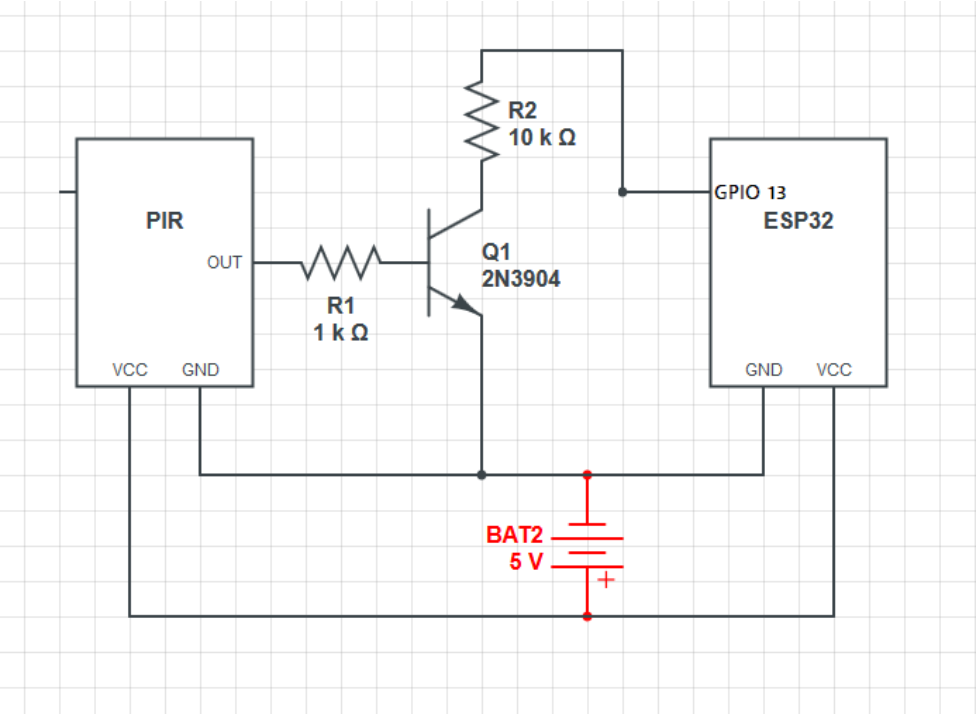
Software Used:

- **Arduino IDE -** Arduino IDE (Integrated Development Environment) is a free and open source software application used to program Arduino boards. It is used for building electronics projects, and the IDE is the primary tool used to write, compile, and upload code to an Arduino board.
  The Arduino IDE is based on the Processing development environment and uses the C++ programming language. It provides a simple interface for writing code, as well as tools for managing libraries, uploading code to the board, and debugging., such as turning on a light or sounding an alarm.

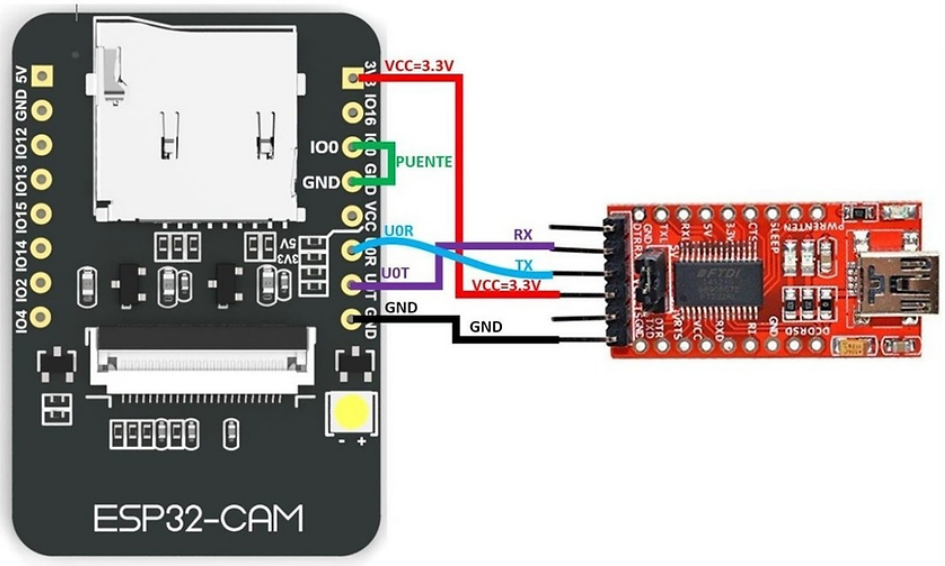
- **Telegram -** Telegram is a cloud-based instant messaging app. It provides an api that can be used to send messages to a specific person, which we used to send the captured images to the user.

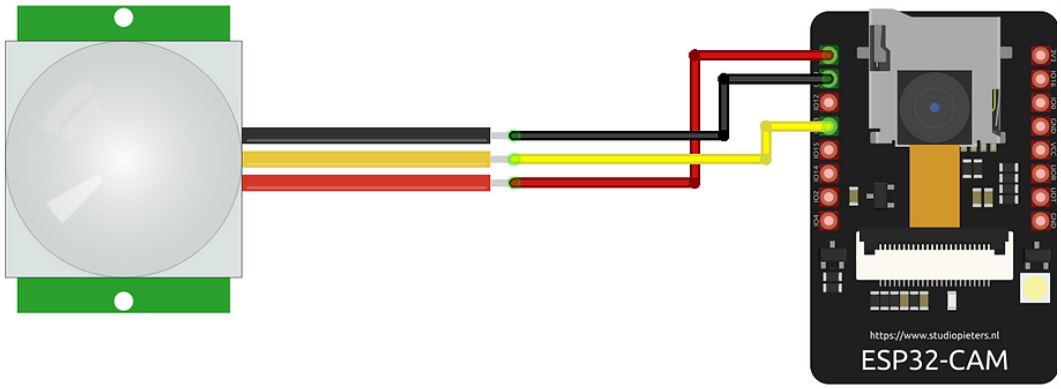
Connection:

- Circuit Diagram:



- PIN Diagrams:





Setting up Telegram API:

- Download the Telegram App and search for “BotFather”.
- Open it and click on “Start”.
- After this type “/newbot” and press enter.
- Now It will ask for a name, enter “MotionDetection” or whatever you like.
- It will ask for a username for the bot. Enter a unique username for the bot, for example “MotionEspCamBot”.
- Now it will give an api token, copy and keep it safe as it will be used in the code.

- Type the exact username of the bot in telegram search to find the bot. Open it and click on “Start”.
- Now, search for “get id” bot. Open it and click on “Start”. It will give a chat id, copy and store it as it will also be used in the code.

Now, our Telegram bot setup is done.

Code:

```
// Enter WiFi ssid and password
const char *ssid = "WIFI_SSID";
const char *password = "WIFI_PASSWORD";
String token = "TELEGRAM_API_TOKEN";
String chat_id = "TELEGRAM_CHAT_ID";

// Importing modules for WiFi and Camera
#include <WiFi.h>
#include <WiFiClientSecure.h>
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"
#include "esp_camera.h"

// CAMERA_MODEL_AI_THINKER, DEFINING THE CONSTANTS
#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

int gpioPIR = 13; // Assigning GPIO PIN number 13 for PIR Sensor

void setup()
{
    WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0);
```

```
Serial.begin(115200);
delay(10);
WiFi.mode(WIFI_STA);
Serial.println("");
Serial.print("Connecting to ");
Serial.println(ssid);
WiFi.begin(ssid, password); // Initiating connection to WiFi
long int StartTime = millis();

// Waiting for WiFi to get connected
while (WiFi.status() != WL_CONNECTED)
{
    delay(500);
    if ((StartTime + 10000) < millis())
        break;
}

Serial.println("");
Serial.println("STAIP address: ");
Serial.println(WiFi.localIP());
Serial.println("");

// Blinking the LED
// 5 times for successful Wi-Fi connection
// and once for connection failure
if (WiFi.status() != WL_CONNECTED)
{
    Serial.println("Reset");

    ledcAttachPin(4, 3);
    ledcSetup(3, 5000, 8);
    ledcWrite(3, 10);
    delay(200);
    ledcWrite(3, 0);
    delay(200);
    ledcDetachPin(3);
    delay(1000);
    ESP.restart();
}
else
{
    ledcAttachPin(4, 3);
    ledcSetup(3, 5000, 8);
    for (int i = 0; i < 5; i++)
    {
        ledcWrite(3, 10);
        delay(200);
```

```
        ledcWrite(3, 0);
        delay(200);
    }
    ledcDetachPin(3);
}

// configuring the camera
camera_config_t config;
config.ledc_channel = LEDC_CHANNEL_0;
config.ledc_timer = LEDC_TIMER_0;
config.pin_d0 = Y2_GPIO_NUM;
config.pin_d1 = Y3_GPIO_NUM;
config.pin_d2 = Y4_GPIO_NUM;
config.pin_d3 = Y5_GPIO_NUM;
config.pin_d4 = Y6_GPIO_NUM;
config.pin_d5 = Y7_GPIO_NUM;
config.pin_d6 = Y8_GPIO_NUM;
config.pin_d7 = Y9_GPIO_NUM;
config.pin_xclk = XCLK_GPIO_NUM;
config.pin_pclk = PCLK_GPIO_NUM;
config.pin_vsync = VSYNC_GPIO_NUM;
config.pin_href = HREF_GPIO_NUM;
config.pin_sscb_sda = SIOD_GPIO_NUM;
config.pin_sscb_scl = SIOC_GPIO_NUM;
config.pin_pwdn = PWDN_GPIO_NUM;
config.pin_reset = RESET_GPIO_NUM;
config.xclk_freq_hz = 20000000;
config.pixel_format = PIXFORMAT_JPEG;

if (psramFound())
{
    // external PSRAM memory found
    // so set image quality to high
    config.frame_size = FRAMESIZE_VGA;
    // 0-63 lower number means higher quality
    config.jpeg_quality = 10;
    config.fb_count = 2;
}
else
{
    // external PSRAM memory not found
    // so set image quality to low
    config.frame_size = FRAMESIZE_QQVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
}
```

```
    // camera initialisation
    esp_err_t err = esp_camera_init(&config);
    if (err != ESP_OK)
    {
        // camera init failed
        Serial.printf("Camera init failed with error 0x%x",
err);
        delay(1000);
        ESP.restart();
    }

    sensor_t *s = esp_camera_sensor_get();
    s->set_framesize(s, FRAMESIZE_XGA);
}

void loop()
{

    pinMode(gpioPIR, INPUT_PULLUP);
    // read digital input from gpioPIR, GPIO PIN 13
    int v = digitalRead(gpioPIR);
    Serial.println(v);
    if (v == 1)
    {
        // Motion Detected
        // Capture image and send to Telegram
        alerts2Telegram(token, chat_id);
        delay(10000);
    }
    delay(1000);
}

// Function to capture image and send it over Telegram
String alerts2Telegram(String token, String chat_id)
{
    const char *myDomain = "api.telegram.org";
    String getAll = "", getBody = "";

    camera_fb_t *fb = NULL;
    // Obtaining the pointer to captured image buffer
    fb = esp_camera_fb_get();
    if (!fb)
    {
        // Image was not captured
        Serial.println("Camera capture failed");
        delay(1000);
        ESP.restart();
```

```
        return "Camera capture failed";
    }

    WiFiClientSecure client_tcp;

    // Establishing TCP/IP connection to
    // "api.telegram.org" on port 443
    if (client_tcp.connect(myDomain, 443))
    {
        Serial.println("Connected to " + String(myDomain));

        // Preparing head and tail data
        String head = "--India\r\nContent-Disposition:
form-data; name=\"chat_id\"; \r\n\r\n" + chat_id +
"\r\n--India\r\nContent-Disposition: form-data; name=\"photo\";
filename=\"esp32-cam.jpg\"\r\nContent-Type: image/jpeg\r\n\r\n";
        String tail = "\r\n--India--\r\n";

        uint16_t imageLen = fb->len;
        uint16_t extraLen = head.length() + tail.length();
        uint16_t totalLen = imageLen + extraLen;

        client_tcp.println("POST /bot" + token + "/sendPhoto
HTTP/1.1");
        client_tcp.println("Host: " + String(myDomain));
        client_tcp.println("Content-Length: " +
String(totalLen));
        client_tcp.println("Content-Type: multipart/form-data;
boundary=India");
        client_tcp.println();
        client_tcp.print(head);

        // Declaring pointer to memory buffer
        uint8_t *fbBuf = fb->buf;
        size_t fbLen = fb->len;

        // Writing the image buffer to the TCP/IP connection
        // in chunks of 1024 bytes
        for (size_t n = 0; n < fbLen; n = n + 1024)
        {

            if (n + 1024 < fbLen)
            {
                client_tcp.write(fbBuf, 1024);
                fbBuf += 1024;
            }
            else if (fbLen % 1024 > 0)
```

```
    {
        size_t remainder = fbLen % 1024;
        client_tcp.write(fbBuf, remainder);
    }
}

// Marking the end of data
client_tcp.print(tail);

// Returning the obtained pointer to image buffer
esp_camera_fb_return(fb);

// set timeout value to 10 seconds
int waitTime = 10000;
long startTime = millis();
boolean state = false;

while ((startTime + waitTime) > millis())
{
    Serial.print(".");
    delay(100);
    // looping until data is available to read
    while (client_tcp.available())
    {
        // reading the data from the TCP/IP connection
        char c = client_tcp.read();
        if (c == '\n')
        {
            if (getAll.length() == 0)
            {
                state = true;
            }
            getAll = "";
        }
        else if (c != '\r')
        {
            getAll += String(c);
        }
        if (state == true)
        {
            getBody += String(c);
        }
        startTime = millis();
    }
    if (getBody.length() > 0) break;
}
// Closing the connection
```

```
        client_tcp.stop();
        Serial.println(getBody);
    }
    else
    {
        getBody = "Connection to telegram failed.";
        Serial.println("Connection to telegram failed.");
    }

    return getBody;
}
```

Setting up Arduino IDE:

- In your Arduino IDE, go to **File**> **Preferences.**
- Enter the following into the **Additional Board Manager URLs** field:
    - https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json
- Go to **Tools** > **Board** > **Boards Manager.**
- Search for **ESP32.**
- For **ESP32 by Espressif Systems,** select version **1.0.4** and press Install.
- Download the following **Universal Arduino Telegram Bot library** to interact with Telegram bot.
    - https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot/archive/master.zip
- Go to **Sketch** > **Include Library** > **Add.ZIP Library** and add the library you've just downloaded.
- Go to **Skech** > **Include Library** > **Manage Libraries.**
- Search for **ArduinoJson** and install the library.
- Go to **Tools > Serial Monitor** and select **115200 baud**.

Uploading the code:

- Go to **Tools** > **Board** and select **ESP32-CAM Wrover Module**.
- Go to **Tools** > **Port** and select the COM port the ESP32 is connected to.
- Go to **Tools** > **Flash Frequency** and select **80MHz**.

- Go to **Tools** > **Flash Mode** and select **QIO**.
- Go to **Tools** > **Partition Scheme** and select **Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)**.
- Go to **Tools** > **Upload Speed** and select **921600**.
- Then, click the upload button to upload the code.
- After a few seconds, the code should be successfully uploaded to the board.
- After code is uploaded, press the ESP32-CAM on-board RST button.

Check Wi-Fi Connection:

Check LED on the ESP32-CAM module

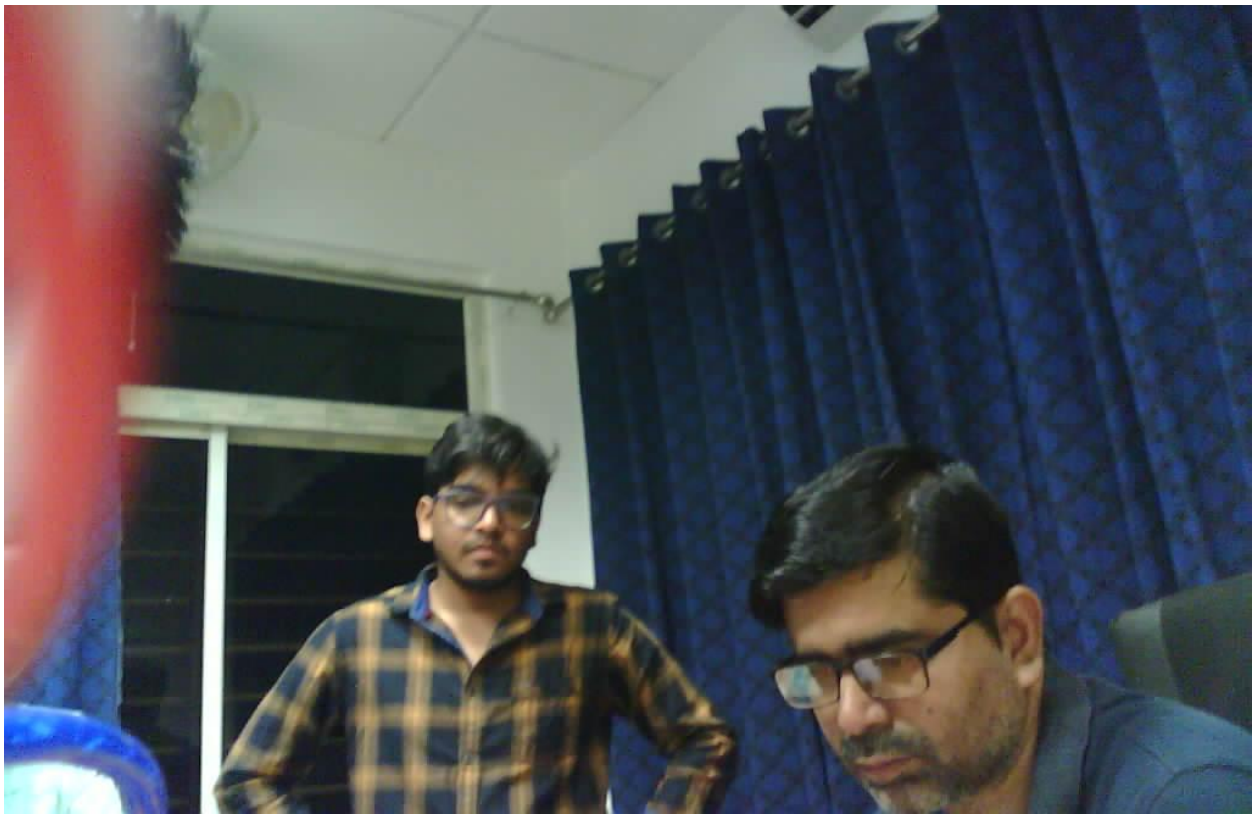
- If it blinks 5 times, then connection to Wi-Fi is successful.
- If it blinks 1 time, then connection to Wifi is unsuccessful.

Input/Output:

After powering on the device, we install it at a suitable position. Whenever a motion is detected, it captures an image and sends it to the telegram chat. Below are attached some images captured by the device.

Captured Image 1:

Captured Image 2:



Captured Image 3:

Serial Monitor Output:

```
Reset
ets Jul 29 2019 12:21:46

rst:0xc (SW_CPU_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:1216
ho 0 tail 12 room 4
load:0x40078000,len:9720
ho 0 tail 12 room 4
load:0x40080400,len:6352
entry 0x400806b8

Connecting to Redmi K50i

STAIP address:
192.168.102.19

0
0
0
0
0
0
1
Connected to api.telegram.org
.....
{"ok":true,"result":{"message_id":54,"from":{"id":6153206549,"is_bot":true,
```

Conclusion

The motion detection and capture project using a PIR sensor and Telegram chat integration is an excellent example of how the Internet of Things (IoT) can be used to create simple but effective home security systems. By detecting movement and sending captured images directly to a Telegram chat, users can remotely monitor their homes and receive instant notifications when motion is detected.

This project can be easily customized and extended with additional features such as integrating with other smart home devices, storing images in the cloud, or adding a camera module for high-quality video capture. Additionally, this project is cost-effective and can be implemented with readily available hardware components and open-source software libraries.

Overall, this project is a great way to learn about the basics of IoT, sensor integration, and software development. It is an excellent example of how IoT can be used to solve practical problems and improve our daily lives.

References

- https://core.telegram.org/
- https://docs.arduino.cc/
- https://docs.ai-thinker.com/en/esp32-cam
- https://cdn-learn.adafruit.com/downloads/pdf/pir-passive-infrared-proximity-motion-sensor.pdf