IoT Based Telegram Security Bot

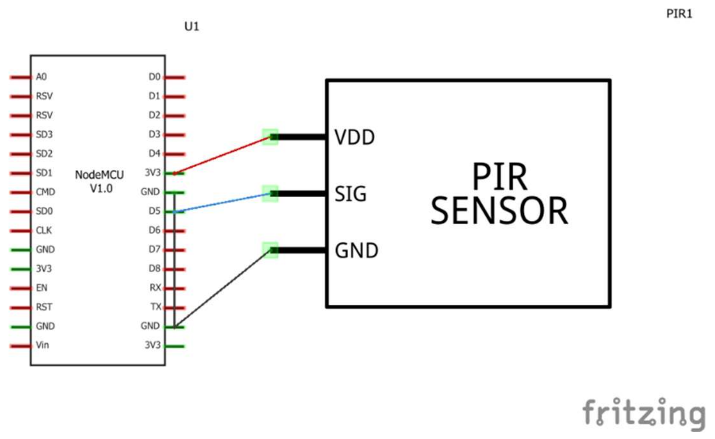
Pranoy Mukherjee
B.E Mechanical Engineering, Sant Longowal Institute of Engineering And Technology
Longowal – 148106

1. Aim: The goal of creating an IoT-based Telegram motion detection security bot is to enhance home or office security through real-time monitoring, instant Telegram alerts for detected motion, and remote system access. Integration with existing systems, cost-effective use of IoT devices, and an intuitive Telegram interface contribute to a comprehensive and user-friendly security solution. Scalability allows customization based on user needs, while energy efficiency is prioritized, particularly for battery-powered devices. Privacy measures ensure user confidentiality. Overall, the project aims to provide an adaptable, efficient, and privacy-respecting security system that empowers users to control and monitor their spaces seamlessly.

2. Introduction: Elevate your security with our cutting-edge IoT-based Telegram motion detection bot. Experience real-time monitoring, instant Telegram alerts, and remote access for unparalleled control. Seamlessly integrate with existing systems, embracing cost-effective IoT solutions with a user-friendly Telegram interface.

3. Implementation: Revolutionize agricultural security by strategically placing IoT motion sensors for real-time monitoring. These sensors detect motion and promptly alert farmers through a Telegram bot. Tailored for the agricultural field, this cost-effective solution integrates seamlessly, offering efficient and scalable security. Prioritizing energy efficiency is crucial for remote areas. The user-friendly Telegram interface ensures easy control, enhancing farmers' ability to monitor and safeguard their agricultural spaces effortlessly.

3.1 Schematic Circuit :



3.2 Equipment Used:

1. HC-SR501 PIR Sensor
2. NODEMCU ESP8266
3. Jumper Wires

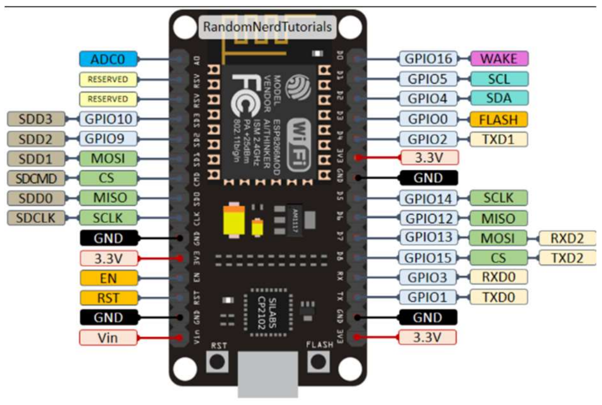
3.3 Power Supply:

Power supply is a reference to a source of electrical power. A device or system that supplies electrical or other types of energy to an output load or group of loads is called a power supply unit or PSU. The term is most applied to electrical energy supplies, less often to mechanical ones, and rarely to others. This power supply section is required to convert AC signal to DC signal and to reduce the amplitude of the signal. The available voltage signal from the main is 230V/50Hz which is an AC voltage, but the required is DC voltage with the amplitude of +5V and +12V for varies applications.

3.4 NODEMCU ESP8266:

The NodeMCU (Node Microcontroller Unit) is an opensource software and hardware development environment that is built around a very inexpensive System-on-a-Chip (SoC) called the ESP8266. And you must program it in low-level machine instructions that can be interpreted by the chip hardware. The ESP-8266 may be a low-cost Wi-Fi microchip with full TCP/IP Transfer control protocol/ Internet protocol). It makes web connectivity possible for the IOT panel. ESP8266 offers a whole and self-contained W-Fi.

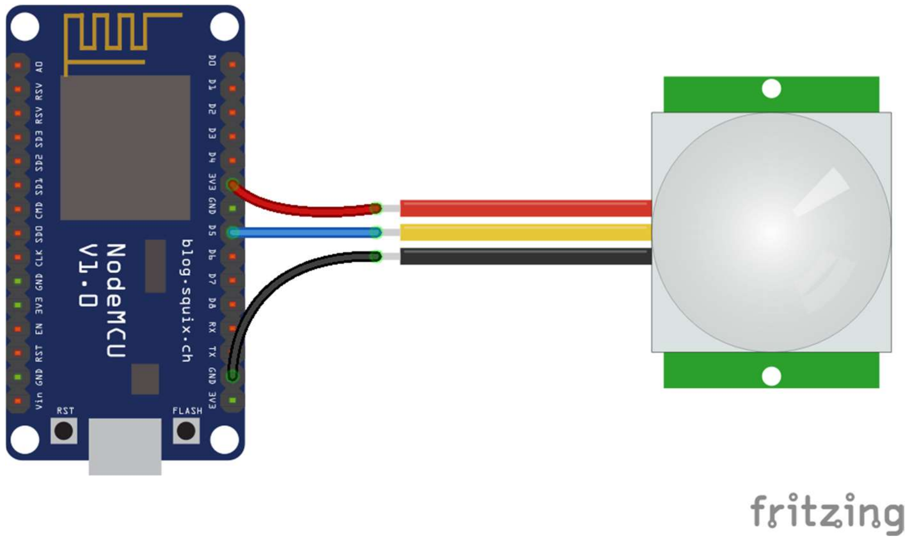
- 2.4 GHz Wi-Fi (802.11 b/g/n, supporting WPA/WPA2).
- General-purpose input/output (16 GPIO).
- Inter-Integrated Circuit (I^2C) serial communication protocol.
- Analog-to-digital conversion (10-bit ADC).
- Serial Peripheral Interface (SPI) serial communication protocol.



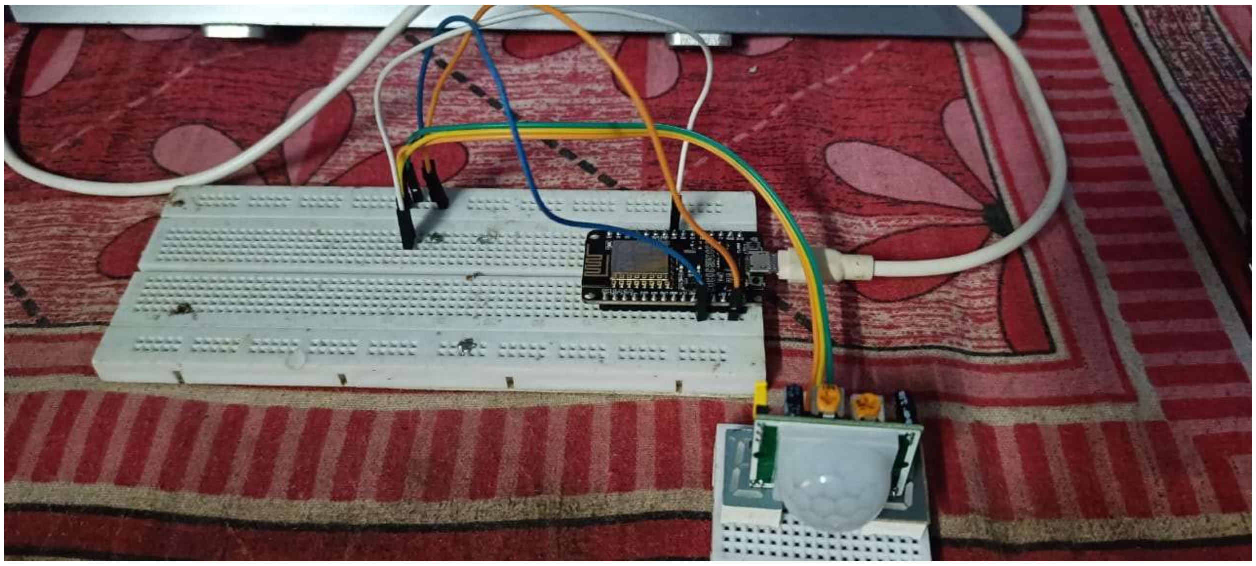
3.5 HC – SR501 PIR Sensor

The HC-SR501 PIR (Passive Infrared) sensor is a motion detector widely used in electronics projects. Compact and affordable, it detects infrared radiation emitted by moving objects. With adjustable sensitivity and time delay settings, it's versatile for various applications. Ideal for security systems, it triggers actions or alarms when motion is detected.

4. Circuit Diagram



4.1 Practical setup:



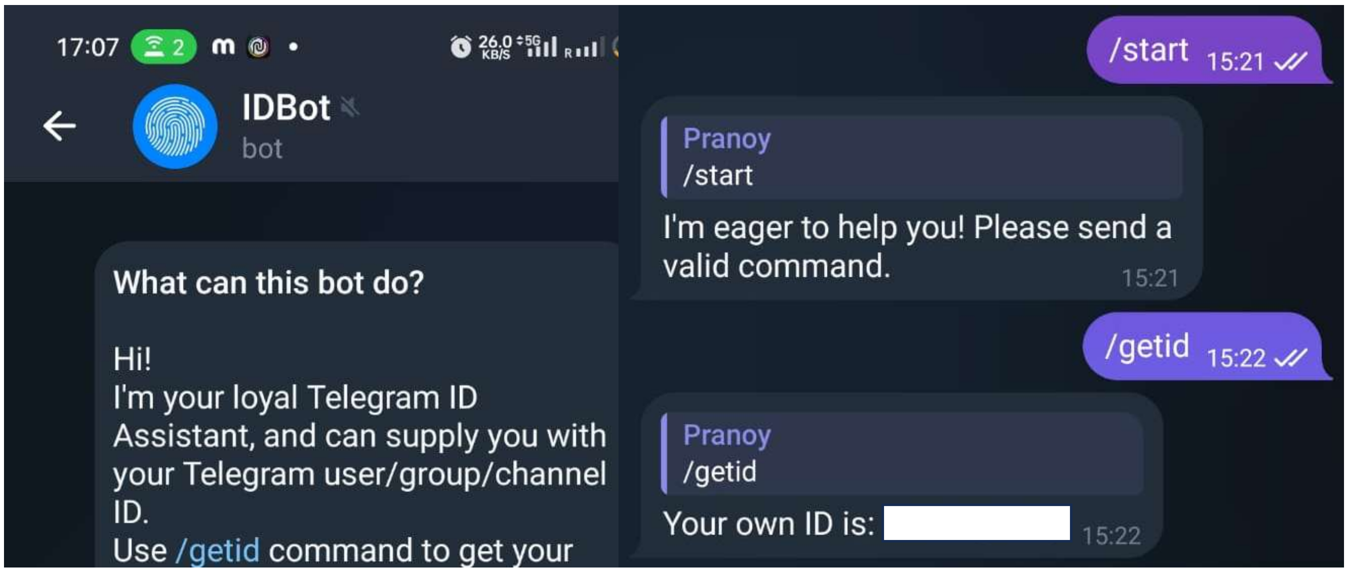
5. Telegram Setup :

5.1 Chat ID Setup:

Anyone that knows your bot username can interact with it. To make sure that we ignore messages that are not from our Telegram account (or any authorized users), you can get your Telegram User ID. Then, when your telegram bot receives a message, the ESP can check whether the sender ID corresponds to your User ID and handle the message or ignore it.

In your Telegram account, search for "**IDBot**" or open this link **t.me/myidbot** in your smartphone.

Start a conversation with that bot and type **/getid**. You will get a reply with your user ID. Save that userID, because you'll need it later in this tutorial.



17:07

IDBot
bot

What can this bot do?

Hi!
I'm your loyal Telegram ID Assistant, and can supply you with your Telegram user/group/channel ID.
Use /getid command to get your

/start 15:21

Pranoy
/start
I'm eager to help you! Please send a valid command. 15:21

/getid 15:22

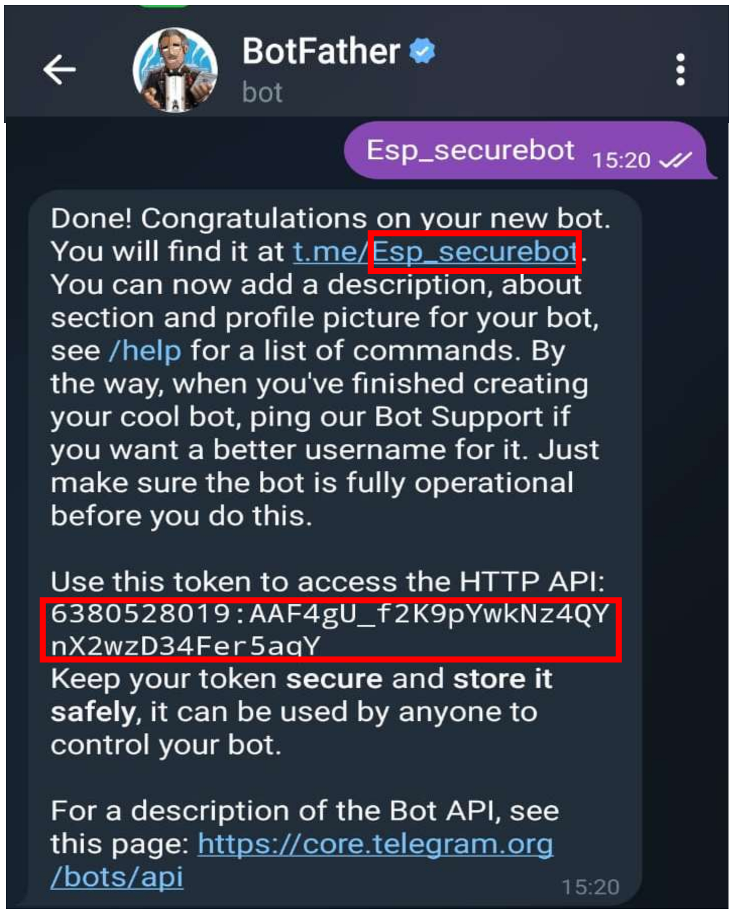
Pranoy
/getid
Your own ID is: [redacted] 15:22

5.2 Bot Setup :

Open Telegram and follow the next steps to create a Telegram Bot. First, search for "**botfather**" and click the BotFather. Or open this link **t.me/botfather** in your smartphone.

Type **/newbot** and follow the instructions to create your bot. Give it a name and username.

If your bot is successfully created, you'll receive a message with a link to access the bot and the bot token. Save the bot token because you'll need it so that the ESP8266 can interact with the bot.



Esp_securebot 15:20

Done! Congratulations on your new bot. You will find it at t.me/Esp_securebot. You can now add a description, about section and profile picture for your bot, see /help for a list of commands. By the way, when you've finished creating your cool bot, ping our Bot Support if you want a better username for it. Just make sure the bot is fully operational before you do this.

Use this token to access the HTTP API:
```
6380528019:AAF4gU_f2K9pYwkNz4QYnX2wzD34Fer5aqY
```
Keep your token **secure** and **store it safely**, it can be used by anyone to control your bot.

For a description of the Bot API, see this page: https://core.telegram.org/bots/api 15:20

6. Setting Up Of Arduino :

6.1 Setting up Universal Arduino Telegram Bot Library

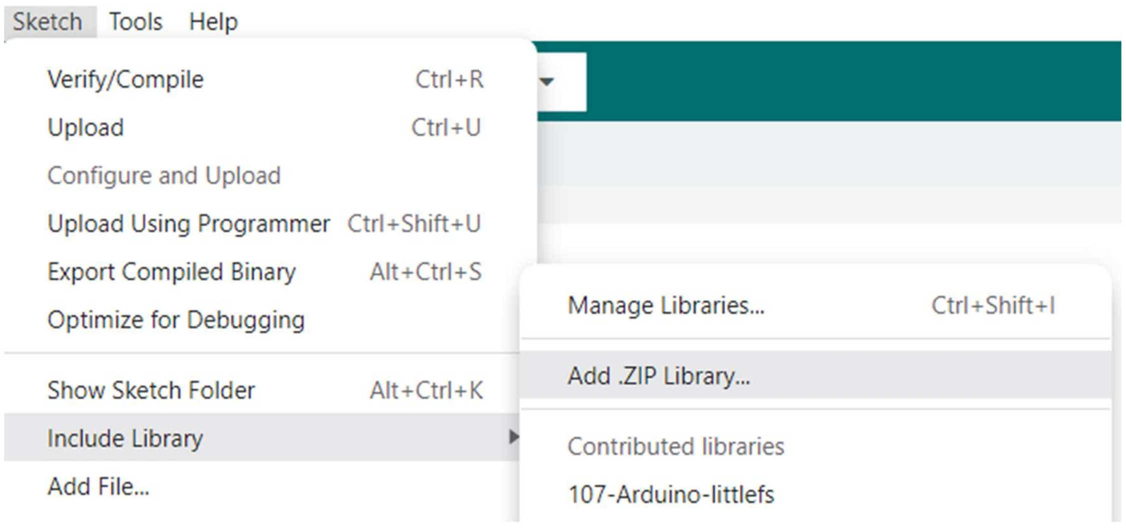
To interact with the Telegram bot, we'll use the **Universal Telegram Bot Library** created by Brian Lough that provides an easy interface for the Telegram Bot API.

Follow the next steps to install the latest release of the library.

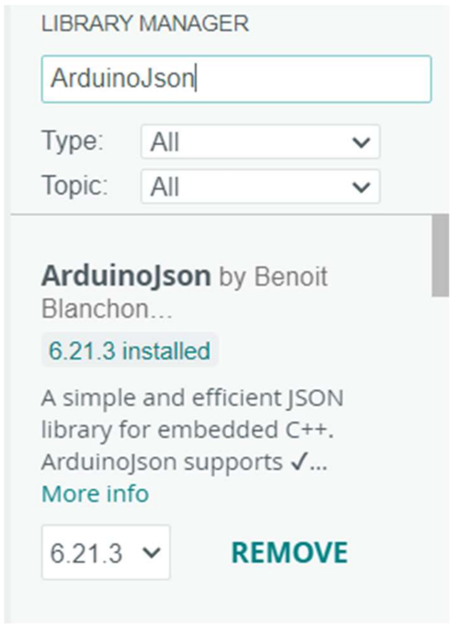
Click here to download the Universal Arduino Telegram Bot library.

Go to Sketch > Include Library > Add.ZIP Library...

Add the library you've just downloaded.



6.2 Inclusion of ArduinoJson Library :


LIBRARY MANAGER
ArduinoJson
Type: All
Topic: All
ArduinoJson by Benoit Blanchon...
6.21.3 installed
A simple and efficient JSON library for embedded C++. ArduinoJson supports ✓...
More info
6.21.3
REMOVE

7. Coding :

7.1 Sketch Uploaded to ESP8266 :

```
//Project created using Brian Lough's Universal Telegram Bot Library:
https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>

const char* ssid = "ESPtest"; //Replace with your WiFi SSID
const char* password = "HarryPotter"; //Replace with your password
#define BOTtoken "6380528019:AAF4gU_f2K9pYwkNz4QYnX2wzD34Fer5aqY"  // your Bot Token
(Get from Botfather)

#define CHAT_ID "1352467909"

X509List cert(TELEGRAM_CERTIFICATE_ROOT);
WiFiClientSecure client;
UniversalTelegramBot bot(BOTtoken, client);

const int motionSensor = 14; // PIR Motion Sensor
bool motionDetected = false;

void ICACHE_RAM_ATTR detectsMovement() {
  //Serial.println("MOTION DETECTED!!!");
  motionDetected = true;
}

void setup() {
  Serial.begin(9600);
  configTime(0, 0, "pool.ntp.org");
  client.setTrustAnchors(&cert);
  pinMode(motionSensor, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(motionSensor), detectsMovement, RISING);

  Serial.print("Connecting Wifi: ");
  Serial.println(ssid);
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  bot.sendMessage(CHAT_ID, "Bot started up", "");
}

void loop() {
  if(motionDetected){
    bot.sendMessage(CHAT_ID, "Motion detected!!", "");
    Serial.println("Motion Detected");
    motionDetected = false;
  }
}
```

7.2 Explanation Of Code :

7.2.1 Start by importing the required libraries.

```
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>
```

7.2.2 Network Credentials

Insert your network credentials in the following variables.

```
const char* ssid = "ESPtest"; //Replace with your WiFi SSID
const char* password = "HarryPotter"; //Replace with your password
```

7.2.3 Telegram Bot Token

Insert your Telegram Bot token you’ve got from Botfather on the BOTtoken variable.

```
#define BOTtoken "[redacted]"  // your Bot Token (Get from Botfather)
```

7.2.4 Telegram User ID

Insert your chat ID. The one you’ve got from the IDBot.

```
#define CHAT_ID "1352467909"
```

7.2.5 Create a new WiFi client with WiFiClientSecure.

```
WiFiClientSecure client;
```

7.2.6 Create a bot with the token and client defined earlier.

```
UniversalTelegramBot bot(BOTtoken, client);
```

7.2.7 Motion Sensor

Define the GPIO that the motion sensor is connected to.

```
const int motionSensor = 14; // PIR Motion Sensor
```

The motionDetected boolean variable is used to indicate whether motion was detected or not. It is set to false by default.

```
bool motionDetected = false;
```

detectsMovement()

The detectsmovement() function is a callback function that will be executed when motion is detected. In this case, it simply changes the state of the motionDetected variable to true.

```
void ICACHE_RAM_ATTR detectsMovement() {
  //Serial.println("MOTION DETECTED!!!");
  motionDetected = true;
}
```

setup()

In the setup(), initialize the Serial Monitor.

```
Serial.begin(9600);
```

For the ESP8266, you need to use the following line:

client.setInsecure();

7.2.8 PIR Motion Sensor Interrupt

Set the PIR motion sensor as an interrupt and set the detectsMovement() as the callback function (when motion is detected, that function will be executed :

```
pinMode(motionSensor, INPUT_PULLUP);
attachInterrupt(digitalPinToInterrupt(motionSensor), detectsMovement, RISING);
```

7.2.9 Init Wi-Fi

Initialize Wi-Fi and connect the ESP8266 to your local network with the SSID and password defined earlier.

```
WiFi.mode(WIFI_STA);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(500);
  }

Serial.println("");
Serial.println("WiFi connected");
Serial.print("IP address: ");
Serial.println(WiFi.localIP());
```

Finally, send a message to indicate that the Bot has started up:

```
bot.sendMessage(CHAT_ID, "Bot started up", "");
```

7.2.10 loop()

In the loop(), check the state of the motionDetected variable.

```
void loop() {
if(motionDetected){
```

If it’s true, it means that motion was detected. So, send a message to your Telegram account indicating that motion was detected.

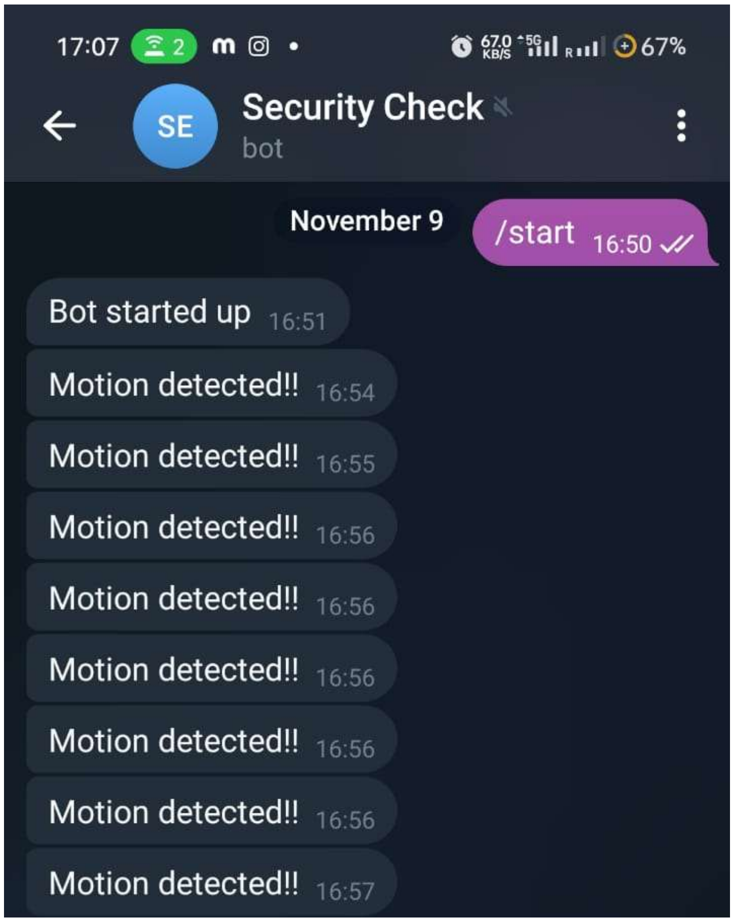
```
bot.sendMessage(CHAT_ID, "Motion detected!!", "");
```

Finally, after sending the message, set the motionDetected variable to false, so it can detect motion again.

```
motionDetected = false;
```

That’s pretty much how the code works.

7.2.11 Final Outcome:



November 9

/start 16:50

Bot started up 16:51

Motion detected!! 16:54

Motion detected!! 16:55

Motion detected!! 16:56

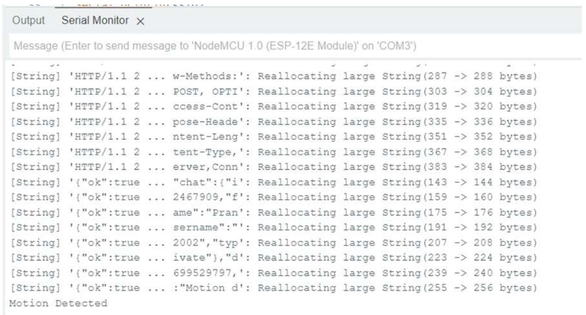
Motion detected!! 16:56

Motion detected!! 16:56

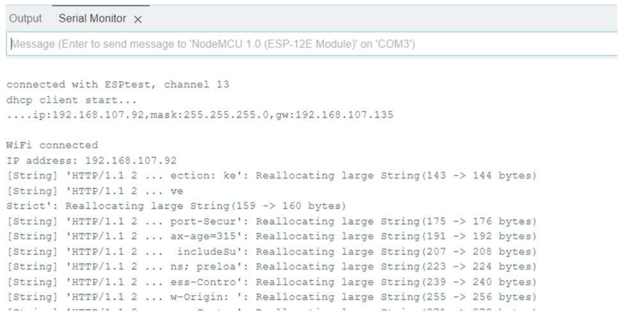
Motion detected!! 16:56

Motion detected!! 16:56

Motion detected!! 16:57


Output
Serial Monitor
Message (Enter to send message to 'NodeMCU 1.0 (ESP-12E Module)' on 'COM3')

```
[String] 'HTTP/1.1 2 ... w-Methods:': Reallocating large String(287 -> 288 bytes)
[String] 'HTTP/1.1 2 ... POST, OPTI': Reallocating large String(303 -> 304 bytes)
[String] 'HTTP/1.1 2 ... ccess-Cont': Reallocating large String(319 -> 320 bytes)
[String] 'HTTP/1.1 2 ... pose-Heade': Reallocating large String(335 -> 336 bytes)
[String] 'HTTP/1.1 2 ... ntent-Leng': Reallocating large String(351 -> 352 bytes)
[String] 'HTTP/1.1 2 ... tent-Type,': Reallocating large String(367 -> 368 bytes)
[String] 'HTTP/1.1 2 ... erver,Conn': Reallocating large String(383 -> 384 bytes)
[String] '{"ok":true ... "chat":{"i': Reallocating large String(143 -> 144 bytes)
[String] '{"ok":true ... 2467909,"f': Reallocating large String(159 -> 160 bytes)
[String] '{"ok":true ... ame":"Pran': Reallocating large String(175 -> 176 bytes)
[String] '{"ok":true ... sername":"': Reallocating large String(191 -> 192 bytes)
[String] '{"ok":true ... 2002","typ': Reallocating large String(207 -> 208 bytes)
[String] '{"ok":true ... ivate"},"d': Reallocating large String(223 -> 224 bytes)
[String] '{"ok":true ... 699529797,': Reallocating large String(239 -> 240 bytes)
[String] '{"ok":true ... :"Motion d': Reallocating large String(255 -> 256 bytes)
Motion Detected
```


Output
Serial Monitor
Message (Enter to send message to 'NodeMCU 1.0 (ESP-12E Module)' on 'COM3')

```
connected with ESPtest, channel 13
dhcp client start...
....ip:192.168.107.92,mask:255.255.255.0,gw:192.168.107.135

WiFi connected
IP address: 192.168.107.92
[String] 'HTTP/1.1 2 ... ection: ke': Reallocating large String(143 -> 144 bytes)
[String] 'HTTP/1.1 2 ... ve
Strict': Reallocating large String(159 -> 160 bytes)
[String] 'HTTP/1.1 2 ... port-Secur': Reallocating large String(175 -> 176 bytes)
[String] 'HTTP/1.1 2 ... ax-age=315': Reallocating large String(191 -> 192 bytes)
[String] 'HTTP/1.1 2 ...  includeSu': Reallocating large String(207 -> 208 bytes)
[String] 'HTTP/1.1 2 ... ns; preloa': Reallocating large String(223 -> 224 bytes)
[String] 'HTTP/1.1 2 ... ess-Contro': Reallocating large String(239 -> 240 bytes)
[String] 'HTTP/1.1 2 ... w-Origin: ': Reallocating large String(255 -> 256 bytes)
```