

# **House Monitoring System Using IOT**

Submitted in partial fulfillment of the requirements for the award of

Bachelor of Engineering Degree in

Computer Science and Engineering

By

**SODUM SIDDARTHA (17SCS0439)**

**SHAIK RAFI (17SCS5183)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SCHOOL OF COMPUTING**

**SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(DEEMED TO BE UNIVERSITY)**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,**

**CHENNAI – 600119, TAMILNADU.**

**MARCH - 2021**



# **SATHYABAMA**

**INSTITUTE OF SCIENCE AND TECHNOLOGY  
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade “A” by NAAC**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600119**

**[www.sathyabamauniversity.ac.in](http://www.sathyabamauniversity.ac.in)**



---

## **DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

### **BONAFIDE CERTIFICATE**

This is to certify that this project report is the bonafide work of **SODUM SIDDARTHA (17SCS0439)** and **SHAIK RAFI (17SCS5183)** who carried out the project entitled “**HOUSE MONITORING SYSTEM USING IOT**” under my supervision from **August 2020** to **March 2021**.

**Internal Guide**

**Ms.Yogitha R, M.E.,Ph.D**

**Head of the Department**

---

**Submitted for Viva voce Examination held on \_\_\_\_\_**

**Internal Examiner**

**External Examiner**

## **DECLARATION**

I **SODUM SIDDARTHA** here by declare that the Project Report entitled “**HOUSE MONITORING SYSTEM USING IOT**” is done by me under the guidance of **Ms.Yogitha R, M.E.,Ph.D** Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

**DATE:**

**PLACE: CHENNAI**

**SIGNATURE OF THE CANDIDATE**

## ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the **Board of management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am are grateful to them.

I convey my thanks to **Dr.T.SASIKALA, M.E., Ph.D., Dean, School of Computing** and **Dr. S. VIGNESWARI, M.E., Ph.D.,Head of the Department, Department of Computer Science and Engineering** for providing me the necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Ms.Yogitha R, M.E.,Ph.D** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the department of **COMPUTER SCIENCE AND ENGINEERING** who were helpful in many ways for the completion of project.

## **ABSTRACT**

The technology is increasing quickly, that ends up in associate upgradation in home security system. Automation in security sector makes it plenty of authentic. There are many electrical instrumentation's in homes within the same manner this equipment is employed to produce security. The house security system is planned at the aspect of the capturing footage of persons through camera sensors. The system provides strengthened security functions that it transfer recorded footage to a user's mobile device once associate invalid user attempt to open or break the door. The illegal operation is happens and it collectively permits a user to see the access information and by practice the commands or selections the user will do the action on that. it's remotely operate the door lock to strengthen the house security. A stand alone system through internet of Things as a network of communication is implemented. Raspberry Pi is utilized as dominant unit coded in Python language.

Keywords- Internet of things, Raspberry pi, Python language.

## TABLE OF CONTENTS

| CHAPTER NO | TITLE                                     | PAGE NO     |
|------------|-------------------------------------------|-------------|
|            | <b>ABSTRACT</b>                           | <b>v</b>    |
|            | <b>LIST OF FIGURES</b>                    | <b>viii</b> |
|            | <b>LIST OF ABEVERSACTIONS</b>             | <b>ix</b>   |
| <b>1</b>   | <b>INTRODUCTION</b>                       | <b>1</b>    |
|            | 1.1 INSIGHT TO PROJECT                    | 1           |
|            | 1.2 STATEMENT OF THE PROBLEM              | 2           |
|            | 1.3 IMPORTANCE OF HOUSE MONITORING SYSTEM | 2           |
| <b>2</b>   | <b>LITERATURE SURVEY</b>                  | <b>3</b>    |
|            | 2.1 INTRODUCTION TO RASPBERRY PI 4        | 3           |
|            | 2.1.1 HARDWARE                            | 4           |
|            | 2.2 PROCESSOR                             | 5           |
|            | 2.3 HISTORY OF PYTHON                     | 7           |
|            | 2.3.1 PYTHON FEATURES                     | 7           |
| <b>3</b>   | <b>AIM AND SCOPE</b>                      | <b>8</b>    |
|            | 3.1 AIM                                   | 8           |
|            | 3.2 EXISTING SYSTEM                       | 8           |
|            | 3.3 PROPOSED SYSTEM                       | 8           |
|            | 3.3.1 MODULE                              | 9           |
| <b>4</b>   | <b>SYSTEM ANALYSIS AND METHODOLOGY</b>    | <b>11</b>   |
|            | 4.1 SYSTEM ANALYSIS                       | 11          |

|          |                                        |           |
|----------|----------------------------------------|-----------|
|          | 4.1.1 USER REQUIREMENTS                | 12        |
|          | 4.2 TOOL WINDOWS                       | 14        |
|          | 4.3 NAVIGATIONS                        | 15        |
|          | 4.4 STYLE AND FORMATTING               | 15        |
|          | 4.5 LIMITATIONS                        | 17        |
| <b>5</b> | <b>RESULT AND DISCUSSION</b>           | <b>18</b> |
|          | 5.1 CONNECTING MODULES TO RASPBERRY PI | 18        |
|          | 5.2 DETECTING THE MOTION               | 18        |
|          | 5.3 SENDING THE MAIL AND RECEIVING SMS | 19        |
|          | 5.4 OPENING THE DOOR LOCK              | 20        |
| <b>6</b> | <b>CONCLUSION AND SUMMARY</b>          | <b>21</b> |
|          | 6.1 CONCUSION                          | 21        |
|          | 6.2 SUMMARY                            | 21        |
|          | <b>REFERENCES</b>                      | <b>22</b> |
|          | <b>APPENDIX</b>                        | <b>23</b> |
|          | A. SOURCE CODE                         | 23        |

## LIST OF FIGURES

| FIGURE NO | FIGURE NAME                         | PAGE NO |
|-----------|-------------------------------------|---------|
| 1         | RASPBERRY PI HARDWARE               | 3       |
| 2.3       | FOUNDER OF PYTHON                   | 5       |
| 2.4       | EXECUTION OF SOURCE CODE            | 7       |
| 3.3       | PROCESS OF THE PROPOSED SYSTEM      | 9       |
| 4.1       | PROJECT FILES IN THONNY             | 14      |
| 4.2       | THE RASPBERRY PI THONNY MAIN WINDOW | 14      |
| 4.4       | CODE BEFORE FORMATTING              | 17      |
| 4.5       | CODE AFTER FORMATTING               | 18      |
| 5.1       | CONNECTING SENSOR TO RASPBERRY PI   | 20      |
| 5.2       | MOTION AND IMAGES ARE CAPTURED      | 21      |
| 5.3       | IMAGES SENT TO MAIL                 | 21      |
| 5.4       | ACTION TAKE THROUGH COMMANDS        | 22      |
| 5.5       | THE DOOR LOCK OPENS                 | 22      |

## LIST OF ABBREVIATIONS

| ABBREVIATION | EXPANSION                          |
|--------------|------------------------------------|
| IDE          | INTEGRATED DEVELOPMENT ENVIRONMENT |
| UG           | UNRESTRICTED GUEST                 |
| PCB          | PRINTED CIRCUIT BOARD              |
| PIR          | PASSIVE INFRARED SENSOR            |
| RFID         | RADIO FREQUENCY IDENTIFICATION     |

# **CHAPTER I**

## **INTRODUCTION**

### **1.1 INSIGHT TO PROJECT**

The present situation ensures the security has become an inevitably essential. There's a regressive progress within the security system because the influence of recent technology is reaching its peak. once there's a contemporary home with minimum human effort, it's renowned as fashionable home. Since there's associate advent of wireless and digital technologies, all at once it introduces a machine-driven intelligent security system. The machine-driven home security system is designed with the camera and motion sensors, and also the use of those sensors are going to be shaping the options of those sensors. The information transmission is happening through the GSM module to security systems that helps the user to manage and monitor the system globally. The new IoT based mostly product and services can grow exponentially in next few years foreseen by the analysts. The IoT involves completely different link layer technologies and an enormous vary of devices. IoT provides open access to specific set of information. Raspberry Pi may be a compact digital computer that is sensible enough to provide the nice property to the web likewise as improve the signals. This framework principally envelopes the house security system from the sensing element, networking, integrates real time information and data management. This planned system has high latency and low value. The system is reliable and consumes very less power as compared with alternative system. The house security system supported some camera connected to the house and also the output for this can be in real time to time within the operation.

respects purchasers subtleties and dealings in facilitate delivered, in addition our things bundle holding up list. All of this knowledge ought to be overseen in an exceedingly productive and value savvy vogue that the association assets may well be viably used. The objective of Vesture companies is it's board framework is to mechanize the administration of the Vesture firm making it more productive and mistake free. It targets normalizing information, solidifying information guaranteeing information respectability and decreasing

irregularities, using profoundly mechanized interaction that is peaceful, dependable and fast using asp.net PC programming language and SQL data set application to both the clients and the staff accountable for the staffing and Vesture the executives measures. XML would be at the front-end and give the graphical UI that relates with the client, while the SQL information base will be at the back-finish to deal with the information stockpiling measure.

## **1.2 STATEMENT OF THE PROBLEM**

The experience of monitoring the house is not pleasant at all. It's the only door to enter our home and this system will help you monitoring.

This system is not completely machine dependent. Every time this system will notify it's owner and he will be deciding the next process.

## **1.3 IMPORTANCE OF HOUSE MONITORING SYSTEM:**

Home monitoring system helps to keep your home safe and secure. They will not only secure the home but it also communicate to it's owner about home. These can make the owner of the home to sleep happily without making him worry about his home when he is far from home.

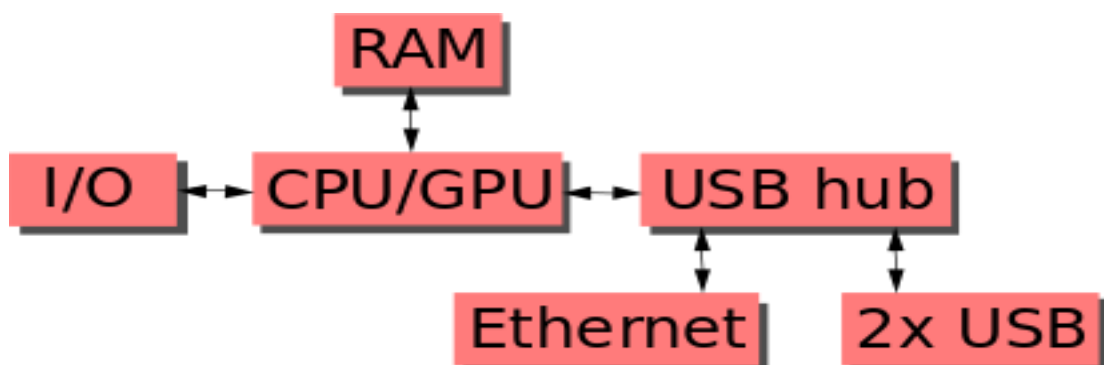
## CHAPTER 2

### LITERATURE SURVEY

#### 2.1 INTRODUCTION TO RASPBERRY PI 4

Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation in association with Broadcom. The Raspberry Pi project originally leaned towards the promotion of teaching basic computer science in schools and in developing countries. The original model became more popular than anticipated, selling outside its target market for uses such as robotics. It is widely used in many areas, such as for weather monitoring, because of its low cost, modularity, and open design. It is typically used by computer and electronic hobbyists, due to its adoption of HDMI and USB devices. After the release of the second board type, the Raspberry Pi Foundation set up a new entity, named Raspberry Pi Trading, and installed Eben Upton as CEO, with the responsibility of developing technology. The Foundation was rededicated as an educational charity for promoting the teaching of basic computer science in schools and developing countries. The Raspberry Pi is one of the best-selling British computers. As of December 2019, more than thirty million boards have been sold. Most Pis are made in a Sony factory in Pencoed, Wales, while others are made in China and Japan.

**2.1.1 Hardware** The Raspberry Pi hardware has evolved through several versions that feature variations in the type of the central processing unit, amount of memory capacity, networking support, and peripheral-device support.



This block diagram describes models B, B+, A and A+. The Pi Zero models are similar, but lack the Ethernet and USB hub components. The Ethernet adapter is internally connected to an additional USB port. In Model A, A+, and the Pi Zero, the USB port is connected directly to the system on a chip (SoC). On the Pi 1 Model B+ and later models the USB/Ethernet chip contains a five-port USB hub, of which four ports are available, while the Pi 1 Model B only provides two. On the Pi Zero, the USB port is also connected directly to the SoC, but it uses a micro USB (OTG) port. Unlike all other Pi models, the 40 pin GPIO connector is omitted on the Pi Zero, with solderable through-holes only in the pin locations. The Pi Zero WH remedies this. Processor speed ranges from 700 MHz to 1.4 GHz for the Pi 3 Model B+ or 1.5 GHz for the Pi 4; on-board memory ranges from 256 MiB to 1 GiB random-access memory (RAM), with up to 8 GiB available on the Pi 4. Secure Digital (SD) cards in MicroSDHC form factor (SDHC on early models) are used to store the operating system and program memory. The boards have one to five USB ports. For video output, HDMI and composite video are supported, with a standard 3.5 mm tip-ring-sleeve jack for audio output. Lower-level output is provided by a number of GPIO pins, which support common protocols like. The B-models have an 8P8C Ethernet port and the Pi 3, Pi 4 and Pi Zero W have on-board Wi-Fi 802.11n and Bluetooth.

## **2.2 PROCESSOR**

The Broadcom BCM2835 SoC used in the first generation Raspberry Pi includes a 700 MHz ARM1176JZF-S processor, VideoCore IV graphics processing unit (GPU), and RAM. It has a level 1 (L1) cache of 16 KiB and a level 2 (L2) cache of 128 KiB. The level 2 cache is used primarily by the GPU. The SoC is stacked underneath the RAM chip, so only its edge is visible. The ARM1176JZ(F)-S is the same CPU used in the original iPhone, although at a higher clock rate, and mated with a much faster GPU. The earlier V1.1 model of the Raspberry Pi 2 used a Broadcom BCM2836 SoC with a 900 MHz 32-bit, quad-core ARM Cortex-A7 processor, with 256 KiB shared L2 cache. The Raspberry Pi 2 V1.2 was upgraded to a Broadcom BCM2837 SoC with a 1.2 GHz 63-bit quad-core ARM Cortex-A53 processor, the same SoC which is used on the Raspberry Pi 3, but underclocked (by default) to the same

900 MHz CPU clock speed as the V1.1. The BCM2836 SoC is no longer in production as of late 2016. The Raspberry Pi 3 Model B uses a Broadcom BCM2837 SoC with a 1.2 GHz 64-bit quad-core ARM Cortex-A53 processor, with 512 KiB shared L2 cache. The Model A+ and B+ are 1.4 GHz. The Raspberry Pi 4 uses a Broadcom BCM2711 SoC with a 1.5 GHz 64-bit quad-core ARM Cortex-A72 processor, with 1 MiB shared L2 cache. Unlike previous models, which all used a custom interrupt controller poorly suited for virtualisation, the interrupt controller on this SoC is compatible with the ARM Generic Interrupt Controller (GIC) architecture 2.0, providing hardware support for interrupt distribution when using ARM virtualisation capabilities. The Raspberry Pi Zero and Zero W use the same Broadcom BCM2835 SoC as the first generation Raspberry Pi, although now running at 1 GHz CPU clock speed. The Raspberry Pi Pico uses the RP2040 running at 133 MHz.

## 2.3 HISTORY OF PYTHON

- Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands. Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Small Talk, Unix shell, and other scripting languages.
- Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL). Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.



***Fig.No.2.3 FOUNDER OF PYTHON***

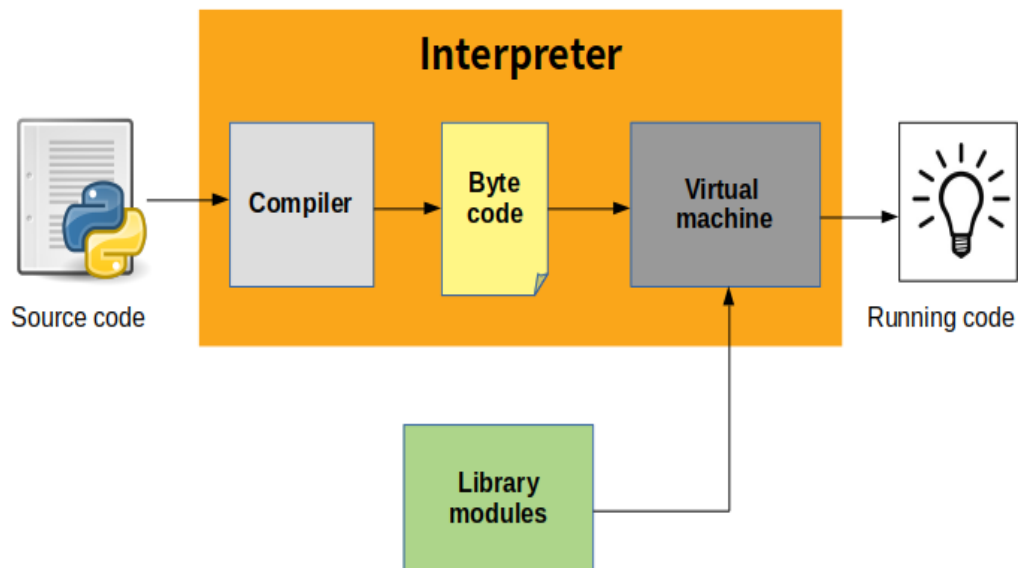
### **2.3.1 PYTHON FEATURES**

Python's features include:

- Easy-to-learn: Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- Easy-to-read: Python code is more clearly defined and visible to the eyes.
- Easy-to-maintain: Python's source code is fairly easy-to-maintain.
- A broad standard library: Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- Interactive Mode: Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- Portable: Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- Extendable: You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- Databases: Python provides interfaces to all major commercial databases.
- GUI Programming: Python supports GUI applications that can be created and ported to many system calls, libraries, and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- Scalable: Python provides a better structure and support for large programs than shell scripting.
- A high-level, interpreted language: Unlike C/C++, you don't have to worry about daunting tasks like memory management, garbage collection and so on. Likewise, when you run Python code, it automatically converts your code to the language your computer understands. You don't need to worry about any lower level

operations.

- Large standard libraries to solve common tasks: Python has a number of standard libraries which makes life of a programmer



***Fig.No.(2.4) Execution of Source code***

- much easier since you don't have to write all the code yourself. For example: Need to connect MySQL database on a Web server You can use MySQL dB library using `import MySQLdB` Standard libraries in Python are well tested and used.

## **CHAPTER 3**

### **AIM AND SCOPE OF THE PROJECT**

#### **3.1 AIM**

The smart lock is an electromechanical lock which is designed to perform locking and unlocking operations on a door when it receives such instructions from an authorised device using a wireless. This is not a complete machine dependent project. It uses many techniques for monitoring the home, By using this we can monitor the home from anywhere around the world. Using these techniques we can reduce the chances of theft and make our home secure.

#### **3.2 EXISTING SYSTEM**

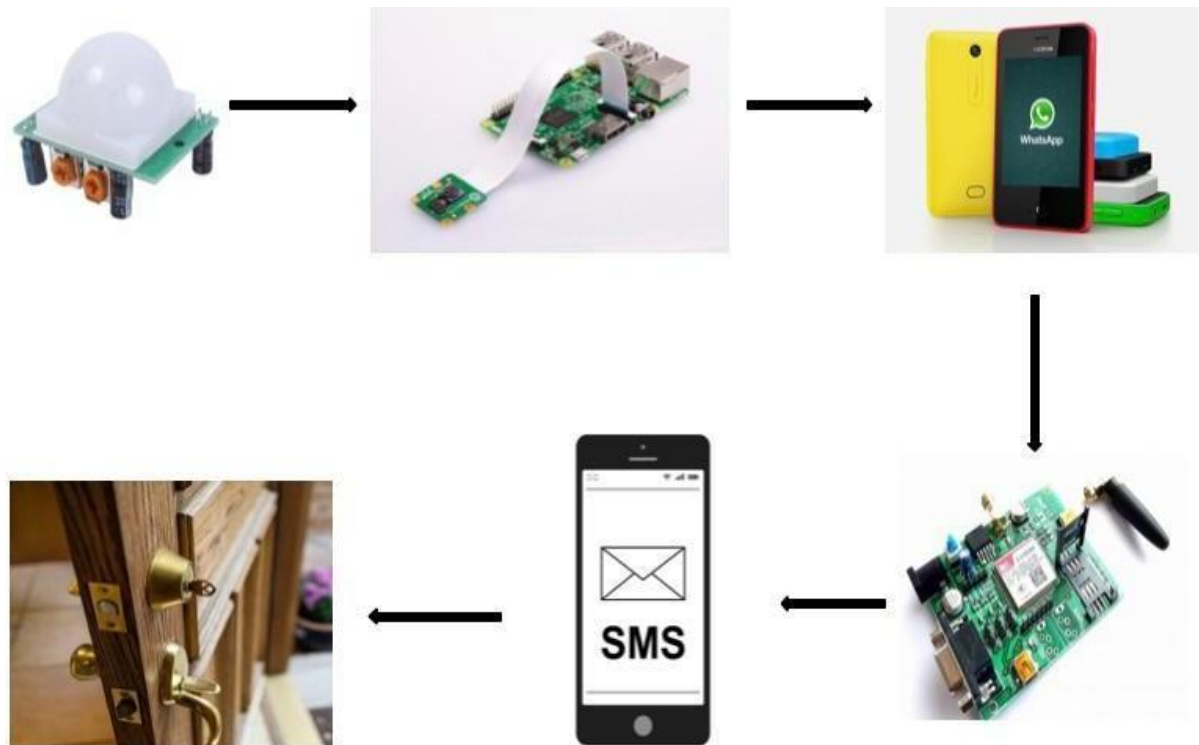
The very first thing we need to do is get our LCD touchscreen working. Depending on which model you purchase for your build, it may just come with drivers to install or it may come with its own Operating System image for you to write to your Raspberry Pi's SD card. That's the case with my one, so I need to use Win32 Disk Imager to write the image to my SD card. Make sure you select the correct device to write to, so you don't accidentally erase any external hard drives.

Once that's complete, pop the microSD card back into the Raspberry Pi and fit the LCD to the GPIO ports like this, starting at the top end which is by the SD card slot and away from the USB and ethernet ports. Next, take the microUSB power lead and plug it in to boot up the Raspberry Pi to confirm that the screen – and the touchscreen element of it – are working correctly.

#### **3.3 PROPOSED SYSTEM**

The Home Monitoring System is designed for any Home to replace their existing system. The new system is connected to the gsm module which helps to provide the information through a sms and process the program. These services are to be provided in an efficient, cost effective manner, with the goal of providing security and to keep the house secure. Since the existing system

makes use of IR sensor which makes us to carry the key all the time if we lost the key it will create another way of trouble so to avoid all these we use GSM module which helps to connect to the phone.



**FIG 3.3: DIAGRAM OF THE PROCESS OF THE PROPOSED SYSTEM**

### **3.3.1 MODULE**

#### **i. PIR**

PIR sensors are more complicated than many of the other sensors explained in these tutorials (like photocells, FSRs and tilt switches) because there are multiple variables that affect the sensors input and output. To begin explaining how a basic sensor works, we'll use this rather nice diagram. The PIR sensor itself has two slots in it, each slot is made of a special material that is sensitive to IR. The lens used here is not really doing much and so we see that the two slots can 'see' out past some distance (basically the sensitivity of the sensor). When the sensor is idle, both slots detect the same amount of IR, the ambient amount radiated from the room or walls

or outdoors. When a warm body like a human or animal passes by, it first intercepts one half of the PIR sensor, which causes a *positive differential* change between the two halves. When the warm body leaves the sensing area, the reverse happens, whereby the sensor generates a negative differential change.

## ii. **Raspberry pi 4**

There are four applications provided: raspistill, raspivid, raspiyuv and raspividuvc. raspistill and raspiyuv are very similar and are intended for capturing images; raspivid and raspividuvc are for capturing video. All the applications are driven from the command line, and written to take advantage of the MMAL API which runs over OpenMAX. The MMAL API provides an easier to use system than that presented by OpenMAX. Note that MMAL is a Broadcom-specific API used only on VideoCore 4 systems.

## iii. **Gsm Module**

A GSM module or a GPRS module is a chip or circuit that will be used to establish communication between a mobile device or a computing machine and a GSM or GPRS system. The modem (modulator-demodulator) is a critical part here.

These modules consist of a GSM module or GPRS modem powered by a power supply circuit and communication interfaces (like RS-232, USB 2.0, and others) for computer. A GSM modem can be a dedicated modem device with a serial, USB or Bluetooth connection, or it can be a mobile phone that provides GSM modem capabilities.

## iv. **Wifi Module**

This module is a inbuilt module in raspberry pi 4 to provide internet connectivity to your projects. The module can work both as a Access point (can create hotspot) and as a station (can connect to Wi-Fi),

hence it can easily fetch data and upload it to the internet making Internet of Things as easy as possible. It can also fetch data from internet using API's hence your project could access any information that is available in the internet, thus making it smarter. Another exciting feature of this module is that it can be programmed using the Arduino IDE which makes it a lot more user friendly. However this version of the module has only 2 GPIO pins (you can hack it to use upto 4) so you have to use it along with another microcontroller like Raspberry pi, else you can look onto the more standalone **ESP-12** or **ESP-32** versions.

v. **12v Electronic Solenoid Door Lock**

This **12V Solenoid lock** has a slug with a slanted cut and a good mounting bracket. It's basically an electronic lock, designed for a basic cabinet, safe or door. When 9-12VDC is applied, the slug pulls in so it doesn't stick out and the door can be opened. It does not use any power in this state. It is very easy to install for automatic door lock systems like electric door lock with the mounting board. This solenoid in particular is nice and strong

#### vi. **Camera Module**

This module is a inbuilt module in raspberry pi 4 to provide internet connectivity to your projects. The module can work both as a Access point (can create hotspot) and as a station (can connect to Wi-Fi), hence it can easily fetch data and upload it to the internet making **Internet of Things** as easy as possible. It can also fetch data from internet using API's hence your project could access any information that is available in the internet, thus making it smarter. Another exciting feature of this module is that it can be programmed using the Arduino IDE which makes it a lot more user friendly. However this version of the module has only 2 GPIO pins (you can hack it to use upto 4) so you have to use it along with another microcontroller like Raspberry pi, else you can look onto the more standalone **ESP-12** or **ESP-32** versions.

#### vii. **Dielectric Door Lock**

This **12V Solenoid lock** has a slug with a slanted cut and a good mounting bracket. It's basically an electronic lock, designed for a basic cabinet, safe or door. When 9-12VDC is applied, the slug pulls in so it doesn't stick out and the door can be opened. It does not use any power in this state. It is very easy to install for automatic door lock systems like electric door lock with the mounting board. This solenoid in particular is nice and strong.

## **CHAPTER 4**

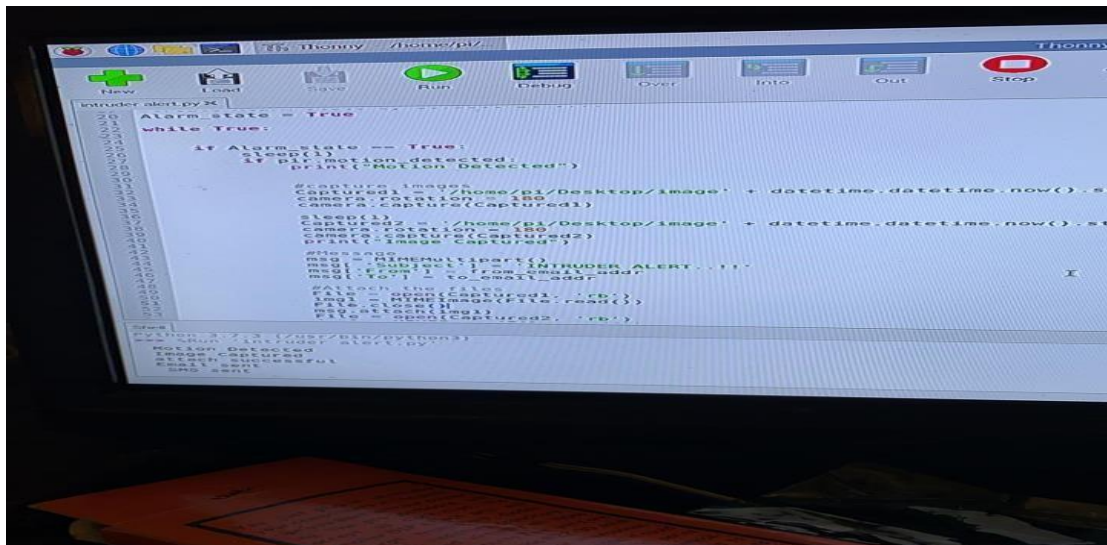
### **SYSTEM ANALYSIS AND METHODOLOGY**

#### **4.1 SYSTEM ANALYSIS**

System Analysis and Design System analysis is a method of problem-solving that deals with the breaking down of a system into components parts in order to study how well the individual parts work and interact to accomplish their purpose. It involves the process of enumerating the existing problems, analyzing the proposed system for costs and benefits, analyzing the system and user requirements and considering possible alternative system. System analysis is important in the design of subsequent systems. System design consists of design activities that produce system specifications which satisfy the functional requirements that have been developed in the system analysis process. System design is basically the structural implementation of system analysis. The proposed system is being designed in such a way that users only need to input their customer data which is then entered into a computer database.

##### ***4.1.1 User Requirements***

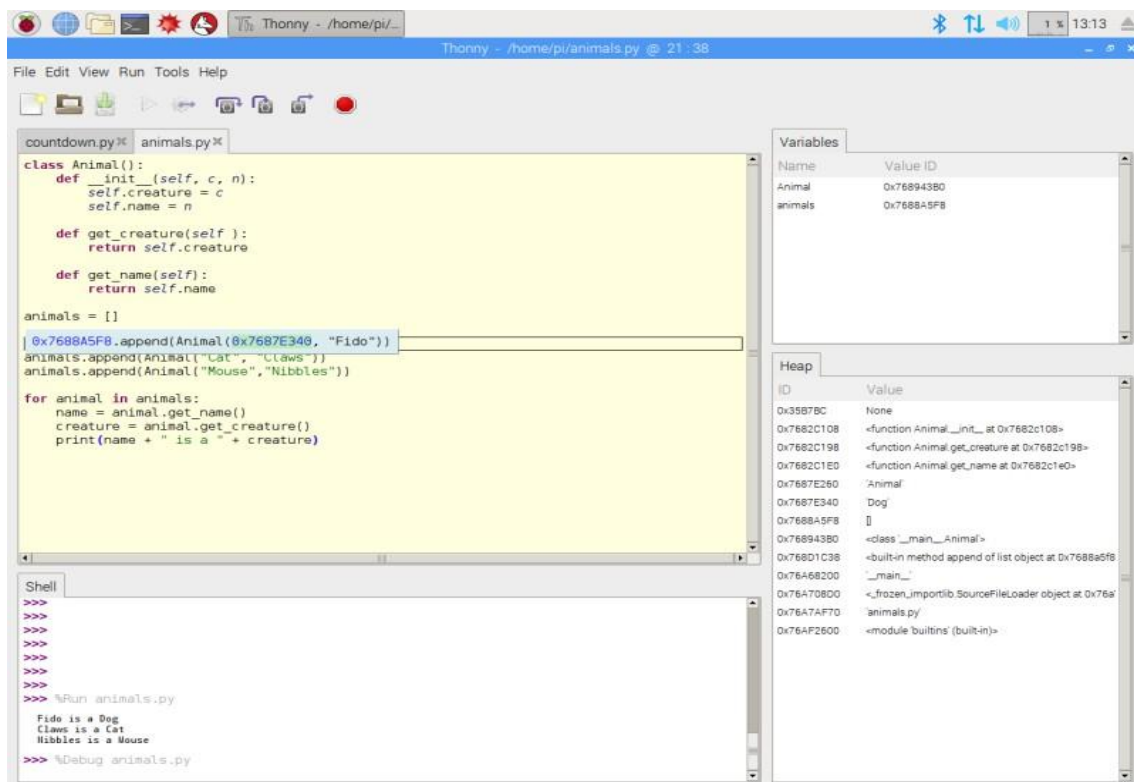
- 1) Raspberry pi 4
- 2) A camera module
- 3) Smart phone
- 4) Gsm module
- 5) Wifi network
- 6) 12v Solenoid Door lock



**Fig4.1: The project files in thonny.**

Each project in Raspberry pi os contains one or more modules with source code files and resource file

The user interface The Thonny main window is made up of several logical areas identified. .



**Fig 4.2: The Raspberry pi Thonny main window.**

1. The **toolbar** lets you carry out a wide range of actions, including running your app and launching Android tools.
2. The **navigation bar** helps you navigate through your project and open files for editing. It provides a more compact view of the structure visible in the **Project** window.
3. The **editor window** is where you create and modify code. Depending on the current file type, the editor can change. For example, when viewing a layout file, the editor displays the Layout Editor.
4. The **tool window bar** runs around the outside of the IDE window and contains the buttons that allow you to expand or collapse individual tool windows.
5. The **tool windows** give you access to specific tasks like project management, search, version control, and more. You can expand them and collapse them.
6. The **status bar** displays the status of your project and the IDE itself, as well as any warnings or messages.

You can organize the main window to give yourself more screen space by hiding or moving toolbars and tool windows. You can also use keyboard shortcuts to access most IDE features.

At any time, you can search across your source code, databases, actions, elements of the user interface, and so on, by double-pressing the Shift key, or clicking the magnifying glass in the upper right-hand corner of the Android Studio window. This can be very useful if, for example, you are trying to locate a particular IDE action that you have forgotten how to trigger.

## 4.2 Tool windows

Instead of using preset perspectives, Android Studio follows your context and automatically brings up relevant tool windows as you work. By default, the most commonly used tool windows are pinned to the tool window bar at the edges of the application window.

- To expand or collapse a tool window, click the tool's name in the tool window bar. You can also drag, pin, unpin, attach, and detach tool windows.
- To return to the current default tool window layout, click **Window > Restore Default Layout** or customize your default layout by clicking **Window > Store Current Layout as Default**.
- To show or hide the entire tool window bar, click the window icon  in the bottom left-hand corner of the Android Studio window.
- To locate a specific tool window, hover over the window icon and select the tool window from the menu.

You can also use keyboard shortcuts to open tool windows.

## 4.3 Navigation

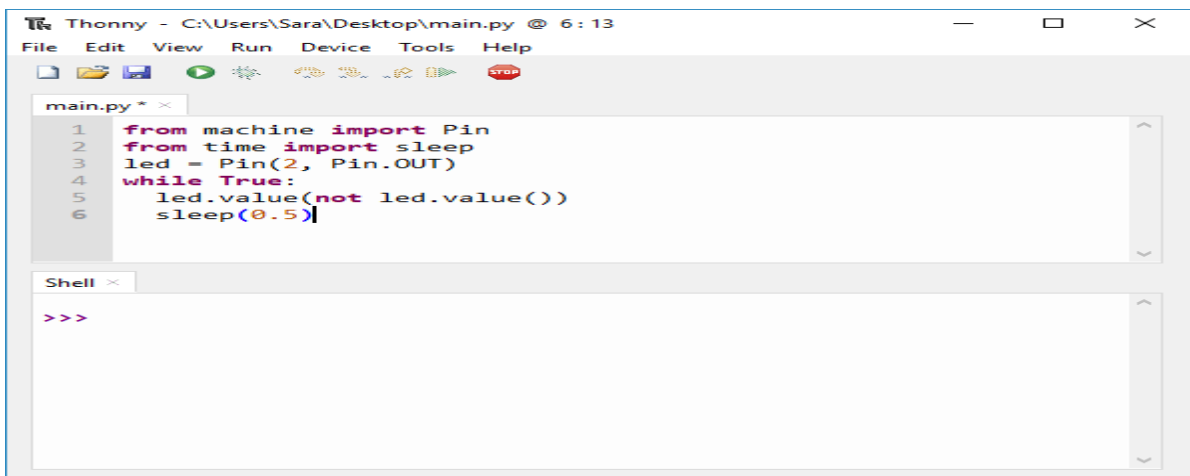
Here are some tips to help you move around Android Studio.

- Switch between your recently accessed files using the *Recent Files* action. Press **Control+E** (**Command+E** on a Mac) to bring up the Recent Files action. By default, the last accessed file is selected. You can also access any tool window through the left column in this action.
- View the structure of the current file using the *File Structure* action. Bring up the File Structure action by pressing **Control+F12** (**Command+F12** on a Mac). Using this action, you can quickly navigate to any part of your current file.

- Search for and navigate to a specific class in your project using the *Navigate to Class* action. Bring up the action by pressing **Control+N** (**Command+O** on a Mac).
- Navigate to a file or folder using the *Navigate to File* action. Bring up the *Navigate to File* action by pressing **Control+Shift+N** (**Command+Shift+O** on a Mac). To search for folders rather than files, add a / at the end of your expression.
- Find all the pieces of code referencing the class, method, field, parameter, or statement at the current cursor position by pressing **Alt+F7** (**Option+F7** on a Mac).

#### 4.4 Style and formatting

As you edit, Android Studio automatically applies formatting and styles as specified in your code style settings. You can customize the code style settings by programming language, including specifying conventions for tabs and indents, spaces, wrapping and braces, and blank lines. To customize your code style settings, Although the IDE automatically applies formatting as you work, you can also explicitly all the *ReformatCode* action by pressing **Control+Alt+L**(**Opt+Command+L** on a Mac), or auto-indent all lines by pressing **Control+Alt+I** (**Control+Option+I** on a Mac).

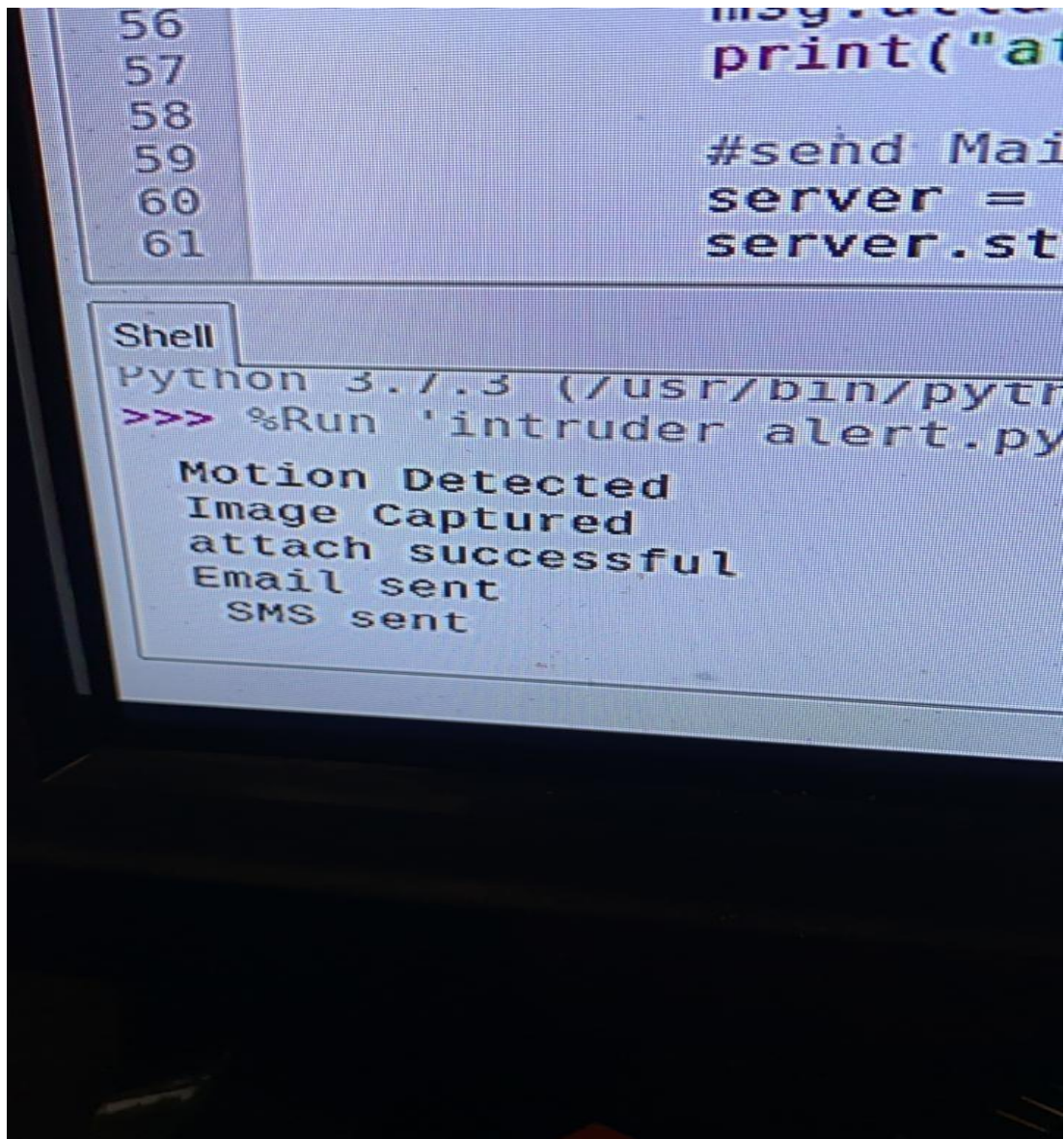


The screenshot shows the Thonny IDE interface. The title bar reads 'Thonny - C:\Users\Sara\Desktop\main.py @ 6:13'. The menu bar includes 'File', 'Edit', 'View', 'Run', 'Device', 'Tools', and 'Help'. Below the menu bar is a toolbar with icons for file operations, running, and debugging. The main editor window displays a Python script in 'main.py' with the following code:

```
1 from machine import Pin
2 from time import sleep
3 led = Pin(2, Pin.OUT)
4 while True:
5     led.value(not led.value())
6     sleep(0.5)
```

Below the editor is a 'Shell' window with the prompt '>>>'.

**Fig4.4:Code before formatting.**



*Fig 4.5: Code after formatting.*

#### 4.5 Limitations:

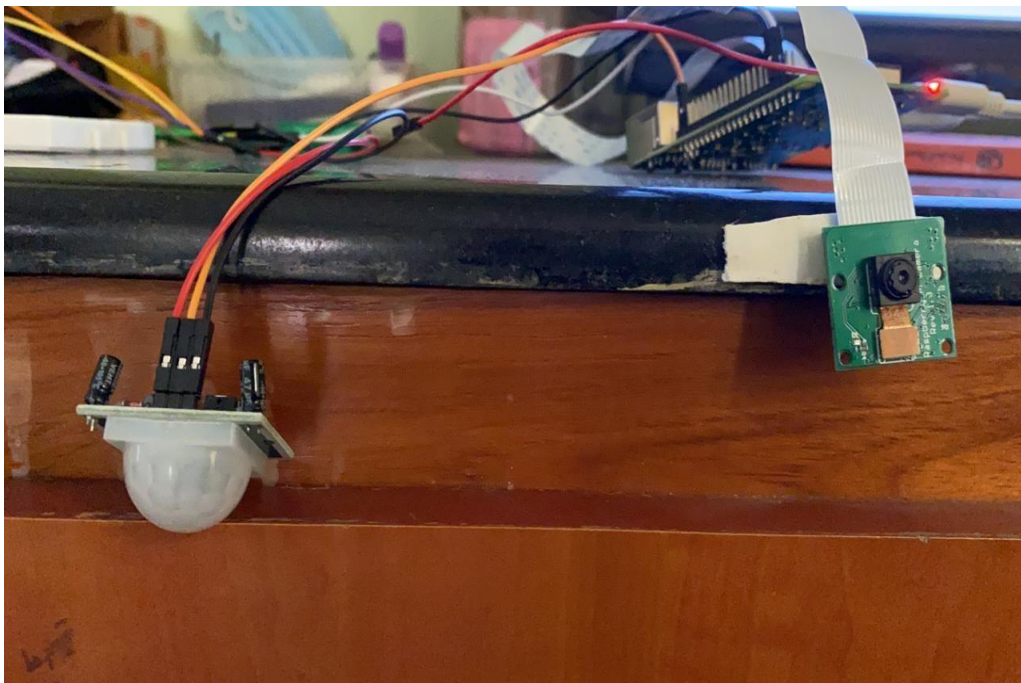
- It is working with an ARM architecture, not like “normal” computers (x86). Some libraries that are working fine on your laptop may give you some surprises when you try to use them on your RPI . Because some dependencies are not ARM-compatibles (or don't exist at all for ARM).
- The GPIOs are an excellent feature, but it doesn't have all the hardware benefits of, let's say, an Arduino. You don't have analog I/O, PWM is not really supported.
- Although you hear a lot about open source around Raspberry Pi, know that the PCB is not open source at all. You just can't get the source file.
- Take care about what you plug on your GPIOs, you can easily burn the RPI (this one is not really a limitation, but I thought it could be useful to remind that)
- The RAM is not expandable. You might have to optimize what programs you launch so you don't use too much.

## CHAPTER 5

### RESULTS AND DISCUSSION

#### 5.1 CONNECTING MODULES TO RASPBERRY PI KIT

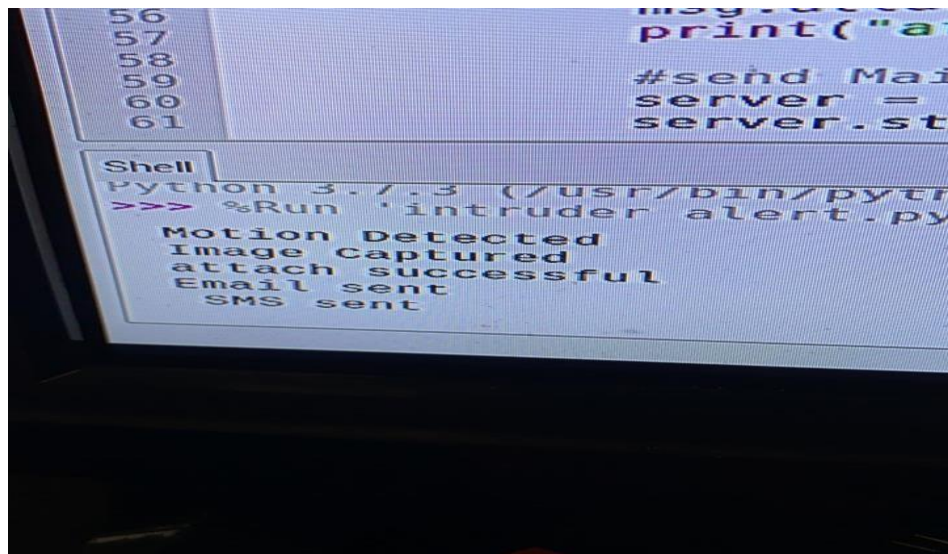
The Raspberry Pi runs Linux system type operating systems further as there's a master version of Linux based mostly kernel accepted as Raspbian which may run nearly all programs that area unit Linux compatible. Raspbian could be a free OS supported Debian optimized for Raspberry Pi hardware. associate OS is that the set of basic programs and utilizes that build raspberry pi run. The Raspberry pi could be a considerably additional powerful device however that power arises with some responsibilities that won't necessities for Arduino devices. For that reason during this project we've created a use of 'python' script for motion detection.



*Fig 5.1: Connecting sensors to Raspberry pi*

#### 5.2 DETECTING THE MOTION THROUGH PIR MOTION SENSOR

The motion device is connected with the raspberry pi kit that is employed to notice the motion of some objects or persons is moving infront of the door and it send request to the camera device that captures the pictures and send that as input to the raspberry pi kit for doing these operations we tend to exploitation the python.

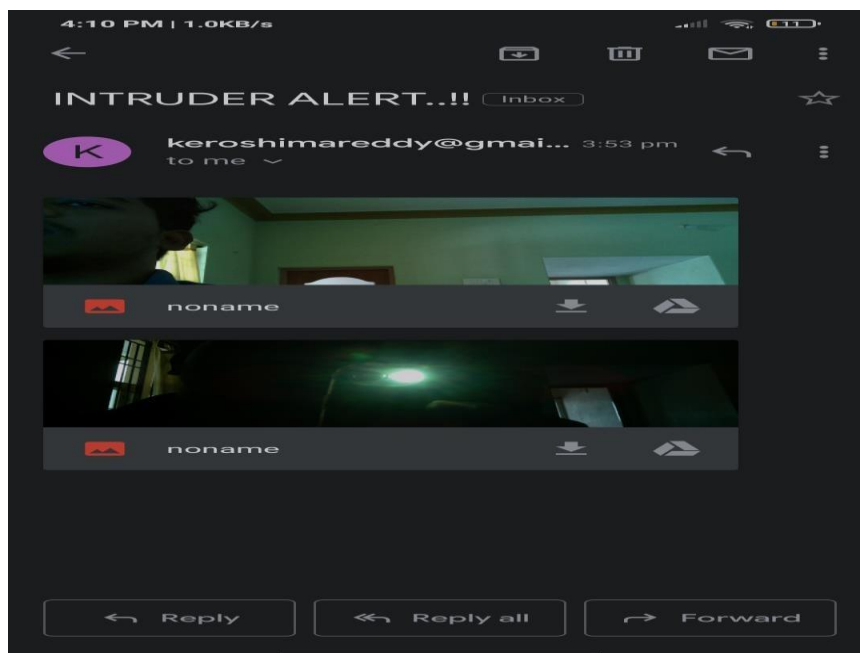


***Fig.5.2: Motion and Images are captured***

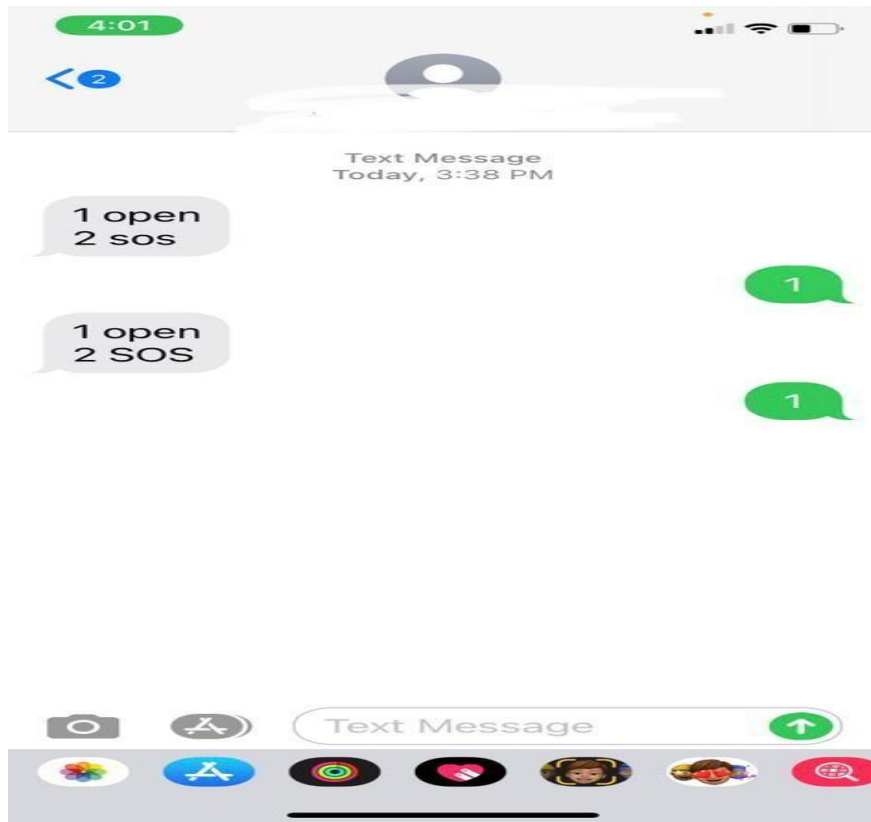
The motion is detected if any things move infront of the sensors and the image was captured and send it through SMS and Email.

### **5.3 SENDING THE MAIL AND SMS TO MOBILE THROUGH GSM**

After capturing the image the raspberry pi sends the image to the connected movable by exploitation GSM electronic equipment the most practicality of electronic equipment is it's connected wireless and acts as wi-fi to transferring the info through wireless communication.



***Fig 5.3: Images sent to mail***



***Fig.5.4: Action take through commands***

We can see the images of the persons who are near the door in Email and take the action through the SMS commands.

#### **5.4 OPENING THE DOOR LOCK THROUGH COMMANDS**

By entering 1 command the door lock is open. The communication is happening through GSM module.



***Fig.5.5: The door lock opens***

## **CHAPTER 6**

### **CONCLUSION AND SUMMARY**

#### **6.1 CONCLUSION**

Digital door locks are user-friendly. These are developed to deliver higher security. This technique makes the system more secure and it is utilized in various places, where there is in high demand for security. Example: analysis institutes, banks, theoretical Laboratories. This technique can even be used for domestic functions. This project helps to prevent all the problems that are connected with thefts, an unauthorized person's entry, and etc. whenever, an unauthorized attempt took place it alerts the owner by sending an SMS. The camera device snaps the photo and sends the picture to the raspberry pi kit as an input, the raspberry is connected with the GSM module that sends the capturing to mobile devices or any social media that it is connected with the device.

#### **6.2 SUMMARY**

The Smart Lock System could run properly and stably. It could implement all the designed and needed features and functions. This could be achieved only by proper software and hardware development, as well as multiple test and debug procedures. The most commonly used system for locking and unlocking the door is a lock and a physical key. The entire process is a mechanical one. The RFID card reader unit is installed near the door. When the card is brought near the reader, it identifies the radio frequency of the card and thus verifies the key.

## REFERENCES

- [1] S. Hong, D. Kim, M. Ha, S. Bae, S. Park, W. Jung, and J. E. Kim, "SNAIL: AN IP-based wireless device network approach to the web of things," *IEEE Wireless Commun.*, vol. 17, no. 6, pp. 34–42, Dec. 2010.
- [2] H. Gharavi and R. Ghafurian, "Smart grid: the electrical energy system of the future", *Proceedings of the IEEE*, 2011, Vol. 99, No. 6, 917 – 921
- [3] D. Surie, O. Laguionie, and T. Pederson, "Wireless device networking of everyday objects during a sensible home setting," in *Proc. Int. Conf. Intell. Sensors, device Netw. Inf. Process.*, 2008, pp. 189–194. 0347
- [4] N. Bui, A. P. Castellani, P. Casari, and M. Zorzi, "The net of energy: A web-enabled sensible grid system," *IEEE Netw.*, vol. 26, no. 4, pp. 39–45, Jul.–Aug. 2012.
- [5] Iera, C. Floerkemeier, J. Mitsugi, and G. Morabito, "The net of things," *IEEE Wireless Commun.*, vol. 17, no. 6, pp. 8–9, Dec. 2010.
- [6] Gluhak, S. Krco, M. Nati, D. Pfisterer, N. Mitton, T. Razafindr alambo, "A survey on facilities for experimental net of things analysis," *IEEE Commun. Mag.*, vol. 49, no. 11, pp. 58–67, Nov. 2011.
- [7] M. Zorzi, A. Gluhak, S. Lange, and A. Bassi, "From today's computer network of things to a future net of things: A wireless- and mobilityrelated read," *IEEE Wireless Commun.*, vol. 17, no. 6, pp. 44– 51, Dec. 2010.
- [8] Pavithra.D, RanjithBalakrishnan "IoT primarily based watching and system for home automation", *Proceedings of 2015 world Conference on Communication Technologies (GCCT 2015)*
- [9] Vishwajeet H. Bhide, "A survey on the sensible homes victimization net of Things (IoT)", *International journal of advance analysis in computing and management studies* volume two, issue 12, Gregorian calendar month 2014.
- [10] M. Mahadi Abdul Jamil, M. ShukriAhmad, "A pilot study: Development of home automation system via Raspberry Pi" *2015 2nd International Conference on Biomedical Engineering (ICoBE)*, 30-31 March 2015.

## APPENDIX

### A.SOURCE CODE

#### PYTHON CODE

```
import RPi.GPIO as gpio
```

```
import serial
```

```
import time
```

```
msg=""
```

```
# 0 7 11 15 19 23 27 32 36 41 42 44 ROLL45
```

```
alpha="1!@.,:~?ABC2DEF3GHI4JKL5MNO6PQRS7TUV8WXYZ90 *#"
```

```
x=0
```

```
y=0
```

```
MATRIX = [
```

```
    ['1','2','3','A'],
```

```
        ['4','5','6','B'],
```

```
        ['7','8','9','C'],
```

```
        ['*','0','#','D']
```

```
    ]
```

```
    ROW = [21,20,16,12]
```

```
    COL = [26,19,13,6]
```

```
moNum=['0','0','0','0','0','0','0','0','0','0']
```

```
m11=17
```

```
m12=27
```

```
led=5
```

```
buz=26
```

```
button=19
```

```
RS =18
```

```
EN =23
```

```
D4 =24
```

```
D5 =25
```

```
D6 =8
```

D7 =7

HIGH=1

LOW=0

```
gpio.setwarnings(False)
gpio.setmode(gpio.BCM)
gpio.setup(RS, gpio.OUT)
gpio.setup(EN, gpio.OUT)
gpio.setup(D4, gpio.OUT)
gpio.setup(D5, gpio.OUT)
gpio.setup(D6, gpio.OUT)
gpio.setup(D7, gpio.OUT)
gpio.setup(led, gpio.OUT)
gpio.setup(buz, gpio.OUT)
gpio.setup(m11, gpio.OUT)
gpio.setup(m12, gpio.OUT)
gpio.setup(button, gpio.IN)
gpio.output(led , 0)
gpio.output(buz , 0)
gpio.output(m11 , 0)
gpio.output(m12 , 0)
for j in range(4):
    gpio.setup(COL[j], gpio.OUT)
    gpio.setup(COL[j],1)

for i in range (4):
    gpio.setup(ROW[i],gpio.IN,pull_up_down=gpio.PUD_UP)
Serial = serial.Serial("/dev/ttyS0", baudrate=9600, timeout=2)
data=""

def begin():
    lcdcmd(0x33)
    lcdcmd(0x32)
    lcdcmd(0x06)
    lcdcmd(0x0C)
    lcdcmd(0x28)
    lcdcmd(0x01)
```

```
time.sleep(0.0005)
```

```
def lcdcmd(ch):  
    gpio.output(RS, 0)  
    gpio.output(D4, 0)  
    gpio.output(D5, 0)  
    gpio.output(D6, 0)  
    gpio.output(D7, 0)  
    if ch&0x10==0x10:  
        gpio.output(D4, 1)  
    if ch&0x20==0x20:  
        gpio.output(D5, 1)  
    if ch&0x40==0x40:  
        gpio.output(D6, 1)  
    if ch&0x80==0x80:  
        gpio.output(D7, 1)  
    gpio.output(EN, 1)  
    time.sleep(0.005)  
    gpio.output(EN, 0)
```

```
#    Low    bits  
gpio.output(D4, 0)  
gpio.output(D5, 0)  
gpio.output(D6, 0)  
gpio.output(D7, 0)  
if ch&0x01==0x01:  
    gpio.output(D4, 1)  
if ch&0x02==0x02:  
    gpio.output(D5, 1)  
if ch&0x04==0x04:  
    gpio.output(D6, 1)  
if ch&0x08==0x08:  
    gpio.output(D7, 1)  
gpio.output(EN, 1)  
time.sleep(0.005)  
gpio.output(EN, 0)
```

```
def lcdwrite(ch):
```

```

gpio.output(RS, 1)
gpio.output(D4, 0)
gpio.output(D5, 0)
gpio.output(D6, 0)
gpio.output(D7, 0)
if ch&0x10==0x10:
gpio.output(D4, 1)
if ch&0x20==0x20:
gpio.output(D5, 1)
if ch&0x40==0x40:
gpio.output(D6, 1)
if ch&0x80==0x80:
gpio.output(D7, 1)
gpio.output(EN, 1)
time.sleep(0.005)
gpio.output(EN, 0)

```

```

# Low bits
gpio.output(D4, 0)
gpio.output(D5, 0)
gpio.output(D6, 0)
gpio.output(D7, 0)
if ch&0x01==0x01:
gpio.output(D4, 1)
if ch&0x02==0x02:
gpio.output(D5, 1)
if ch&0x04==0x04:
gpio.output(D6, 1)
if ch&0x08==0x08:
gpio.output(D7, 1)
gpio.output(EN, 1)
time.sleep(0.005)
gpio.output(EN, 0)

```

```

def lcdclear():
    lcdcmd(0x01)

```

```

def lcdprint(Str):

```

```

l=0;
l=len(Str)
for i in range(l):
    lcdwrite(ord(Str[i]))
def setCursor(x,y):
    if y == 0:
        n=128+x
    elif y == 1:
        n=192+x
    lcdcmd(n)

def keypad():
    for j in range(4):
        gpio.setup(COL[j], gpio.OUT)
        gpio.output(COL[j], 0)
        ch=0
        for i in range(4):
            if gpio.input(ROW[i])==0:
                ch=MATRIX[i][j]
                #lcdwrite(ord(ch))
                # print "Key Pressed:",ch
                # time.sleep(2)
                return ch
            while (gpio.input(ROW[i]) == 0):
                pass
        gpio.output(COL[j],1)
        # callNum[n]=ch

def serialEvent():
    data = Serial.read(20)
    #if data != '\0':
    print data
    data=""

def gsmlInit():
    lcdclear()
    lcdprint("Finding Module");
    time.sleep(1)

```

```

while 1:
    data=""
    Serial.write("AT\r");
    data=Serial.read(10)
    print data
    r=data.find("OK")
    if r>=0:
        break
    time.sleep(0.5)

```

```

while 1:
    data=""
    Serial.write("AT+CLIP=1\r");
    data=Serial.read(10)
    print data
    r=data.find("OK")
    if r>=0:
        break
    time.sleep(0.5)

```

```

lcdclear()
lcdprint("Finding Network")
time.sleep(1)
while 1:
    data=""
    Serial.flush()
    Serial.write("AT+CPIN?\r");
    data=Serial.read(30)
    print data
    r=data.find("READY")
    if r>=0:
        break
    time.sleep(0.5)

```

```

lcdclear()
lcdprint("Finding Operator")
time.sleep(1)
while 1:

```

```

data=""
Serial.flush()
Serial.read(20)
Serial.write("AT+COPS?\r");
data=Serial.read(40)
#print data
r=data.find("+COPS:")
if r>=0:
    l1=data.find(",\r")+2
    l2=data.find("\r")
    operator=data[l1:l2]
    lcdclear()
    lcdprint(operator)
    time.sleep(3)
    print operator
    break;
    time.sleep(0.5)
Serial.write("AT+CMGF=1\r");
time.sleep(0.5)
# Serial.write("AT+CNMI=2,2,0,0,0\r");
# time.sleep(0.5)
Serial.write("AT+CSMP=17,167,0,0\r");
time.sleep(0.5)

def receiveCall(data):
    inNumber=""
    r=data.find("+CLIP:")
    if r>0:
        inNumber=""
        inNumber=data[r+8:r+21]
        lcdclear()
        lcdprint("incoming")
        setCursor(0,1)
        lcdprint(inNumber)
        time.sleep(1)
        return 1

def receiveSMS(data):

```

```

print data
r=data.find("\",")
print r

if r>0:
    if data[r+4] == "\r":
        smsNum=data[r+2:r+4]
    elif data[r+3] == "\r":
        smsNum=data[r+2]
    elif data[r+5] == "\r":
        smsNum=data[r+2:r+5]
    else:
        print "else"
    print smsNum
    if r>0:

        lcdclear()
        lcdprint("SMS Received")
        setCursor(0,1)
        lcdprint("Press Button B")
        print "AT+CMGR="+smsNum+"\r"
        time.sleep(2)
        return str(smsNum)
    else:
        return 0

def attendCall():
    print "Attend call"
    Serial.write("ATA\r")
    data=""
    data=Serial.read(10)
    l=data.find("OK")
    if l>=0:
        lcdclear()
        lcdprint("Call attended")
        time.sleep(2)
        flag=-1;
        while flag<0:

```

```

        data=Serial.read(12);
        print data
        flag=data.find("NO CARRIER")
        #flag=data.find("BUSY")
        print flag
        lcdclear()
        lcdprint("Call Ended")
        time.sleep(1)
        lcdclear()

def readSMS(index):
    print index
    Serial.write("AT+CMGR="+index+"\r")
    data=""
    data=Serial.read(200)
    print data
    r=data.find("OK")
    if r>=0:
        r1=data.find("\r\n")
        msg=""
        msg=data[r1+3:r-4]
        lcdclear()
        lcdprint(msg)
        print msg
        time.sleep(5)
        lcdclear();
        smsFlag=0
        print "Receive SMS"

def getChar(Key, ind, maxInd):
    ch=0
    ch=ind
    lcdcmd(0x0e)
    Char=""
    count=0
    global msg
    global x
    global y

```

```

while count<20:
    key=keypad()
    print key
    if key== Key:
        setCursor(x,y)
        Char=alpha[ch]
        lcdwrite(ord(Char))
        ch=ch+1
        if ch>maxInd:
            ch=ind
        count=0
        count=count+1
        time.sleep(0.1)
    msg+=Char
    x=x+1
    if x>15:
        x=0
        y=1
        lcdcmd(0x0f)

def alphaKeypad():
    lcdclear()
    setCursor(x,y)
    lcdcmd(0x0f)
    msg=""
    while 1:
        key=0
        count=0
        key=keypad()
        if key == '1':
            ind=0
            maxInd=6
            Key='1'
            getChar(Key, ind, maxInd)

        elif key == '2':
            ind=7
            maxInd=10

```

```
    Key='2'
    getChar(Key, ind, maxInd)

elif key == '3':
    ind=11
    maxInd=14
    Key='3'
    getChar(Key, ind, maxInd)

elif key == '4':
    ind=15
    maxInd=18
    Key='4'
    getChar(Key, ind, maxInd)

elif key == '5':
    ind=19
    maxInd=22
    Key='5'
    getChar(Key, ind, maxInd)

elif key == '6':
    ind=23
    maxInd=26
    Key='6'
    getChar(Key, ind, maxInd)

elif key == '7':
    ind=27
    maxInd=31
    Key='7'
    getChar(Key, ind, maxInd)

elif key == '8':
    ind=32
    maxInd=35
    Key='8'
    getChar(Key, ind, maxInd)
```

```

elif key == '9':
    ind=36
    maxInd=40
    Key='9'
    getChar(Key, ind, maxInd)

elif key == '0':
    ind=41
    maxInd=42
    Key='0'
    getChar(Key, ind, maxInd)

elif key == '*':
    ind=43
    maxInd=43
    Key='*'
    getChar(Key, ind, maxInd)

elif key == '#':
    ind=44
    maxInd=44
    Key='#'
    getChar(Key, ind, maxInd)

elif key == 'D':
    return

def sendSMS():
    print"Sending sms"
    lcdclear()
    lcdprint("Enter Number:")
    setCursor(0,1)
    time.sleep(2)
    moNum=""
    while 1:
        key=0;
        key=keypad()

```

```

#print key
if key>0:
    if key == 'A' or key== 'B' or key== 'C':
        print key
        return
    elif key == 'D':
        print key
        print moNum
        Serial.write("AT+CMGF=1\r")
        time.sleep(1)
        Serial.write("AT+CMGS=\""+91"+moNum+"\"")
        time.sleep(2)
        data=""
        data=Serial.read(60)
        print data
        alphaKeypad()
        print msg
        lcdclear()
        lcdprint("Sending ....")
        Serial.write(msg)
        time.sleep(1)
        Serial.write("\x1A")
        while 1:
            data=""
            data=Serial.read(40)
            print data
            l=data.find("+CMGS:")
            if l>=0:
                lcdclear()
                lcdprint("SMS Sent.")
                time.sleep(2)
                return;

            l=data.find("Error")
            if l>=0:
                lcdclear()
                lcdprint("Error")
                time.sleep(1)

```

```

        return
    else:
        print key
        moNum+=key
        lcdwrite(ord(key))
        time.sleep(0.5)

def call():
    print "Call"
    n=0
    moNum=""
    lcdclear()
    lcdprint("Enter Number:")
    setCursor(0,1)
    time.sleep(2)
    while 1:
        key=0;
        key=keypad()
        #print key
        if key>0:
            if key == 'A' or key== 'B' or key== 'D':
                print key
                return
            elif key == 'C':
                print key
                print moNum
                Serial.write("ATD+91"+moNum+"\r")
                data=""
                time.sleep(2)
                data=Serial.read(30)
                l=data.find("OK")
                if l>=0:
                    lcdclear()
                    lcdprint("Calling ....")
                    setCursor(0,1)
                    lcdprint("+91"+moNum)
                    time.sleep(30)
                    lcdclear()

```

```

        return
    #l=data.find("Error")
    #if l>=0:
    else:
        lcdclear()
        lcdprint("Error")
        time.sleep(1)
        return
    else:
        print key
        moNum+=key
        lcdwrite(ord(key))
        n=n+1
        time.sleep(0.5)

begin()
lcdcmd(0x01)
lcdprint(" Mobile Phone ")
lcdcmd(0xc0)
lcdprint(" Using RPI ")
time.sleep(3)
lcdcmd(0x01)
lcdprint("Circuit Digest")
lcdcmd(0xc0)
lcdprint("Welcomes you")
time.sleep(3)
gsmInit()
smsFlag=0
index=""

while 1:

    key=0
    key=keypad()
    print key
    if key == 'A':
        attendCall()
    elif key == 'B':

```

```

        readSMS(index)
        smsFlag=0
    elif key == 'C':
        call()
    elif key == 'D':
        sendSMS()

data=""
Serial.flush()
data=Serial.read(150)
print data
l=data.find("RING")
if l>=0:
    callstr=data
    receiveCall(data)
    l=data.find("\\"SM\\")
    if l>=0:
        smsstr=data
        smsIndex=""
        (smsIndex)=receiveSMS(smsstr)
        print smsIndex
        if smsIndex>0:
            smsFlag=1
            index=smsIndex
if smsFlag == 1:
    lcdclear()
    lcdprint("New Message")
    time.sleep(1)

setCursor(0,0)
lcdprint("C--> Call <--A");
setCursor(0,1);
lcdprint("D--> SMS <--B")

}

```

```

from gpiozero import MotionSensor, Button, LED
from picamera import PiCamera

```

```

from email.mime.image import MIMEImage
from email.mime.multipart import MIMEMultipart
import datetime
import smtplib
import time
from time import sleep

pir = MotionSensor(20)
camera = PiCamera()
led = LED(17)
button = Button(16)

#replace the next three lines with your credentials
from_email_addr = 'Sender_Email@gmail.com'
from_email_password = 'Sender_Password'
to_email_addr = 'Receiver_Email@gmail.com'

#Create Alarm State by default it off.
Alarm_state = False

while True:

    if button.is_pressed:
        Alarm_state = True
        print('Alarm ON')

    if Alarm_state == True:
        led.on()
        sleep(1)
        if pir.motion_detected:
            print("Motion Detected")
            led.off()

            #capture images
            Captured1 = '/home/pi/Desktop/image' +
datetime.datetime.now().strftime('%Y-%m-
%d%H:%M:%S') + '.png'
            camera.rotation = 180
            camera.capture(Captured1)

            sleep(1)
            Captured2 = '/home/pi/Desktop/image' +
datetime.datetime.now().strftime('%Y-%m-
%d%H:%M:%S') + '.png'
            camera.rotation = 180
            camera.capture(Captured2)
            print("Image Captured")

```

```

#Message
msg = MIMEMultipart()
msg['Subject'] = 'INTRUDER ALERT...!!'
msg['From'] = from_email_addr
msg['To'] = to_email_addr

#Attach the files
File = open(Captured1, 'rb')
img1 = MIMEImage(File.read())
File.close()
msg.attach(img1)
File = open(Captured2, 'rb')
img2 = MIMEImage(File.read())
File.close()
msg.attach(img2)
print("attach successful")

#send Mail
server = smtplib.SMTP('smtp.gmail.com',
587)
server.starttls()
server.login(from_email_addr,
from_email_password)
server.sendmail(from_email_addr,
to_email_addr, msg.as_string())
server.quit()
print('Email sent')
Alarm_state = False

```