**CHAPTER-1**

**INTRODUCTION**

In this technological era the world is changing dynamically with a call for modernization. In the technological field, this modernization may lead to increase in crime followed. As a result, Indians became sensitive to security needs and were eager to find ways to protect themselves and their property. Additionally, many insurance companies began offering premium discounts to alarm subscribers. These events produced a consumer demand for alarm systems. Fast forward to today, homeowners have traded indoor shakers for [automated, sophisticated systems](http://www.alarm.org/HomeSafety/CreatingaSmartHome.aspx). Now, to ensure that doors are locked and the alarm is set, users simply log in remotely via a web-enabled device and check the status of their homes. They can lock doors, arm the system, or adjust the thermostat with the touch of a button. In the past all measures seemed to be ineffective therefore modern day technology is used to curb this menace.

In this project the main objective of security is for restricting the entrance of unknown or unauthorized person to the restricted –zone and providing access to authorized person. Security is being employed by various societies, colonies as well as individual houses to safeguard the materials of their interests.

The data are made available in the Command center, irrelevant of its source of origin. The command center in return sends this data to an authorized person. To access the data or information at any corner of the world, various wireless technologies have emerged and are used widely. Data is sent from one time-periodically to another either through a wired medium or wireless medium. However, the latter is more used comparatively. To access data through wireless medium in various protocols can be designed. Besides, transmission and reception of data, though wireless channel for automated systems can also be achieved with the help of readily available wireless medium using IoT (Internet of Things).

* 1. **Motivation**

The motivation is to prevent unwanted circumstances/conditions that occur during the absence of an individual such as intrusion with a motive of burglary, the loss incurred can be prevented using an efficient home security system.

* 1. **Problem statement**
* The basic problem to be solved is to implement an algorithm for detection of face in an image. Facial recognition, used in combination with another biometric method can improve verification and identification results dramatically.
* Interfacing RFIDreader which can read RFID tags to Node MCUwhen an RFID tag is shown near an RFID reader, it collects the unique tag(a combination of digits and character)from the RFID tag.
* Interfacing USB camera to Raspberry pi module, so as to detect face in captured video frame and recognize face in real time through efficient face recognition approach.
  1. **Objective**
* To design a home security system
* To implement real time, face recognition
* Obtain a wireless alert system for the house
* To build a data display to the user
* Obtain an exclusive secure access using RFID

**1.4 Scope**

This project result is a home security system which has two major applications adumbrated below:

1. The house security system can be implemented in each and every individual house premises where there is a possibility of theft (alarm situation) as it is very user friendly and economical. In this project, the house is continuously checked by the wireless control unit which sends proper alert messages in the case of emergency.
2. Real time face recognition has many applications and various improvements are being implemented. This technology can be used for monitoring the people entering a particular place and storing the images of the people.

**1.5 Organization of the project**

This report is organized in to five chapters. The first chapter gives an in-depth introduction of the project. The second chapter discusses about the theoretical background of the task and all concepts which inspires to take up the project. The third chapter is the heart of the project which gives the details about the design, and implementation of the security system. The fourth chapter deals with information about the results and the fifth chapter gives information about the future scope of the project.

**1.6 Summary**

In this chapter we gave introduction about crimes arising due to modernization. To overcome from this, people became more sensitive towards secure their property and their home.we discussed about the basic problem is to be solved followed by problem statement.

Then objective of our project is listed and two major applications for home security system were discussed in scope.

**CHAPTER-2**

**THEORITICAL BACKGROUND**

**2.1 Hardware description**

This chapter discusses about the basic functioning of the units that are employed in this project and the theoretical background associated with them. Various components required to develop this system are discussed here.

**2.2 Node MCU**

****

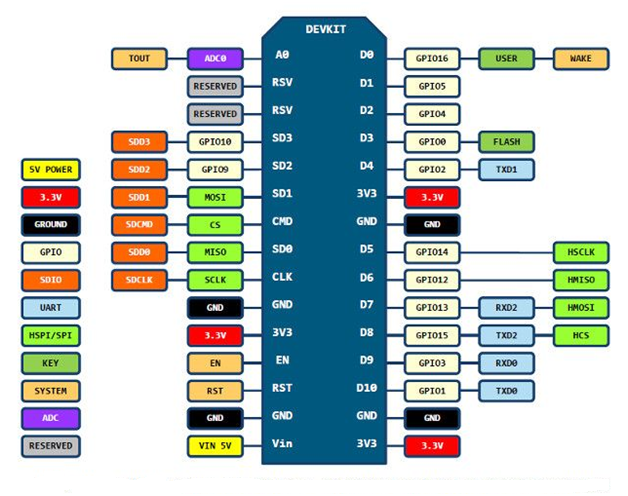
**Fig 2.2 Node MCU**

Node mcu is a firmware on ESP8266. Its basically an soc( system on chip).A system on a chip or system on chip(soc) is an integrated circuit that integrated all components of a computer or other electronic system.

**Features:**

1. Developer: ESP8266 Open source community
2. Type: single-board microcontroller
3. Operating system:XTOS
4. CPU:ESP8266(LX106)
5. Memory 128kbytes
6. Storage :4MBytes
7. Power: USB

**Pin diagram**

****

**Fig 2.2.1 Node MCU Pin diagram**

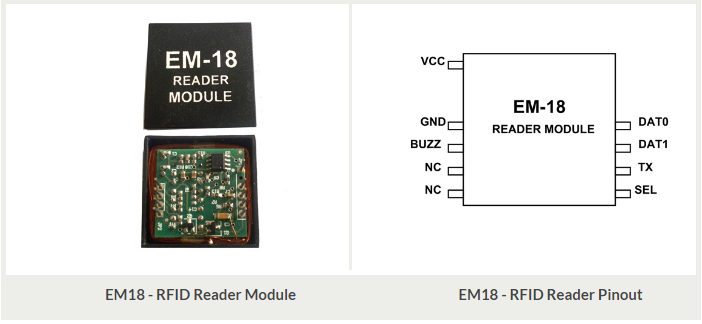
**2.3 RFID Reader EM-18**

****

**Fig 2.3 RFID reader EM-18**

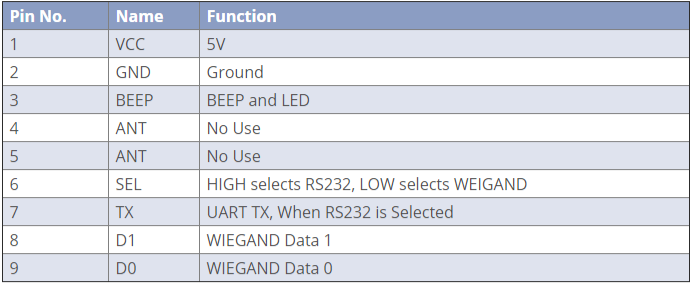
EM-18 RFID reader module uses an RFID reader that can read 125 KHz tags. So, it can be called as a low frequency RFID reader. It gives out a serial output and has a range of about 8-12 cm. There is a built-in **antenna** and it can be connected to the PC with help of **RS232**.

**EM18 RFID Reader** is a module which reads the ID information stored in RFID tag. This ID information is unique for every TAG which cannot be copied.

****

**Fig 2.3.1 EM-18 RFID Reader module and pinout**

**Table 2.3.2 : EM-18 Pin configuration**

****

**Features**

• Low cost.

• Low power.

• Form factor being small.

• Usability is easy.

• Direct interfacing with UART, PC is possible using RS232.

• Serial and TTL output.

• Excellent read performance without an external circuit.

• Two RFID cards.

• Cost-effective and compact size.

**Specifications**

• Read distance 10cm

• Current <50mA

• Operating frequency 125khz

• Parameter Value

• Operating Voltage 5v

**2.4 Buzzer**

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric(piezo for short).Typical uses of buzzer and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke.



**Fig 2.4 Buzzer**

**2.5 Servo motor**

****

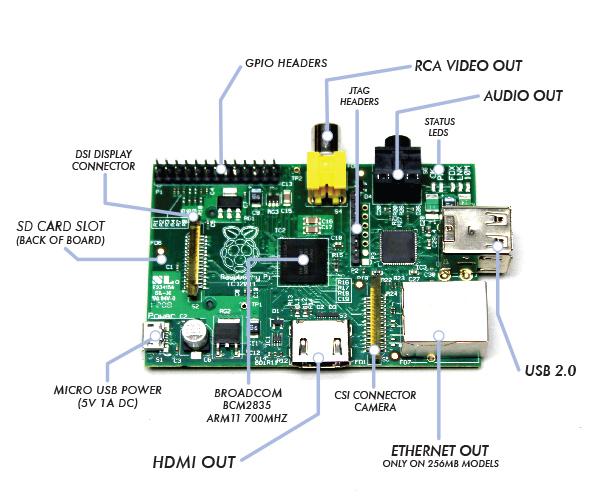
**Fig 2.5 Servo Motor**

A servomotor is a rotary actuator that allows for precise control of angular position, velocity and acceleration. It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servo motors.It is used to operate the door of the house

**Specification**

* Operating Voltage : 4.8~6.0V
* Operating Speed : 0.12sec/60 degree(4.8V)~0.1sec/60 degree(6.0V)
* Torque : 1.6kg/cm(4.8V)
* Dead Band Width : 5usec
* Temperature Range : -30~+60℃
* Cable Length : 25cm
* Servo Type : Analog Servo
* Brand Model : Tower Pro SG

**2.6 Raspberry pi**

****

**Fig 2.6 Raspberry pi**

Pi is a low-cost, basic computer that was originally intended to help spur interest in computing among school-aged children. The Raspberry Pi is contained on a single circuit board and features ports for:

 4 USB 2.0

 Wi-Fi module

 Analog audio

 Power

 Ethernet

 SD Card

1. CPU: This 700Mhz central Processor with ARM V6 architecture, is also known as SoC, System on Chip as the single CHIP is the CPU, 512MB system RAM and GPU, Graphic Processor Unit
2. Micro-USB power intake: This powers the RPi. Consider a power supply unit that can supply at least 1AMP. Dedicated PSU can be purchased from RS Components or Farrell‘s but mobile phone chargers mat work well. An iPhone charger has an output of 1AMP and an iPad 2.5 AMP. Both seems to work well on RPi.
3. HDMI: High definition Multimedia Interface: This can be the output to your TV or Monitor. If your monitor does not have HDMI input, then you may be able to obtain and HDMI to VGA or HDMI to DVI connection. These almost certainly will be cheapest purchased online rather than high street stores. The price different can be quite large.

Some monitors and TVs may have the composite video connection often used to connect gaming consoles to the TV.

1. 10/100 RJ45 Ethernet: For wired access to the network the 10/100 Ether-net connection exists. This is a single port that can work with 10Mbps or 100 Mbps connections.
2. USB Ports: Dual USB ports exist. This could be for a keyboard or mouse or connection through to a USB Hub. This would be a powered USB hub that then could drive external hard disks and the like.
3. Status LEDS: The pretty lights that flash when there is activity.
4. Audio Jack: If you are not using an HDMI to HDMI connection to your TV then audio can be used from this jack.
5. Composite Video: The yellow connect can be used to plug into some TVs and Monitors.
6. GPIO: The General Purpose Input / Output interface that can be used for simple electronics projects.
7. Wi-Fi: The board is accompanied with a Wi-Fi module to enable wireless access or communication to or from the control canter. This can also be achieved by using an Ethernet cable for localized movement.
8. Power: Raspberry Pi can be powered three different ways namely Micro USB port, SPIO connector and USB device.

**2.7 SD CARD**

****

**Fig2.7 SD Card**

The SD card is a key part of the Raspberry Pi; it provides the initial storage for the Operating System and files. Storage can be extended through many types of USB connected peripherals.

When the Raspberry Pi is 'switched on', i.e. connected to a power supply, a special piece of code called the bootloader is executed, which reads more special code from the SD card that is used to start up the Raspberry Pi. If there is no SD card inserted, it will not start. Do NOT push in or pull out an SD card while the Raspberry Pi is connected to the power, as this is likely to corrupt the SD card data.

The SD card must be formatted, or written to, in a special way that means the Raspberry Pi can read the data it needs to start properly. One advantage to using an SD card like this is that you can have several SD cards, each with a different operating system, or a different purpose. Simply power off, switch cards, and reconnect the power.

**SD Card pinout**



**Fig 2.7.1 SD Card pinout**

**2.8 USB Camera**



**Fig 2.8 USB Camera**

USB Cameras are imaging cameras that use USB 2.0 or USB 3.0 technology to transfer image data. USB Cameras are designed to easily interface with dedicated computer systems by using the same USB technology that is found on most computers. The accessibility of USB technology in computer systems as well as the 480 Mb/s transfer rate of USB 2.0 makes USB Cameras ideal for many imaging applications. An increasing selection of USB 3.0 Cameras is also available with data transfer rates of up to 5 Gb/s.

Edmund Optics offers a variety of USB Cameras suited to meet many imaging needs. EO USB Cameras are available in both CMOS as well as CCD sensor types making them suitable across a larger range of applications. USB Cameras contain out-of-the-box functionality for quick setup. USB Cameras using low power USB ports, such as on a laptop, may require a separate power supply for operation

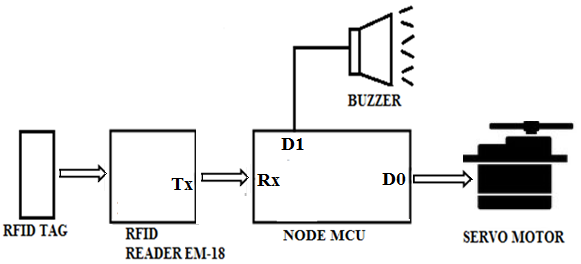
**2.9 Summary**

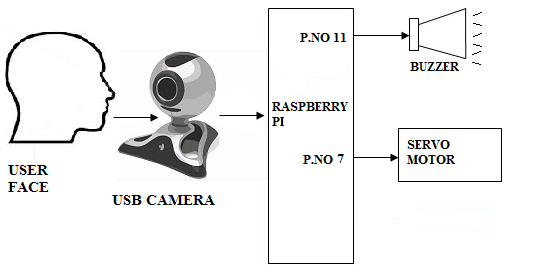
In this chapter we discussed about functional unit that are employed in the project and the theoretical background associated with them.Then discussed about various hardware components that are required to develope our system .

**CHAPTER-3**

**SYSTEM DESIGN AND IMPLEMENTATION**

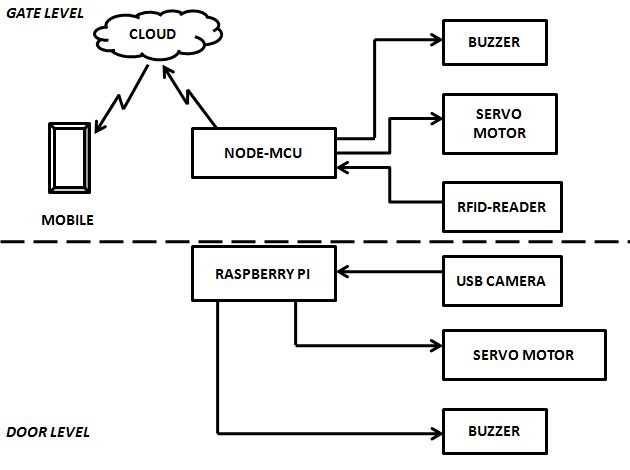
**3.1 Block diagram**

****

****

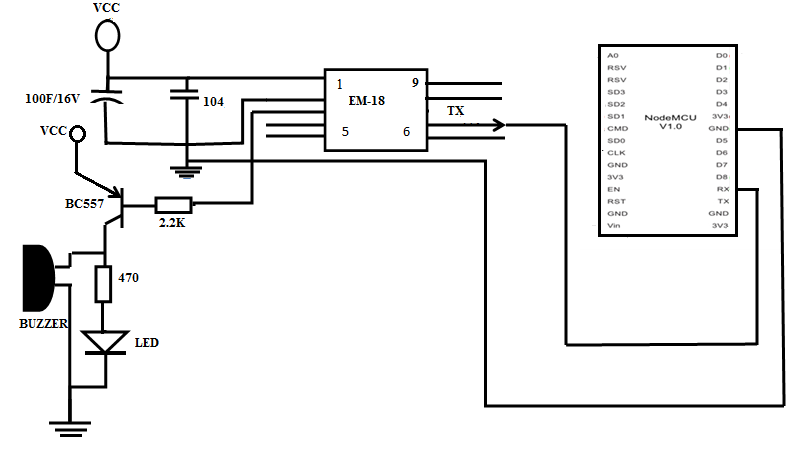
**Fig 3.1 Block diagram level**-**1 security and level**-**2 security**

**3.1.2 Block diagram of entire system**

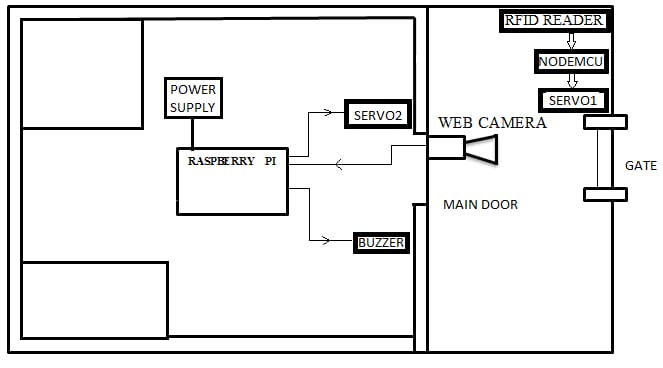
****

**Fig 3.1.2 Block diagram of entire system**

**3.1.3 Application circuit**



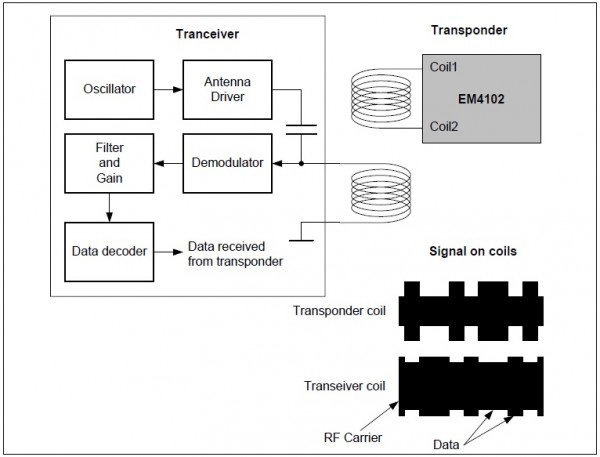
**3.2 System layout**

****

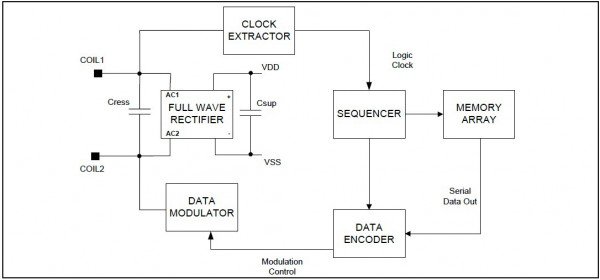
**Fig 3.2 System layout of project**

**3.3 RFID System**

The EM-18 RFID Reader module generates and radiates RF Carrier Signals of frequency 125KHz through its coils. When a 125KHz Passive [RFID](https://electrosome.com/rfid-radio-frequency-identification/) Tag (have no battery) is brought in to this field, will get energized from it. These RFID Tags are usually made using a CMOS IC EM4102. It gets enough power and master clock for its operations from the electromagnetic fields produced by RFID Reader. By changing the modulation current through the coils, tag will send back the information contained in the factory programmed memory array.

[](https://electrosome.com/wp-content/uploads/2014/11/RFID-System-Principle.jpg)

**Fig 3.3 RFID System principle**

[](https://electrosome.com/wp-content/uploads/2014/11/Passive-RFID-Tag-Block-Diagram.jpg)

**Fig 3.3.1 Passive RFID Tag block diagram**

**3.4 Getting Started with Raspberry Pi**

**3.4.1Configuring the raspberry pi**

1. Download NOOBS from the raspberry pi website and extract it on the sd card on your laptop.

2. Now, insert the micro SD card on the raspberry pi.

3. Connect pi to the monitor using a HDMI cable.

4. Connect a usb mouse and keyboard to the pi‘s usb ports.

5. Plug in the WI-Fi adapter to one of the pi‘s USB ports.

6. Plug in the power supply.

7. A start up screen should appear showing you a list of operating systems available for installation choose Raspbian OS and install it.

8. It will 15-20 minutes for completion.

9. The raspberry pi must be configured for local time zone and set date time etc.

. Type in your user name and password when asked.

. To go to the visual appearance of the OS type the command: startx

**3.4.2. Network configuration**

Network configuration should be done for the Pi to enable installation of different packages, dependencies and to regularly update and upgrade the operating system.

**3.4.3 List of packages**

1. Following packages are to be installed for implementing the proposed model. Installation commands have been listed below.

i. sudo apt-get install camorama

ii. sudo apt-get install python-dev

iii. sudo apt-get install libjpeg62-dev

iv. sudo apt-get install libpngl2-dev

v. sudo apt-get install python-matplotlib

vi. sudo apt-get install python-numpy

vii. sudo apt-get install python-scipy

viii. sudo apt-get install python-imaging

ix. sudo apt-get install python-tk

**3.4.4 OpenCV installation**

We use OpenCV to implement the proposed algorithm. Before installing OpenCV following dependencies were installed in the following manner. Menu –> Accessories -> LX Terminal

-> this opens a terminal window in RPi graphical interface. Then we use following commands.

i. sudo apt-get install build-essential

ii. sudo apt-get install cmake

iii. sudo apt-get install pkg-config

iv. sudo apt-get install libpngI2-0 libpngI2-dev libpng++dev libpng3

v. sudo apt-get install zlibig-dbg zlibig zlibig-dev

vi. sudo apt-get install libpnglite-dev libpngwriterO-dev IibpngwriterOc2

vii. sudo apt-get install pngtools libtiff4-dev libtiff4 libtiffxxOc2 libtiff-tools

viii. sudo apt-get install libjpeg8 libjpeg8-dev libjpeg8-dbg libjpeg-progs

ix. sudo apt-get install ffrnpeg libavcodec-dev libavcodec52 libavformat52 libavformat-dev

x. sudo apt-get install libgstreamerO.I O-O-dbg IibgstreamerO.IO-O IibgstreamerO.I O-dev

xi. sudo apt-get install libxinel-ffmpeg libxine-dev Iibxine I-bin

xii. sudo apt-get install Iibunicap2 libunicap2-dev

xiii. sudo apt-get install libdc I394-22-dev libdc 1394-22 IibdcI394-utils

xiv. sudo apt-get install libv41-0 libv41-dev

xv. sudo apt-get install python-numpy

xvi. sudo apt-get install libpython2. 7 python-dev python2.7-dev

xvii. sudo apt-get install libgtk2.0-dev pkg-config

2. OpenCV was downloaded from the internet using the

following command

wget http://sourceforge . net/projects/ opencv library/files/opencv-unixl2.3 . 1I0penCV2.3. 1 a.tar.bz2/download

3. Then we make, install and configure the OpenCV installed by using following commands.

i. cmake -D CMAKE- BUILD- TYPE = RELEASE -D

CMAKE \_ INST ALL ]REFIX = <path to OpenCV folder which has been extracted from the downloaded file> -D BUILD- NEW- PYTHON SUPPORT ON -D BUILD EXAMPLES=ON ..

ii. sudo make

iii. sudo make

**3.5. Face Recognition**

**3.5.1. Face detection using Voila John Algorithm**

The Viola–Jones object detection framework is the first object detection framework to provide competitive object detection rates in real-time proposed in 2001 by Paul Viola and Michael Jones. Although it can be trained to detect a variety of object classes, it was motivated primarily by the problem of face detection. The basic problem to be solved is to implement an algorithm for detection of faces in an image. This can be solved easily by humans. However, there is a stark contrast to how difficult it actually is to make a computer successfully solve this task. In order to ease the task Viola–Jones limit themselves to full view frontal upright faces. That is, in order to be detected the entire face must point towards the camera and it should not be tilted to any side. This may compromise the requirement for being unconstrained a little bit, but considering that the detection algorithm most often will be succeeded by a recognition algorithm these demands seem quite reasonable. Feature types and evaluation. The main characteristics of Viola–Jones algorithm which makes it a good detection algorithm are:

 Robust – very high detection rate (true-positive rate) & very low false-positive rate always.

 Real time – For practical applications at least 2 frames per second must be processed.

 Face detection and not recognition - The goal is to distinguish faces from non-faces (face detection is the first step in the identification processThe algorithm has mainly 4 stages:

1. Haar Features Selection

2. Creating Integral Image

3. Adaboost Training algorithm

4. Cascaded Classifiers

The features employed by the detection framework universally involve the sums of image pixels within rectangular areas. As such, they bear some resemblance to Haar basis functions, which have been used previously in the realm of image-based object detection. However, since the features used by Viola and Jones all rely on more than one rectangular area, they are generally more complex. The figure at right illustrates the four different types of features used in the framework. The value of any given feature is always simply the sum of the pixels within clear rectangles subtracted from the sum of the pixels within shaded rectangles. As is to be expected, rectangular features of this sort are rather primitive when compared to alternatives such as steerable filters. Although they are sensitive to vertical and horizontal features, their feedback is considerably coarser.

**3.5.2. Face recognition using Principal Component Analysis**

Face Recognition is implemented using Principal Components Analysis method. PCA is dimensionality reduction technique which extracts principal components. The first principal components are linear combinations of highest variability components while nth principal components are linear combinations of maximum variability among features. This scheme is based on an approach where each face is decomposed in to small sets of significant features called ―Eigenfaces‖. These are nothing but principal components. Afterwards, Eigenvectors are computed and later Euclidean distance and thresholding are used for recognition of faces.

PCA is most simple algorithm while highly sensitive to variations. The recognition flow using PCA is detailed below.



**Fig 3.5 Face Recognition using PCA algorithm**

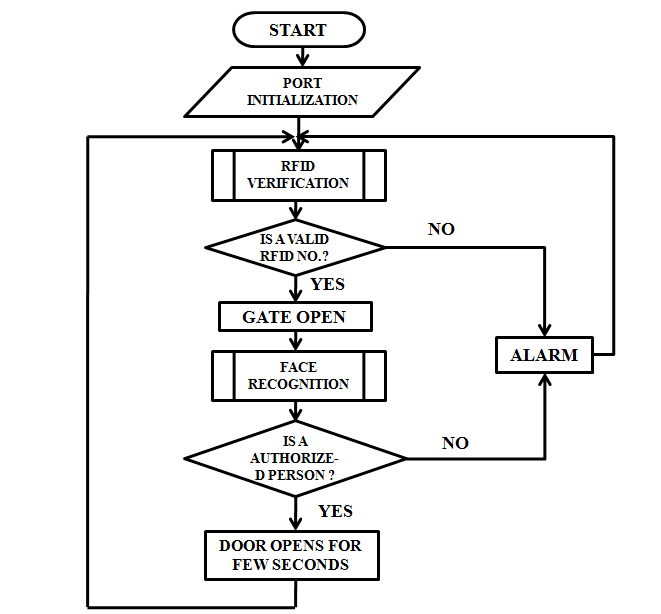
**3.5.3 Classification using confidence measure**

Confidence measure is for rejection of unknown faces and detection of recognition errors.

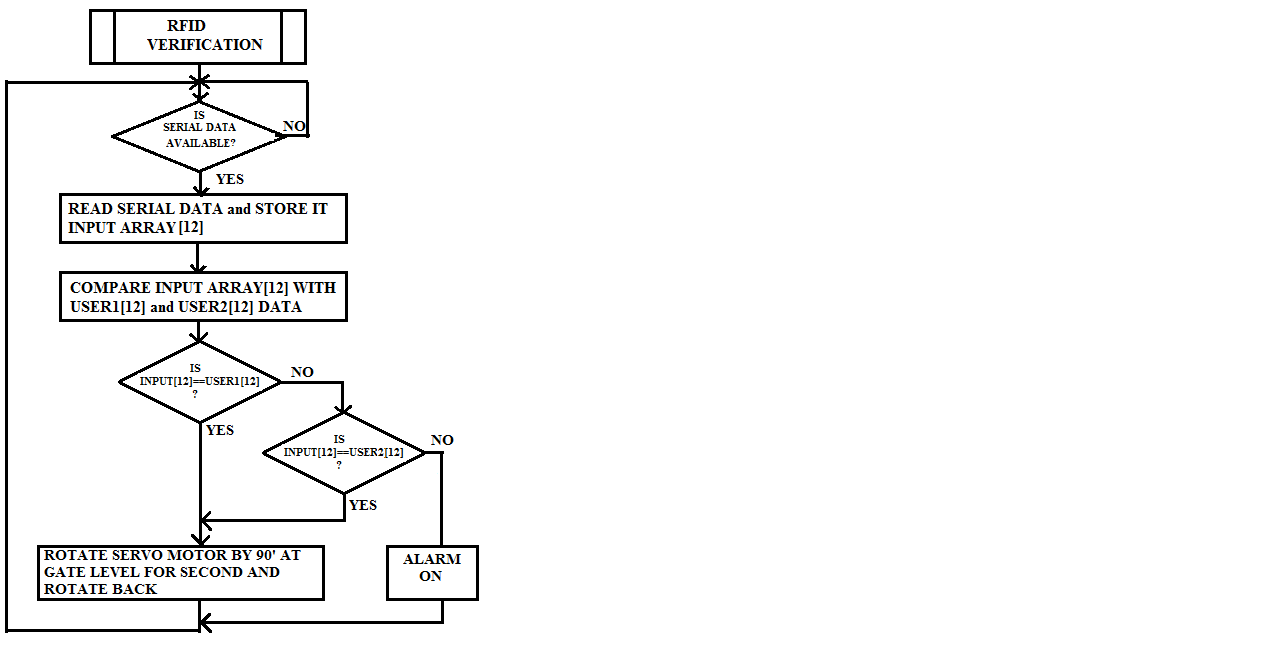
The confidence is a value like 1321.35 or 3987.77 above and represents the distance of the captured face from the predicted face--a lower distance value means the captured face is more similar to the predicted face. Positive threshold value is 2000.Predicted POSITIVE face with confidence 1321.35253959 (lower is more confident) will recognize the face. Predicted NEGATIVE face with confidence 3987.76625152 will not recognize the face. If a single face couldn't be detected (because none is visible, or there are multiple faces detected) an error message will be displayed like with the training script.

When a face is found, you'll notice the prediction message includes how the face was recognized (i.e. matching either the positive or negative training data), and the confidence of the recognition.

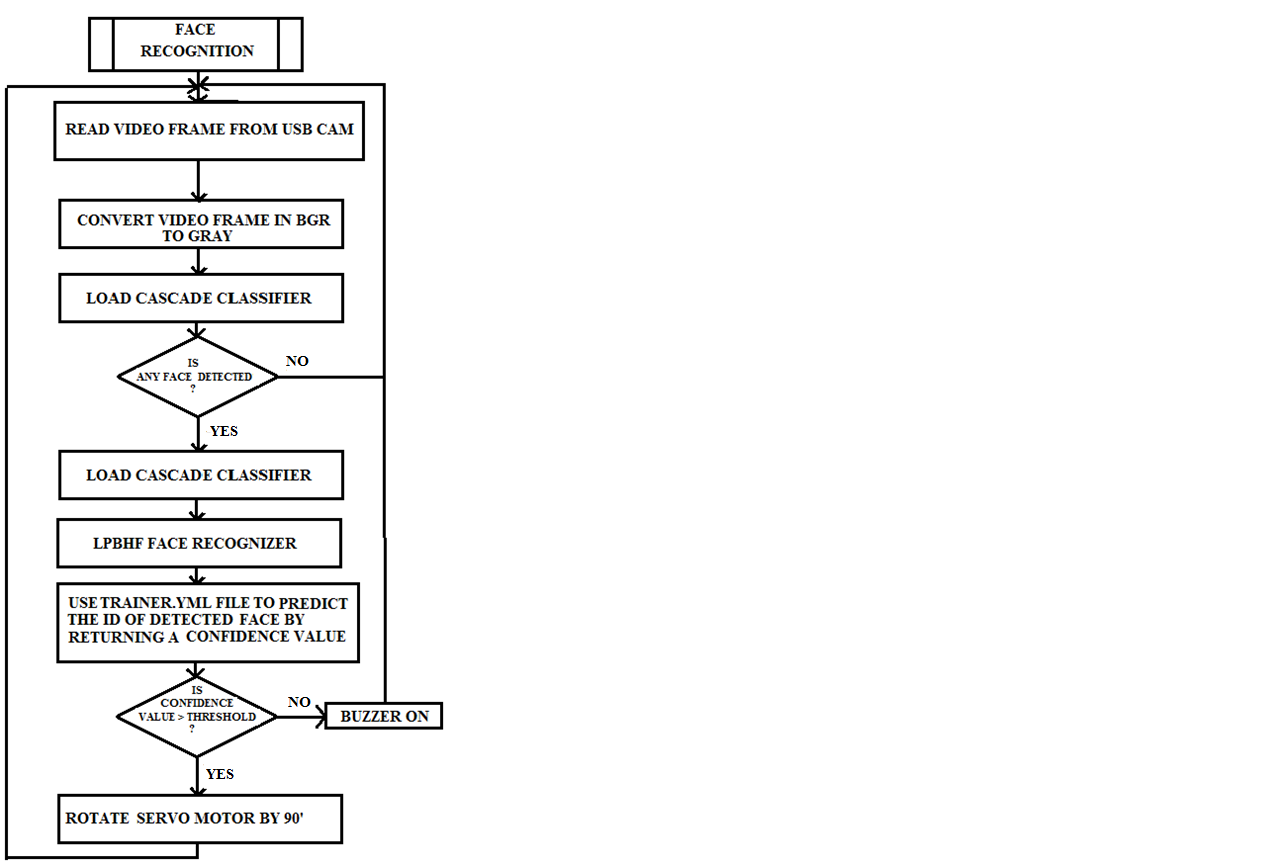
**3.6 Flowchart**



**Fig 3.6.1 Main System flow chart**



**Fig 3.6.2 Flow chart Level-1 security**



**Fig 3.6.3 Flow chart Level-2 security**

**3.7 Summary**

In this chapter we discussed about the system design and implementation, functional and block diagram of entire system of our project. Then we discussed configuration, packages used in Raspberry Pi and OpenCV installation. Later face recognition methods are discussed.

**CHAPTER-4**

**SOFTWARE REQUIREMENTS**

The software that were used are listed below:

The data fetched and transmitted through Wi-Fi medium are tended to be processed to extract

useful information. In this project, OPENCV are the application software used for signal analysis.

**4.1. Raspbian Jessie**

Raspbian Jessie is a Raspberry Pi operating system which is installed on SD card. It is nothing but bootable image file. Installing Jessie involves only few steps. The components required for installation are a SD card (4GB or more), windows or Mac computers and Raspbian image file.

**4.2. OPENCV**

[Open CV (Open Source Computer Vision)](http://opencv.org/) is a popular computer vision library started by [Intel](http://www.intel.com/) in 1999. The cross-platform library sets its focus on real-time image processing and includes patent-free implementations of the latest computer vision algorithms. In 2008 [Willow Garage](http://www.willowgarage.com/) took over support and Open CV 2.3.1 now comes with a programming interface to C, C++, [Python](http://www.python.org/) and [Android](http://www.android.com/). Open CV is released under a BSD license so it is used in academic projects and commercial products alike.

Open CV 2.4.9 now comes with the very new [Face Recognizer](http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_api.html#FaceRecognizer : public Algorithm) class for face recognition, so you can start experimenting with face recognition right away. This document is the guide. The currently available algorithms are:

* Eigen faces (see [create Eigen Face Recognizer()](http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_api.html#Ptr<FaceRecognizer> createEigenFaceRecognizer(int num_components , double threshold)))
* Fisher faces (see [create Fisher Face Recognizer()](http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_api.html#Ptr<FaceRecognizer> createFisherFaceRecognizer(int num_components , double threshold)))
* Local Binary Patterns Histograms (see [create LBPH Face Recognizer()](http://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_api.html#Ptr<FaceRecognizer> createLBPHFaceRecognizer(int radius, int neighbors, int grid_x, int grid_y, double threshold)))

There are 3 main stages in face recognition: -

1. Face detection: Viola Jones method has been used to extract Haar features and cascade them to train the system what a face is.
2. Feature extraction: Principal component analysis method has been used to extract the Eigen faces from the given images. It reduces data dimensionality.
3. Classification: Confidence measures method is used. Based on a threshold value two faces are compared. The threshold value is the distance between the test and trained image.

**4.3. Python IDE**

Python is widely used general-purpose, high-level programming language. The language provides constructs intended to enable clear programs on both a small and large scale. Its design philosophy emphasizes code readability and its syntax allows programmers to express concepts in few lines of code that would be possible in languages such as C++ or Java.

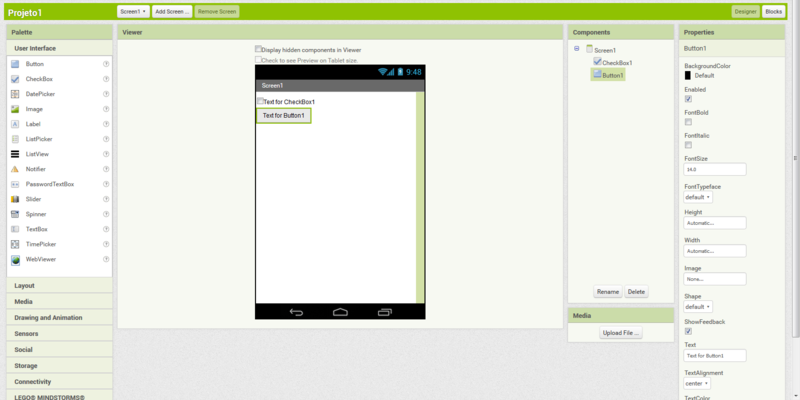
Python interpreters are available for installation on many operating systems, allowing Python code execution on a wide variety of systems. Python IDE provides an integrated environment for compilation and execution of python code.

**4.4 Embedded C**

Embedded C is a set of language extensions for the [C Programming language](https://en.wikipedia.org/wiki/C_Language) by the [C Standards committee](https://en.wikipedia.org/wiki/SC22) to address commonality issues that exist between C extensions for different [embedded systems](https://en.wikipedia.org/wiki/Embedded_systems). Historically, embedded C programming requires nonstandard extensions to the C language in order to support exotic features such as [fixed-point arithmetic](https://en.wikipedia.org/wiki/Fixed-point_arithmetic), multiple distinct [memory banks](https://en.wikipedia.org/wiki/Memory_bank), and basic [I/O](https://en.wikipedia.org/wiki/Input/output) operations.In 2008, the C Standards Committee extended the C language to address these issues by providing a common standard for all implementations to adhere to. It includes a number of features not available in normal C, such as, fixed-point arithmetic, named address spaces, and basic I/O hardware addressing.

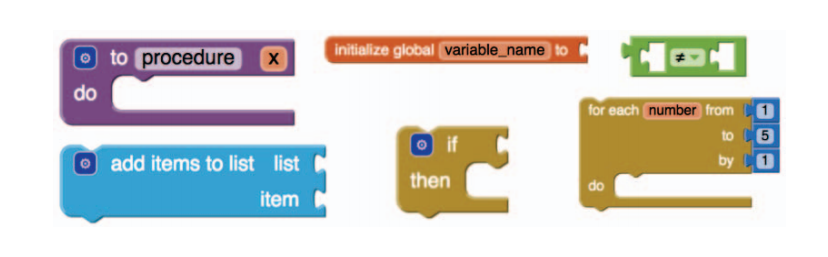
**4.5 MIT app inventor**

MIT app inventor is an open source web application originally provided by Google , and now maintained by the Massachusetts institute of technology(MIT).



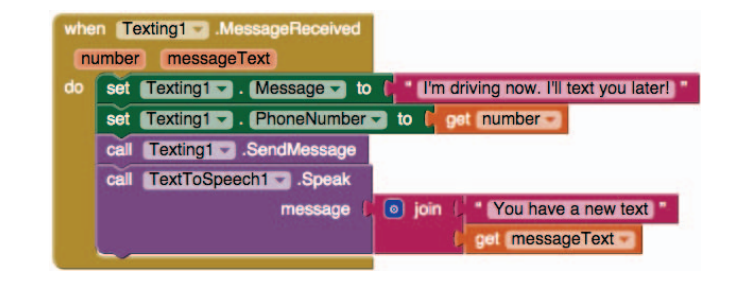
**Fig 4.5 Google App inventor**

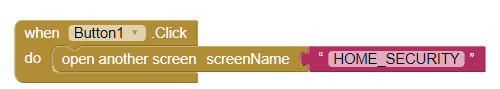
MIT App Inventor is an environment that leverages a blocks-based visual language to enable people to create mobile applications (”apps”) for Android devices.An App Inventor project consists of a set of components and a set of program blocks that provide functionality to these components (Blockly,)Components include items visible on the phone screen (e.g. buttons, text boxes) as well as non-visible items (e.g. camera, database, sensors). Figure 1 shows blocks used in an app that responds to text messages and reads them aloud. Upon receiving a text message, this program replies to the sender with a default message and reads the received message aloud.



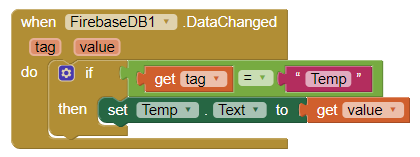
**Fig 4.5.1 blocks programming language in MIT App Inventor**

We adapt computational concepts from the framework for assessing computational thinking in Scratch for use with App Inventor .We define six computational concepts for App Inventor: {procedure, variable, logic, loop, conditional, list}. Of the 1,333 different types of blocks found in projects we analyzed, 39 of them are computational concept blocks (CC blocks) that relate to computational concepts. Figure 2 shows examples of CC blocks from each of the six concepts.

**Fig. 4.5.2 Computational Concept (CC) block**



**Fig 4.5.3 Blocks used for the front screen in mobile application**



**Fig 4.5.4 Blocks used for the displaying the user information from fire base database in mobile application**

****

**Fig 4.5.5 Screenshot of MIT App Inventor**

**4.6 Fire base cloud**

Firebase Cloud Messaging (FCM) is a cross-platform interactive application which lets you build your application on multiple platforms like android, ios ,tablet and many more, that allow you to deliver messages in more reliable manner at no cost. Using FCM, you can notify a client app that new email or other data is available to sync, all this is done in real time. FCM allows multiple parties to send



**Fig 4.6 Firebase cloud messaging**

An FCM implementation includes an app server that interacts with FCM via HTTP or XMPP protocol, and a client app. You can compose and send messages using the app server or the notification console. Firebase Cloud Messaging allows us to build notification of firebase that shares the same software development kit for client development. For testing or for sending marketing or engagement messages with powerful built-in targeting and analytics, you can use Notifications. Firebase can update the data within nanoseconds.

With FCM, you can send two types of messages to clients:

• Notification messages, sometimes --thought of as "display messages."

• Data messages, which are handled by the client app.

**Features**

• Real time Database: A schema less and cloud hosted database that is structured in Java Script Object Notation format.

• Authentication: Authentication in firebase provides a very good service of authenticating all the users without even using any client side coding.

• Cloud messaging: Firebase cloud messaging allows messages to be delivered from any platform other than the one on which it is developed in a reliable manner.

• Storage: Files can be uploaded and downloaded in a secure manner without being affected by network quality

• Hosting: Firebase web applications can be statically hosted in a secure manner in a very fast and quick manner.

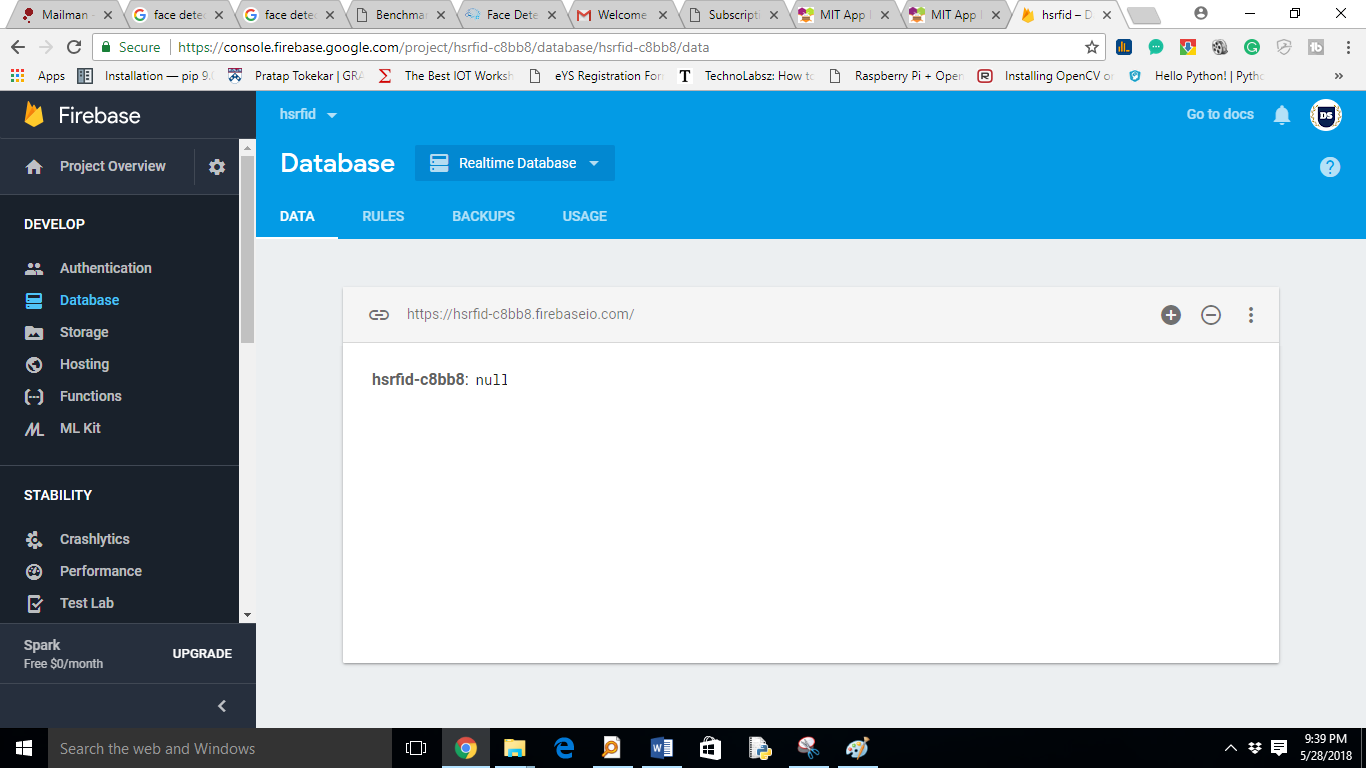
• Remote config: This feature allows to update the user application without the need of deploying the current updated app.

• Test lab: It involves testing of applications with different configurations and with variety of devices.

• Crash reporting: This feature is used to prepare a report of all the app crashes and errors in application.



**Fig 4.6.1 Firebase analysis**

****

**Fig 4.6.2 Screenshot of firebase database**

**4.7 SERVO MOTOR CODE**

#include <Servo.h>

#include <FirebaseArduino.h>

#include <ESP8266WiFi.h>

#define FIREBASE\_HOST "hsrfid-c8bb8.firebaseio.com"

#define WIFI\_SSID "itsurarm"// Change the name of your WIFI

#define WIFI\_PASSWORD "ka14ej4004" // Change the password of your WIFI

char input[12];

Servo myservo;

void setup()

{

Serial.begin(9600);

WiFi.begin (WIFI\_SSID, WIFI\_PASSWORD);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println ("");

Serial.println ("WiFi Connected!");

Firebase.begin(FIREBASE\_HOST);

myservo.attach(D0);

}

void loop()

{

if(Serial.available())

{

int count = 0;

while(Serial.available() && count < 12) // Read 12 characters and store them in input array

{

input[count] = Serial.read();

count++;

delay(5);

}

Serial.print(input);

//Serial.print(input);

if(input[10]=='A')

{

Firebase.setString ("Temp","user1");

//user1 gate open code

int pos;

for(pos = 0; pos <= 90; pos += 1) // goes from 0 degrees to 90 degrees

{ // in steps of 1 degree

myservo.write(pos); // tell servo to go to position in variable 'pos'

delay(15); // waits 15ms for the servo to reach the position

}

delay(2000); // waits 2 sec

for(pos = 90; pos>=0; pos-=1) // goes from 90 degrees to 0 degrees

{

myservo.write(pos); // tell servo to go to position in variable 'pos'

delay(15); // waits 15ms for the servo to reach the position

}

}

else if(input[10]=='D')

{

Firebase.setString ("Temp","user2");

//user2 gate open

int pos;

for(pos = 0; pos <= 90; pos += 1) // goes from 0 degrees to 90 degrees

{ // in steps of 1 degree

myservo.write(pos); // tell servo to go to position in variable 'pos'

delay(15); // waits 15ms for the servo to reach the position

}

delay(2000); // waits 2 sec

for(pos = 90; pos>=0; pos-=1) // goes from 90 degrees to 0 degrees

{

myservo.write(pos); // tell servo to go to position in variable 'pos'

delay(15); // waits 15ms for the servo to reach the position

}

}

else

{

Firebase.setString ("Temp","invalid");

}

}

delay(200);

}

**4.7.1 Python code to prepare data set of face**

import time

import cv2

vid\_cam = cv2.VideoCapture(-1)

time.sleep(0.1)

##return\_value, image = camera.read()

face\_detector = cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')

# For each person, one face id

face\_id = 1

# Initialize sample face image

count = 0

# Start looping

while(1):

# Capture video frame

ret, image\_frame = vid\_cam.read()

cv2.imwrite("image.jpg", image\_frame)

gray = cv2.cvtColor(image\_frame, cv2.COLOR\_BGR2GRAY)

# Detect frames of different sizes, list of faces rectangles

faces = face\_detector.detectMultiScale(gray, 1.3, 5)

# Loops for each faces

for (x,y,w,h) in faces:

# Crop the image frame into rectangle

cv2.rectangle(image\_frame, (x,y), (x+w,y+h), (255,0,0), 2)

# Increment sample face image

count += 1

# Save the captured image into the datasets folder

cv2.imwrite("dataset/User." + str(face\_id) + '.' + str(count) + ".jpg", gray[y:y+h,x:x+w])

# Display the video frame, with bounded rectangle on the person's face

cv2.imshow('frame', image\_frame)

# To stop taking video, press 'q' for at least 100ms

if cv2.waitKey(100) & 0xFF == ord('q'):

break

# If image taken reach 100, stop taking video

elif count>50:

break

# Stop video

vid\_cam.release()

# Close all started windows

cv2.destroyAllWindows()

**4.7.2 Python code to training the neural network**

# Import OpenCV2 for image processing

# Import os for file path

import cv2, os

# Import numpy for matrix calculation

import numpy as np

# Import Python Image Library (PIL)

from PIL import Image

# Create Local Binary Patterns Histograms for face recognization

##recognizer = cv2.face.createLBPHFaceRecognizer()

recognizer = cv2.face.LBPHFaceRecognizer\_create()

# Using prebuilt frontal face training model, for face detection

detector = cv2.CascadeClassifier("haarcascade\_frontalface\_default.xml");

# Create method to get the images and label data

def getImagesAndLabels(path):

# Get all file path

imagePaths = [os.path.join(path,f) for f in os.listdir(path)]

# Initialize empty face sample

faceSamples=[]

# Initialize empty id

ids = []

# Loop all the file path

for imagePath in imagePaths:

# Get the image and convert it to grayscale

PIL\_img = Image.open(imagePath).convert('L')

# PIL image to numpy array

img\_numpy = np.array(PIL\_img,'uint8')

# Get the image id

id = int(os.path.split(imagePath)[-1].split(".")[1])

print(id)

# Get the face from the training images

faces = detector.detectMultiScale(img\_numpy)

# Loop for each face, append to their respective ID

for (x,y,w,h) in faces:

# Add the image to face samples

faceSamples.append(img\_numpy[y:y+h,x:x+w])

# Add the ID to IDs

ids.append(id)

# Pass the face array and IDs array

return faceSamples,ids

# Get the faces and IDs

faces,ids = getImagesAndLabels('dataset')

# Train the model using the faces and IDs

recognizer.train(faces, np.array(ids))

# Save the model into trainer.yml

##recognizer.save('trainer/trainer.yml')

recognizer.write('trainer/trainer.yml')

**4.7.3 Face recognition code**

# Import OpenCV2 for image processing

import cv2

# Import numpy for matrices calculations

import numpy as np

import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BOARD)

GPIO.setwarnings(False)

GPIO.setup(7,GPIO.OUT) ## servo

GPIO.setup(11,GPIO.OUT) ##buzzer

# Create classifier from prebuilt model

def servocall(int):

p = GPIO.PWM(7,50)

p.start(7.5)

try:

while True:

p.ChangeDutyCycle(7.5)

time.sleep(2)

p.ChangeDutyCycle(2.5)

time.sleep(2)

print "1"

time.sleep(1)

break

except KeyboardInterrupt:

p.stop()

GPIO.cleanup()

return;

faceCascade = cv2.CascadeClassifier('haarcascade\_frontalface\_default.xml')

# Initialize and start the video frame capture

cam = cv2.VideoCapture(0)

# Create Local Binary Patterns Histograms for face recognization

recognizer = cv2.face.LBPHFaceRecognizer\_create()

# Load the trained mode

##recognizer.read('trainer/trainer.yml')

Recognizer.read('trainer/trainer.yml')

# Load prebuilt model for Frontal Face

##cascadePath = "haarcascade\_frontalface\_default.xml"

id=0

# Set the font style

##font = cv2.InitFont(cv2.CV\_FONT\_HERSHEY\_COMPLEX\_SMALL,5,1,0,4)

# Loop

while True:

# Read the video frame

ret, im =cam.read()

# Convert the captured frame into grayscale

gray = cv2.cvtColor(im,cv2.COLOR\_BGR2GRAY)

# Get all face from the video frame

faces = faceCascade.detectMultiScale(gray, 1.2,5)

# For each face in faces

for(x,y,w,h) in faces:

# Create rectangle around the face

## cv2.rectangle(im, (x-20,y-20), (x+w+20,y+h+20), (0,255,0), 4)

cv2.rectangle(im, (x-20,y-20), (x+w,y+h), (0,0,255), 2)

# Recognize the face belongs to which ID

id,conf = recognizer.predict(gray[y:y+h,x:x+w])

print "conf is",conf

if(conf > 55):

# Check the ID if exist

if(id == 1):

id = "benak"

print "1"

servocall(1)

# If not exist, then it is Unknown

else:

id = "Unknown"

print "unknown"

GPIO.output(11,GPIO.HIGH)

time.sleep(1)

GPIO.output(11,GPIO.LOW)

time.sleep(1)

print "ex"

# Put text describe who is in the picture

## cv2.rectangle(im, (x-22,y-90), (x+w+22, y-22), (0,255,0), -1)

cv2.putText(im, str(id), (x,y+h),cv2.FONT\_HERSHEY\_SIMPLEX,1,(0,0,255))

# Display the video frame with the bounded rectangle

cv2.imshow('im',im)

# If 'q' is pressed, close program

if cv2.waitKey(10) & 0xFF == ord('q'):

break

# Stop the camera

cam.release()

# Close all windows

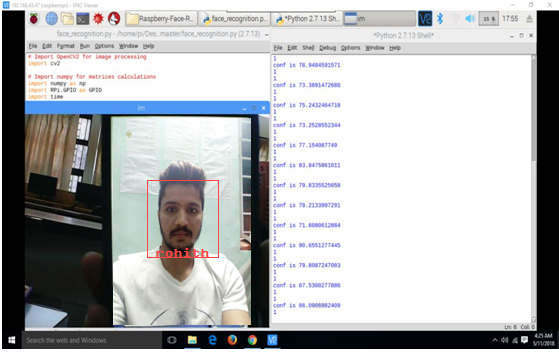
cv2.destroyAllWindows()

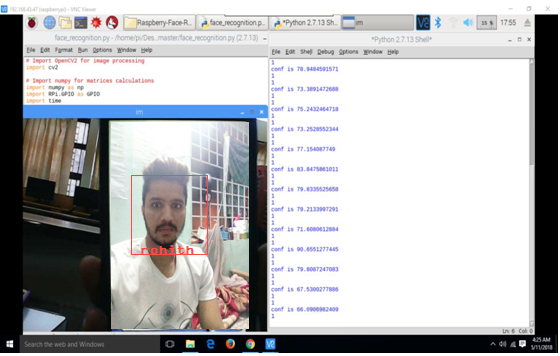
**4.7.4 Summary**

In this chapter we discussed about the software which are used in our project such as Raspbian Jessie, OpenCV,Python IDE and embedded-C.Then MIT app inventor, firebase cloud we have discussed about the codes which are used in our project.

**CHAPTER** -**5**

**RESULT**

****

****

**CHAPTER-6**

**CONCLUSION**

**6.1 Advantages**

* **Protects Valuables**

This is, an important benefit that most people immediately think of. The reason to install home security systems is to avoid avoid any loos of electronics, jewellery, or other high-value items in the case of home invasion. The tragedy is compounded when the item is an irreplaceable family heirloom. A home security system has an alarm that scares off many would-be burglars and can notify the local authorities if someone does attempt a break-in.

* **Deters Crime**

Studies have found that, as the number of home security systems increased in an area, the number of residential robberies decreased in that area, even for people who didn’t have their own security system. Having a security system not only protects you, but helps your neighbourhood be a safer place for everyone

* **Notifies You of Fire or Gas Problems**

You can opt to receive notifications if your smoke or carbon monoxide alarms go off when you’re away from home. Depending on the requirement, you can even set it up so authorities are instantly notified of these emergencies.

* **Lowers Homeowner’s Insurance**

Yes, you may be paying a monthly fee for your security system, but having the system in your home can lower your homeowner’s insurance by up to 20%. That, combined with the other benefits, makes home security a pretty good deal

* **Makes Room for Peace of Mind**

The sense of security and peace you gain with a home security system is perhaps the greatest benefit of all. Next to being safe, the confidence of feeling safe will help you be a more productive, healthy, and focused person. Adding a security system to your home gives you an extra layer of defence against any potential intruders.

**6.2 Applications**

* Residential complex/societies are secured using home security system that helps to keep a control/check of any potential intrusion.
* Commercial campuses installed with security system will have an advantage to avoid any valuable or management losses.

Military properties provided with security system prevent any unauthorized intrusion or loss of documents

**6.3 Future Enhancement**

3D facial recognition can be implemented to captures the class-specific properties of faces. The face recognition process can be made more reliable by increasing its efficiency and performance. Video processing can be implemented for live feed and surveillance. Occlusion can be implemented to recognize face during improper illumination. An Android app can be developed to control the microcontroller and sensors.

**6.4 Scope**

This project result is a home security system which has two major applications adumbrated below:

1. The house security system can be implemented in each and every individual house premises where there is a possibility of theft (alarm situation) as it is very user friendly and economical. In this project, the house is continuously checked by the wireless control unit which sends proper alert messages in the case of emergency.
2. Real time face recognition has many applications and various improvements are being implemented. This technology can be used for monitoring the people entering a particular place and storing the images of the people.

**6.5 Summary**

In this chapter we conclude the project by saying about the advantages,applications,future enhancement and scope of the project.

**CHAPTER-7**

**BIBLOGRAPHY**

1.Rajesh Kannan Megalingam, Ramesh Nammily Nair, Sai Manoj Prakhya and Mithun Mohan, “Low Power, Intelligent, Wireless, Home Security System for the Elderly People”,2011 IEEE.

2. Ma. Christina D. Fernandez, Kristina Joyce E. Gob, Aubrey Rose M. Leonidas,

Ron Jason J. Ravara, Argel A. Bandala and Elmer P. Dadios, “Simultaneous Face Detection and Recognition using Viola-Jones Algorithm and Artificial Neural Networks for Identity

Verification”, IEEE Region 10 Symposium, 2014.

3. Mrutyunjaya Sahani, Susanta Kumar Rout, Atul Kumar Sharan, and Subbradeep Dutta, “Real Time Color Image Enhancement with a high regard for restoration of skin color by using Raspberry Pi”, 2014 IEEE.

4. Dhiraj Sunehra and Ayesha Bano, “An Intelligent Surveillance with Cloud Storage for

Home Security”, 2014 Annual IEEE India Conference (INDICON).

5. Mrutyunjaya Sahani, Chiranjiv Nanda, Abhijeet Kumar Sahu and Biswajeet Pattnaik,

“Web-Based Online Embedded Door Access Control and Home Security System Based on Face Recognition”, International Conference on Circuit, Power and Computing Technologies [ICCPCT].