# ACKNOWLEDGEMENT

The successful of this project would not have been possible without the kind support and assistance of many individuals and organizations, and we are immensely blessed to have achieved this all along the duration of our project. We would like to extend our profound gratitude to each and every one of them.

We are highly indebted to **New Summit College** for constant guidance and supervision, as well as for providing all the necessary ICT infrastructure and friendly environment for the successful completion of the project. We are also appreciative of the efforts of BSc. CSIT coordinator **Mr. Chok Raj Dawadi**, without his supporting role, the project would have been nowhere near completion.

We would like to express our gratitude to our project supervisor **Mr. Roshan Khatri** who took keen interest on our project and guided us throughout the project by providing all the necessary ideas, information and knowledge for developing a system that secures a home.

# ABSTRACT

Home automation is a topic which is gaining popularity day by day, because of large advantages. Nowadays, technology is an integral part of everyone's lives. It influences several facts of everyday life and allows improved capability of indulging in entertainment. One can achieve home automation by simply connecting home appliance electrical devices to the internet or cloud storage. Door Lock System is a hardware-based system focused on the security of smart homes. This system uses a Webcam to record the live motion of the surroundings and unlock or lock the door according to the data trained using OpenCV. The Webcam is powered and controlled by Raspberry Pi. It uses HaarCascade Algorithm to detect the face of the user. The output of the door lock system either locks or unlocks the door according to the data trained using OpenCV and HaarCascade Classifier.

**Keywords: Face Detection, HaarCascade**

**Table of Contents**

[ACKNOWLEDGEMENT I](#_Toc65140296)

[ABSTRACT II](#_Toc65140297)

[LIST OF FIGURES VI](#_Toc65140298)

[LIST OF TABLES VII](#_Toc65140299)

[LIST OF ABBREVIATIONS VIII](#_Toc65140300)

[CHAPTER 1 1](#_Toc65140301)

[INTRODUCTION 1](#_Toc65140302)

[1.1 INTRODUCTION 1](#_Toc65140303)

[1.2 Problem Definition 1](#_Toc65140304)

[1.3 Objectives 2](#_Toc65140305)

[1.4 Scopes and Limitation 2](#_Toc65140306)

[1.4.1 Scope 2](#_Toc65140307)

[1.4.2 Limitation 2](#_Toc65140308)

[1.5. Report Organization 3](#_Toc65140309)

[Chapter 1: Introduction 3](#_Toc65140310)

[Chapter 2: Requirement analysis 3](#_Toc65140311)

[Chapter 3: System Design 3](#_Toc65140312)

[Chapter 4: Implementation and Testing 3](#_Toc65140313)

[Chapter 5: Result and Analysis 3](#_Toc65140314)

[Chapter 6: Conclusion and Enhancement 3](#_Toc65140315)

[CHAPTER 2 4](#_Toc65140316)

[LITERATURE REVIEW AND SYSTEM ANALYSIS 4](#_Toc65140317)

[2.1 Literature Review 4](#_Toc65140318)

[2.2 Requirement Analysis 5](#_Toc65140319)

[2.2.1 Functional Requirements 5](#_Toc65140320)

[2.2.2 Non-Functional Requirements 7](#_Toc65140321)

[2.3 Feasibility Analysis 7](#_Toc65140322)

[2.3.1 Economic Feasibility 7](#_Toc65140323)

[2.3.2 Technical Feasibility 7](#_Toc65140324)

[2.3.3 Operational Feasibility 7](#_Toc65140325)

[2.3.4 Schedule Feasibility 8](#_Toc65140326)

[CHAPTER 3 9](#_Toc65140327)

[SYSTEM DESIGN 9](#_Toc65140328)

[3.1 System Architecture 9](#_Toc65140329)

[3.2 Flow Diagram of System 10](#_Toc65140330)

[3.3 Sequence Diagram 11](#_Toc65140331)

[3.4 Class Diagram 12](#_Toc65140332)

[CHAPTER 4 13](#_Toc65140333)

[IMPLEMENTATION AND TESTING 13](#_Toc65140334)

[4.1 Implementation 13](#_Toc65140335)

[4.2 Tools used 14](#_Toc65140336)

[4.2.1 Hardware Requirements 14](#_Toc65140337)

[4.2.2 Software Requirements 16](#_Toc65140338)

[4.3 Data Collection and Algorithms 16](#_Toc65140339)

[4.3.1 Data Collection 17](#_Toc65140340)

[4.3.2 Algorithm implemented 17](#_Toc65140341)

[4.4 Testing 20](#_Toc65140342)

[4.4.1 Unit Testing 20](#_Toc65140343)

[4.4.2 Integration Testing 23](#_Toc65140344)

[4.4.3 System Testing 24](#_Toc65140345)

[CHAPTER 5 26](#_Toc65140346)

[RESULTS AND ANALYSIS 26](#_Toc65140347)

[CHAPTER 6 27](#_Toc65140348)

[CONCLUSION AND FUTURE ENHANCEMENT 27](#_Toc65140349)

[6.1 Conclusion 27](#_Toc65140350)

[6.2 Future Enhancement 27](#_Toc65140351)

[REFERENCE 28](#_Toc65140352)

# LIST OF FIGURES

[Figure 2. 1 Use Case Diagram 6](#_Toc65140444)

[Figure 2. 2 Gantt chart 8](#_Toc65140445)

[Figure 3. 1 System Architecture 9](#_Toc65137937)

[Figure 3. 2 Flow Diagram of the System 10](#_Toc65137938)

[Figure 3. 3 Sequence Diagram of the System 11](#_Toc65137939)

[Figure 3. 4 Class Diagram of the System 12](#_Toc65137940)

[Figure 4. 1 Incremental Prototyping Model 13](#_Toc65138501)

[Figure 4. 2 Raspberry Pi 3B+ 14](#_Toc65138502)

[Figure 4. 3 Webcam 15](#_Toc65138503)

[Figure 4. 4 Breadboard with led 15](#_Toc65138504)

[Figure 4. 5 HaarFeatures 18](#_Toc65138505)

[Figure 4. 6 Good features 19](#_Toc65138506)

[Figure 5. 1 Test Subjects 26](#_Toc65138606)

[Figure 5. 2 True Positive 26](#_Toc65138607)

# LIST OF TABLES

[Table 4. 1 Unit Testing of Dataset Script 20](#_Toc65139009)

[Table 4. 2 Unit Testing of Trainer Script 21](#_Toc65139010)

[Table 4. 3 Unit Testing of Recognizer Script 22](#_Toc65139011)

[Table 4. 4 Integration Testing of Scripts and Raspberry Pi 23](#_Toc65139012)

[Table 4. 5 System Testing 24](#_Toc65139013)

# LIST OF ABBREVIATIONS

|  |  |
| --- | --- |
| IoT | Internet of Things |
| HDM | Human Detection Module |
| LCD | Liquid-Crystal Display |
| RFID | Radio-Frequency Identification |
| NFC | Near Field Communication |
| GSM | Global System for Mobile Communications |
| PIC | Peripheral Interface Controller |
| OOP | Object Oriented Programming |
| OpenCV | Open-Source Computer Vision |
| LAN | Local Area Network |
| IDE | Integrated Development Environment |
| OS | Operating System |

# CHAPTER 1

# INTRODUCTION

## 1.1 INTRODUCTION

Home automation using IOT is a project that automates the house of the user. It can be used to increase the security and comfort of the house while reducing the cost of living. This allows the user to convert their home into a machine. This machine doesn’t need constant supervision from the user and can operate on its own to provide comfort to the user. It can control all the electric appliances of the house and also provide some extra applications to the user.

This project aims to free the user from the worries of home when going out for different purposes such as work, vacations, going out to meet with friends etc. This is achieved by the door lock system that has been implemented in this project. The door lock system in this project uses face recognition to unlock the door only when the authorized people are detected and send notification to the user when there are any signs of forced entry. Basically, the system is the door lock and the face of the user is the key to the lock. The system architecture, algorithms used in each stage and the implementation of the system is described in this report in detail.

## 1.2 Problem Definition

A house becomes a home only when it is secure and has no worries of threats like thieves and robbers. Many people in Nepal are still unaware of new technologies that are ignorant of such threats. They use traditional security measures like latches, locks and keys which are very easy to bypass. There are also some keyless door lock systems that use passcode instead of traditional key and lock but still have many flaws.

The common problems that arise in the mentioned door lock systems are:

* Traditional door locks and keys are easy to brute force with a hammer.
* Easy to lose keys.
* Breaching of passcodes/passwords.
* Lack of notification system in case of breach.
* Lack of delay system.

## 1.3 Objectives

The objectives of this project are:

* To secure the home of the user from the intruders
* To develop a door lock system based on face recognition.
* To notify the user about any forced entries.
* To delay the intruder as much as possible.

## 1.4 Scopes and Limitation

### 1.4.1 Scope

Home automation using IOT allows the owner/user to monitor and control the home appliances connected to the Arduino from remote places. The user is able to see the status (on/off state) and change the state of the home appliances through the internet from anywhere. This project is focused on the security part of home automation. This is achieved by an automatic door lock system which unlocks the door when the face of the user or authorized people is detected and if there is forced entry then the system locks the door and notifies the users.

### 1.4.2 Limitation

The limitations of this project are:

* The dataset depends on the quality of camera.
* The position of the user when image is captured affects face recognition.
* System couldn’t distinguish picture from actual person.

## 1.5. Report Organization

Our report is organized into 5 chapters:

### Chapter 1: Introduction

In this section the brief introduction of our project, statement of the problem, its objectives, scopes and limitations are discussed.

### Chapter 2: Requirement analysis

The previous work related to our projects were studied, system requirements. the functional and nonfunctional requirements are discussed and different feasibility analysis are summarized in this section.

### Chapter 3: System Design

In this section, we have designed the structuring system requirement like activity diagram, sequence diagram etc.

### Chapter 4: Implementation and Testing

In this section, various implementation methods and tools are described. This part also contains the description of various testing and results we got after performing.

### Chapter 5: Result and Analysis

In this section, the findings of this project are discussed.

### Chapter 6: Conclusion and Enhancement

In this section, a summary of the project and the future enhancements are discussed.

# CHAPTER 2

# LITERATURE REVIEW AND SYSTEM ANALYSIS

## 2.1 Literature Review

Several remote systems have been proposed whether for the academic or business domain. Such systems were intended to provide remote control and monitoring tasks. For instance, a system has been proposed by which is based on ZigBee technology. This system is composed of multiple modules such as the human detection module (HDM) which aims to detect the user at the door[1].

Digital door lock in home automation systems provides proper control and home environment monitoring to the user. A system has been proposed based on the RFID technology which provides a touch LCD monitor[2].

Another system consists of a build in NFC capabilities of a smart phone which would eventually be the key to open the door by means of logical link control protocol, which then matches the user's own set of passwords to verify that the user should be given permission or not[3].

A system proposed based on the design of GSM digital door lock system using PIC platform.5-digit password is used to lock/unlock the door. If the user submits an incorrect password the system notifies the owner[4].

In[5] an intelligent system for home security using illumination sensitive background models is presented. Such a system enables tracking and detection of intruders and it is based on providing home security. For this purpose, a face recognition technique is utilized to identify the intruder and on finding him, an image of the intruder is sent on the owner mail id for further action.

The main difference between these projects and our Door lock system is that it has high Image transformation in frequency domain, which gives better classification. The Haar Classifier uses the different stages i.e., Haar feature Selection, Creating Integral Images, Adaboost Training and Cascading Classifier. The image is passed through these steps to train and classify the objects. Result of our Door Lock System will show precision detection and classify the object with great accuracy.

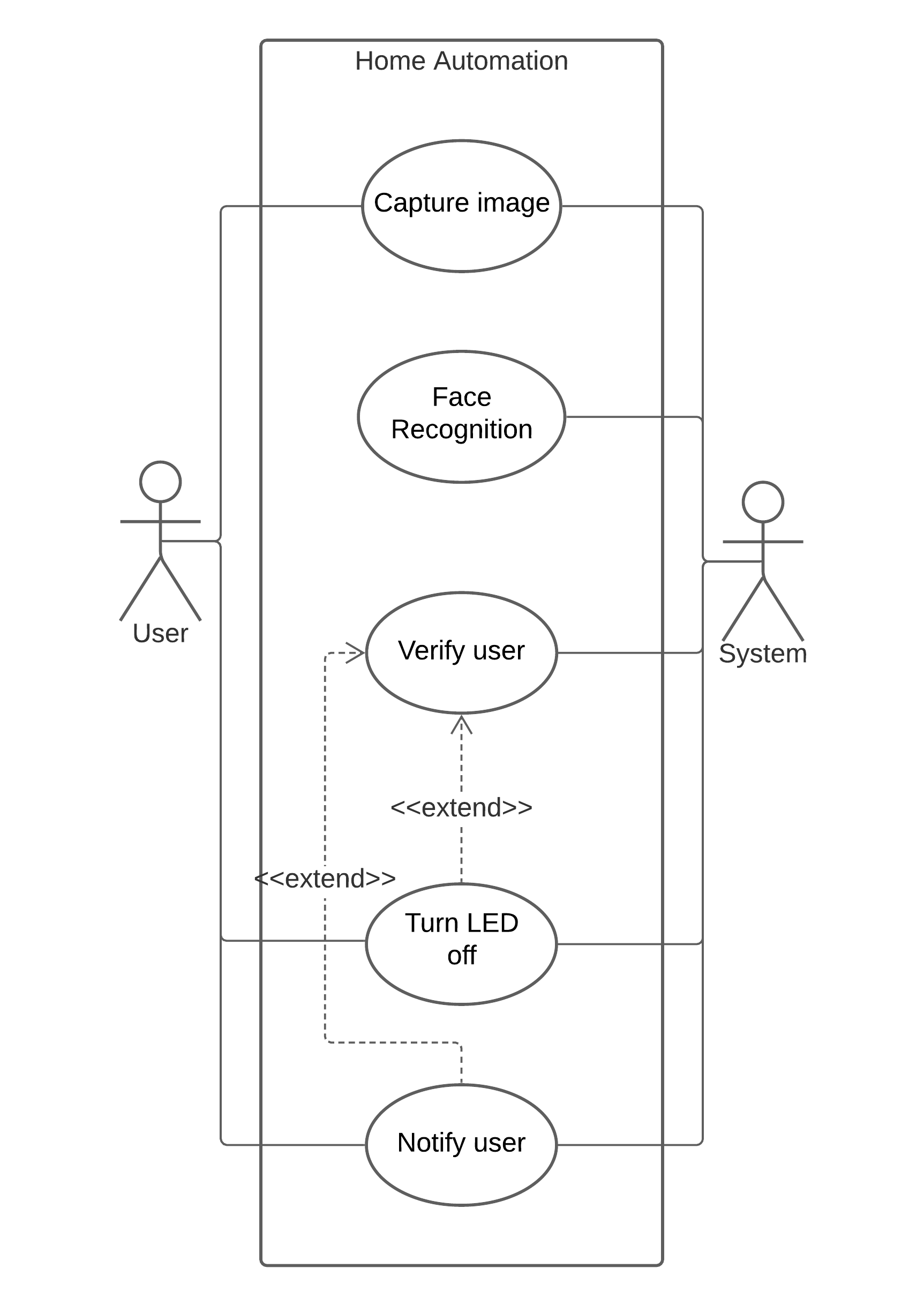
## 2.2 Requirement Analysis

### 2.2.1 Functional Requirements

Functional requirements identify the provision of the system and the system’s reaction to the certain output and how the system should behave on a day-to-day basis. This system is focused on capturing the image of people, recognizing the people and making decisions on locking or unlocking a door.

The functional requirements of “Door Lock System” includes the following tasks: -

* System should capture the image of people and classify them.
* System should detect the people correctly.
* The door must be unlocked to the user after detection.
* The user must get email notification when the intruder is detected.

****

*Figure 2. 1 Use Case Diagram*

### 2.2.2 Non-Functional Requirements

* **Usability:** The system needs to be usable by every user.
* **Maintenance:** The system needs to be maintainable.
* **Extendable and Scalability:** The system is extendable and scalable for futureenhancements.

## 2.3 Feasibility Analysis

### 2.3.1 Economic Feasibility

The purpose of economic feasibility assessment is to determine the positive economic benefits to the organization that the proposed system will provide. It includes quantification and identification of all the benefits expected. This assessment typically involves a cost/benefit analysis. The probable cost at the first step of the project includes all the hardware and software costs, which are minimal for our project. Hence, we can say that this project has been found to be economically feasible.

### 2.3.2 Technical Feasibility

This assessment is based on an outline design of system requirements to determine whether the company/team has the technical expertise to handle completion of the project. It is an evaluation of the hardware and software and how it meets the needs of the proposed system.

As per the requirement analysis, it was observed that technical expertise in the programming language Python, Machine Learning, is involved in the project. Owing to our familiarity with OOP based programming language we found the project will be technically feasible.

### 2.3.3 Operational Feasibility

Operational Feasibility is the measure of how a proposed system solves a problem and takes advantage of the opportunities identified during scope definitions and how it satisfies the requirements identified in the requirement analysis phase of the project development. To ensure success, desired operational outcomes must be imparted during design and development. These include such design-dependent parameters such as reliability, maintainability, usability, productivity, sustainability and others. Hence, we will be able to say that our project is operationally feasible.

### 2.3.4 Schedule Feasibility

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Work Time** | **Sep**  **15** | **Sep**  **30** | **Oct**  **1** | **Oct**  **20** | **Oct**  **21** | **Oct**  **25** | **Nov**  **8** | **Dec**  **1** | **Dec**  **2** | **Feb**  **2** |
| Planning |  |  |  |  |  |  |  |  |  |  |
| Research & Analysis |  |  |  |  |  |  |  |  |  |  |
| Design |  |  |  |  |  |  |  |  |  |  |
| Coding |  |  |  |  |  |  |  |  |  |  |
| Implementation |  |  |  |  |  |  |  |  |  |  |
| Testing |  |  |  |  |  |  |  |  |  |  |
| Documentation |  |  |  |  |  |  |  |  |  |  |

Figure 2. 2 Gantt chart

# CHAPTER 3

# SYSTEM DESIGN

## 3.1 System Architecture

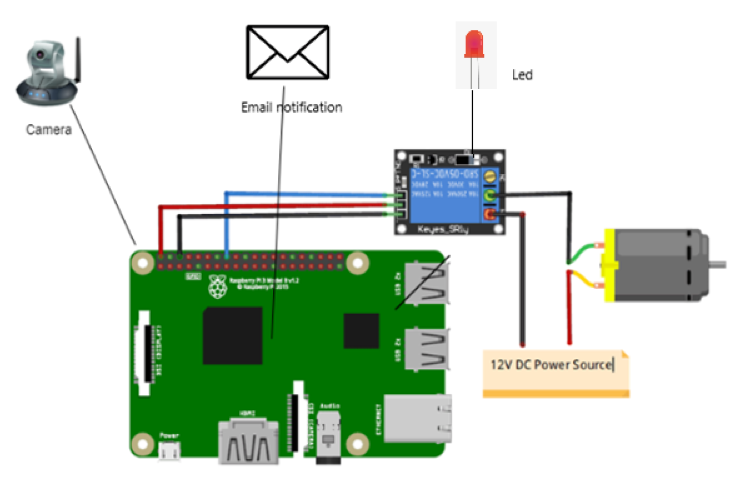


Figure 3. 1 System Architecture

## 3.2 Flow Diagram of System

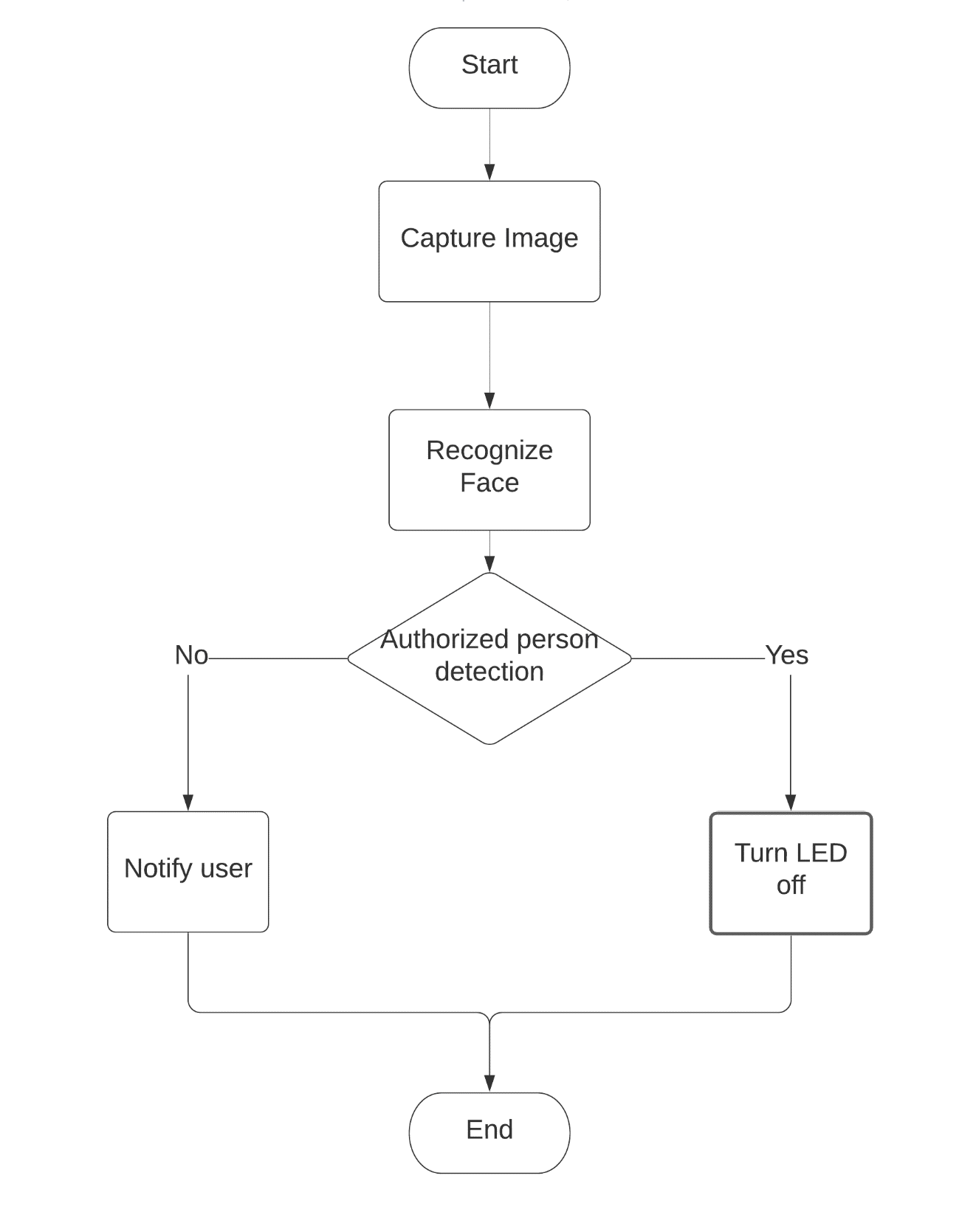


Figure 3. 2 Flow Diagram of the System

## 3.3 Sequence Diagram

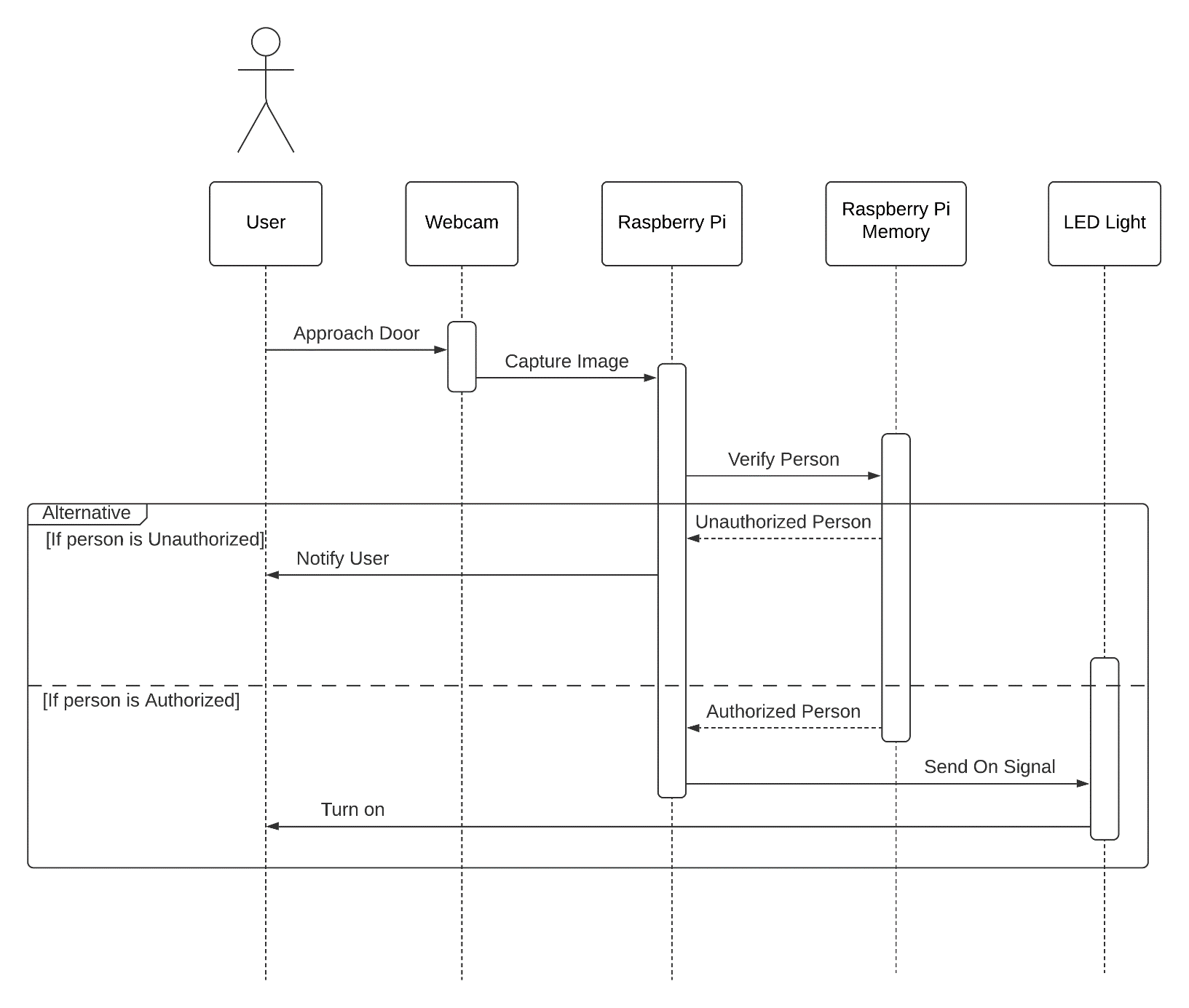


Figure 3. 3 Sequence Diagram of the System

The user approaches the door and stands in front of the webcam. Then the webcam captures the image of the user. After that this image is sent to the Raspberry Pi which checks its trained dataset to see if the user is authorized or not. If the person is authorized then the Raspberry pi sends the turn off signal to the LED light and the LED is turned off. But if the user is unauthorized then the Raspberry Pi notifies the user about the unauthorized user.

## 3.4 Class Diagram

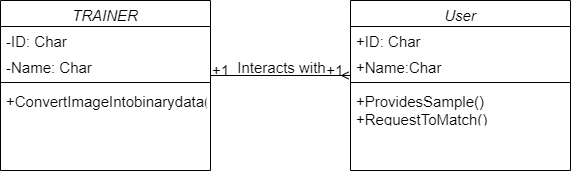


Figure 3. 4 Class Diagram of the System

The User class has attributes ID and Name. The user provides the sample for the system so that the system can learn the user’s face. Multiple users can interact with the Trainer class to request to match with the data in the Trainer class. The Trainer also has attribute ID and Name. It converts the image provided by the User into binary data.

# CHAPTER 4

# IMPLEMENTATION AND TESTING

## 4.1 Implementation

**Incremental Prototyping Model**

In this project, we used incremental prototyping to first design a prototype of a system that uses HaarCascade classifier that detects faces using a webcam to identify the user and unlock the door.

At first some image of the user is captured using the webcam, which is then used to train the system so that the system can recognize the user. After the system has been trained the system is able to detect faces and unlock the door if it recognizes the user. Here, the HaarCascade classifier algorithm is implemented to train the system.

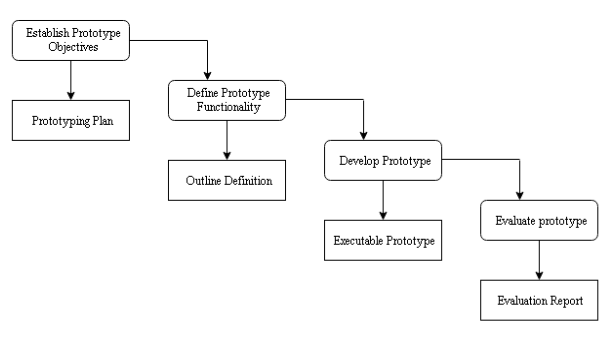


Figure 4. 1 Incremental Prototyping Model

## 4.2 Tools used

### 4.2.1 Hardware Requirements

**Raspberry Pi 3B+:**

Raspberry Pi 3B+ is a low-cost computer with 1.4GHz 64-bit quad-core processor, dual-band wireless LAN, Bluetooth 4.2/BLE, faster Ethernet, and Power-over-Ethernet support. In this project, the Raspberry Pi is used for computing the image received from the webcam, identifying the authorized people and controlling the solenoid lock to unlock the door for the authorized only.



Figure 4. 2 Raspberry Pi 3B+

**Webcam:**

Webcam stands for World Wide Web Camera. A webcam is a digital video device commonly built into a computer. Its main function is to transmit pictures over the Internet. It is popularly used with instant messaging services and for recording images. In this project, the webcam is used to take pictures of people who are trying to enter the house.



Figure 4. 3 Webcam

**Bread Board:** A thin plastic board used to hold electronic components that are wired together. The purpose of the breadboard is to make quick electrical connections between components- like resistors, LEDs, capacitors, etc. so that you can test your circuit



Figure 4. 4 Breadboard with led

### 4.2.2 Software Requirements

**OpenCV:**

OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. It deals with modeling and replicating human vision using computer software and hardware. Today by using Computer Vision we can explain how to reconstruct, interrupt, and understand a 3D scene from its 2D images, in terms of the properties of the structure present in the scene. It is used in this project for image processing and face detection.

**PyCharm:**

PyCharm is an Integrated Development Environment (IDE) used for programming in Python. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems, and supports web development with Django. In this project PyCharm is used for coding in python.

**Raspbian OS:**

Raspbian is a free operating system based on Debian optimized for the Raspberry Pi hardware. An operating system is the set of basic programs and utilities that make the Raspberry Pi run. This project used the 32-bit Raspbian OS to create a working environment in the Raspberry Pi 3B+.

**Microsoft Teams:**

Microsoft Teams is a proprietary business communication platform developed by [Microsoft](https://en.wikipedia.org/wiki/Microsoft). In this project, webhook based notification is used to send user a notification on Microsoft Teams when the system recognizes an intruder.

## 4.3 Data Collection and Algorithms

### 4.3.1 Data Collection

The data used in this project are the images of users taken from webcam which are fed into the model for the training purpose and some of the data were used for testing purposes. These collected data are trained by using HaarCascade Classifier for the Face Detection. During data collection the HaarCascade Algorithm is used which collects data from a webcam in inverted format. The main feature of this algorithm is an effective machine learning in which a cascade function is trained using a sample that contains a lot of positive and negative images. In this algorithm scale factor and minimum neighbors are the major proportion for security since the recognition is done under the minimum neighbors.

### 4.3.2 Algorithm implemented

**HaarCascade Classifier Algorithm**

HaarCascade Classifier is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect faces in other images. This algorithm needs a lot of positive images (images with faces) and negative images (images without faces) to train the classifier. Then the algorithm extracts certain features called Haar features from the images used to train the cascade function. Haar features shown in below images are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting the sum of pixels under white rectangle from the sum of pixels under black rectangle.

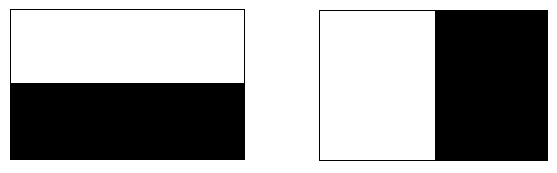
∆ = sum of dark pixels – sum of white pixels +

Where,

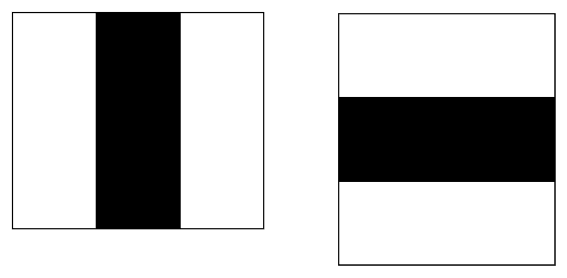
∆ = single value of each feature

I(x) = pixel intensity of a given pixel

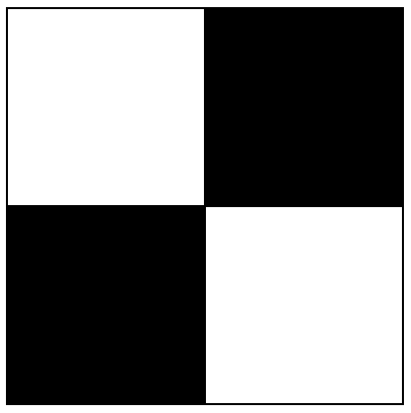
n = number of pixels



(a)Edge features



(b) Line Features



(c) Four-rectangle Features

Figure 4. 5 HaarFeatures

Then all possible sizes and locations of each kernel is used to calculate plenty of features. For each feature calculation, the sum of pixels under white and black rectangles is needed to be calculated using Integral images. It simplifies calculation of sum of pixels to an operation involving just four pixels, regardless of the number of pixels.

Most of the features calculated are irrelevant. For example, consider the image below. Top row shows two good features. The first feature selected seems to focus on the property that the region of the eyes is often darker than the region of the nose and cheeks. The second feature selected relies on the property that the eyes are darker than the bridge of the nose. But the same windows applying on cheeks or any other place is irrelevant.

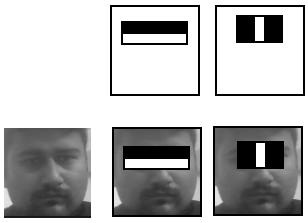


Figure 4. 6 Good features

So, to select the best features Adaboost training is used. For each feature, it finds the best threshold which will classify the faces to positive and negative. To do this, each image is given an equal weight in the beginning. After each classification, weights of misclassified images are increased. Then again, the same process is done. New error rates are calculated and also new weights. The process is continued until required accuracy or error rate is achieved or required number of features are found. Then weak classifiers that can’t classify the image alone are summed up together to form a strong final classifier that can provide face detection with 95% accuracy[6].This results in reduction of features from 160000+ in a 24x24 window to 6000[6] features.

After this step, the concept of Cascade of Classifiersis used to check whether the region of the image is not a face region and discard that region. Instead of applying all the 6000 features on a window, group the features into different stages of classifiers and apply one-by-one. If a window fails the first stage, it is discarded. If it passes, the second stage of features is applied and the process is continued. The window which passes all stages is a face region[6].

## 4.4 Testing

A series of testing were conducted to check the differences between the given input and the expected output of the system.

### 4.4.1 Unit Testing

After the work was divided and the coding was completed each script was tested and any bugs detected were fixed.

Table 4. 1 Unit Testing of Dataset Script

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Unit | Pre-Condition/Test Data | Expected Outcome | Actual Outcome | Status |
| 1 | Test if the script captures the image of face | Webcam must be on | Capture 30 images of face and save images in dataset folder | Webcam was not detected | Fail |
| 2 | Test if the script captures the image of face | Webcam must be on | Capture 30 images of face and save images in dataset folder | Captures 25 images of face and 5 of non-face region | Fail |
| 3 | Test if the script captures the image of face | Webcam must be on | Capture 30 images of face and save images in dataset folder | Captures 30 images of face but doesn’t save the images in dataset folder | Fail |
| 4 | Test if the script captures the image of face | Webcam must be on | Capture 30 images of face and save images in dataset folder | Captures 30 images of face and saves images in dataset folder | Pass |

Table 4. 2 Unit Testing of Trainer Script

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Unit | Pre-Condition/Test Data | Expected Outcome | Actual Outcome | Status |
| 1 | Test if the script trains the dataset images | Dataset folder must have images to train | Display image trained and create trainer.yml file | Displays image trained but doesn’t create trainer.yml file | Fail |
| 2 | Test if the script trains the dataset images | Dataset folder must have images to train | Display image trained and create trainer.yml file | Displays image trained and creates trainer.yml file | Pass |

Table 4. 3 Unit Testing of Recognizer Script

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Unit | Pre-Condition/Test Data | Expected Outcome | Actual Outcome | Status |
| 1 | Test if the script recognizes the authorized user | Webcam must be on | Display name of user | Displays name of user | Pass |
| 2 | Test if the script turns off led for authorized user | Webcam must be on | Display unlock door | Displays intruder alert | Fail |
| 3 | Test if the script turns off led for authorized user | Webcam must be on | Display door unlocked | Displays door unlocked | Pass |
| 4 | Test if script recognizes intruder | Webcam must be on | Display intruder alert | Displays door unlocked | Fail |
| 5 | Test if script recognizes intruder | Webcam must be on | Display intruder alert | Displays intruder alert | Pass |
| 6 | Test if script sends email notification to user about intruder | Internet must be available | User receives webhook notification | User receives webhook notification | Pass |

### 4.4.2 Integration Testing

After Unit testing was conducted and all the bugs were fixed, the Scripts and Raspberry Pi were integrated. Then integration testing was conducted on the integrated system.

Table 4. 4 Integration Testing of Scripts and Raspberry Pi

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Unit | Pre-Condition/Test Data | Expected Outcome | Actual Outcome | Status |
| 1 | Run the Dataset script by integrating into Raspberry Pi | Webcam should be connected to Raspberry Pi | Capture 30 images of face and save images in dataset folder in the Raspberry Pi | Captures 30 images of face and saves images in dataset folder in the Raspberry Pi | Pass |
| 2 | Run the Trainer script by integrating into Raspberry Pi | Dataset folder must have images to train | Create trainer.yml file | Creates trainer.yml file | Pass |
| 3 | Run the Recognizer script by integrating into Raspberry Pi to detect user | Webcam and LED light (initially on) must be connected to Raspberry Pi | Turn off LED light | Turn off LED light | Pass |
| 4 | Run the Recognizer script by integrating into Raspberry Pi to detect intruder | Webcam and LED light (initially on) must be connected to Raspberry Pi | No changes in LED light and user receives webhook notification about intruder | Turns off LED light | Fail |
| 5 | Run the Recognizer script by integrating into Raspberry Pi to detect intruder | Webcam and solenoid lock (initially locked) must be connected to Raspberry Pi | No changes in LED light and user receives webhook notification about intruder | No changes in LED light and user receives webhook notification about intruder | Pass |

### 4.4.3 System Testing

After the integration was done and all bugs were fixed a final system testing was conducted. This was done to check if the system worked correctly and after finding few bugs and fixing them the system was completed.

Table 4. 5 System Testing

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test Unit | Pre-Condition/Test Data | Expected Outcome | Actual Outcome | Status |
| 1 | Test if the system recognizes user | Raspberry pi should be connected to LED light | Turn off LED light | Turn off LED light | Pass |
| 2 | Test if the system recognizes intruder | Raspberry pi should be connected to locked LED light and internet | No changes in LED light and user receives webhook notification | No changes in LED light but user didn’t receive webhook notification | Fail |
| 3 | Test if the system recognizes intruder | Raspberry pi should be connected to locked LED light and internet | No changes in LED light and user receives webhook notification | No changes in LED light and user received webhook notification | Pass |

# 

# CHAPTER 5

# RESULTS AND ANALYSIS

In this section, each step of methodology is implemented and results are described. This system uses a door lock system based on face recognition by verifying the facial images of the test subjects. This system is always connected to the internet. This setup allows the system to send alert notification to user in case of intruder detection. The door locks automatically after 30 seconds.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| ID: 1 | ID: 2 | ID: 3 | ID: 4 |
| Prabin Sitaula | Rimjhim Sarraf | Alish Twanabasu | Sahil Ansari |

Figure 5. 1 Test Subjects

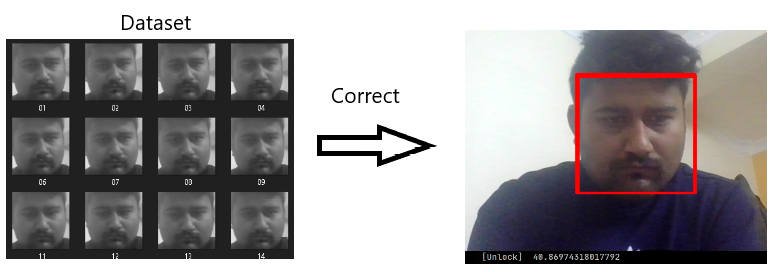


Figure 5. 2 True Positive

We found out that light, environment, stability and position of user affects the Face Detection. Also, the distance of the user from the camera is inversely proportional to the clarity of the face detection.

# CHAPTER 6

# CONCLUSION AND FUTURE ENHANCEMENT

## 6.1 Conclusion

The main aim of our project is to develop an embedded system for the opening and closing of the door. Thus, we have made the protocol of smart doors. First of all, we became familiar with the raspberry pi. During the programming section how to make the program practically, how to make the program efficient and error free.

Summarizing, a home security system using facial recognition and the Internet of Things was built using only low-budget hardware and open-source software. IoT communication in both directions, Android to Raspberry Pi and vice versa, is coded and can be used in further development. False positives have been reduced to a minimum with the recognition showing reliable results. Further improvements on the recognition part will depend on the improvements of the algorithms through neural networks and deep learning.

## 6.2 Future Enhancement

This work can further be extended by informing the users about the suspicious activity with an image of the intruder through email. It can be extended to notify the concerned authority i.e., the police department about the intruder as well as lock other alternative exits so that the intruder is delayed as much as possible. We can add sensors in addition so that the door can automatically open in case of any unexpected incidents like fire, earthquakes etc.

# 

# REFERENCE

[1] Gill, Khusvinder & Yang, Shuang-Hua & Yao, Fang & Lu, Xin. (2009). A ZigBee-Based Home Automation System. Consumer Electronics, IEEE Transactions on. 55. 422 - 430. 10.1109/TCE.2009.5174403

[2] Huang, Yo-Ping & Wang, Shan-Shan & Deng, Zheng-Hong. (2010). An Interactive RFID-based Multi-touch Product Display System.

[3]Alshehri, A. and Schneider, S., 2014. Addressing NFC Mobile Relay Attacks: NFC User Key Confirmation Protocols. International Journal of RFID Security and Cryptography, 3(2), pp.137-147.

[4] Qu, Qiao, Zhao Guohao, and Wei Baohua.” Design of Home Safeguard System Based on GSM Technique.” Electronic Engineer 32.11 (2006): 76-78.

[5]Young-Keun Choi, Ki-Man Kim, Ji-Won Jung, Seung-Yong Chun and Kyu-Sik Park, 2005. Acoustic intruder detection system for home security. IEEE Transactions on Consumer Electronics, 51(1), pp.130-138.

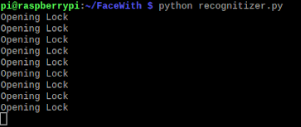
[6]"Face Detection using Haar Cascades — OpenCV-Python Tutorials 1 documentation", *Opencv-python-tutroals.readthedocs.io*, 2021. [Online]. Available: https://opencv-python-tutroals.readthedocs.io/en/latest/py\_tutorials/py\_objdetect/py\_face\_detection/py\_face\_detection.html. [Accessed: 06- Feb- 2021].

**APPENDICES**

1. **Screenshot of Dataset collection**
2. **Screenshot of Training**



1. **Screenshot of Recognition of User**





1. **Source code**

**Code for Dataset Script**

import cv2

import os

cam = cv2.VideoCapture(0)

cam.set(3, 640) # set video width

cam.set(4, 480) # set video height

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

# For each person, enter one numeric face id

face\_id = input('\n enter user id end press <return> ==> ')

print("\n [INFO] Initializing face capture. Look the camera and wait ...")

# Initialize individual sampling face count

count = 0

while(True):

ret, img = cam.read()

img = cv2.flip(img, 1) # flip video image vertically

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

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

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

cv2.rectangle(img, (x,y), (x+w,y+h), (255,0,0), 2)

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])

cv2.imshow('image', img)

k = cv2.waitKey(100) & 0xff # Press 'ESC' for exiting video

if k == 27:

break

elif count >= 30: # Take 30 face sample and stop video

break

# Do a bit of cleanup

print("\n [INFO] Exiting Program and cleanup stuff")

cam.release()

cv2.destroyAllWindows()

**Code for Trainer Script**

import cv2

import numpy as np

from PIL import Image

import os

# Path for face image database

path = 'dataset'

recoginizer = cv2.face.createLBPHFaceRecognizer()

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

# function to get the images and label data

def getImagesAndLabels(path):

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

faceSamples=[]

ids = []

for imagePath in imagePaths:

PIL\_img = Image.open(imagePath).convert('L') # convert it to grayscale

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

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

faces = detector.detectMultiScale(img\_numpy)

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

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

ids.append(id)

return faceSamples,ids

print ("\n [INFO] Training faces. It will take a few seconds. Wait ...")

faces,ids = getImagesAndLabels(path)

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

recoginizer.save("trainer.yml")

#Save the matrix as YML file

# Print the numer of faces trained and end program

print("\n [INFO] {0} faces trained. Exiting Program".format(len(np.unique(ids))))

**Code for Recognizer Script**

import cv2

import numpy as np

import os

import RPi.GPIO as GPIO

import time

import smtplib

from smtplib import SMTP

relay = 23

GPIO.setwarnings(False)

GPIO.setmode(GPIO.BCM)

GPIO.setup(relay, GPIO.OUT)

GPIO.output(relay ,1)

recognizer = cv2.face.createLBPHFaceRecognizer()

recognizer.load('trainer.yml')

cascadePath = "haarcascade\_frontalface\_default.xml"

faceCascade = cv2.CascadeClassifier(cascadePath);

font = cv2.FONT\_HERSHEY\_SIMPLEX

#iniciate id counter

id = 0

# names related to ids: example ==>

names = ['None','Prabin','Sahil']

# Initialize and start realtime video capture

cam = cv2.VideoCapture(0)

cam.set(3, 640) # set video widht

cam.set(4, 480) # set video height

# Define min window size to be recognized as a face

minW = 0.1\*cam.get(3)

minH = 0.1\*cam.get(4)

while True:

ret, img =cam.read()

img = cv2.flip(img, 1) # Flip vertically

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

faces = faceCascade.detectMultiScale(

gray,

scaleFactor = 1.2,

minNeighbors = 5,

minSize = (int(minW), int(minH)),

)

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

cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 2)

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

# Check if confidence is less them 100 ==> "0" is perfect match

if (confidence < 100):

id = names[id]

confidence = " {0}%".format(round(100 - confidence))

GPIO.output(relay, 0)

print("Opening Lock")

time.sleep(1)

GPIO.output(relay, 1)

else:

id = "unknown"

confidence = " {0}%".format(round(100 - confidence))

server\_ssl = smtplib.SMTP\_SSL("smtp.gmail.com", 465)

server\_ssl.ehlo()

server\_ssl.login('monikagc17@gmail.com', 'HelloWorld123')

server\_ssl.sendmail('monikagc17@gmail.com', 'sspravingmail.com', 'Intruder

Detected')

server\_ssl.close()

print('Message sent Successfully')

GPIO.output(relay, 1)

cv2.putText(img, str(id), (x+5,y-5), font, 1, (255,255,255), 2)

cv2.putText(img, str(confidence), (x+5,y+h-5), font, 1, (255,255,0), 1)

cv2.imshow('camera',img)

k = cv2.waitKey(10) & 0xff # Press 'ESC' for exiting video

if k == 27:

break

# Do a bit of cleanup

print("\n [INFO] Exiting Program and cleanup stuff")

cam.release()

cv2.destroyAllWindows()