**MOUNT KENYA UNIVERSITY**

****

**SCHOOL OF ENGINEERING, ENERGY AND THE BUILT ENVIRONMENT**

**DEPARTMENT OF ENGINEERING**

**PROJECT TITLE: FIRE DETECTION SYSTEM.**

**PRESENTED BY: FELIX KIPKIRUI**

**REG NO: BTCES/2018/85527**

**This project is submitted in partial fulfillment of requirement for the Mount Kenya**

**University award of Degree in Bachelor of Technology in Computer and Electronics System**

**DECLARATION**

I hereby declare that this project report bases on my original work except for citations and quotation, which have been acknowledged duly. I also declare that it has not been previously and submitted concurrently for any other degree award at Mount Kenya University.

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name: FELIX KIPKIRUI

Registration No.: BTCES/2018/85527

ID No.: 38610925

Date: .

**SUPERVISOR**

I the undersigned, do hereby certify that this is a true report for the project undertaken by the above-named student under my supervision and that it has been submitted to Mount Kenya University with my approval

Signature……………………………………………………. Date…………………………….

**DEDICATION**

I would like to dedicate this project to my parents who have supported me through all of my studies financially and also through their words of encouragement has been a blessing to my life. It was also a great

**ACKNOWLEDGEMENTS**

I would like to thank everyone who has contributed to the successful completion of this project. I would like to express my gratitude to my project supervisor, Mr. Raymond Nyaata for his invaluable advice, guidance and his enormous patience throughout the development of the project. In addition, I would also like to express my gratitude to my loving parents and friends who had helped and given me encouragement

**ABSTRACT**

Fire is one of the most dangerous and destructive disasters that can occur in anyplace. Authorities would be able to detect and put out fires before they became out of hand if they had a means to detect fire. The majority of existing fire detection systems, on the other hand, rely on temperature or smoke sensors, which take time to respond. Furthermore, these systems are expensive and ineffective if the fire is located distant from the detectors. As a result, I considered alternatives such as computer-vision-based approaches. One of the most cost-effective techniques would be to utilize surveillance cameras to detect fires and alert the appropriate parties. The proposed project work proposes a way for monitoring fires anywhere within camera range using surveillance cameras. The proposed method for fire detection in video pictures is based on the color and motion aspects of fire in this research. Color segmentation is employed in this method. The method locates the moving region's boundary in the color segmented image and estimates the number of fire pixels in that area. Then, based on this methodology, a fire detection system is built to efficiently detect fire and save lives and property from fire hazards.

Table of Contents

[CHAPTER 1 8](#_Toc103246412)

[1.1 INTRODUCTION. 8](#_Toc103246413)

[1.2 PROBLEM STATEMENT. 9](#_Toc103246414)

[1.3OBJECTIVES. 9](#_Toc103246415)

[1.3.1 Main objective 9](#_Toc103246416)

[1.3.2 Specific objectives 9](#_Toc103246417)

[1.4 JUSTIFICATION. 9](#_Toc103246418)

[1.5 APPLICATION FLOW CHART. 10](#_Toc103246419)

[CHAPTER 2 12](#_Toc103246420)

[2.1 LITERATURE REVIEW. 12](#_Toc103246421)

[2.1.1 What is fire? 12](#_Toc103246422)

[2.1.2 Fire triangle. 13](#_Toc103246423)

[2.1.3 Fire Detection Systems currently in use. 14](#_Toc103246424)

[2.1.4 Why Fire Detection System Based on Computer Vision. 19](#_Toc103246425)

[CHAPTER 3 20](#_Toc103246426)

[3.1 SYSTEM DESIGN. 20](#_Toc103246427)

[3.1.1 Steps for fire detection. 20](#_Toc103246428)

[3.1.2 Installing all the necessary libraries and modules. 21](#_Toc103246429)

[3.1.3 Image Acquisition. 22](#_Toc103246430)

[3.1.4 Motion Detection. 22](#_Toc103246431)

[3.1.5 Smoothing. 23](#_Toc103246432)

[3.1.6 Conversion to HSV. 24](#_Toc103246433)

[3.1.7 Fire Detection. 25](#_Toc103246434)

[3.1.8 Alarm prompts. 26](#_Toc103246435)

[3.1.9 Fire Alarm Sound. 26](#_Toc103246436)

[3.2 Code implementation of the Fire Alarm. 27](#_Toc103246437)

[3.2.1 Alarm Sound. 27](#_Toc103246438)

[3.2.2 SMS and Email Notification. 29](#_Toc103246439)

[3.2.3 Email Notification. 36](#_Toc103246440)

[CHAPTER 4 39](#_Toc103246441)

[4.1 SYSTEM IMPLEMENTATION AND TESTING. 39](#_Toc103246442)

[4.1.1 Selecting method of development. 39](#_Toc103246443)

[4.1.2 What is Python. 39](#_Toc103246444)

[3.1.3 What is Open Computer Vision Library (OpenCV) 39](#_Toc103246445)

[4.1.4 Why need for OpenCV Library? 40](#_Toc103246446)

[4.1.5 Tools used for Development. 40](#_Toc103246447)

[4.1.6 Important Sources. 40](#_Toc103246448)

[4.1.7 Application Structure Flow Chart. 41](#_Toc103246449)

[4.1.8 TESTING THE APPLICATION. 43](#_Toc103246450)

[CHAPTER 5 44](#_Toc103246451)

[5.1 LIMITATIONS, CONCLUSIONS AND RECOMMENDATIONS. 44](#_Toc103246452)

[5.1.1 Limitations. 44](#_Toc103246453)

[5.1.2 Conclusion. 44](#_Toc103246454)

[5.1.3 Recommendations. 44](#_Toc103246455)

[REFERENCE. 45](#_Toc103246456)

[APPENDIX. 46](#_Toc103246457)

List of Tables.

[**Table 1: Requirements Table**. 45](#_Toc103079535)

.

**List of Figures.**

[**Figure 1:Application Flow Chart** 10](file:///C:\Users\felix\OneDrive\Documents\Fire%20Detection%20System%20Report.docx#_Toc103206512)

[**Figure 2: Fire Triangle** 13](#_Toc103206513)

[**Figure 3: Heat Detector.** 14](#_Toc103206514)

[**Figure 4:Light Scattering Smoke Detector.** 15](#_Toc103206515)

[**Figure 5:Light Obscuring Smoke Detector.** 16](#_Toc103206516)

[**Figure 6:Carbon Monoxide Detector**. 17](#_Toc103206517)

[**Figure 7:Multi-Sensor Detector.** 17](#_Toc103206518)

[**Figure 8:Manual Call Points.** 18](#_Toc103206519)

[**Figure 9 Installing libraries and modules.** 21](#_Toc103206520)

[**Figure 10:Image Acquisition.** 22](#_Toc103206521)

[**Figure 11:Motion Detection.** 22](#_Toc103206522)

[**Figure 12:Smoothing.** 23](#_Toc103206523)

[**Figure 13:Conversion to HSV** 24](#_Toc103206524)

[**Figure 14:Fire Detection.** 25](#_Toc103206525)

[**Figure 15:Playsound status.** 27](#_Toc103206526)

[**Figure 16:Alarm Function.** 27](#_Toc103206527)

[**Figure 17:Calling Alarm Function.** 28](#_Toc103206528)

[**Figure 18: Africastalking login.** 29](#_Toc103206529)

[**Figure 19: Creating a team.** 30](#_Toc103206530)

[**Figure 20:Application Dashboard.** 31](#_Toc103206531)

[**Figure 21:Initializing the SDK**. 32](#_Toc103206532)

[**Figure 22:Generating API Key**. 33](#_Toc103206533)

[**Figure 23:Initializing**. 33](#_Toc103206534)

[**Figure 24:Initializing the service.** 34](#_Toc103206535)

[**Figure 25:SMS Sending function.** 34](#_Toc103206536)

[**Figure 26: Declaring a function when there is a fire.** 35](#_Toc103206537)

[**Figure 27:Email function.** 36](#_Toc103206538)

[**Figure 28:Send mail function.** 37](#_Toc103206539)

[**Figure 29:Release function.** 38](#_Toc103206540)

[**Figure 30:Application structure flow chart** 41](file:///C:\Users\felix\OneDrive\Documents\Fire%20Detection%20System%20Report.docx#_Toc103206541)

# CHAPTER 1

## 1.1 INTRODUCTION.

Fire detection based on computer vision has the potential to be beneficial in situations when traditional methods aren't feasible. Fires are distinguished from these visible stimuli by their visual features such as brightness, color, spectral texture, spectral flicker, and edge trembling. Many algorithms for fire detection make use of these qualities. In addition to typical UV and infrared sampling, most traditional fire detection techniques rely on particle sampling, temperature sampling, relative humidity sampling, air transparency testing, and smoke analysis.

## 1.2 PROBLEM STATEMENT.

Being too close to a fire can result in burns, and the smoke released by the fires can result in asphyxiation. Fire, like other natural calamities, damages property. When an object is set on fire, it can release hazardous gases, and if a big amount of flammable material is placed on fire, it can cause a massive explosion. Overheating electrical appliances, candles, smoking, cooking appliances, and heating appliances are all common fire threats. These areas should be checked for fire more frequently since they have a proclivity for causing fires and are regular places where fires occur. There are already a variety of smoke detectors available on the market. Optical smoke detectors, in which a light travel from one point to another and is dispersed by smoke; ionization smoke detectors, in which smoke particles prevent current from flowing inside a circuit; and air sampling smoke detectors, in which the air is sampled over a period of time in order for a large system to detect trace amounts of smoke. It should be noted that these detectors are rather costly to install and are normally only used in enclosed locations. Another issue with them is that they take a long time to respond since the smoke and heat must dissipate first. As a result, the goal of this project is to develop a fire detection system that can be utilized everywhere, especially in open places, and that can provide a timely response to a fire.

## 1.3OBJECTIVES.

### 1.3.1 Main objective

To develop a system that can detect fires.

### 1.3.2 Specific objectives

To develop a system that alerts users when there is a fire.

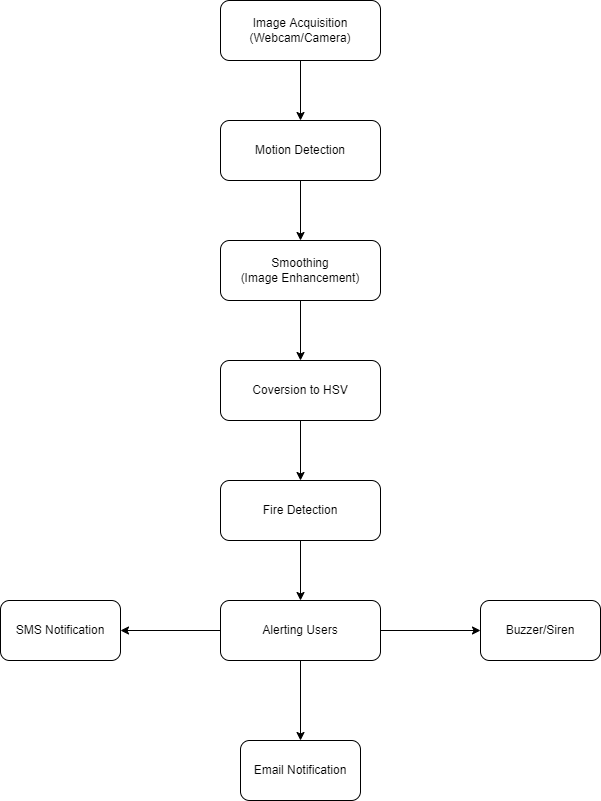
To develop a system that can be used both indoors and outdoors.

To develop an affordable fire detection system in real time.

## 1.4 JUSTIFICATION.

Fire detection using computer vision is an emerging technology that can help prevent calamities brought to by fire.

## 1.5 APPLICATION FLOW CHART.

******

**Figure 1:Application Flow Chart**

1. Image acquisition-The video camera is turned on first, and the first frame captured is used as the initial background image. The camera then continuously captures frames.
2. Motion detection-The system examines subsequent frames to see if there is any motion. Background subtraction is used for this.
3. Smoothing-Image quality is reduced due to noise in a variety of situations. So, prior to any further processing, the image is cleaned of noise in this step.
4. Conversion to HSV-The input is in RGB form, and then RGB form is converted to HSV form. Then the range representing HSV form of fire is applied to detect only fire characteristics.
5. Fire detection-The system counts number of ﬁre pixels. If this number is higher than some threshold value, there is a ﬁre with very high probability.
6. Alerting the users- this implemented through a buzzer/siren, SMS or an email notification.

# CHAPTER 2

## 2.1 LITERATURE REVIEW.

### 2.1.1 What is fire?

Fire is the visible effect of the process of combustion – a special type of chemical reaction. It occurs between oxygen in the air and some sort of fuel. The products from the chemical reaction are completely different from the starting material. The fuel must be heated to its ignition temperature for combustion to occur. The reaction will keep going as long as there is enough heat, fuel and oxygen. Combustion is when fuel reacts with oxygen to release heat energy. Combustion can be slow or fast depending on the amount of oxygen available. Combustion that results in a flame is very fast and is called burning. Combustion can only occur between gases. Fuels can be solids, liquids or gases. During the chemical reaction that produces fire, fuel is heated to such an extent that (if not already a gas) it releases gases from its surface. Only gases can react in combustion. The heat generated by the reaction is what sustains the fire. The heat of the flame will keep remaining fuel at ignition temperature. The flame ignites gases being emitted, and the fire spreads. As long as there is enough fuel and oxygen, the fire keeps burning. There are two types of combustion, complete and incomplete combustion. In complete combustion, the burning fuel will produce only water and carbon dioxide (no smoke or other products). The flame is typically blue. For this to happen, there needs to be enough oxygen to combine completely with the fuel gas. Many of us use methane gas (CH4), commonly known as natural gas, at home for cooking. When the gas is heated (by a flame or spark) and if there is enough oxygen in the atmosphere, the molecules will break apart and reform totally as water and carbon dioxide. In incomplete combustion, it produces products such as carbon (C) and carbon monoxide (CO) as well as water and carbon dioxide. The burning flame is typically yellow or orange and there is smoke. If there is not enough oxygen available during a chemical reaction, incomplete combustion occurs, and products such as carbon (C) and carbon monoxide (CO) as well as water and carbon dioxide are produced. Less heat energy is released during incomplete combustion than complete combustion. In incomplete combustion, the burning flame is typically yellow or orange and there is smoke.

### 2.1.2 Fire triangle.

The fire triangle, or combustion triangle, is the three components needed to ignite and sustain a fire. The three ingredients of a fire triangle are; heat, fuel and oxygen. If just one of these components is removed, the fire triangle will collapse and the fire will be extinguished.

The components of the fire triangle in details:

#### 2.1.2.1 Heat

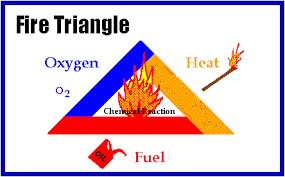
A source of heat is required in order for ignition to occur, and different materials have different ‘flash points’ e.g., the lowest temperature at which they ignite. Unfortunately, combustion reactions also produce heat as they burn, further increasing the temperature of the fuel. For some types of fire, the heat can be cooled with the application of water.

#### 2.1.2.2 Fuel

A fire cannot begin if there is no material to burn. Homes and businesses are full of flammable materials, such as paper, oil, wood and fabrics. Any of these can serve as a fuel for a fire. Some materials burn more easily than others. Fuels are probably the most difficult ‘side’ of the fire triangle to remove, so it’s wise to store them appropriately to prevent them becoming a fire hazard.

#### 2.1.2.3 Oxygen

To sustain the combustion reaction, oxygen (or an oxidizing agent) is needed, as it reacts with the burning fuel to release heat and CO2. Earth’s atmosphere consists of 21% oxygen, so there is plenty available to trigger a fire if the other two components are present. Fire blankets and certain fire extinguishers remove the oxygen ‘side’ of the triangle by removing it or displacing it, causing suffocation and thereby ceasing the combustion reaction.



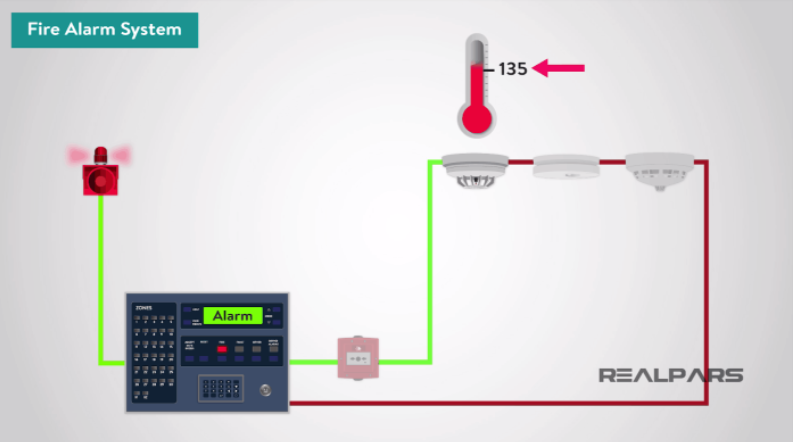
**Figure 2: Fire Triangle**

### 2.1.3 Fire Detection Systems currently in use.

Fire alarms are found in Offices, Factories, and public buildings, they are a part of our everyday routine, but are often overlooked until there is an emergency at which point, they might just save our lives. Whatever the method of detection is, if the alarm is triggered, sounders will operate to warn people in the building that there may be a fire and to evacuate. The fire alarm system may also incorporate a remote signal system which could then alert the fire brigade via a central station. At the core of a fire alarm system are the detection devices, from sophisticated intelligent smoke detectors to simple manually operated break glass units, there are a wide array of different types, but we can divide them into groups including:

#### 2.1.3.1 Heat Detectors.

Heat detector can either work on a fixed temperature basis, where it will trigger an alarm if the temperature exceeds a pre-set value or they can work on the rate of change in temperature. Commonly Heat detectors work in a similar way to an electrical fuse, the detectors contain a eutectic alloy which is heat sensitive when a certain temperature is reached the alloy turns from a solid to a liquid which in turn triggers the alarm.



**Figure 3: Heat Detector.**

#### 2.1.3.2 Smoke Detectors.

There are three basic types of smoke detectors including:

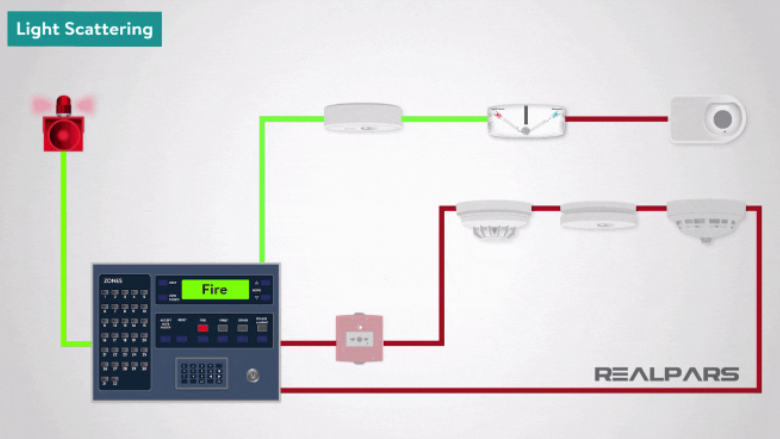
1. Ionization Light
2. Scattering
3. Light Obscuring

#### 2.1.3.3 Ionization Smoke Detector.

Ionization Smoke detector generally contains two chambers. The first is used as a reference to compensate for changes in ambient temperature, humidity or pressure.The second chamber contains a radioactive source, usually alpha particle, which ionizes the air passing through the chamber where a current flows between two electrodes.When smoke enters the chamber, the current flow decreases. This drop in current flow is used to initiate an alarm.

#### 2.1.3.4 Light Scattering Smoke Detector.

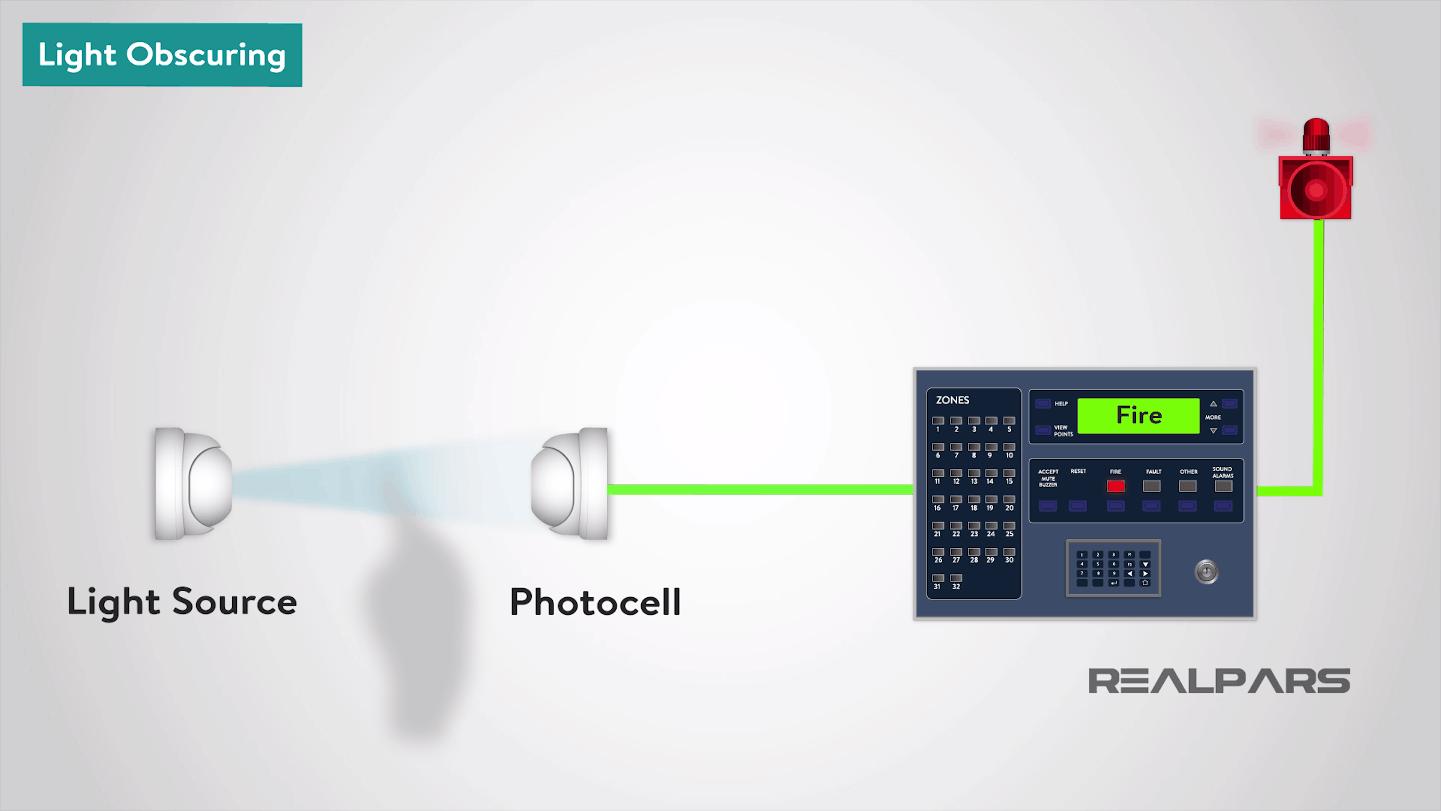
The light scattering smoke detector operates on the Tyndall effect; a photocell and light source are separated from each other by a darkened chamber such that the light source does not fall on the photocell. The passage of smoke into the chamber causes the light from the source to be scattered and fall on the photocell. The photocell output is being used to initiate an alarm.



**Figure 4:Light Scattering Smoke Detector.**

#### 2.1.3.5 Light Obscuring Smoke Detector.

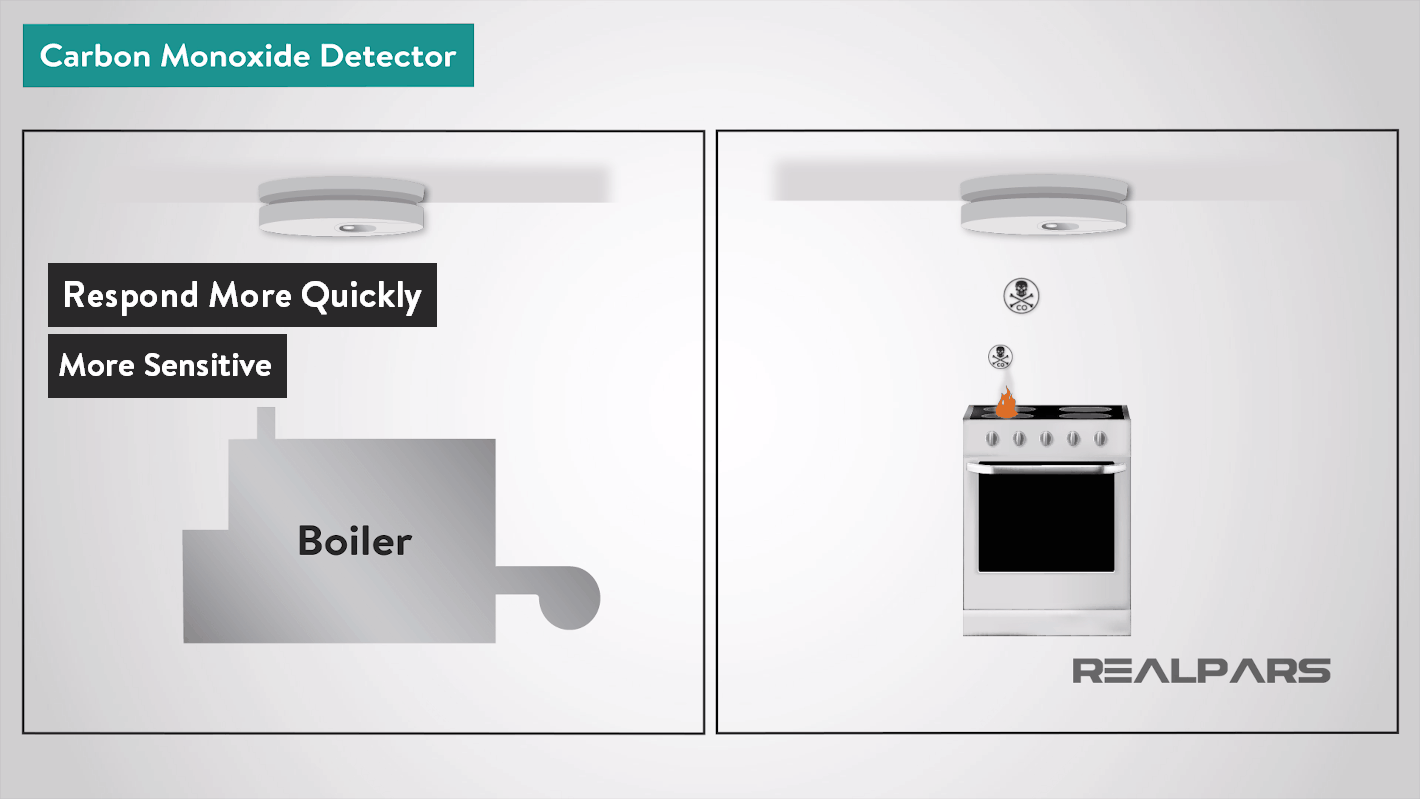
In the Light obscuring smoke detector, smoke interferes with a light beam between a light source and photocell. The photocell measures the amount of light it receives. The variation in photocell output, is being used to initiate an alarm. This type of fire detection equipment can be used to protect large areas with the light source and photocell positioned some distance apart.



**Figure 5:Light Obscuring Smoke Detector.**

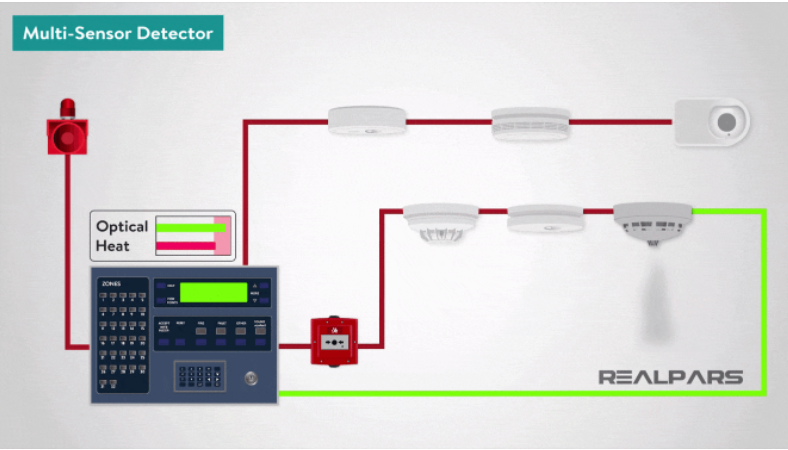
#### 2.1.3.6 Carbon Monoxide Detectors.

Carbon monoxide detectors are known also as CO fire detectors are electronic detectors used to indicate the outbreak of fire by sensing the level of carbon monoxide in the air. Carbon monoxide is a poisonous gas produced by combustion. In this instance, these detectors are not the same as Carbon monoxide detectors used in the home for protecting residents against carbon monoxide produced by incomplete combustion in appliances such as gas fires or boilers. Carbon Monoxide fire detectors use the same type of sensor as those in the home but are more sensitive and respond more quickly. Carbon monoxide detectors have an electrochemical cell, which senses carbon monoxide, but not smoke or any other combustion products.

**Figure 6:Carbon Monoxide Detector**.

#### 2.1.3.7 Multi-Sensor Detectors.

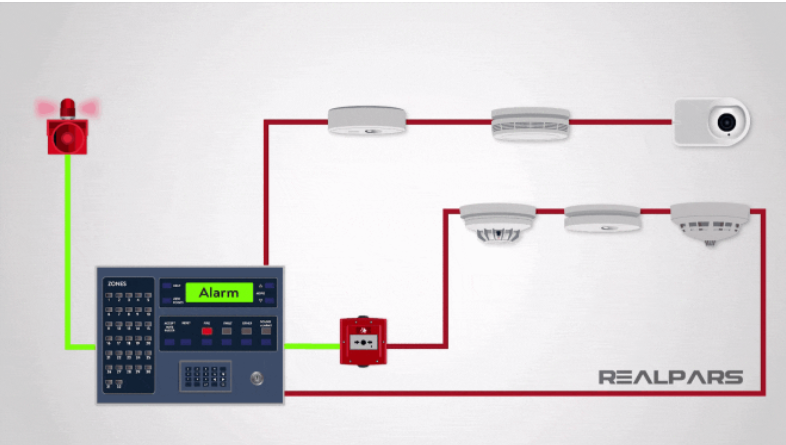
The Multi-sensor detectors combine inputs from both optical and heat sensors and process them using a sophisticated algorithm built into the detector circuitry. When polled by the control panel the detector returns a value based on the combined responses from both the optical and heat sensors. They are designed to be sensitive to a wide range of fires.



**Figure 7:Multi-Sensor Detector.**

#### 2.1.3.8 Manual Call Points.

A Manual Call Point or Break Glass Call Point is a device which enables personnel to raise the alarm by breaking the frangible element on the fascia; this then triggers the alarm.



**Figure 8:Manual Call Points.**

Limitations of existing systems are:

1. Only after detection of smoke, the fire is detected.
2. Even if there is any fire, the smoke may be generated quite later after burning the surroundings. For some fires, smoke may not be generated.
3. It takes long time for the smoke detectors to detect the smoke.
4. Surrounding materials will be burnt till the next precautionary measure is taken.

### 2.1.4 Why Fire Detection System Based on Computer Vision.

One of humanity's oldest and most pressing problems is the detection of fires. Several approaches and methods for implementing a fire detection system have been proposed to date. Many studies are still being conducted to overcome the limitations of existing systems. The methods used to detect fires differ greatly. Video-based fire detection systems can be useful for detecting fire in covered areas such as auditoriums, tunnels, and atriums, where conventional chemical fire sensors cannot provide quick responses to fire. In addition, closed circuit television (CCTV) surveillance systems are now installed in a variety of public places to monitor both indoors and outdoors. Such systems may gain early fire detection capability by utilizing fire detection software that processes the outputs of CCTV cameras in real-time.

# CHAPTER 3

## 3.1 SYSTEM DESIGN.

With everything set up, I started work on the code. In my vision based ﬁre detection system, at ﬁrst the video is captured. Since the project focus on a low-cost solution, I used my webcam to capture the video. Then the captured video is converted to suitable color space. Noise reduction is performed to the video. Then based on the detection algorithm various information of the image is analyzed. Color and motion information of the video is analyzed in almost every ﬁre detection system. In this approach, color and motion information are computed from video sequences to detect ﬁre. According to RGB color information of pixels, ﬁre colored pixel are detected. Fire colored pixels are possible in the ﬁre regions. To be certain about ﬁre, number of ﬁre pixel variation in subsequent frames are calculated. If variation is above some level, ﬁre is detected. After the detection a buzzer goes off and a notification is sent to the necessary parties.

### 3.1.1 Steps for fire detection.

In this system my approach was to come up with a detection system that uses color segmentation. The steps include:

1. Image Acquisition.
2. Motion Detection.
3. Smoothing.
4. Conversion to HSV.
5. Fire Detection.
6. Alerting the users.

### 3.1.2 Installing all the necessary libraries and modules.

import cv2

import numpy as np

import smtplib  # for sending email

import playsound

import threading

import africastalking

**Figure 9 Installing libraries and modules.**

CV2 is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more.

NumPy can be used to perform a wide variety of mathematical operations on arrays. It adds powerful data structures to Python that guarantee efficient calculations with arrays and matrices and it supplies an enormous library of high-level mathematical functions that operate on these arrays and matrices.

The smtplib module defines an SMTP client session object that can be used to send mail to any internet machine with an SMTP or ESMTP listener daemon.

The playsound module is used to play sound.

Threading is used to run multiple threads (tasks, function calls) at the same time.

Africastalking is an Application Program Interface (API) used to send SMS notifications.

After installing and requiring these modules, I started using them on my codebase.

### 3.1.3 Image Acquisition.

The video camera is turned on first, and the first frame captured is used as the initial background image. The camera then continuously captures frames. The photograph is taken in RGB color space.

The procedure is given below:

import cv2

video = cv2.VideoCapture(0)

**Figure 10:Image Acquisition.**

### 3.1.4 Motion Detection.

The system examines subsequent frames to see if there is any motion. Background subtraction is used for this. Calculating the difference between an image in the sequence and the background image yields foreground objects (previously obtained). The movement of these foreground objects between frames is then determined. We can find the area where motion occurred by subtracting subsequent frames. The number of frames captured by the camera is counted. The first frame is regarded as the initial background frame. As time passes, the background image is updated in accordance with the foreground image. There are several techniques for distinguishing the foreground from the background. Because video is used, some built-in OpenCV functions are used directly to achieve faster performance.

while True:

    (grabbed, frame) = video.read()  # extract frames from the video

    if not grabbed:

        break

    frame = cv2.resize(frame, (960, 540))  # resizing the output window

**Figure 11:Motion Detection.**

### 3.1.5 Smoothing.

Image quality is reduced due to noise in a variety of situations. So, prior to any further processing, the image is cleaned of noise in this step. Smoothing is frequently used to reduce image noise or to produce a less pixelated image. The captured frames are then blurred. Convert the grayscale frame to a Gaussian Blur frame.

# applying blur to remove noises

    blur = cv2.GaussianBlur(frame, (21, 21), 0)

**Figure 12:Smoothing.**

### 3.1.6 Conversion to HSV.

The input is in RGB form, and then RGB form is converted to HSV form. Then the range representing HSV form of fire is applied to detect only fire characteristics. Fire in HSV form is then displayed. HSV color space is chosen purposely because it has ability to differ illumination information from chrominance more effectively than the other color spaces. Threshold values for the fire are loaded in to the system, as par the threshold values color detection system display result only if the fire is detected. The steps are as follows:

    # applying blur to remove noises

    blur = cv2.GaussianBlur(frame, (21, 21), 0)

    # converting the flame into hsv formats

    hsv = cv2.cvtColor(blur, cv2.COLOR\_BGR2HSV)

    # defining the colour of the fire

    lower = [18, 50, 50]

    upper = [35, 255, 255]

    lower = np.array(lower, dtype="uint8")  # convert into numpy value

    upper = np.array(upper, dtype="uint8")  # convert into numpy value

    mask = cv2.inRange(hsv, lower, upper)  # looking for the two colors

    output = cv2.bitwise\_and(frame, hsv, mask=mask)

    no\_red = cv2.countNonZero(mask)  # measuring the size of the fire

**Figure 13:Conversion to HSV**

### 3.1.7 Fire Detection.

The system counts number of ﬁre pixels. If this number is higher than some threshold value, there is a ﬁre with very high probability. This means that ﬁre is detected. Initially, the state of Fire\_Reported, that states when there is a fire, is set to 0. When fire has been reported it is set to 1. The procedure is given below:

Fire\_Reported = 0

    if (

        int(no\_red) > 15000

    ):  # if the size of the fire exceeds 1500 then fire is reported

        Fire\_Reported = Fire\_Reported + 1

**Figure 14:Fire Detection.**

### 3.1.8 Alarm prompts.

After the system has detected presence of a fire hazard people occupying premises and relevant authorities for instance fire fighters within or outside the institution need to be notified. For this, the following prompts were used:

1. Fire Alarm Sound.
2. Sending an SMS notification.
3. Email notification.

### 3.1.9 Fire Alarm Sound.

Fire alarm sounds are important as they alert us to stay from a place or situation which may put our lives in danger. The use of fire alarms is expected that people in the premise do the following:

1. Immediately evacuate the building to the outside.
2. NEVER go back to retrieve personal belongings.
3. Move away from the front of the building to allow the fire fighters and their trucks to access the building.
4. If there is an incident on the upper floors, the area underneath is the hazard zone and that is where you will be injured by falling glass and debris.

## 3.2 Code implementation of the Fire Alarm.

### 3.2.1 Alarm Sound.

I found a suitable sound of an alarm sound on YouTube and I download an mp3 sound file of it to my local storage.

3.2.1.1 Setting status of alarm.

I first import the playsound module. I then set the status of the alarm to false.

import playsound

Alarm\_Status = False

**Figure 15:Playsound status.**

#### 3.2.1.2 Alarm function.

I then initiated a function “play\_alarm\_sound\_function” such that when the Alarm\_Status is True, the sound is then played but when false, no sound is played. The function contains a while statement that only get called when true.

def play\_alarm\_sound\_function():

    while True:

        playsound.playsound("alarm-sound.mp3", True)//name of the sound extension

**Figure 16:Alarm Function.**

#### 3.2.1.3 Calling the function.

This is simply done by tapping into the threading module that will allow for multiple function calls. The result of calling this function is that the alarm sound through my laptops speakers.

        if Alarm\_Status == False:

            threading.Thread(target=play\_alarm\_sound\_function).start()

            Alarm\_Status = True

**Figure 17:Calling Alarm Function.**

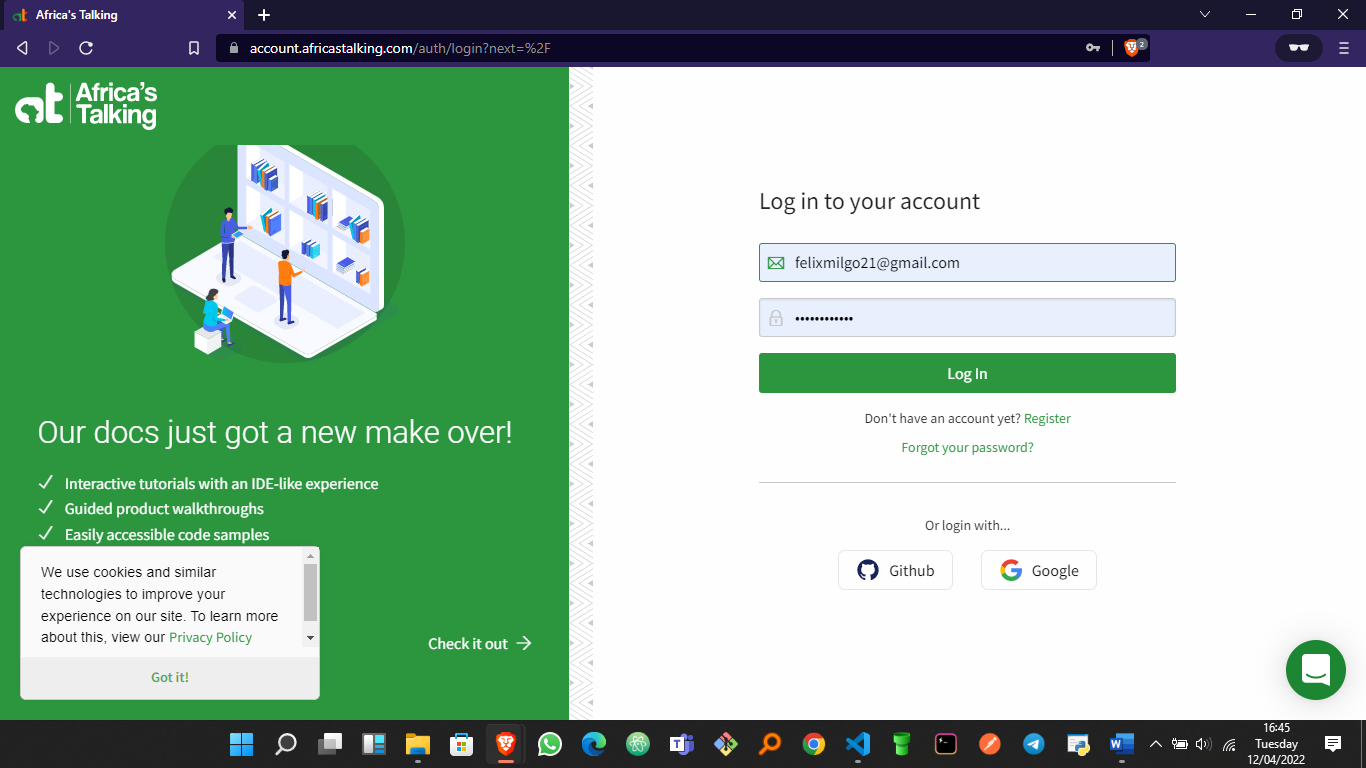
### 3.2.2 SMS and Email Notification.

The main purpose of the SMS and email is not only to inform the occupants of the building but also the relevant authorities who can act accordingly and offer help in evacuating the building, putting out the fire and also offer medical support to those who might be injured.

#### 3.2.2.1 SMS Implementation.

#### 3.2.2.2 Africastalking SDK

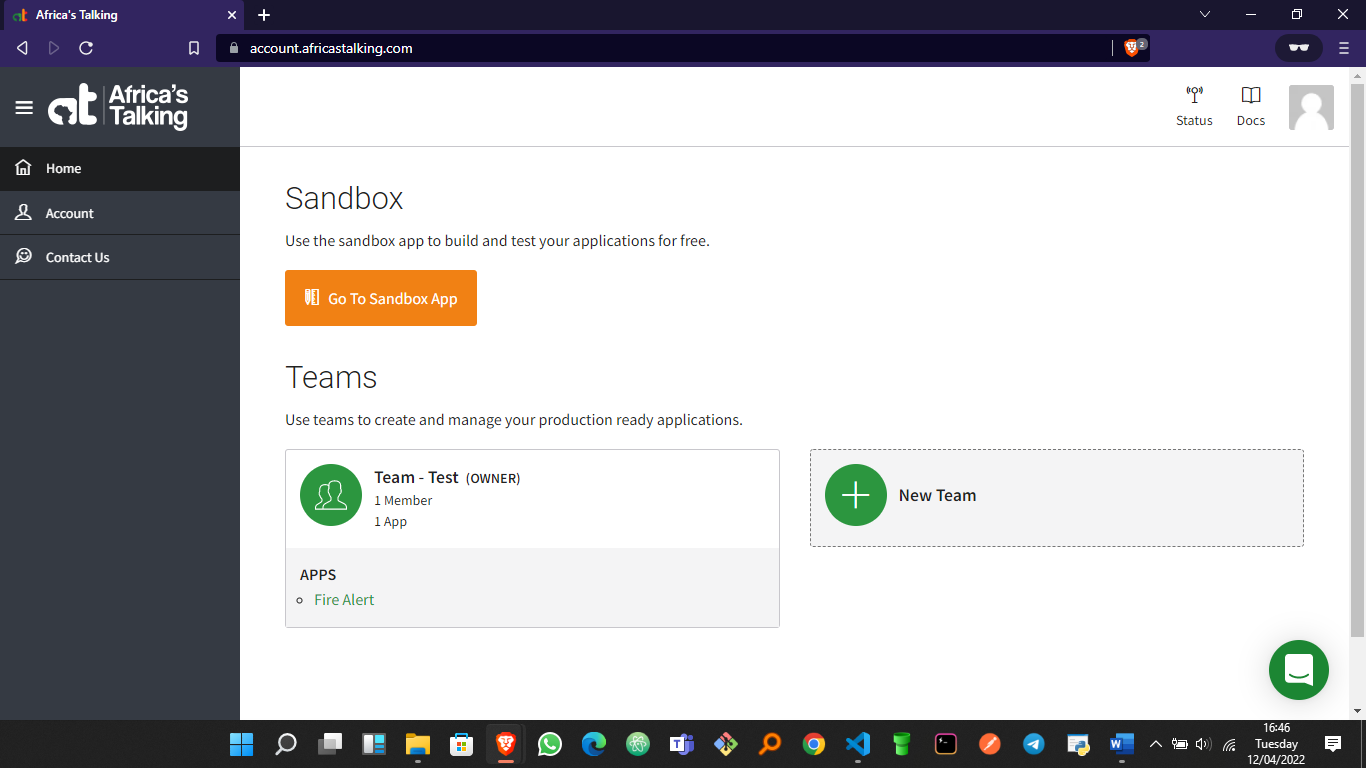
To use africastalking SDK you first need to create an account, that is an email, username and password.



**Figure 18: Africastalking login.**

#### 3.2.2.3 Creating a Team.

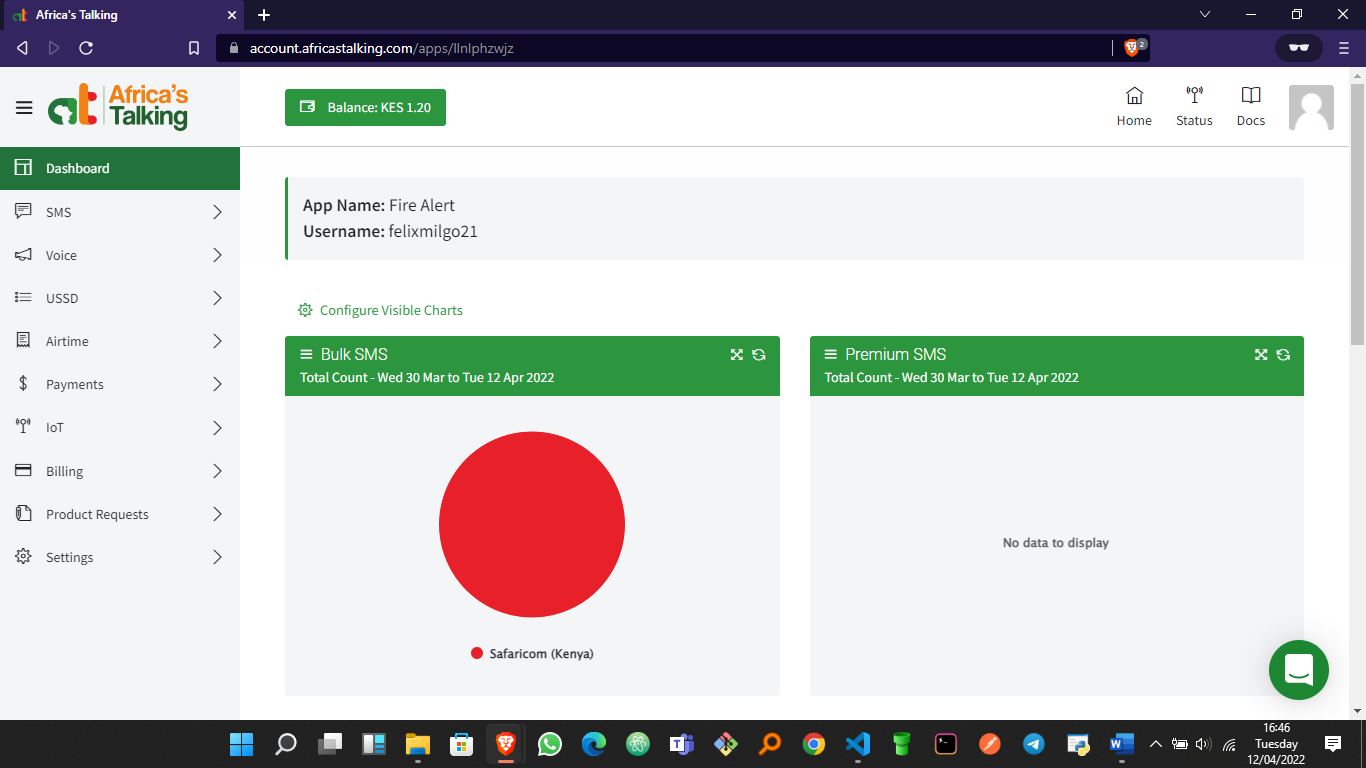
After logging in, I created a team, “Test”, which then I will use create and manage production ready applications. In my team I created the application “Fire Alert”



**Figure 19: Creating a team.**

#### 3.2.2.4 Application Dashboard.

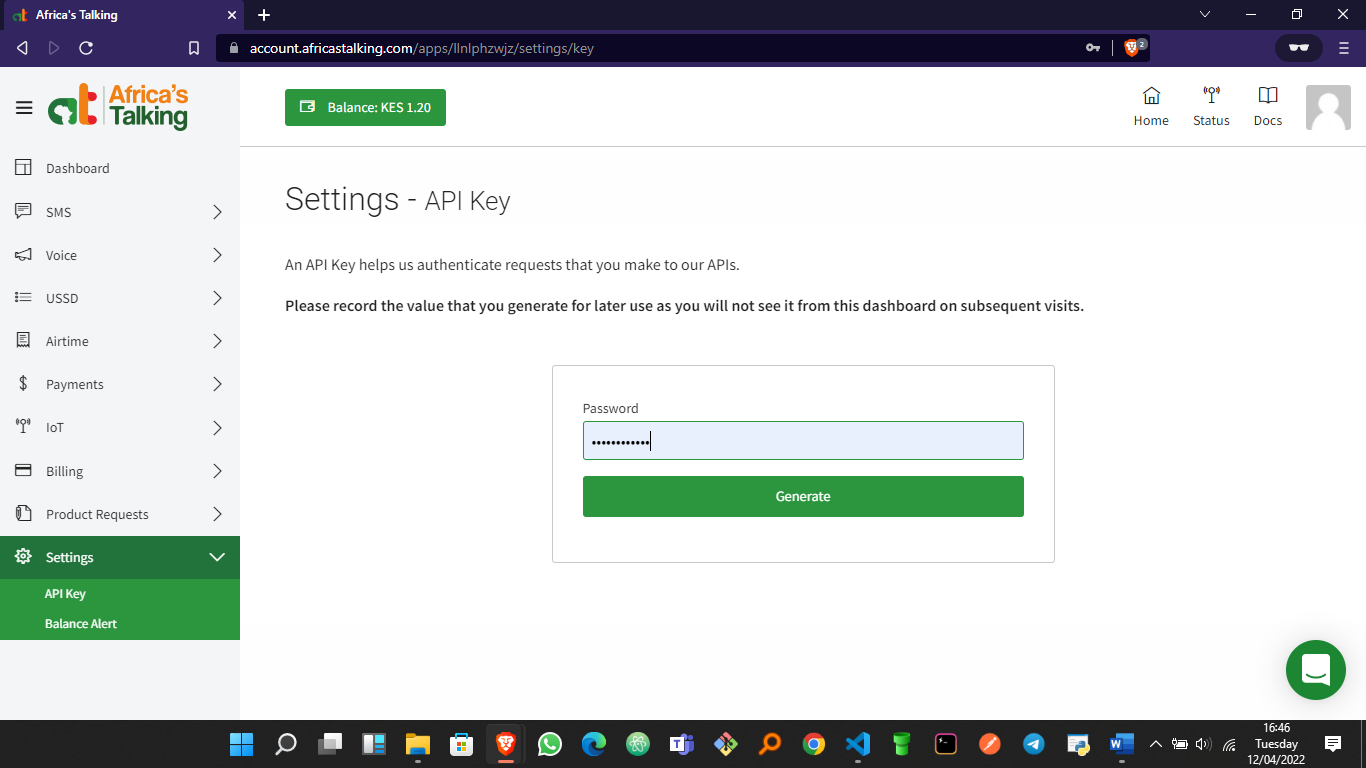
On opening my dashboard, it shows me my statistics including total number of SMS that has been sent. It also has the balance of the remaining amount of money that I still have in my account. The charges are 0.8 KSH per SMS.



**Figure 20:Application Dashboard.**

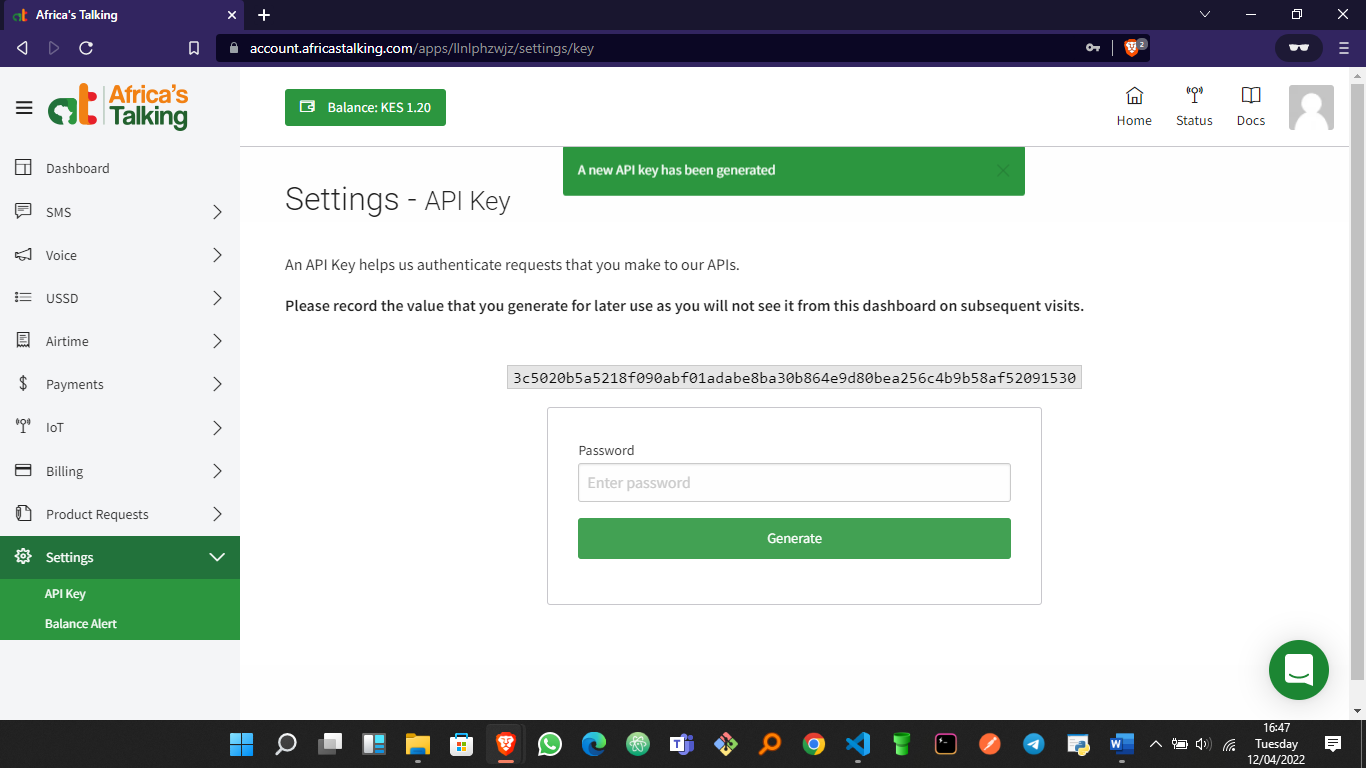
#### 3.2.2.5 Initializing the SDK.

For this, an API key is needed alongside my account username. Automatically, an API key is always created for you but for security reasons I need to generate one for myself. Inside my teams, at the bottom left there is settings upon click there is an option “API Key”. On clicking, it requires you to enter your password for which I did and then you click generate

**Figure 21:Initializing the SDK**.

#### 3.2.2.6 API Key.

A unique key is created as shown below. This key acts as an identification for its use in my appplication. I then copied it for use in my code.



**Figure 22:Generating API Key**.

#### 3.2.2.7 Initialize the SDK.

First, I imported the package africastalking the initialized it. This is where I use the API Key and my account username.

import africastalking

africastalking.initialize(

    username="felixmilgo21",

    api\_key="3c5020b5a5218f090abf01adabe8ba30b864e9d80bea256c4b9b58af52091530",

)

**Figure 23:Initializing**.

#### 3.2.2.8 Initializing the service.

Africastalking offers a wide range of services that is from USSD, Voice, Airtime, SMS and Internet of Things (IOT). For this application, the service I am using is SMS.

sms = africastalking.SMS

**Figure 24:Initializing the service.**

#### 3.2.2.9 Using the Service.

#### 3.2.2.10 Defining the function.

I created a function sending. From the documentation is as shown below. I set the recipient phone number in international format. After that I created a custom message that would be used to alert the recipient when the function is called. When there is an error, a custom message “Houston, we have a problem: {e}” that is printed whenever there is an error.

def sending():

    # Set the numbers in international format

    recipients = ["+254727541692"]

    # Set your message

    message = (

        "There is a FIRE in Diamond Heights Floor Number 2! Evacuate the building now! "

    )

    try:

        response = sms.send(message, recipients)

        print(response)

    except Exception as e:

        print(f"Houston, we have a problem: {e}")

**Figure 25:SMS Sending function.**

#### 3.2.2.11 Calling the function when there is a fire.

Initially, I set the SMS status to False so that when fire is reported, the status of SMS is set to true therefore calls the function and sends the warning alert. This works the same way as the one for the alarm sound.

    if Fire\_Reported >= 1:

        if Alarm\_Status == False:

            threading.Thread(target=play\_alarm\_sound\_function).start()

            Alarm\_Status = True

        if Email\_Status == False:

            threading.Thread(target=send\_mail\_function).start()

            Email\_Status = True

**Figure 26: Declaring a function when there is a fire.**

#### 3.2.2.12 Notification on my phone.

As soon as fire is reported, the message I receive on my mobile phone is as shown below.

### 3.2.3 Email Notification.

This works similarly to the one for SMS. The smtplib module defines an SMTP client session object that can be used to send mail to any internet machine with an SMTP or ESMTP listener daemon. I first define the function send\_mail\_function then add recipient email. After that, I login to the account I like to receive my mail from. This requires the email account name and password. After that, I then requested the server to send a mail to the stated recipient mail with a custom warning message indicated. This works when the function is called.

def send\_mail\_function():

    recipientEmail = "felixmilgo21@gmail.com"

    recipientEmail = recipientEmail.lower()

    try:

        server = smtplib.SMTP("smtp.gmail.com", 587)

        server.ehlo()

        server.starttls()

        server.login("ropkibetphilemon22@gmail.com", "+254728624952")

        server.sendmail(

            "felixmilgo21@gmail.com",

            recipientEmail,

            "Warning A Fire Accident has been reported on your premises.",

        )

        print("sent to {}".format(recipientEmail))

        server.close()

    except Exception as e:

        print(e)

**Figure 27:Email function.**

#### 3.2.3.1 Sending the mail when there is fire.

I set the status of email to false such that when I indicate it to be true in calls the function send\_mail\_function. The status changes to be true when there is fire reported.

import smtplib  # for sending email

Email\_Status = False

def send\_mail\_function():

    recipientEmail = "felixmilgo21@gmail.com"

    recipientEmail = recipientEmail.lower()

    try:

        server = smtplib.SMTP("smtp.gmail.com", 587)

        server.ehlo()

        server.starttls()

        server.login("ropkibetphilemon22@gmail.com", "+254728624952")

        server.sendmail(

            "felixmilgo21@gmail.com",

            recipientEmail,

            "Warning A Fire Accident has been reported on your premises ",

        )

        print("sent to {}".format(recipientEmail))

        server.close()

    except Exception as e:

        print(e)

    if Fire\_Reported >= 1:

        if Alarm\_Status == False:

            threading.Thread(target=play\_alarm\_sound\_function).start()

            Alarm\_Status = True

        if Email\_Status == False:

            threading.Thread(target=send\_mail\_function).start()

            Email\_Status = True

        if Sms\_Status == False:

            threading.Thread(target=sending).start()

            Sms\_Status = True

if cv2.waitKey(1) & 0xFF == ord("q"):

        break

**Figure 28:Send mail function.**

#### 3.2.3.2 Terminating the process.

When the notifications have been sent and fire hazard cleared, the code breaks, that is it starts again the process of surveillance. cv2.waitKey(1) will display a frame for 1 ms, after which display will be automatically closed. Since the OS has a minimum time between switching threads, the function will not wait exactly 1 ms, it will wait at least 1 ms, depending on what else is running on the computer at that time. Python OpenCV destroyAllWindow() function allows me to destroy all windows at any time. It doesn't take any parameters and doesn't return anything.

    if Fire\_Reported >= 1:

        if Alarm\_Status == False:

            threading.Thread(target=play\_alarm\_sound\_function).start()

            Alarm\_Status = True

        if Email\_Status == False:

            threading.Thread(target=send\_mail\_function).start()

            Email\_Status = True

        if Sms\_Status == False:

            threading.Thread(target=sending).start()

            Sms\_Status = True

    if cv2.waitKey(1) & 0xFF == ord("q"):

        break

cv2.destroyAllWindows()

video.release()

**Figure 29:Release function.**

# CHAPTER 4

## 4.1 SYSTEM IMPLEMENTATION AND TESTING.

### 4.1.1 Selecting method of development.

As technology advances, one of the frontiers and potentially revolutionary technologies in computer science is computer vision. It is a subset of machine learning, which is a subset of Artificial Intelligence (AI). There are several programming languages that support Computer Vision and there is no single programming language that is best for it. It will always depend on the project’s requirements and what is the suitable programming language for developers. There is already CV support for other major programming languages like Python-OpenCV, PHP-OpenCV for PHP, ruby-OpenCV for Ruby, common-cv for Common Lisp, gocv for Go, cv-Rs for Rust, and EmguCV for C#.

Some of the conditions I had to consider before deciding on the method to use in the development:

1. Ease of coding.
2. How fast will I deliver a Minimum Viable Product (MVP)
3. Availability of learning resources.
4. My proficiency in the selected language.

### 4.1.2 What is Python.

Python is an interpreted, high-level and general-purpose programming Language. Python's design philosophy emphasizes code readability with its Notable use of significant indentation. Python because it is currently the most prevalent, mature, and well-supported among programming languages in the area of machine learning.

### 3.1.3 What is Open Computer Vision Library (OpenCV)

OpenCV is an open-source computer vision library. The library is written in Cand C++ and runs under Linux, Windows and Mac OS X. There is active development on interfaces for Python, Ruby, MATLAB, and other languages. OpenCV was designed for computational eﬀiciency and with a strong focus on real-time applications. OpenCV is written in optimized C and can take advantage of multi core processors. The OpenCV library contains over 500 functions that span many areas in vision, including factory product inspection, medical imaging, security, user interface, camera calibration, stereo vision, and robotics.

### 4.1.4 Why need for OpenCV Library?

The OpenCV Library is a way of establishing an open-source vision community that will make better use of up-to-date opportunities to apply computer vision in the growing PC environment. The software provides a set of image processing functions, as well as image and pattern analysis functions. The functions are optimized for Intel architecture processors, and are particularly eﬀective at taking advantage of MMX technology.

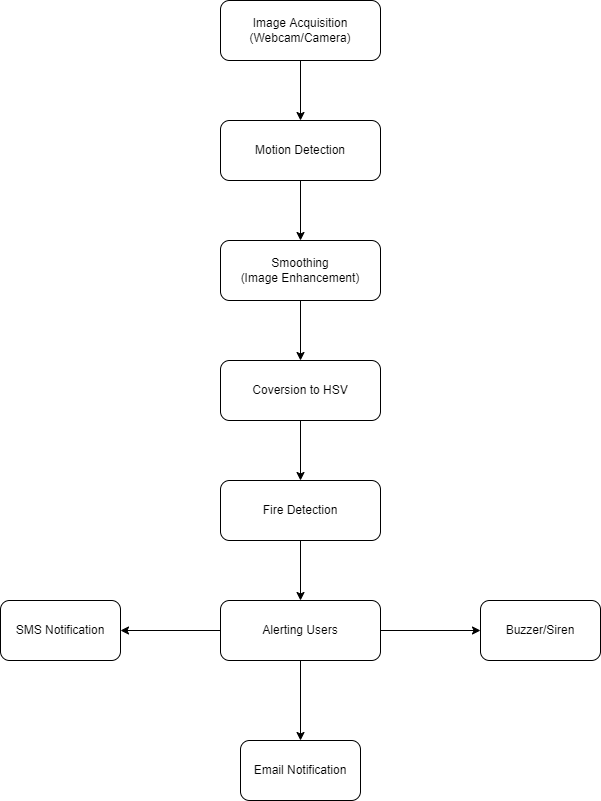
### 4.1.5 Tools used for Development.

1. Visual Studio Code.
2. Git/GitHub.
3. Python 3.10.0.
4. Open Computer Vision Library (OpenCV)

### 4.1.6 Important Sources.

1. Stack Overflow – It is a question-and-answer website that I used whenever I was stuck with my code.
2. YouTube - I referred to it to learn how to implement some of the code.
3. Africastalking Documentation- I used it in developing a SMS alert system for when there is a fire.

### 4.1.7 Application Structure Flow Chart.

****

**Figure 30:Application structure flow chart**

1. Image acquisition-The video camera is turned on first, and the first frame captured is used as the initial background image. The camera then continuously captures frames.
2. Motion detection-The system examines subsequent frames to see if there is any motion. Background subtraction is used for this.
3. Smoothing-Image quality is reduced due to noise in a variety of situations. So, prior to any further processing, the image is cleaned of noise in this step.
4. Conversion to HSV-The input is in RGB form, and then RGB form is converted to HSV form. Then the range representing HSV form of fire is applied to detect only fire characteristics.
5. Fire detection-The system counts number of ﬁre pixels. If this number is higher than some threshold value, there is a ﬁre with very high probability.
6. Alerting the users- this implemented through a buzzer/siren, SMS or an email notification.

## 4.1.8 TESTING THE APPLICATION.

The testing of the application was done in various environments. Majorly I tested it under low and intense lighting. Under both conditions the application proved to work. I only made adjustments to the suitable number of pixels required to alert when there is fire.

Thankfully, after working tirelessly on handling any runtime errors, I was able to test the app against more conditions with zero fails and errors. I picked conditions to test the application basing on the current placement of CCTV cameras.

The testing phase completed successfully after a few builds and going back to development and fixing codes.

# CHAPTER 5

## 5.1 LIMITATIONS, CONCLUSIONS AND RECOMMENDATIONS.

### 5.1.1 Limitations.

The primary focus of the project was to develop a reliable and low cost ﬁre detection system with satisfactory performance. As every system have some limitations, some of them are described below.

One of the weaknesses in this system is that it cannot detect what it cannot see. If the ﬁre exists in the range of the video which is being blocked by an object, the system will not be able to detect it.

The method used also has its weaknesses, as can be seen from the results. It gives false alarm in some cases when there is no ﬁre but an extreme ﬁre colored object exists.

### 5.1.2 Conclusion.

This project proposed a fire detection algorithm which is free from sensors as the ordinary fire detection systems contain. The objective of this project was to create a system which would be able to detect fire as early as possible from a live video feed. System is expected to detect fire while it is still small and has not grown to mammoth proportions. Also, the hardware is minimal and has been already existent in places, thus saving capital. It also saves cost by getting rid of expensive temperature and heat sensors etc. Based on the results produced, the system has proven to be effective at detecting fire.

### 5.1.3 Recommendations.

1. Smoke detection along with fire detection can be added as a feature
2. System Optimization and Delay Reduction i.e., lesser latency may be achieved
3. System can be used to detect forest fires and may be embedded on a drone or any other UAV for surveillance purposes of property
4. The system can have military applications
5. The system can be used for rescue operations on land and in sea

## REFERENCE.

1. Van Rossum, G., 2007, June. Python Programming language. In *USENIX annual technical conference* (Vol. 41, No. 1, pp. 1-36).
2. Sanner, M.F., 1999. Python: a programming language for software integration and development. *J Mol Graph Model*, *17*(1), pp.57-61.
3. Zhang, D., Han, S., Zhao, J., Zhang, Z., Qu, C., Ke, Y. and Chen, X., 2009, April. Image based forest fire detection using dynamic characteristics with artificial neural networks. In *2009 International Joint Conference on Artificial Intelligence* (pp. 290-293). IEEE.
4. Kawu, A.A., Abdullahi, A., Joseph, E., Mishra, A. and Abdulrahman, A., 2020, February. MIRCS: A Mobile USSD-SMS Interactive Result Checking System for Resource-Constrained Settings. In *Proceedings of the 2020 9th International Conference on Software and Computer Applications* (pp. 264-268).
5. Joshi, P., 2015. *OpenCV with Python by example*. Packt Publishing Ltd.

## APPENDIX.

#### Requirements.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Category** | **Price(Kshs)** | **Details** |
| A laptop | Hardware | 30,000 | Preferably 8GB RAM, Intel core i5 processor |
| Lighter | Hardware | 30 | A lighter that can produce a flame. |
| Visual Studio Code | Software | free | To connect android to laptop for debugging. |
| Python | Software | free | Version 3.10.0 because of compatibility issues |

**Table 1: Requirements Table**.

This is an example of syntax used in the coding phase,

import cv2

import numpy as np

import smtplib # for sending email

import playsound

import threading

Alarm\_Status = False

Email\_Status = False

Fire\_Reported = 0

def play\_alarm\_sound\_function():

while True:

playsound.playsound('alarm-sound.mp3', True)

def send\_mail\_function():

recipientEmail = "felixmilgo21@gmail.com"

recipientEmail = recipientEmail.lower()

try:

server = smtplib.SMTP('smtp.gmail.com', 587)

server.ehlo()

server.starttls()

server.login("ropkibetphilemon22@gmail.com",

'+254728624952')

server.sendmail('felixmilgo21@gmail.com', recipientEmail,

"Warning A Fire Accident has been reported on your premises.")

print("sent to {}".format(recipientEmail))

server.close()

except Exception as e:

print(e)

video = cv2.VideoCapture(0) # bonfire.mp4 capture video

while True:

(grabbed, frame) = video.read() # extract frames from the video

if not grabbed:

break

frame = cv2.resize(frame, (960, 540)) # resizing the output window

# applying blur to remove noises

blur = cv2.GaussianBlur(frame, (21, 21), 0)

# converting the flame into hsv formats

hsv = cv2.cvtColor(blur, cv2.COLOR\_BGR2HSV)

# defining the colour of the fire

lower = [18, 50, 50]

upper = [35, 255, 255]

lower = np.array(lower, dtype="uint8") # convert into numpy value

upper = np.array(upper, dtype="uint8") # convert into numpy value

mask = cv2.inRange(hsv, lower, upper) # looking for the two colors

output = cv2.bitwise\_and(frame, hsv, mask=mask)

no\_red = cv2.countNonZero(mask) # measuring the size of the fire

if int(no\_red) > 10000: # if the size of the fire exceeds 1500 then fire is reported

Fire\_Reported = Fire\_Reported + 1

cv2.imshow("fire\_detector", output) # the output window

# turn on the alarm sound

if Fire\_Reported >= 1:

if Alarm\_Status == False:

threading.Thread(target=play\_alarm\_sound\_function).start()

Alarm\_Status = True

if Email\_Status == False:

threading.Thread(target=send\_mail\_function).start()

Email\_Status = True

if cv2.waitKey(1) & 0xFF == ord('q'):

break

cv2.destroyAllWindows()

video.release()