A PROJECT REPORT ON

**CAM MOTION DETECTOR**

**SUBMITTED BY**

**SHUBHAM MAURYA**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **TITLE** | **REMARK** |
| 1 | Objective |  |
| 2 | Introduction |  |
| 3 | Software and Hardware requirements |  |
| 4 | block diagram, methodology and modules used |  |
| 5 | Source code |  |
| 6 | Output and screenshots |  |
| 7 | Conclusion |  |

**OBJECTIVE**

The project aims to provide a user with an interface that can allow it to easily monitor any motion in a particular area with the help of a camera. it helps to set up a surveillance system easily without any effort.

This app provides an interface by which user can check when an object enters in the surveillance area and exits the surveillance area and it also captures the image of perpetrator. it helps to provide safety or help to protector our self from unwanted people.

The main objective of this application is to provide an easy to use, easy to set up and cost-efficient surveillance system.

**INTRODUCTION**

CAM MOTION DETECTOR is an interface that provides the user with a surveillance system which is powered by python3, it helps to detect any type of motion in the surveillance area with the help of python 3 library OpenCV.

this tool detects any type of motion after recording any type of motion it captures the image of the detected object and records the entering date and time of object and leaving date and time of the detected object.

**Why cam motion detector -**

* Easy to use
* Easy to set up
* Cost-efficient

**The main features of the system are –**

* A log or every motion detected
* Captures images of the perpetrator

**LOGS –**

It is a doc file which records the time interval of a detected object in the surveillance area. Whereas the time interval contains two value starts its the date and time when the object is detected and end it the date and time when the object leaves the surveillance area.

**IMAGES –**

These are the captured images the object which enters in the surveillance area the images are stored in the format of JPG.

**SOFTWARE AND HARDWARE REQUIREMENTS**

**HARDWARE REQUIREMENTS -**

* Microsoft Windows 7, 8, 8.1, 10.
* Decent amount of RAM
* Decent amount of HDD space to storing images.
* A webcam or any type of camera connected to the system.

**SOFTWARE REQUIREMENTS -**

* Python 3 is used as the primary language for the project. Python provides a robust and efficient interface for programming and can also be used with various other applications.
* The GUI is created with the help of Tkinter module which is unarguably the most efficient graphical interface module till date.

**BLOCK DIAGRAM**

**CAM MOTION DETECTOR**

**IMAGES**

**CHECK LOGS**

**START**

**log.doc present**

**open images folder**

**camera present**

NO

YES NO

YES

**error message**

**error message**

**open log.doc**

**main image**

**image to grey image**

**capture current image**

**capture main image**

**save current image and record current time**

**current! = main image**

YES

NO

**write time into log.doc**

**‘q’ pressed**

NO YES

**MODULES USED**

**TKINTER MODULE –**

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit. Creating a GUI application using Tkinter is an easy task.

**OPENCV MODULE –**

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

**DATE TIME MODULE –**

The datetime module supplies classes for manipulating dates and times. While date and time arithmetic are supported, the focus of the implementation is on efficient attribute extraction for output formatting and manipulation.

**OS MODULE –**

The OS module in python provides functions for interacting with the operating system. OS, comes under Python's standard utility modules. This module provides a portable way of using operating system dependent functionality. ... path\* modules include many functions to interact with the file system.

**SOURCE CODE**

#################################################################################

#### Import files ->

import cv2, time

import tkinter as tk

import os

from tkinter import \*

from tkinter import messagebox

from datetime import datetime

#################################################################################

##############  COMMANDS / functions

#####  start button command

def camera():

    first\_frame = None

    status\_list = [None,None]

    times = []

    image =0

    try :

        camera = cv2.VideoCapture(0)

        #video = cv2.VideoCapture(0,cv2.CAP\_DSHOW)

        try:

            os.remove("data.doc")

        except:

            pass

        while True:

            status = False

            return\_value, frame = camera.read()

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

            gray = cv2.GaussianBlur(gray,(21,21),0)

            if first\_frame is None:

                first\_frame = gray

                continue

            current\_frame = cv2.absdiff(first\_frame,gray)

            thresh\_frame = cv2.threshold(current\_frame,30,255,cv2.THRESH\_BINARY)[1]

            thresh\_frame = cv2.dilate(thresh\_frame,None, iterations=2)

            (cnts,\_) = cv2.findContours(thresh\_frame.copy(),

                                        cv2.RETR\_EXTERNAL,

                                        cv2.CHAIN\_APPROX\_SIMPLE)

            for contour in cnts:

                if cv2.contourArea(contour) < 5000 :

                    continue

                status = True

                (x,y,w,h) = cv2.boundingRect(contour)

                cv2.rectangle(frame, (x, y), (x+w, y+h), (0,255,0), 3)

            status\_list.append(status)

            if status\_list[-1] == True and status\_list[-2] == False:

                cv2.imwrite("image%04i.jpg"%image, frame)

                image += 1

                times.append(datetime.now())

            if status\_list[-1] == False and status\_list[-2] == True:

                times.append(datetime.now())

            cv2.imshow("grey frame",gray)

            cv2.imshow("current frame",current\_frame)

            cv2.imshow("threshold frame",thresh\_frame)

            cv2.imshow("color frame",frame)

            key = cv2.waitKey(1)

            if key == ord('q'):

                if status == True:

                    times.append(datetime.now())

                break

    except:

        tk.messagebox.showerror(title="CAM MOTION DETECTOR", message="Camera Not found")

    with open("data.doc","a") as data:

        for i in range(0,len(times),2):

            data.write(f'[start: {times[i]},end: {times[i+1]}]\n')

    camera.release()

    cv2.destroyAllWindows()

##### log button command

def log():

    try :

        os.system("start data.doc")

    except:

        tk.messagebox.showerror(title=None, message="Logs Not created yet.", \*\*options)

##### image button command

def images():

    import subprocess

    subprocess.Popen(f'explorer /select,"{os.getcwd()}"')

#####  about button command

def about():

    about\_win = tk.Toplevel()

    about\_win.iconbitmap(r"i

cons/icon.ico")

    about\_win.geometry('300x400')

    about\_win.resizable(width=False,height=False)

    about\_win.title("ABOUT")

    about\_img = tk.PhotoImage(file='icons/icon.png')

    about\_con = tk.Label(about\_win,image=about\_img)

    about\_con .image = about\_img

    about\_con.pack(side=tk.TOP,pady=10)

    about\_title = tk.Label(about\_win,text="CAM MOTION DETECTOR\n")

    about\_title.pack(side=tk.TOP,pady=10)

    about\_des = tk.Label(about\_win,text="Version : 1.0.0\n \

Created By : SHUBHAM MAURYA")

    about\_des.pack(fill=tk.X)

#################################################################################

############# GUI PROGRAMING  #############

################# ROOT SETTING

root = Tk()

root.geometry('600x400')

root.title("CAM MOTION DETECTOR")

root.configure(bg="white")

root.resizable(width=False,height=False)

root.iconbitmap(r"icons/icon.ico")

################ BUTTONS

##### info button

info\_button\_image = tk.PhotoImage(file='icons/info.png')

info\_button = tk.Button(root,width=40,height=40,bg="white",image=info\_button\_image,command = about)

info\_button.place(x=540,y=15)

##### start button

start\_button\_image = tk.PhotoImage(file='icons/run2.png')

start\_button = tk.Button(root,width=100,height=40,bg="white",fg="green",text ="START",image=start\_button\_image, compound="left", command = camera)

start\_button.place(x=250,y=100)

##### log button

log\_button\_image = tk.PhotoImage(file='icons/logs2.png')

log\_button = tk.Button(root,width=140,height=40,bg="white",fg="#39A0FB",text ="CHECK LOGS",image=log\_button\_image, compound="left", command = log)

log\_button.place(x=130,y=180)

##### image button

images\_button\_image = tk.PhotoImage(file='icons/images.png')

images\_button = tk.Button(root,width=140,height=40,bg="white",fg="#546E7A",text ="IMAGES",image=images\_button\_image, compound="left", command = images)

images\_button.place(x=330,y=180)

##### Note

label = tk.Label( root,bg="white",fg="red",height=2,text="\*Press 'q' to quit the camera window", relief=RAISED )

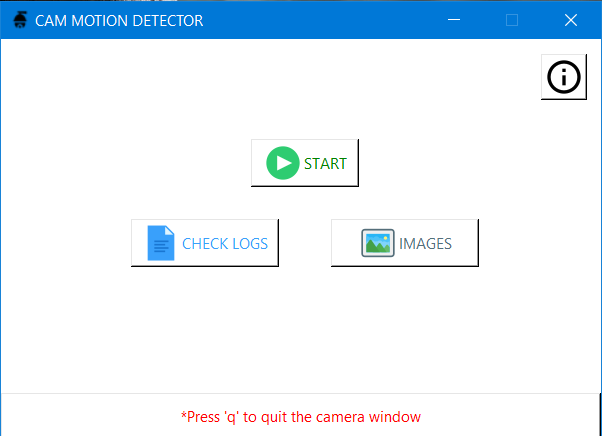
label.pack(side=tk.BOTTOM,fill=tk.X)

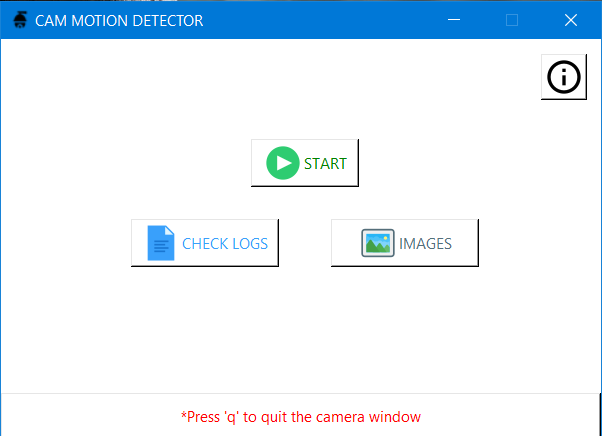
### ending

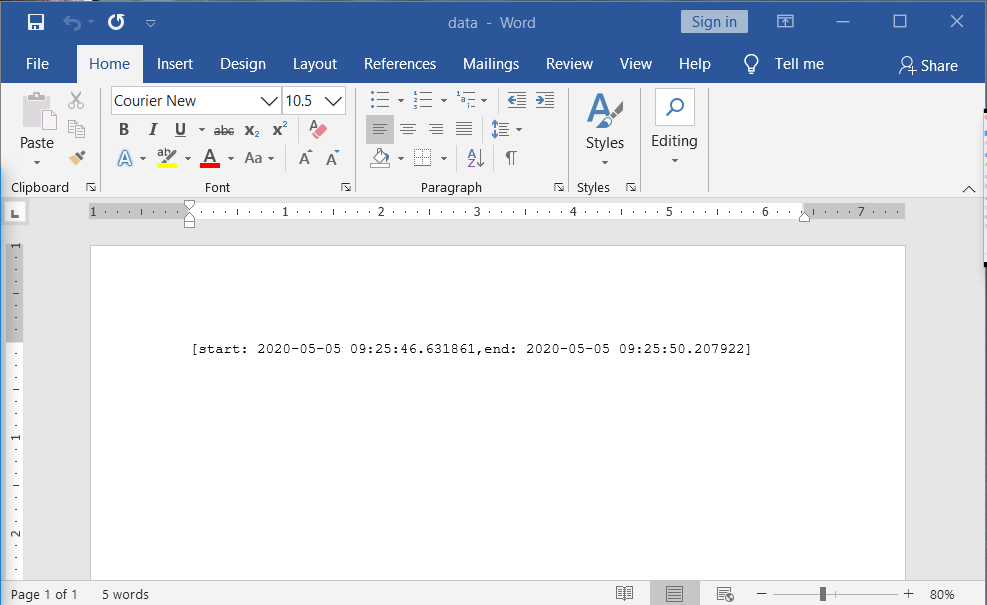
root.mainloop()

**OUTPUT AND SCREENSHOTS**

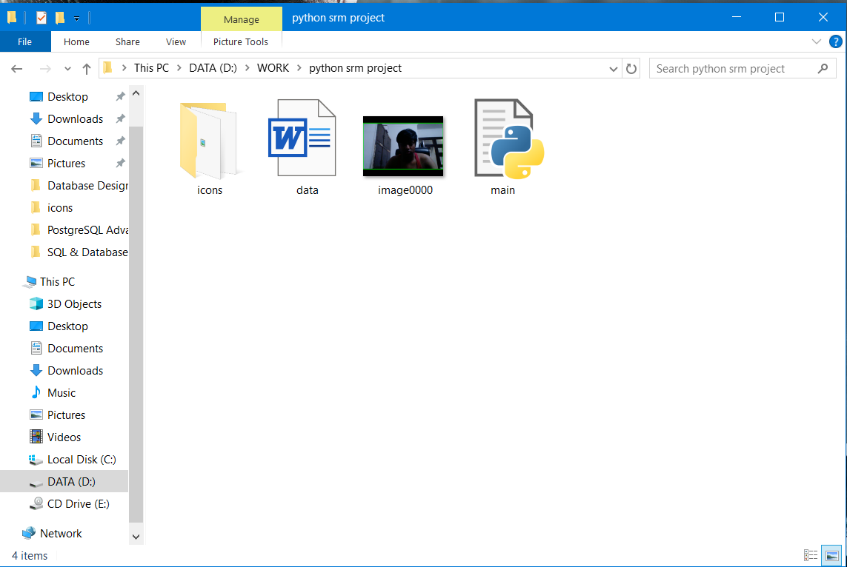
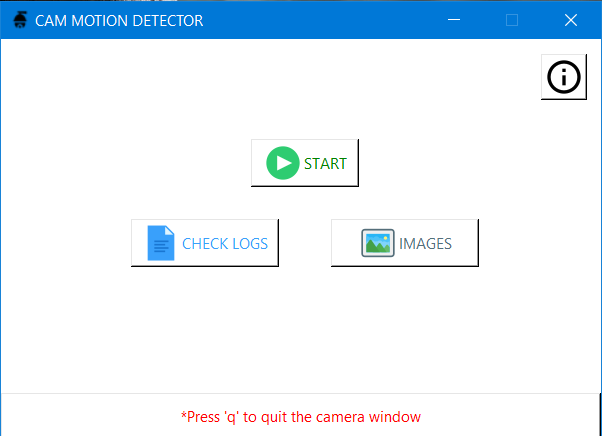
**SCREENSHOTS**

* **Cam motion detector**
* **Log button**

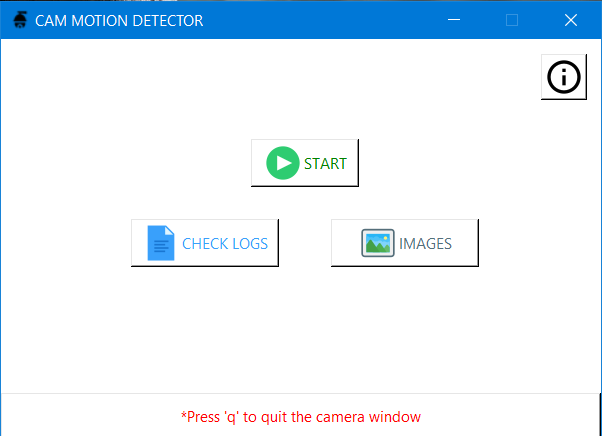
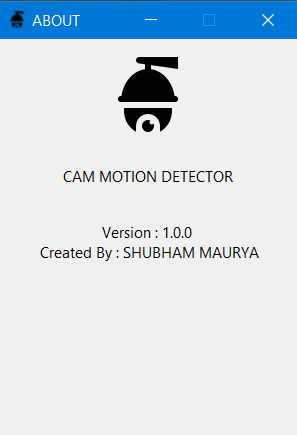




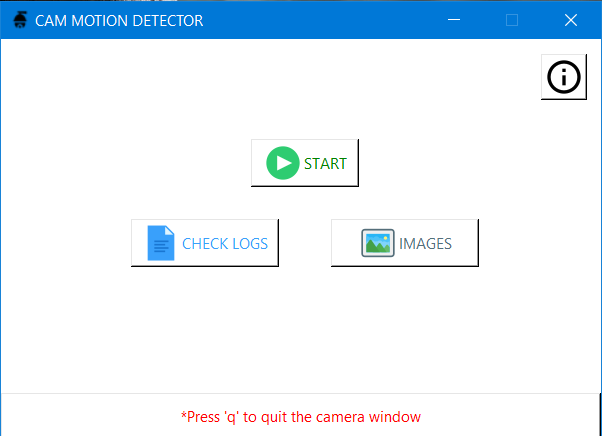
* **Image button**

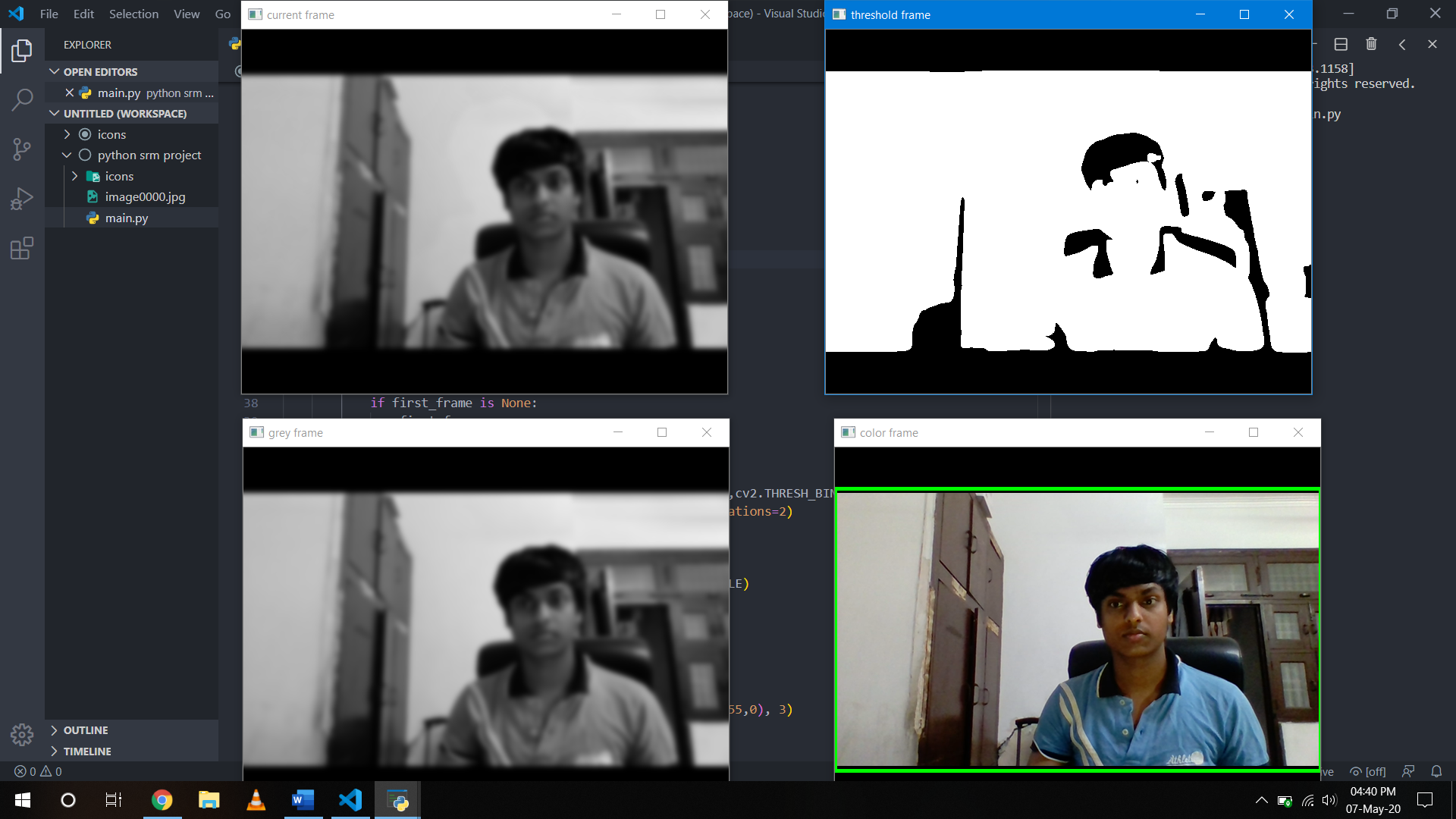


* **About/info button**



* **Start button**





**CONCLUSION**

With this report we conclude that my project cam motion detector has

been completed successfully. This project at current stage of completion can perform every designed operation.

It does not have any memory limits; it uses the memory of the device it is being used on.

Thus, I have learned about python, integration between python and tkinter which provides the best possible graphical user interface.

This project is designed in such a way that a normal person with no background knowledge of computers can easily use it without facing any difficulty.