```
"""
                    Welcome to
        <<<< :. Third EYE Surveillance System >>>>
        By Vikas Patel @ DTech
        www.villageprogrammer.tech
"""
import sys
# sys.path.append('../')
import cv2
import imutils
import time
import tkinter as tk
from threading import Thread
from datetime import datetime

now = datetime.now()
time = now.strftime("REC %d-%m-%Y %H-%M")
# local Module
from Alert.alert import Alert
from Databases.database import SettingsDatabase


class SurveillanceSystem:

    """
                        Welcome to
            <<<< Home Surveillance System >>>>
            By Vikas Patel @ DTech
            https://www.villageprogrammer.tech
    """

    """
        This file handles the video analysis task
    """

    mode = 0

    def __init__(self,videoSource=0):
        self.cap = cv2.VideoCapture(videoSource)
        self.cap.set(3, 640)  # set Width
        self.cap.set(4, 480)  # set Height
        self.run = True
        self.alert = Alert()
        self.db = SettingsDatabase()
        self.video_name = "../Outputs/video/"+str(time) + ".avi"
```

```
        self.maxThresh = int(self.db.get_sensitivity())
        frame_width = int(self.cap.get(3))
        frame_height = int(self.cap.get(4))
        fps = 10
        # Define the codec and create VideoWriter object.The output is
        stored in      'REC xxxxxxx.avi' file.
        self.out = cv2.VideoWriter(self.video_name, cv2.VideoWriter_fourcc(
        'M', 'J', 'P', 'G'), fps,(frame_width, frame_height))

    def alert_me(self):
        Thread(target=self.alert.alertSMS).start()
        print("Alert Me")
        Thread(target=self.alert.alertSound).start()
        print("Sending Email 111")
        Thread(target=self.alert.send_email).start()
        print("Alerted From here")

    def change(self):
        global mode
        self.mode = 1

    def video_analysis(self):
        """ ********************************************
        Analyzing the LIVE VIDEO file coming from the camera
        ******************************************** """

        '''con -> frame difference (the sensitivity of motion)'''
        con = 0

        '''mode -> alarm activation mode '''
        global mode

        '''read the live feed (frames)'''
        e, f_start = self.cap.read()
        f_start = imutils.resize(f_start, width=500)
        gray = cv2.cvtColor(f_start, cv2.COLOR_BGR2GRAY)
        f_start = cv2.GaussianBlur(gray, (21, 21), 0)
        imgn = 0 # image number(frame number)
        while self.run:
            ret, frame = self.cap.read()
            self.out.write(frame)
            '''Write the current frame into video file'''
            # self.out.write(frame)

            '''Resize the frame'''
```

```
            frame = imutils.resize(frame, width=500)

            '''
            save all the photos(frames) inside images folder
            '''
            name = "../Outputs/images/"+"img_"+str(imgn)+".png"
            cv2.imwrite(name,frame)


            imgn += 1

            ''' mode = 1 indicates that the alarm has been activated'''
            if self.mode == 1:
                '''Convert the current frame into GRAY (black and white)'''
                gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
                # cv2.imshow("GRAY",gray)
                '''apply some gaussian blur'''
                gray = cv2.GaussianBlur(gray, (5, 5), 0)
                # gaus = cv2.GaussianBlur(gray, (5, 5), 0)
                # cv2.imshow("GAUSSIAN",gaus)
                #    f.append(gray)
                #     time.sleep(2)
                ''' calculate frame delta (for frame difference)  '''
                frameDelta = cv2.absdiff(gray, f_start)
                # cv2.imshow("frame delta",frameDelta)
                '''Find the threshhold of the frame Delta'''
                thresh = cv2.threshold(frameDelta, 25, 255, cv2.
                THRESH_BINARY)[1]
                # cv2.imshow("THreshhold",thresh)
                f_start = gray

                if thresh.sum() > 100:
                    # print(thresh.sum(),con)
                    con += 1
                else:
                    if con > 0:
                        con -= 1
                        # print("subs")

                Thread(target=cv2.imshow('Surveillance System::Analyze
                Video Frame', thresh)).start()

                if con > self.maxThresh:
                    print("Alerted")
                    self.mode = 0
```

```
                    con = 0
                    ##### ALERT OPTIONS #####
                    # call()
                    Thread(target=self.alert_me).start()
                    cv2.imwrite("INTRUDER.png", frame)
                    cv2.destroyWindow('Surveillance System::Analyze Video
                    Frame')
                else:
                    pass

            if self.mode == 0 or self.mode==1:
                # print("showing")
                cv2.imshow('Third Eye :: Live Video Feed', frame)

            # print(mode)
            k = cv2.waitKey(30) & 0xff
            if k == 27:  # press 'ESC' to quit
                self.run= False
                break
            elif k == ord('a'):
                self.mode = 1
            elif k == ord('s'):
                # print("Stop System")
                self.stop_system()


        if not self.run:
            self.out.release()
            self.cap.release()
            print("Released")
            cv2.destroyAllWindows()

def start_alarm(s):
    s.mode = 1

def stop_system(s):
    s.run = False

def start_system(address):
    s = SurveillanceSystem(address)
    Thread(target=s.video_analysis).start()
    return s
```