

Project Report  
on  
**SMART SURVEILLANCE USING COMPUTER  
VISION**

*Submitted in the partial fulfilment of the requirements for  
the award of the degree of*

**BACHELOR OF TECHNOLOGY**  
In  
**ELECTRONICS AND COMMUNICATION ENGINEERING**

By  

<b>P SAI LOHITH</b>	<b>17311A04T0</b>
<b>L VIGNAN GOUD</b>	<b>17311A04T4</b>
<b>G SHIVASAI</b>	<b>17311A04T7</b>

**UNDER THE GUIDANCE OF**

**Mrs.CH. LOWKYA**

**Assistant Professor**

**ECE - Department**



**DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING  
SREENIDHI INSTITUTE OF SCIENCE AND TECHNOLOGY  
(Autonomous)  
Yamnampet (V), Ghatkesar (M), Hyderabad – 501 301.  
June 2021**



## DECLARATION

We hereby declare that the work described in this thesis titled “**SMART SURVEILLANCE USING COMPUTER VISION**” which is being submitted by us in partial fulfilment for the award of Bachelor of Technology in the Department of **Electronics and Communication Engineering**, Sreenidhi Institute Of Science and Technology is the result of investigations carried out by us under the guidance of **Mrs.Ch. Lowkya, Department of ECE, Sreenidhi Institute of Science and Technology, Hyderabad.**

No part of the thesis is copied from books/ journals/ internet and whenever the portion is taken, the same has been duly referred. The report is based on the project work done entirely by us and not copied from any other source. The work is original and has not been submitted for any Degree/Diploma of this or any other university.

Place: Hyderabad

Date:

**P SAI LOHITH**

**17311A04T0**

**L VIGNAN GOUD**

**17311A04T4**

**G SHIVASAI**

**17311A04T7**

## **ACKNOWLEDGEMENTS**

We would like to take this opportunity to express our sincere gratitude to our Internal Guide – **Mrs.CH.LOWKYA** (Assistant Professor, Department of ECE), Sreenidhi Institute of Science and Technology, Hyderabad, for his invaluable support and guidance throughout the duration of this project phase.

We thank our Project coordinator - **Dr. S.I.KHAN** (Associate Professor, Department of ECE), Sreenidhi Institute of Science and Technology, Hyderabad, for his valuable comments and suggestions that greatly helped in improving the quality of our project.

We express our heartfelt gratitude to **Dr. Narsimha Reddy** (Executive Director, SNIST), **Dr. T. Ch. Shiva Reddy** (Principal, SNIST), **Dr. S. P. V. Subba Rao**, (Prof. & Head, Department of ECE, SNIST), for supporting us in executing this project throughout the course of this semester.

We also wish to extend a special thanks to our colleagues, friends and family for their unceasing encouragement and support and for helping us indirectly or directly to complete this project.

## **ABSTRACT**

This is a Project built using latest Programming Language and highly evolving Computer Science field which is “Computer Vision”. Which means this project allow computer to watch or in other words it gives vision capability to computers.

While this technique creates better execution, it has the weakness that it's anything but a great deal of clear space data. For instance, subtleties are required on what various types of untamed life might be available in each program region. This number of information turns out to be huge particularly where potential models (for example climate results) are remembered for the system.

Wellbeing and security are a significant issue in present day times. Individuals use safety efforts to ensure their property at home or in the organization. Current security frameworks incorporate the utilization of an assortment of movement sensors and video reconnaissance cameras, in particular, Border Interruption Discovery Frameworks. This task means to give a solitary viewpoint to guarantee the wellbeing and security of individual property. In this paper we propose to give a keen CCTV Observation framework to distinguish interruption location. Numerous USB cameras have been introduced in different spaces of live streaming and observing. This program empowers face acknowledgment as a confirmation interaction and advises the proprietor when a mysterious face is recognized by sending an email with unknown face acknowledgment and SMS. Live feeds from different cameras can be seen on Cell phones, PCs.

# INDEX

<b>CHAPTER 1 INTRODUCTION</b>	<b>8</b>
<b>CHAPTER 2 LITERATURE SURVEY</b>	<b>9</b>
2.1 Computer Vision For Security Applications By Kingsley Sage	<b>9</b>
2.2 Smart Cctv Surveillance System For Intrusion Detection With Live Streaming By Akshay Bharadwaj K	<b>9</b>
2.3 Smart Surveillance: Applications, Technologies And Implications By Arun Hampapur	<b>9</b>
<b>CHAPTER 3 OBJECTIVES</b>	<b>10</b>
3.1 Monitor	<b>10</b>
3.2 Identify the person	<b>12</b>
3.3 Detect the noise	<b>17</b>
3.4 In and Out detection	<b>17</b>
<b>CHAPTER 4 METHODOLOGY</b>	<b>18</b>
<b>CHAPTER 5 APPARATUS DESCRIPTION</b>	<b>19</b>
5.1 Waterfall Model	<b>20</b>
5.2 Software Requirements	<b>21</b>
5.3 Hardware Requirements	<b>21</b>
5.4 Libraries used	<b>22</b>
5.4.1 OpenCV	<b>22</b>
5.4.2 Numpy	<b>22</b>
5.4.3 Skimage	<b>22</b>
<b>CHAPTER 6 IMPLEMENTATION</b>	<b>25</b>
6.1 Code	<b>25</b>
6.1.1 spot_diff.py	<b>25</b>
6.1.2 find_motion.py	<b>27</b>

6.1.3 in_out.py	30
6.1.4 motion.py	33
6.1.5 record.py	35
6.1.6 rect_noise.py	36
6.1.7 identify.py	39
6.1.8 main.py	44
<b>CHAPTER 7 TEST AND EVALUATION</b>	<b>48</b>
<b>CHAPTER 8 RESULTS</b>	<b>49</b>
<b>CHAPTER 9 ANALYSIS</b>	<b>70</b>
<b>CHAPTER 10 FUTURE SCOPE</b>	<b>71</b>
<b>CHAPTER 11 REFERENCES</b>	<b>72</b>

## **LIST OF FIGURES**

Figure 3.1 The structural similarity index	12
Figure 3.2 Detecting faces in the frames	13
Figure 3.3 Haar features	14
Figure 3.4 Haar Features	14
Figure 3.5 LBPH for Face Recognition	16
Figure 3.6 Detect for noises in the frame	17
Figure 3.7 Frames and its differences	17
Figure 5.1 Waterfall Model	21
Figure 8.1 Command prompt	49
Figure 8.2 GUI Interface	50
Figure 8.3 Monitor the area it covers	51
Figure 8.4 Book has been removed from frame	52
Figure 8.5 It detects that book had been removed	53
Figure 8.6 GUI of identification	54
Figure 8.7 Asks for New Member details	55
Figure 8.8 Asks for New Member details	56
Figure 8.9 Capturing of images of new member	57
Figure 8.10 Training part is being initiated	58
Figure 8.11 Person is being identified	59
Figure 8.12 No motion is being identified in Rectangle Box	60
Figure 8.13 Motion is being identified in Rectangle Box	61
Figure 8.14 No motion is being identified in whole frame	62
Figure 8.15 Motion is being identified	63
Figure 8.16 Commands for in and out motion	64
Figure 8.17 Person is moving in	65
Figure 8.18 Person is moving out	66
Figure 8.19 Visitors file	67
Figure 8.20 Complete Database	68
Figure 8.21 Captured images with date and time	69



# **CHAPTER-1**

## **INTRODUCTION**

In present world things has made a change in the security worldview from "examination of episodes" to "counteraction of conceivably disastrous incidents". Available advanced video observation frameworks gives the foundation just to catch, store and disseminate video, while leaving the undertaking of danger discovery especially to human capacities. Human seeing of reconnaissance video is a human concentrated errand. It is fundamentally concurred that watching video takes care of requires a more significant level of visual consideration than most regular errands. Especially carefulness, the capacity to hold consideration and to react to infrequently happening occasions, is amazingly requesting and inclined to blunder because of breaches in consideration. Face acknowledgment innovation has progressed significantly over the most recent twenty years. These days, machines can consequently approve personality data for secure exchanges, for observation and security assignments, and for access control to structures. These applications ordinarily work in controlled conditions and acknowledgment calculations that can exploit the natural requirements to achieve high acknowledgment precision. Escalated research is going on in the field of interruption recognition. In our task we have figured out how to develop a powerful, adaptable face acknowledgment framework with off the rack gears.

Picture preparing introductory started to affect security innovation in the mid 1980s with the starter of Video Movement Discovery (VMD) frameworks which professed to alter Edge Gatecrasher Identification Frameworks (PIDS). Lackluster showing (specifically, high bogus caution paces) of early working establishments prompted research exertion coordinated towards delivering more refined arrangements. Ideas from man-made consciousness, like neural organizations and master frameworks, have been joined into frameworks which are more PC vision orientated.

In a days which takes the stand concerning a creation of Shut Circuit TV (CCTV) cameras for security and reconnaissance observing, the utilization of picture preparing and PC vision strategies which were given as top end customized arrangements would now be able to be acknowledged utilizing work area PC handling. Business Video Movement Recognition (VMD) and Savvy Scene Observing (ISM) frameworks are getting progressively modern, helped, in no little way, by an innovation move from beforehand solely military exploration areas.

## **CHAPTER-2**

### **LITERATURE SURVEY**

#### **2.1 COMPUTER VISION FOR SECURITY APPLICATIONS BY KINGSLEY SAGE**

To seek after enhancements in video based PIDS innovation, PSDB chose to seek after investigate in 3 regions:

- To improve framework viability by co-ordinating non-video sensors with video sensors: specifically, to utilize the video frameworks as a methods for giving computerized confirmation. This methodology (the AMETHYST undertaking) is accounted for.
- To improve the complexity and execution of fundamental picture preparing techniques.
- To receive a frameworks approach utilizing various types of video handling and join them into a bound together understanding. Such a framework could join development examination, optic stream estimation, prescient climate model and human walk investigation and different highlights.

#### **2.2 SMART CCTV SURVEILLANCE SYSTEM FOR INTRUSION DETECTION WITH LIVE STREAMING BY AKSHAY BHARADWAJ K**

He proposed a use of a security lock framework utilizing a face acknowledgment technique. Head Segment Investigation (PCA) is chosen for the face acknowledgment calculation and when the face acknowledgment returns a positive reaction a sign is shipped off the Arduino UNO to open the framework.

#### **2.3 SMART SURVEILLANCE: APPLICATIONS, TECHNOLOGIES AND IMPLICATIONS BY ARUN HAMPAPUR**

Shrewd observation, is the utilization of programmed video study advancements in video reconnaissance applications. This paper endeavors to answer various questions about shrewd observation: What are the uses of keen reconnaissance? What are the framework plans for shrewd observation? What are the key advances? What are the a portion of the key commonsense difficulties? furthermore, What are the ramifications of brilliant observation, both to security and protection? These are the things examined by the creator about his task.

## CHAPTER- 3

### OBJECTIVES

Below are the different features which can performed by using this project:

1. Monitor
2. Identify the person
3. Detect the noise
4. In and Out detection

#### 1. MONITOR:

This feature is used to find what is the thing which is stolen from the frame which is visible to webcam. Meaning It constantly monitors the frames and checks which object or thing from the frame has been taken away by the thief.

This uses **Structural Similarity** to find the differences in the two frames. The two frames are captured first when noise was not happened and second when noise stopped happening in the frame.

SSIM is utilized as a measurement to quantify the closeness between two given pictures. As this strategy has been around since 2004, a great deal of material exists clarifying the hypothesis behind SSIM however not very many assets delve profound into the subtleties, that too explicitly for an angle based execution as SSIM is regularly utilized as a misfortune work.

The Structural Similarity Index (SSIM) metric extracts 3 key *features* from an image:

- Luminance
- Contrast
- Structure

*The comparison* between the two images is performed on the basis of these 3 features.

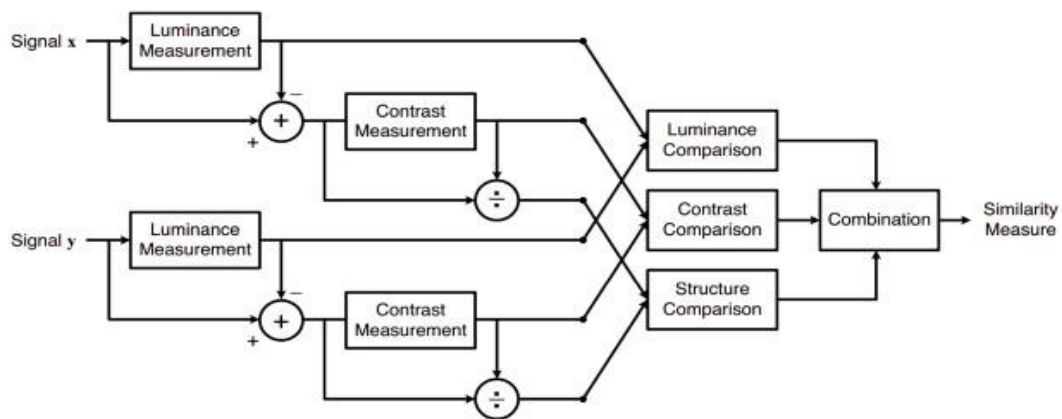


Figure 3.1 The structural similarity index

The system calculates the structural similarity index between two given images, with a value between 1 and +1. A value of +1 indicates that the two image data are very similar or identical, and a value of 1 indicates that the two image data are very similar. These values are usually adjusted so that they are in the range  $[0, 1]$ , where the extreme values have the same value.

**LUMINANCE:** Luminance is measured by *averaging* over all the pixel values. Its denoted by  $\mu$  (Mu).

**CONTRAST:** Contrast is measured by calculating the standard deviation of the all pixel values.

**STRUCTURE:** The structural comparison is done using a comprehensive formula (more on that later), but in essence, we divide the input signal by its standard deviation, so the result is only one standard deviation, so that a more reliable comparison can be made.

Luckily , thanks to skimage package in python we dont have to replicate all this mathematical calculation in python since skimage has pre build feature that d o all of these tasks for us with just calling its in-built function.

We just have to feed in two images/frames which we have captured earlier, so we just feed them in and it gives us out the masked image with score.

## 2. IDENTIFY THE PERSON:

This feature is a very useful feature of our minor project, It is used to find if the person the frame is known or not. It does this in two steps:

1. Find the faces in the frames
2. Use LBPH face recognizer algorithm to predict the person from already trained model.

So let's divide this in following categories,

### 1 – DETECTING FACES IN THE FRAMES:

This is done via Haar cascade classifiers which are again in-built in open CV module of python.

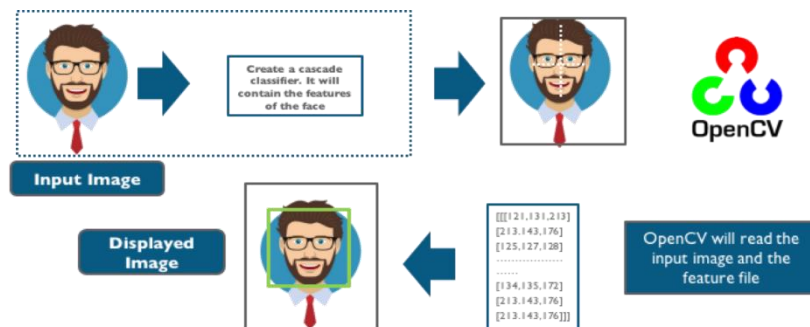


Figure 3.2 Detecting faces in the frames

Cascade classifiers, or cascades of closed classifiers that process hair-like features, are a special case of ensemble learning, called authorization. It is usually based on the Adaboost classifier (and other models such as Real Adaboost, Gentle Adaboost, or Logitboost). They train on hundreds of sample images containing the object to be detected and other images that do not contain these images.

There are some common features that we find on most common human faces:

- a dark eye region compared to upper-cheeks
- a bright nose bridge region compared to the eyes
- some specific location of eyes, mouth, nose...

The characteristics are called **Haar Features**. The feature extraction process will look like this:

Haar features are similar to these **convolution kernels** which are used to detect the presence of that feature in the given image.

For doing all this stuff openCV module in python language has inbuilt

function called cascade classifier which we have used in order to detect for faces in the frame

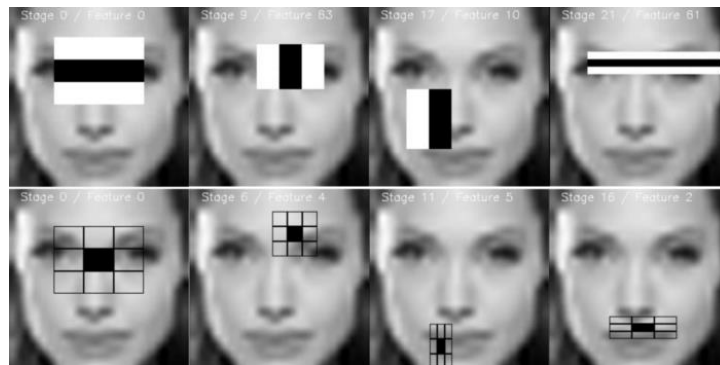


Figure 3.3 Haar features

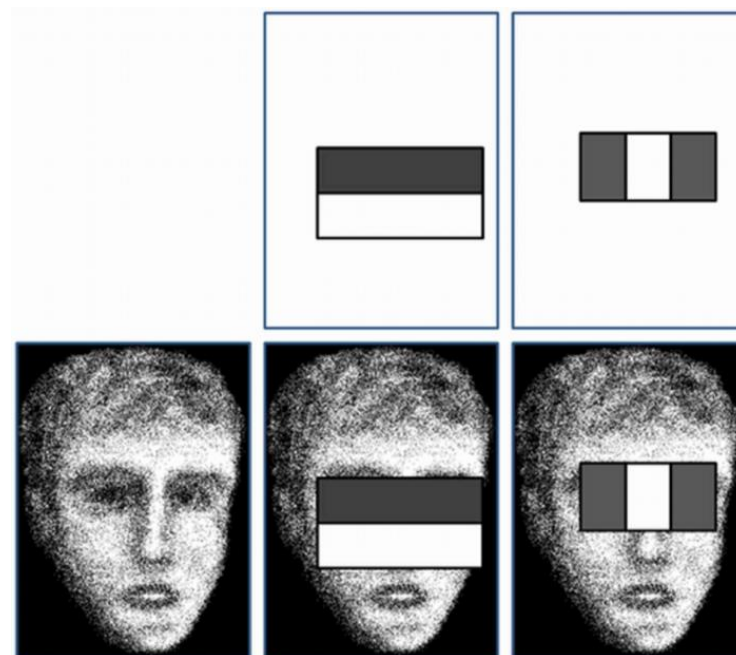


Figure 3.4 Haar Features

AdaBoost is an acronym for measurable grouping of Versatile BOOSTING meta-calculus, defined by Yoav Freund and Robert Schapir, who won the Gödel Prize for their work in 2003. It can be used well with many different types of training calculations to improve performance. The learning calculations ("weak learners") are combined into a weighted integer that takes

into account the latest auxiliary classifier results. AdaBoost is universal, because the resulting weak learner will switch to the case where the previous classifier misclassified. Compared with other learning calculations, it is less prone to overfitting. However, individual learners may be powerless, as long as their testimony is slightly better than the vague hypothesis, the latter model can unite persistent learners.

Generally speaking, calculation is more suitable for certain types of problems than other problems, and it usually has various limitations and mechanisms that must be changed before the data set can perform at its best. AdaBoost (with a selection tree like the helpless student) is often called the best out-of-bass classifier. When used with selection tree training, AdaBoost collects data at each stage. The calculation of the relative "stiffness" of each preliminary test is carried out in the calculation of tree development, so that subsequent trees tend to focus on the more rigorous group model.

## **2 -USING LBPH FOR FACE RECOGNITION:**

So now we have detected for faces in the frame and this is the time to identify it and check if it is in the dataset which we've used to train our lbph model.

The LBPH uses 4 parameters:

- Radius: The radius is used to construct a circular local binary pattern, which represents the radius around the central pixel. Usually set to .

Neighbors: The number of sample points used to create a circular local binary pattern. Please note that the more sample points you include, the greater the calculation workload. Usually set to 8.

- X-grid: the number of cells in the horizontal direction. The more cells and the finer the mesh, the larger the size of the generated object. Usually set to 8.

- Grid Y-the number of cells in the vertical direction. The more cells and the finer the grid, the larger the dimension of the resulting feature vector.Usually its value is 8.

The first step of LBPH calculation is to create an intermediate image that better describes the original image by emphasizing facial features. To this end, the algorithm uses a sliding window concept based on radius and neighbor parameters, as shown in the figure above.Which is shown perfectly via the above image.

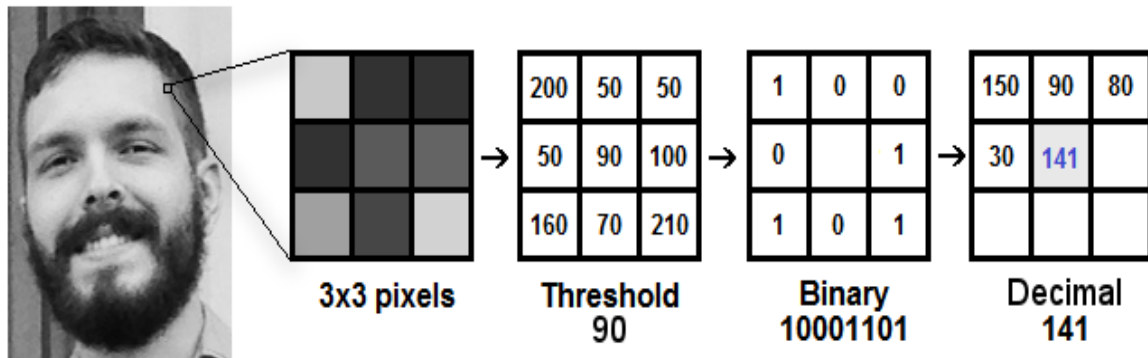


Figure 3.5 LBPH for Face Recognition

**Extracting the Histograms:** Now, using the image generated in the last step, we can use the **Grid X** and **Grid Y** parameters to divide the image into multiple grids, as can be seen in the following image:

And after all this the model is trained and later on when we want to make predictions the same steps are applied to the make and its histograms are compared with already trained model and in such way this feature works.



### 3. DETECT FOR NOISES IN THE FRAME:

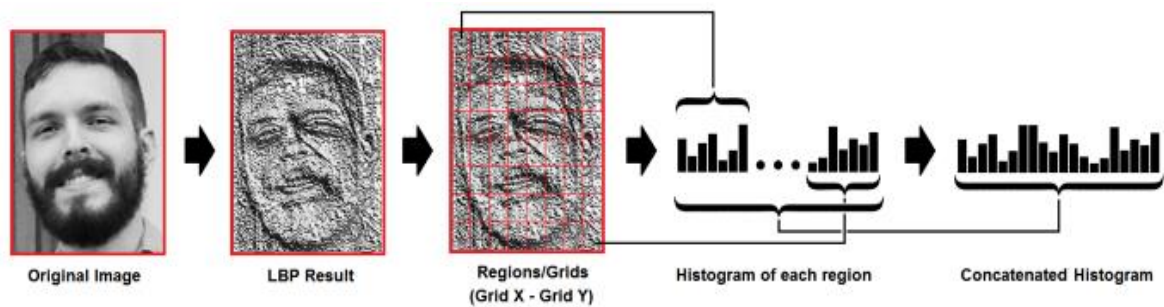


Figure 3.6 Detect for noises in the frame

This feature is used to find the noises in the frames well this is something you would find in most of the cctv's but in this module we'll see how it works.

Talking in simple way all the frames are continuously analyzed and checked for noises. Noise is checked in the consecutive frames. Simply we do the absolute difference between two frames and in this way the difference of two images are analyzed and Contours (boundaries of the motion are detected) and if there are no boundaries then no motion and if there is any there is

frame1	frame2	frame2 - frame1	abs (frame2 - frame1)
10 90 16 16	10 90 16 16	0 0 0 0	0 0 0 0
0 11 11 11	0 13 17 11	0 2 6 0	0 2 6 0
18 30 33 33	18 34 31 33	0 4 -2 0	0 4 2 0
18 18 18 18	18 17 19 18	0 -1 1 0	0 1 1 0

motion.

Figure 3.7 Frames and its differences

As you would know all images are just integer/ float values of pixels which tells the brightness of pixel and similarly every pixel has that values of brightness.

So we just do simply absolute difference because negative will make no sense at all.

### 4. VISITORS IN ROOM DETECTION:

This is the feature which can detect if someone has entered in the room or gone out.

So it works using following steps:

- 1.** It first detect for noises in the frame.
- 2.** Then if any motion happen it find from which side does that happen either left or right.
- 3.** Last if checks if motion from left ended to right then its will detect it as entered and capture the frame.

Or vice-versa.

So there is not complex mathematics going on around in this specific feature.

So basically to know from which side does the motion happened we first detect for motion and later on we draw rectangle over noise and last step is we check the co-ordinates if those points lie on left side then it is classified as left motion.

## **CHAPTER – 4**

### **METHODOLOGY**

A large number of cameras (USB cameras/webcams) are used to take pictures. Newly recognized faces can be distinguished based on the reflection of wild boars, and these faces can be recorded. These recognized faces are rich in contrast. The faces in the information database are calculated by face recognition. Assuming the comparison between the recognized faces and the faces in the data set, the procedure continues, but the condition is that the test gives the wrong level. From all The email and SMS notifications sent by the user's device have a dark face on the shell. When this happens, the customer/owner can remotely stream the video by simply collecting the IP address of the frame running the program. For example, here all data collection components must be assigned to a similar organization. Once inserted into the frame, the camera's sensor will send out circuit diagrams encoded in JPEG structure, and each circuit diagram is dissected according to the position of the face. Calculate, and then process these recognized faces through a face recognition calculation, which checks the comparable face codes in the appearance data set. Confirmed alarms are triggered by email and SMS.

The video received after the camera is hacked can be viewed remotely through the local network. The module and its function is the face detection module: this module is used to detect every face within the camera. This module uses Borov's glare for face recognition. Face recognition module: This module uses the face recognized by the face recognition module. The face of the tourist spot is mainly obtained by calculating the uneven forest. If the recognized face is available in our database, then these codes and our database are compared in the same way at this stage. Information, mark "name", if the person who is marked is not in our record, mark it as "unknown". Real-time streaming module: This module is used to broadcast the captured profile/real-time video from the camera to the restricted staff. It can be transmitted remotely using a mobile phone or PC. We use Flask WSGI as a documented web worker. Workers and web workers are included in the video stream to access the port of the Ready Messages module: As mentioned in the face recognition section, when an unknown person is within the range of the camera that started Twilio, we will send an SMS message to the owner r. An email with a photo of the attacker will also be sent as a link to the owner's email.

## CHAPTER – 5

### APPARATUS DESCRIPTION

For this model we have used **waterfall model**, since it was not huge project at all.

Reasons behind choosing waterfall model -

1. Good for minor projects.
2. Easy to follow.
3. Well tracking for small projects.
4. Well time managed.

#### **WATERFALL MODEL:**

Traditional cascade model is the fundamental programming advancement life cycle model. It is extremely straightforward yet optimistic. Prior this model was well known yet these days it's anything but utilized. Yet, it is vital on the grounds that the wide range of various programming advancement life cycle models depend on the traditional cascade model.

Classical waterfall:

This model partitions the existence cycle into a bunch of stages. This model thinks about that one stage can be begun after culmination of the past stage. That is the yield of one stage will be the contribution to the following stage. Consequently the improvement interaction can be considered as a successive

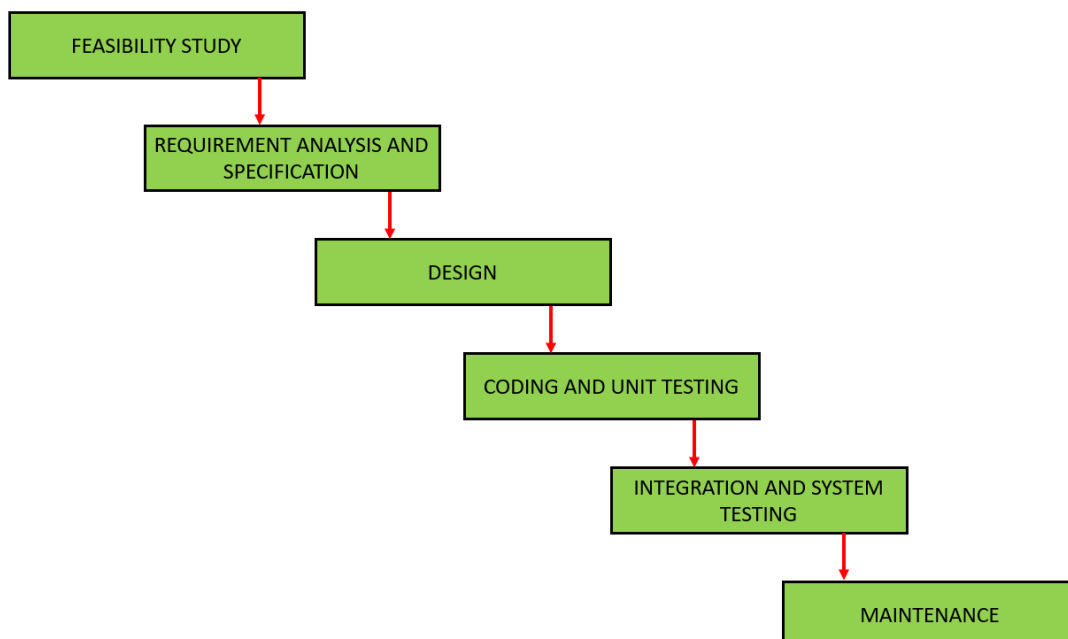


Figure 5.1 Waterfall Model

stream in the cascade. Here the stages don't cover with one another. The distinctive consecutive stages are

### **SOFTWARE REQUIREMENTS:**

Since this is a software hence it will have to run on some hardware and operating system obviously, so below are the requirements to run this software :

- Windows/Linux/Mac OS any version, hence it can run on any platform.
- Python3, it need python to be installed in your system to run this successfully.
- Packages in python -
  - openCV
  - skimage
  - numpy
  - tkinter

### **HARDWARE REQUIREMENTS:**

In terms of hardware requirements there is not much required at all but still below requirements are must :

- Working PC or Laptop
- Webcam with drivers installed
- Flashlight/ LED if using this at night.

### **Technology used to make this project :**

- as mentioned earlier **Python** language is used.
- Sublime Text Editor is used to write the code.
- Linux Mint os is used to run and create this minor project.
- HP-ay503tx laptop is used.
  - Core i5 dual core
  - 240GB ssd
  - 8GB RAM
  - In-built webcam hp-truevision
- Terminal to run the code

### **Platforms already tested:**

It is tested on Linux Mint, Linux Ubunutu, Windows 7, Windows 10.

### **OPENCV:**

OpenCV was launched by Gary Bradsky of Intel in 1999, and its first version was released in 2000. Gary Bradsky collaborated with Vadim Risharevsky to lead the Russian software development team Intel OpenCV, which was deployed in 2005 and the Windows machine was deployed in 2005. Support various projects. These algorithms are connected to the Windows machine learning computer, so they are spread every day.

OpenCV supports multiple programming languages such as C++, Python, Java, etc. It also provides multiple programs, including Windows, Linux, and pre-authorized application services provided by top-level applications. They are also actively developing. OpenCVpython has a library of predetermined connections that can solve python's visual problems with fewer lines of code without sacrificing readability. Compared with languages such as C/C++, Python is slower. + And create a Python writer that can be used as a Python module. This brings us two benefits: First, the code is as fast as co-source C/co-code, and programming in Python is easier than in C/C++. OpenCV Python is the author of the original OpenCV C++ implementation of Python.

OpenCV Python allows the use of numbers, which is a deeply optimized library number format, and all track synthesizers are spelled structures. It can also be more easily integrated with other libraries that use NumPy (such as SciPy and Matplotlib). Introducing the OpenCV Python tutorial OpenCV provides a new set of OpenCV Python tutorials, OpenCV, which will help you understand the various OpenCVPython functions available.

NumPy and Numpy are recommended, because they are not found in this help. Skills using Numpy should use OpenCV Python to create an Optimized Opod. The plan led by Alexander Mordvintsev. OpenCV needs you! ! Since OpenCV is a joint project, anyone can contribute to the library, documentation, and guidance.

You can resolve this issue by closing OpenCV on GitHub and submitting the complete application. OpenCV developers will check your application thoroughly, provide you with important information and (combined comments will be displayed later). Become a new employee. Because the modules added to OpenCV Python need to be extended. this way.

## **SKIMAGE:**

### **What is skimage and Why Should We Use it?**

Python has several libraries and frameworks that allow us to process images.

Why use skimage? If you use sklear for a lot of work, skimage will become part of the framework.

skimage has a well-structured document for registering all modules, submodules, and functions in skimage.

The following is a summary of all sub-modules and functions in skimage.

## **NUMPY:**

Numpy is fundamental package for scientific computing in Python. It is Python library that provides a multidimensional array object determined various objects (such as masked arrays and matrices) and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I / O, discrete transforms.

Fourier, basic linear algebra, basic statistical operations, random simulation and much more. At the core of Numpy package is an array object. This encapsulates ndimensional arrays of homogeneous data types, many presets programmed in perfectly composed code. There are some important differences between number series and the standard period order:

Many files are fixed in size when they are created, nothing about memory registers (which can grow dynamically). Changing the size of an array creates a new register and clears its components. The same kind of data, and that way it will have the same size in memory. types other operations on large numbers data.

Typically, such operations are more efficiently executed and less than code is possible using Python sequences. A underlying growing plethora of scientific and mathematical Python-based packages are arrays Numpy used; though this typically support Python sequence input the convert such input to Numpy arrays prior to processing that and often output Numpy arrays.

In other words in order to use efficiently much (perhaps even most) today of scientific / mathematical Python based software easy knowing The type and wise using the inherent sequence types of Python is not enough for someone who also needs to know how to use Numpy arrays.

The points about sequence size and speed are particularly important in scientific computing. As a simple example, consider the case of multiplying each component in a 1-D sequence with the corresponding component in another sequence of the same length. On the off chance that the data are stored in two Python records, a and b, we could iterate over each component:

For what reason is Numpy Fast?

Vectorization describes the absence of any explicit looping, ordering, etc., in the code - these things are taking place, of course, just "behind the scenes" in

optimized, pre-compiled C code. Vectorized code has many advantages, among which are:

vectorized code is more concise and easier to read

less lines of code generally means less bugs

the code more closely takes after standard mathematical notation (making it easier, typically, to correctly code mathematical constructs)

vectorization results in more "Pythonic" code. Without vectorization, our code would be covered with inefficient and difficult to read for loops.

Broadcasting is a term used to describe important components related to the behavior of the components in the formula; in general, all versions, not only arithmetic, but also classical, magazine, functional group equivalents or ladders and a group or even two different Name the group, and the blacksmith "expands" to a later group. For the "rules" for classifying gaps, see Classification of Classifications.

Who Else Uses Numpy?

Numpy completely supports ONE object-oriented approach, again starting from ndarray. For example, ndarray has a collection of many method attributes. This flexibility makes Numpy and Numpy ndarray dialects the authoritative languages for multidimensional data of different sizes.



## **CHAPTER-6**

### **IMPLEMENTATION**

#### **CODE:**

##### **SPOT\_DIFF.py:**

```
import cv2
import time
from skimage.metrics import structural_similarity
from datetime import datetime
import beepy

def spot_diff(frame1, frame2):

    frame1 = frame1[1]
    frame2 = frame2[1]

    g1 = cv2.cvtColor(frame1, cv2.COLOR_BGR2GRAY)
    g2 = cv2.cvtColor(frame2, cv2.COLOR_BGR2GRAY)

    g1 = cv2.blur(g1, (2,2))
    g2 = cv2.blur(g2, (2,2))

    (score, diff) = structural_similarity(g2, g1, full=True)

    print("Image similarity", score)

    diff = (diff * 255).astype("uint8")
    thresh = cv2.threshold(diff, 100, 255, cv2.THRESH_BINARY_INV)[1]
```

```

    contours = cv2.findContours(thresh, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)[0]

    contours = [c for c in contours if cv2.contourArea(c) > 50]

    if len(contours):
        for c in contours:

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

            cv2.rectangle(frame1, (x,y), (x+w, y+h), (0,255,0), 2)

        else:
            print("nothing stolen")
            return 0

    cv2.imshow("diff", thresh)
    cv2.imshow("win1", frame1)
    beepy.beep(sound=4)
    cv2.imwrite("stolen/"+datetime.now().strftime('%y-%m-%d-
%H:%M:%S')+".jpg", frame1)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

    return 1

```

### **Find\_Motion.py**

```
import cv2
from spot_diff import spot_diff
import time
import numpy as np

def find_motion():

    motion_detected = False
    is_start_done = False

    cap = cv2.VideoCapture(0)

    check = []

    print("waiting for 2 seconds")
    time.sleep(2)
    frame1 = cap.read()

    _, frm1 = cap.read()
    frm1 = cv2.cvtColor(frm1, cv2.COLOR_BGR2GRAY)

    while True:
        _, frm2 = cap.read()
        frm2 = cv2.cvtColor(frm2, cv2.COLOR_BGR2GRAY)

        diff = cv2.absdiff(frm1, frm2)
```

```

_, thresh = cv2.threshold(diff, 30, 255, cv2.THRESH_BINARY)

contors = cv2.findContours(thresh, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)[0]

#look at it
contors = [c for c in contors if cv2.contourArea(c) > 25]

if len(contors) > 5:
    cv2.putText(thresh, "motion detected", (50,50),
cv2.FONT_HERSHEY_SIMPLEX, 2, 255)
    motion_detected = True
    is_start_done = False

elif motion_detected and len(contors) < 3:
    if (is_start_done) == False:
        start = time.time()
        is_start_done = True
        end = time.time()

    end = time.time()

print(end-start)
if (end - start) > 4:
    frame2 = cap.read()
    cap.release()
    cv2.destroyAllWindows()
    x = spot_diff(frame1, frame2)
    if x == 0:
        print("running again")

```

```

        return

    else:
        print("found motion")
        return

    else:
        cv2.putText(thresh, "no motion detected", (50,50),
cv2.FONT_HERSHEY_SIMPLEX, 2, 255)

    cv2.imshow("winname", thresh)

    _, frm1 = cap.read()
    frm1 = cv2.cvtColor(frm1, cv2.COLOR_BGR2GRAY)

    if cv2.waitKey(1) == 27:

        break

return

```

### **In\_Out.py**

```
import cv2

from datetime import datetime

def in_out():

    cap = cv2.VideoCapture(0)


    right, left = "", ""


    while True:

        _, frame1 = cap.read()
        frame1 = cv2.flip(frame1, 1)
        _, frame2 = cap.read()
        frame2 = cv2.flip(frame2, 1)


        diff = cv2.absdiff(frame2, frame1)


        diff = cv2.blur(diff, (5,5))


        gray = cv2.cvtColor(diff, cv2.COLOR_BGR2GRAY)


        _, threshd = cv2.threshold(gray, 40, 255, cv2.THRESH_BINARY)


        contr, _ = cv2.findContours(threshd, cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)


        x = 300

        if len(contr) > 0:

            max_cnt = max(contr, key=cv2.contourArea)
```

```

x,y,w,h = cv2.boundingRect(max_cnt)
cv2.rectangle(frame1, (x, y), (x+w, y+h), (0,255,0), 2)
cv2.putText(frame1, "MOTION", (10,80),
cv2.FONT_HERSHEY_SIMPLEX, 2, (0,255,0), 2)

if right == "" and left == "":
    if x > 500:
        right = True

    elif x < 200:
        left = True

elif right:
    if x < 200:
        print("to left")
        x = 300
        right, left = "", ""
        cv2.imwrite(f'visitors/in/{datetime.now().strftime("%Y-%m-%d-%H:%M:%S")}.jpg', frame1)

elif left:
    if x > 500:
        print("to right")
        x = 300
        right, left = "", ""
        cv2.imwrite(f'visitors/out/{datetime.now().strftime("%Y-%m-%d-%H:%M:%S")}.jpg', frame1)

```

```
cv2.imshow("", frame1)
```

```
k = cv2.waitKey(1)
```

```
if k == 27:
```

```
    cap.release()
```

```
    cv2.destroyAllWindows()
```

```
    break
```



## **Motion.py**

```
import cv2
```

```
def noise():
```

```
    cap = cv2.VideoCapture(0)
```

```
    while True:
```

```
        _, frame1 = cap.read()
```

```
        _, frame2 = cap.read()
```

```
        diff = cv2.absdiff(frame2, frame1)
```

```
        diff = cv2.cvtColor(diff, cv2.COLOR_BGR2GRAY)
```

```
        diff = cv2.blur(diff, (5,5))
```

```
        _, thresh = cv2.threshold(diff, 25, 255, cv2.THRESH_BINARY)
```

```
        contr, _ = cv2.findContours(thresh, cv2.RETR_TREE,  
cv2.CHAIN_APPROX_SIMPLE)
```

```
        if len(contr) > 0:
```

```
            max_cnt = max(contr, key=cv2.contourArea)
```

```
            x,y,w,h = cv2.boundingRect(max_cnt)
```

```
            cv2.rectangle(frame1, (x, y), (x+w, y+h), (0,255,0), 2)
```

```
            cv2.putText(frame1, "MOTION", (10,80),  
cv2.FONT_HERSHEY_SIMPLEX, 2, (0,255,0), 2)
```

```
        else:
```

```
            cv2.putText(frame1, "NO-MOTION", (10,80),  
cv2.FONT_HERSHEY_SIMPLEX, 2, (0,0,255), 2)
```

```
cv2.imshow("esc. to exit", frame1)
```

```
if cv2.waitKey(1) == 27:
```

```
    cap.release()
```

```
    cv2.destroyAllWindows()
```

```
    break
```

## **Record.py**

```
import cv2

from datetime import datetime

def record():

    cap = cv2.VideoCapture(0)

    fourcc = cv2.VideoWriter_fourcc(*'XVID')

    out = cv2.VideoWriter(f'recordings/{datetime.now().strftime("%H-%M-%S")}.avi', fourcc,20.0,(640,480))

    while True:

        _, frame = cap.read()

        cv2.putText(frame, f'{datetime.now().strftime("%D-%H-%M-%S")}',
(50,50), cv2.FONT_HERSHEY_COMPLEX,
                0.6, (255,255,255), 2)

        out.write(frame)

    cv2.imshow("esc. to stop", frame)

    if cv2.waitKey(1) == 27:
        cap.release()
        cv2.destroyAllWindows()
        break
```

### **Rect\_noise.py**

```
import cv2

donel = False
doner = False
x1,y1,x2,y2 = 0,0,0,0

def select(event, x, y, flag, param):
    global x1,x2,y1,y2,donel, doner
    if event == cv2.EVENT_LBUTTONDOWN:
        x1,y1 = x,y
        donel = True
    elif event == cv2.EVENT_RBUTTONDOWN:
        x2,y2 = x,y
        doner = True
        print(doner, donel)

def rect_noise():

    global x1,x2,y1,y2, donel, doner
    cap = cv2.VideoCapture(0)

    cv2.namedWindow("select_region")
    cv2.setMouseCallback("select_region", select)
```

```
while True:
```

```
    _, frame = cap.read()
```

```
    cv2.imshow("select_region", frame)
```

```
    if cv2.waitKey(1) == 27 or doner == True:
```

```
        cv2.destroyAllWindows()
```

```
        print("gone--")
```

```
        break
```

```
while True:
```

```
    _, frame1 = cap.read()
```

```
    _, frame2 = cap.read()
```

```
    frame1only = frame1[y1:y2, x1:x2]
```

```
    frame2only = frame2[y1:y2, x1:x2]
```

```
    diff = cv2.absdiff(frame2only, frame1only)
```

```
    diff = cv2.cvtColor(diff, cv2.COLOR_BGR2GRAY)
```

```
    diff = cv2.blur(diff, (5,5))
```

```
    _, thresh = cv2.threshold(diff, 25, 255, cv2.THRESH_BINARY)
```

```
    contr, _ = cv2.findContours(thresh, cv2.RETR_TREE,  
cv2.CHAIN_APPROX_SIMPLE)
```

```
    if len(contr) > 0:
```

```
        max_cnt = max(contr, key=cv2.contourArea)
```

```
        x,y,w,h = cv2.boundingRect(max_cnt)
```

```

        cv2.rectangle(frame1, (x+x1, y+y1), (x+w+x1, y+h+y1), (0,255,0), 2)
        cv2.putText(frame1, "MOTION", (10,80),
cv2.FONT_HERSHEY_SIMPLEX, 2, (0,255,0), 2)

    else:
        cv2.putText(frame1, "NO-MOTION", (10,80),
cv2.FONT_HERSHEY_SIMPLEX, 2, (0,0,255), 2)

cv2.rectangle(frame1, (x1,y1), (x2, y2), (0,0,255), 1)
cv2.imshow("esc. to exit", frame1)

if cv2.waitKey(1) == 27:
    cap.release()
    cv2.destroyAllWindows()
    break

```

## **Identify.py**

```
count = 1

ids = input("Enter ID: ")

cap = cv2.VideoCapture(0)

filename = "haarcascade_frontalface_default.xml"

cascade = cv2.CascadeClassifier(filename)

while True:
    _, frm = cap.read()

    gray = cv2.cvtColor(frm, cv2.COLOR_BGR2GRAY)

    faces = cascade.detectMultiScale(gray, 1.4, 1)

    for x,y,w,h in faces:
        cv2.rectangle(frm, (x,y), (x+w, y+h), (0,255,0), 2)
        roi = gray[y:y+h, x:x+w]

        cv2.imwrite(f'persons/{name}-{count}-{ids}.jpg', roi)
        count = count + 1
        cv2.putText(frm, f'{count}', (20,20),
cv2.FONT_HERSHEY_PLAIN, 2, (0,255,0), 3)
        cv2.imshow("new", roi)

    cv2.imshow("identify", frm)
```

```

if cv2.waitKey(1) == 27 or count > 300:
    cv2.destroyAllWindows()
    cap.release()
    train()
    break

```

```

def train():
    print("training part initiated !")

    recog = cv2.face.LBPHFaceRecognizer_create()

    dataset = 'persons'

    paths = [os.path.join(dataset, im) for im in os.listdir(dataset)]

    faces = []
    ids = []
    labels = []
    for path in paths:
        labels.append(path.split('/')[-1].split('-')[0])

        ids.append(int(path.split('/')[-1].split('-')[2].split('.')[0]))

        faces.append(cv2.imread(path, 0))

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

```



```

    recog.save('model.yml')

    return

def identify():
    cap = cv2.VideoCapture(0)

    filename = "haarcascade_frontalface_default.xml"

    paths = [os.path.join("persons", im) for im in os.listdir("persons")]
    labelslist = {}
    for path in paths:
        labelslist[path.split('/')[-1].split('-')[2].split('.')[0]] = path.split('/')[-1].split('-')[0]

    print(labelslist)
    recog = cv2.face.LBPHFaceRecognizer_create()

    recog.read('model.yml')

    cascade = cv2.CascadeClassifier(filename)

    while True:
        _, frm = cap.read()

        gray = cv2.cvtColor(frm, cv2.COLOR_BGR2GRAY)

        faces = cascade.detectMultiScale(gray, 1.3, 2)

```

```

    for x,y,w,h in faces:
        cv2.rectangle(frm, (x,y), (x+w, y+h), (0,255,0), 2)
        roi = gray[y:y+h, x:x+w]

        label = recog.predict(roi)

        if label[1] < 100:
            cv2.putText(frm, f'{labelslist[str(label[0])]} +
{int(label[1])}', (x,y), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,255), 3)
        else:
            cv2.putText(frm, "unkown", (x,y),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,255), 3)

    cv2.imshow("identify", frm)

    if cv2.waitKey(1) == 27:
        cv2.destroyAllWindows()
        cap.release()
        break

def maincall():

    root = tk.Tk()

    root.geometry("480x100")
    root.title("identify")

    label = tk.Label(root, text="Select below buttons ")

```

```

label.grid(row=0, columnspan=2)
label_font = font.Font(size=35, weight='bold',family='Helvetica')
label['font'] = label_font

btn_font = font.Font(size=25)

button1 = tk.Button(root, text="Add Member ", command=collect_data,
height=2, width=20)
button1.grid(row=1, column=0, pady=(10,10), padx=(5,5))
button1['font'] = btn_font

button2 = tk.Button(root, text="Start with known ",
command=identify, height=2, width=20)
button2.grid(row=1, column=1,pady=(10,10), padx=(5,5))
button2['font'] = btn_font
root.mainloop()

return

```

## **Main.py**

```
import tkinter as tk
import tkinter.font as font
from in_out import in_out
from motion import noise
from rect_noise import rect_noise
from record import record
from PIL import Image, ImageTk
from find_motion import find_motion
from identify import maincall

window = tk.Tk()
window.title("Smart cctv")
window.iconphoto(False, tk.PhotoImage(file='mn.png'))
window.geometry('1080x700')

frame1 = tk.Frame(window)

label_title = tk.Label(frame1, text="Smart cctv Camera")
label_font = font.Font(size=35, weight='bold',family='Helvetica')
label_title['font'] = label_font
label_title.grid(pady=(10,10), column=2)

icon = Image.open('icons/spy.png')
icon = icon.resize((150,150), Image.ANTIALIAS)
icon = ImageTk.PhotoImage(icon)
```

```
label_icon = tk.Label(frame1, image=icon)
label_icon.grid(row=1, pady=(5,10), column=2)
```

```
btn1_image = Image.open('icons/lamp.png')
btn1_image = btn1_image.resize((50,50), Image.ANTIALIAS)
btn1_image = ImageTk.PhotoImage(btn1_image)
```

```
btn2_image = Image.open('icons/rectangle-of-cuttet-line-geometrical-
shape.png')
btn2_image = btn2_image.resize((50,50), Image.ANTIALIAS)
btn2_image = ImageTk.PhotoImage(btn2_image)
```

```
btn5_image = Image.open('icons/exit.png')
btn5_image = btn5_image.resize((50,50), Image.ANTIALIAS)
btn5_image = ImageTk.PhotoImage(btn5_image)
```

```
btn3_image = Image.open('icons/security-camera.png')
btn3_image = btn3_image.resize((50,50), Image.ANTIALIAS)
btn3_image = ImageTk.PhotoImage(btn3_image)
```

```
btn6_image = Image.open('icons/incognito.png')
btn6_image = btn6_image.resize((50,50), Image.ANTIALIAS)
btn6_image = ImageTk.PhotoImage(btn6_image)
```

```
btn4_image = Image.open('icons/recording.png')
btn4_image = btn4_image.resize((50,50), Image.ANTIALIAS)
btn4_image = ImageTk.PhotoImage(btn4_image)
```

```
btn7_image = Image.open('icons/recording.png')
```

```

btn7_image = btn7_image.resize((50,50), Image.ANTIALIAS)
btn7_image = ImageTk.PhotoImage(btn7_image)

# ----- Button -----#
btn_font = font.Font(size=25)
btn1 = tk.Button(frame1, text='Monitor', height=90, width=180,
fg='green',command = find_motion, image=btn1_image, compound='left')
btn1['font'] = btn_font
btn1.grid(row=3, pady=(20,10))

btn2 = tk.Button(frame1, text='Rectangle', height=90, width=180,
fg='orange', command=rect_noise, compound='left', image=btn2_image)
btn2['font'] = btn_font
btn2.grid(row=3, pady=(20,10), column=3, padx=(20,5))

btn_font = font.Font(size=25)
btn3 = tk.Button(frame1, text='Noise', height=90, width=180, fg='green',
command=noise, image=btn3_image, compound='left')
btn3['font'] = btn_font
btn3.grid(row=5, pady=(20,10))

btn4 = tk.Button(frame1, text='Record', height=90, width=180, fg='orange',
command=record, image=btn4_image, compound='left')
btn4['font'] = btn_font
btn4.grid(row=5, pady=(20,10), column=3)

btn6 = tk.Button(frame1, text='In Out', height=90, width=180, fg='green',
command=in_out, image=btn6_image, compound='left')

```

```
btn6['font'] = btn_font
btn6.grid(row=5, pady=(20,10), column=2)

btn5 = tk.Button(frame1, height=90, width=180, fg='red',
command=window.quit, image=btn5_image)
btn5['font'] = btn_font
btn5.grid(row=6, pady=(20,10), column=2)

btn7 = tk.Button(frame1, text="identify", fg="orange",command=maincall,
compound='left', image=btn7_image, height=90, width=180)
btn7['font'] = btn_font
btn7.grid(row=3, column=2, pady=(20,10))

frame1.pack()
window.mainloop()
```

## **CHAPTER-7**

### **TEST AND EVALUATION**

#### **7.1 OPEN THE COMMAND PROMPT**

- Now enter main.py.
- It opens the Graphical User Interface that we created.
- We can see the 7 options we created in it.
- Those 7 options are Monitor, Identification, Rectangle, Noise, In and Out, Record, Exit.

#### **7.2 IF WE CHOOSE MONITOR BUTTON**

- It takes a photo initially just after starting.
- Count down starts after something is added or removed from the initially taken photo.
- The added or removed thing is shown in rectangle box.

#### **7.3 IF WE CHOOSE IDENTIFICATION BUTTON**

- After choosing identification button it shows further two more options.
- They are either the person is New or Existing user.
- It takes upto 300 photos continuously if it is a new user.
- And it trains the system in such a way that it recognize the person whenever that person appears again.

#### **7.4 IF WE CHOOSE RECTANGLE BUTTON**

- We have to fix the two points using the cursor and right and left keys.
- And we consider those two points as a diagonal of a rectangle.
- If some motion is appeared in that rectangle region then it captures the image.

#### **7.5 IF WE CHOOSE NOISE BUTTON**

- It results in capturing the whole screen not a particular region that make this feature different from rectangular button.

#### **7.6 IF WE CHOOSE RECORD BUTTON**

- It starts recording when this button is pressed.

#### **7.7 IF WE CHOOSE EXIT BUTTON**

- We get exit from that window.



## CHAPTER – 8

### RESULTS

This is where we open the command prompt and enter to the GUI we created.

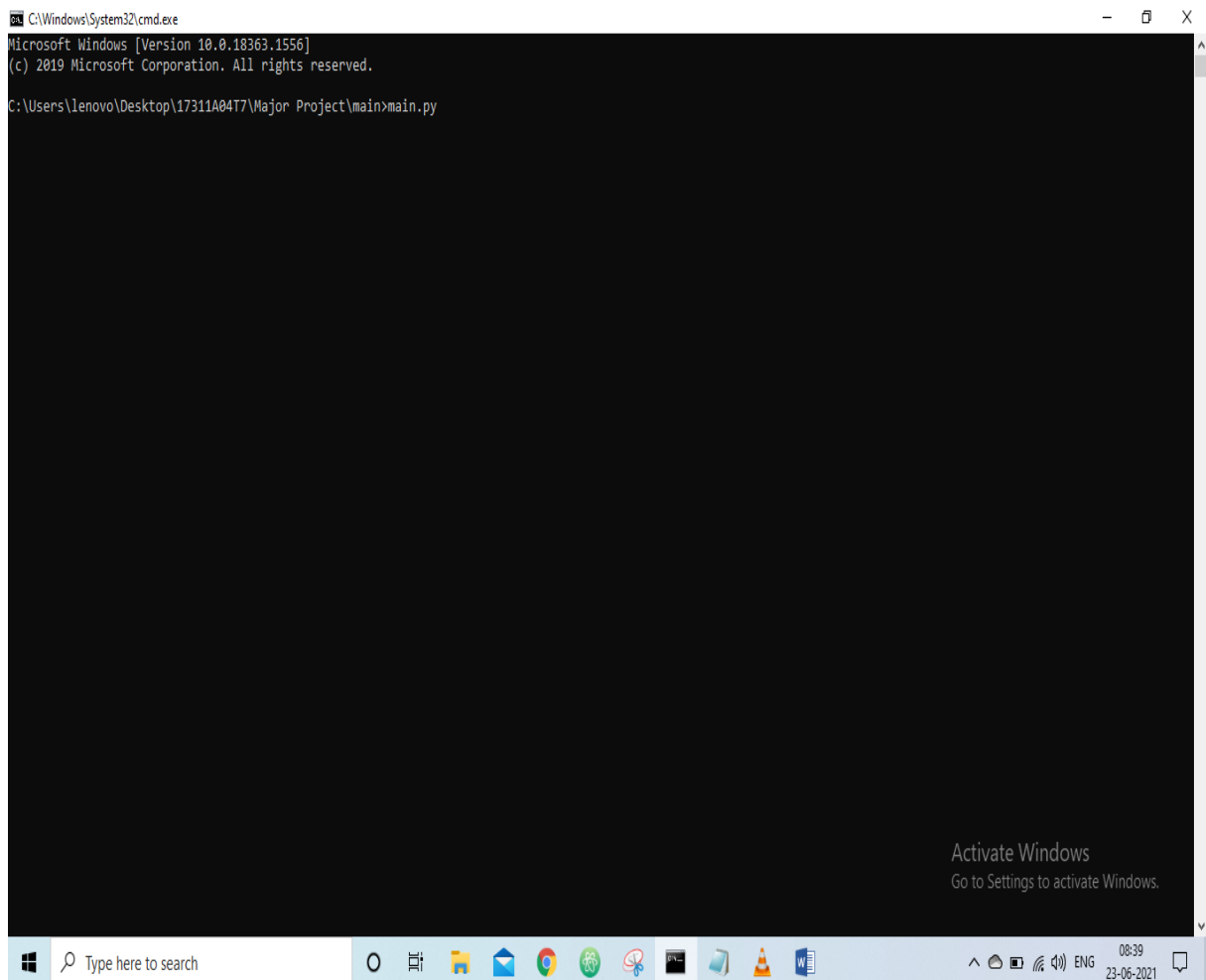


Figure 8.1 Command prompt

This is the interface which we created to check those features which we added for a cctv camera.



Figure 8.2 GUI Interface

## 8.1 MONITOR

This is the initial picture taken before the book in the frame is not moved.

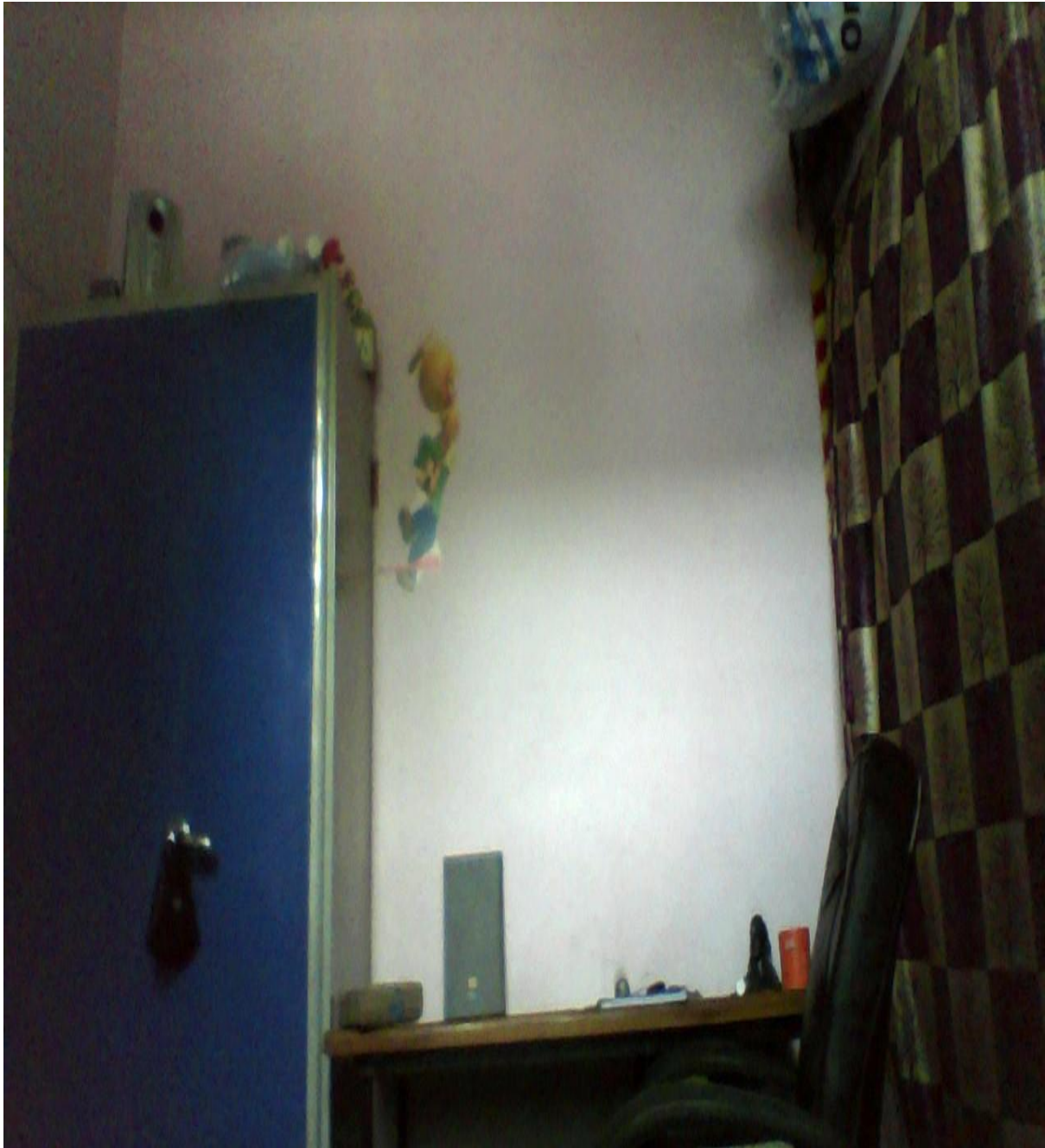


Figure 8.3 Monitor the area it covers

This is the picture taken after book is being taken off from the frame



Figure 8.4 Book has been removed from frame



This is the result image that mainly concentrates on the rectangular box that consists of book that is not present in the 2 fig.



Figure 8.5 It detects that book had been removed

## 8.2 IDENTIFICATION

This is the GUI of identification of a person.

There are the two more options where we have to choose one of it.

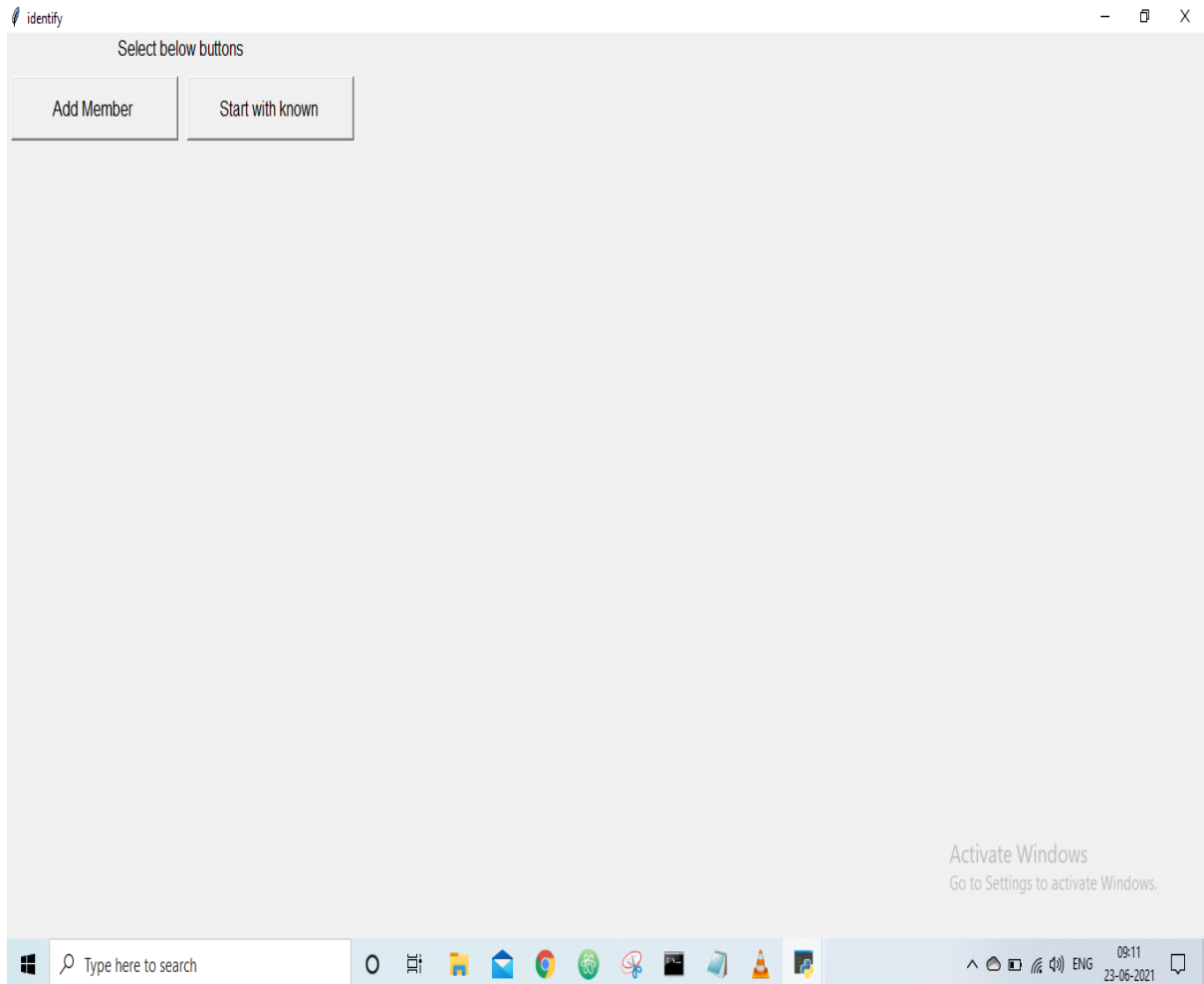


Figure 8.6 GUI of identification

If we choose the Add Member then it shows as that of below shown picture.  
There we have to enter the name of the new member that we want to add.

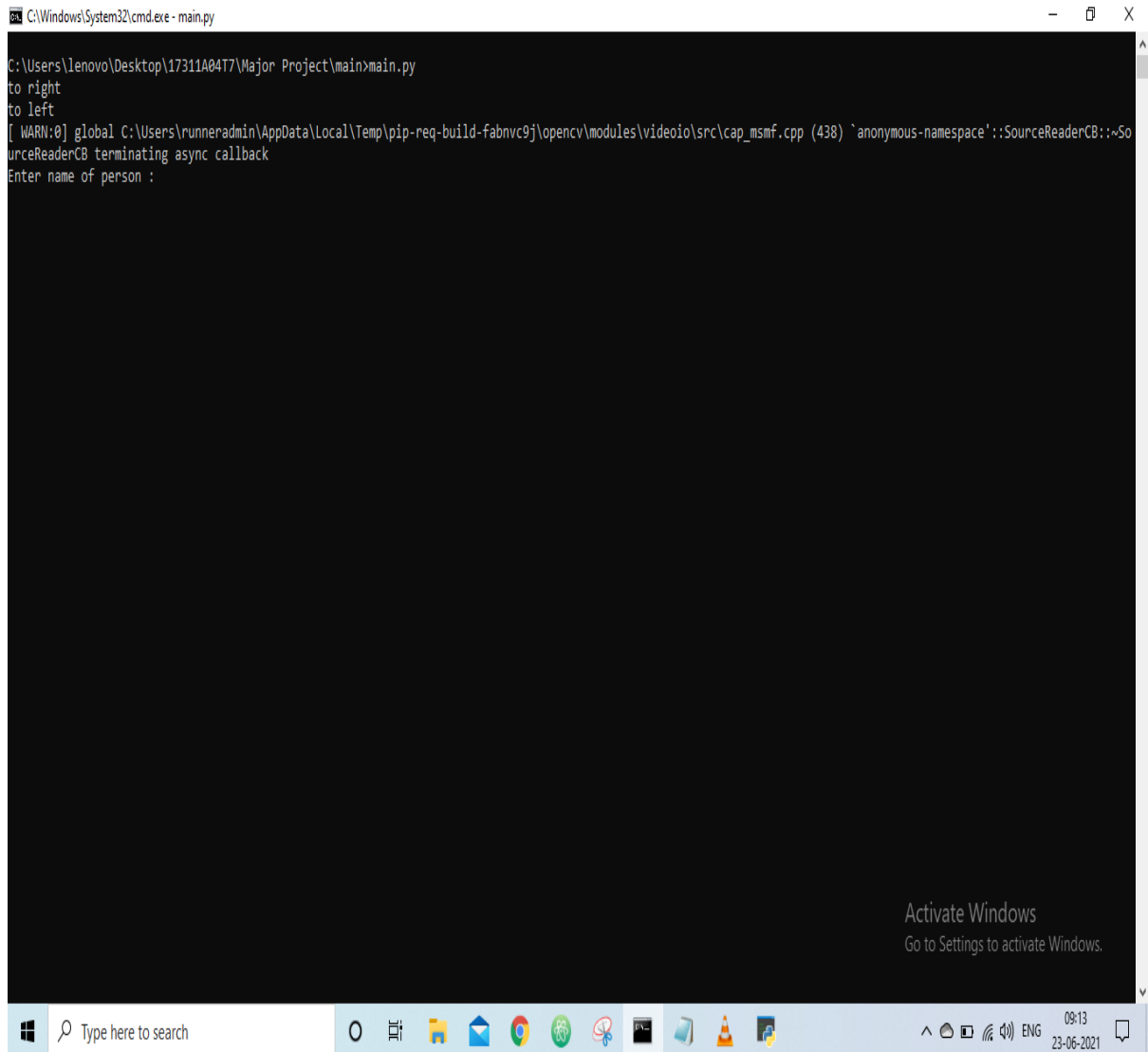
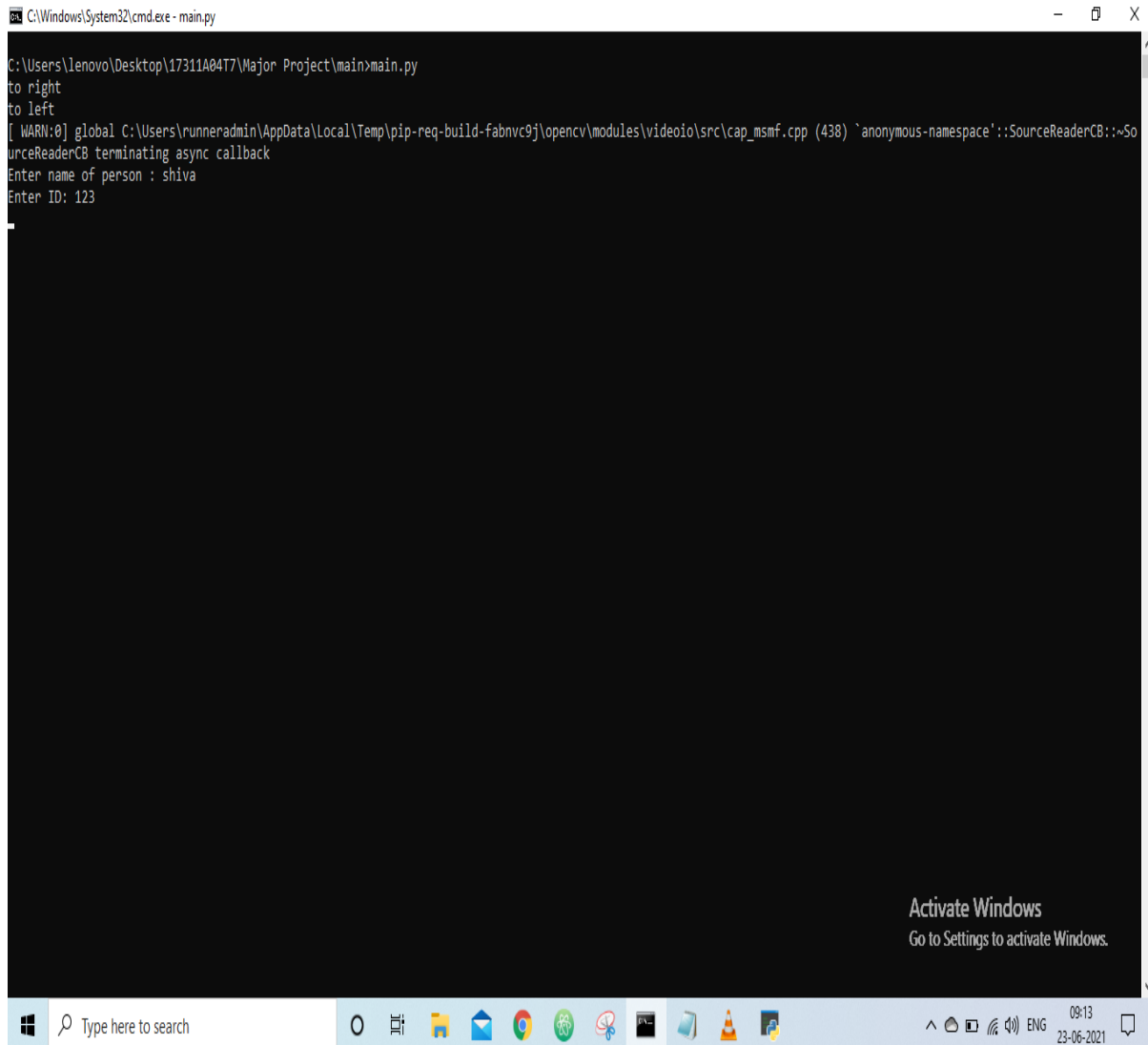


Figure 8.7 Asks for New Member details

Here we are going to add a new member and giving them a unique ID.



```
C:\Windows\System32\cmd.exe - main.py

C:\Users\lenovo\Desktop\17311A0477\Major Project\main>main.py
to right
to left
[ WARN:0] global C:\Users\runneradmin\AppData\Local\Temp\pip-req-build-fabnvc9j\opencv\modules\videoio\src\cap_msmf.cpp (438) `anonymous-namespace'::SourceReaderCB::~SourceReaderCB terminating async callback
Enter name of person : shiva
Enter ID: 123
```

Figure 8.8 Asks for New Member details



Here the camera captures the 300 photos immediately and trains itself to remember the new member.

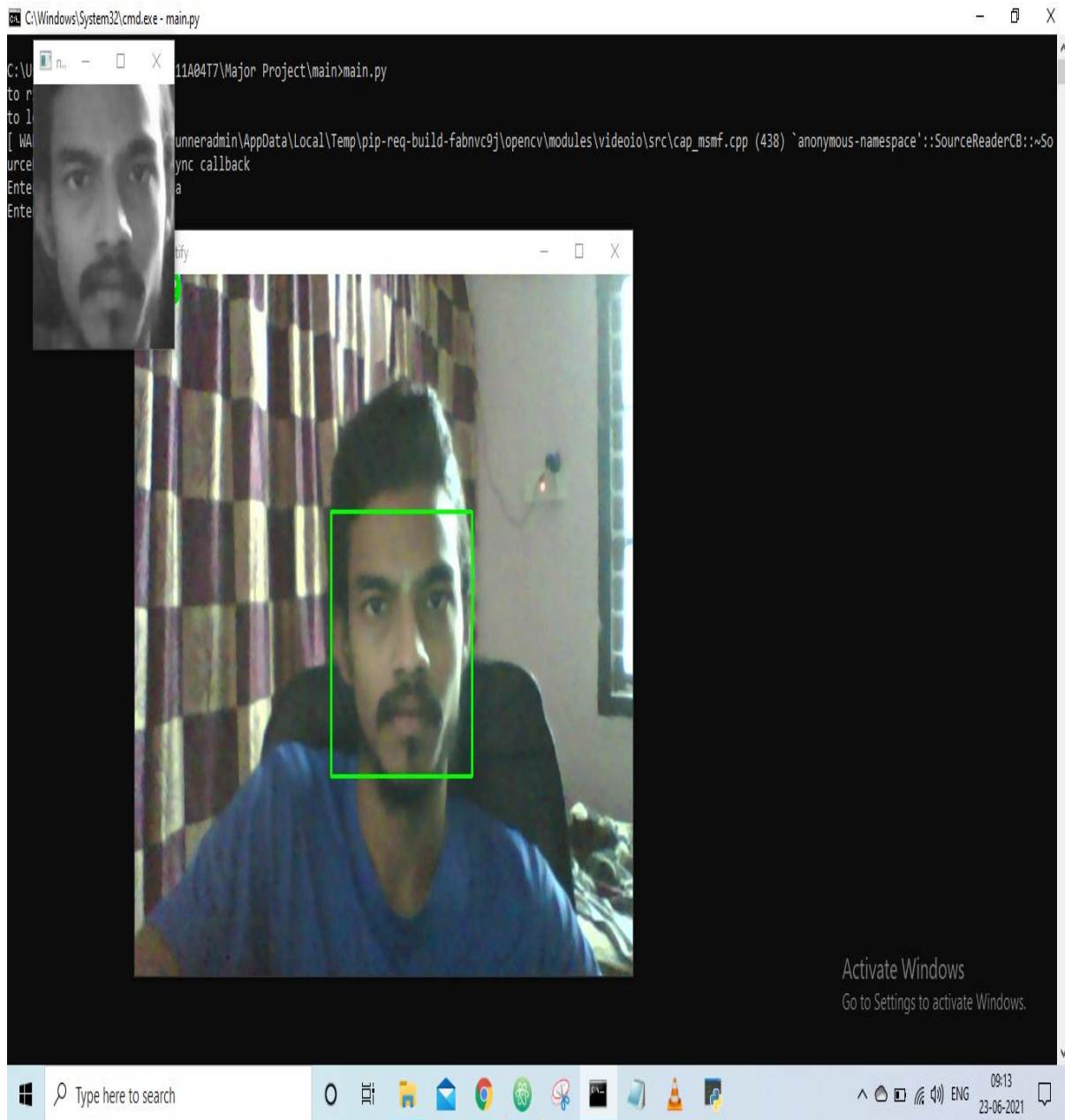
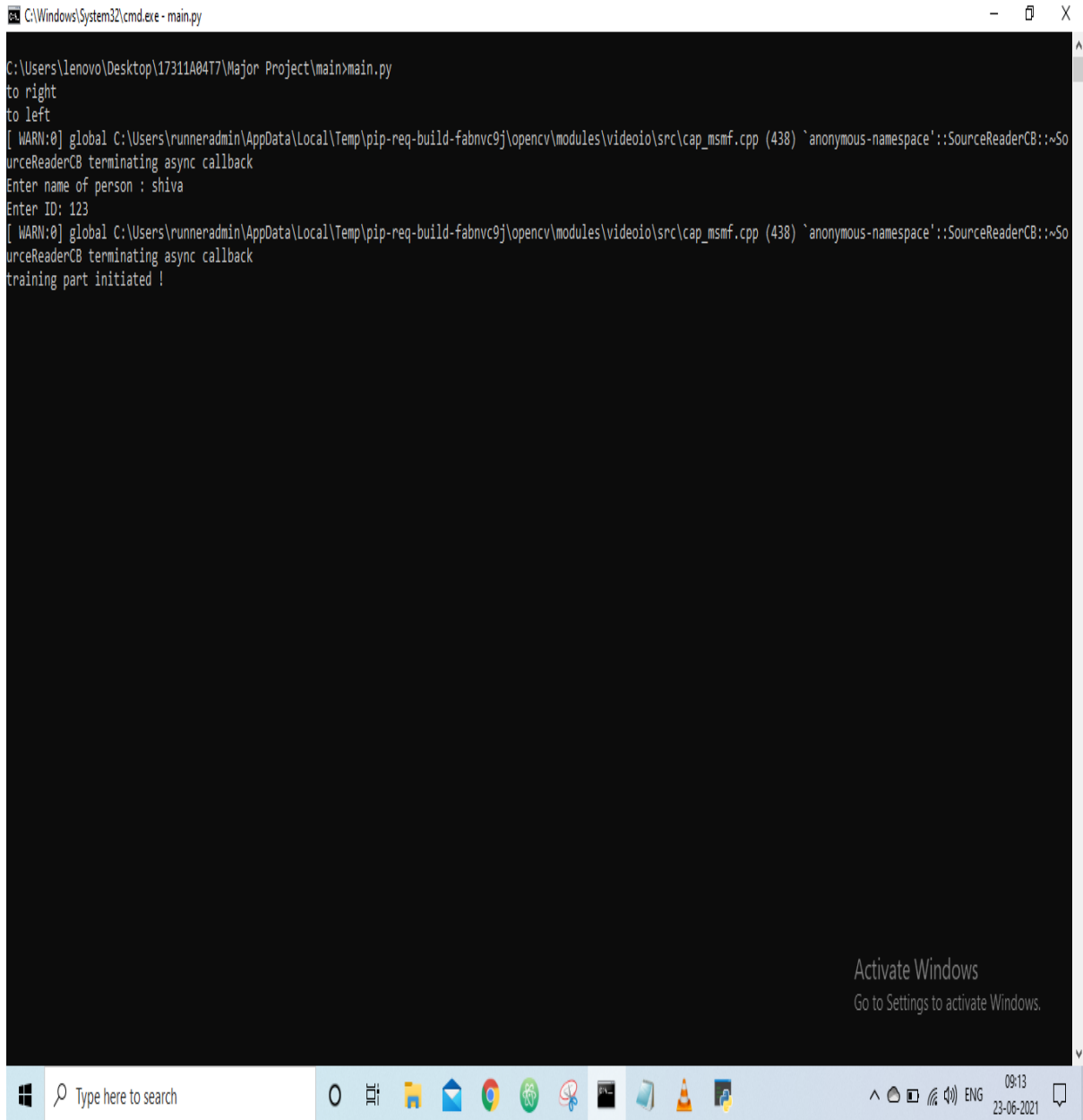


Figure 8.9 Capturing of images of new member

Here the training part is being initiated.



```
C:\Windows\System32\cmd.exe - main.py

C:\Users\lenovo\Desktop\17311A04T7\Major Project\main>main.py
to right
to left
[ WARN:0] global C:\Users\runneradmin\AppData\Local\Temp\pip-req-build-fabnvc9j\opencv\modules\videoio\src\cap_msmf.cpp (438) 'anonymous-namespace':SourceReaderCB::~SourceReaderCB terminating async callback
Enter name of person : shiva
Enter ID: 123
[ WARN:0] global C:\Users\runneradmin\AppData\Local\Temp\pip-req-build-fabnvc9j\opencv\modules\videoio\src\cap_msmf.cpp (438) 'anonymous-namespace':SourceReaderCB::~SourceReaderCB terminating async callback
training part initiated !

Activate Windows
Go to Settings to activate Windows.
```

Figure 8.10 Training part is being initiated

Here the person is being identified where he appeared in the frame

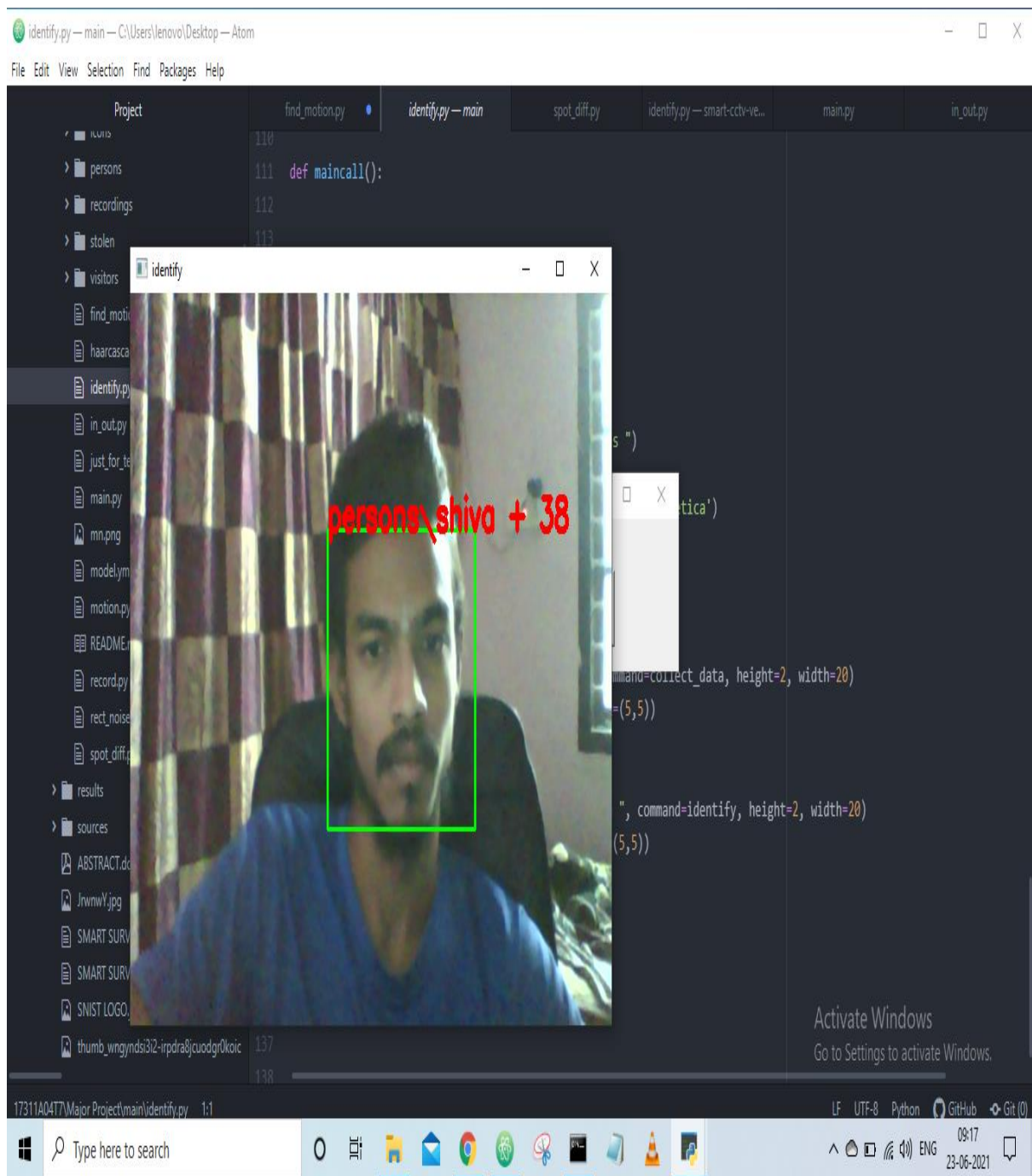


Figure 8.11 Person is being identified

### 8.3 RECTANGLE NOISE

It is the third feature it senses the motion in a particular region that region we choose by fixing the two points by right and left keys.



Figure 8.12 No motion is being identified in Rectangle Box

Here the motion of hand is being detected and that motion happened in the rectangular box.



Figure 8.13 Motion is being identified in Rectangle Box



## 8.4 NOISE

Here it shows that there is no motion in this picture.

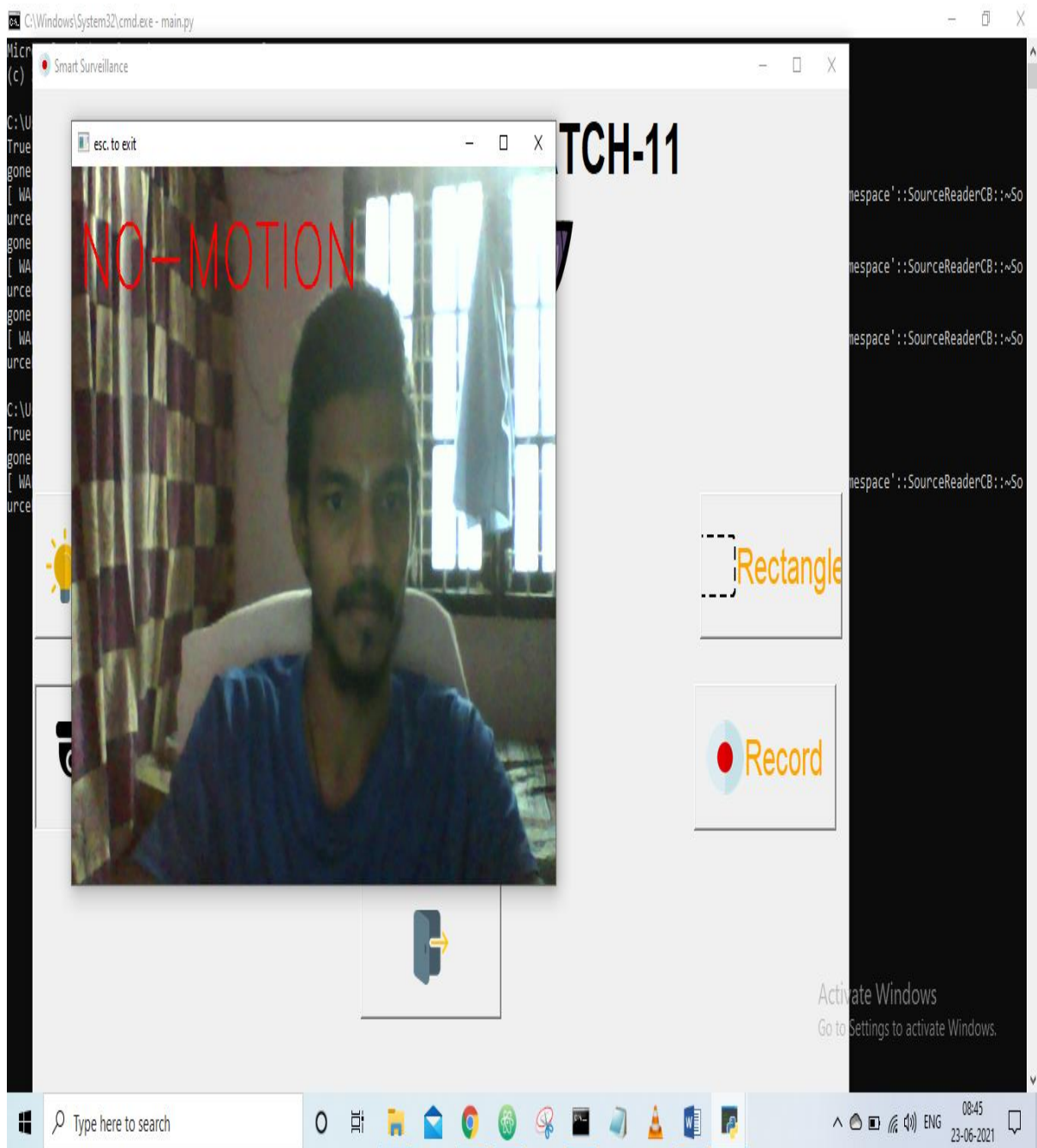


Figure 8.14 No motion is being identified in whole frame

In this picture there slight motion of is being observed by the camera and it detected it.

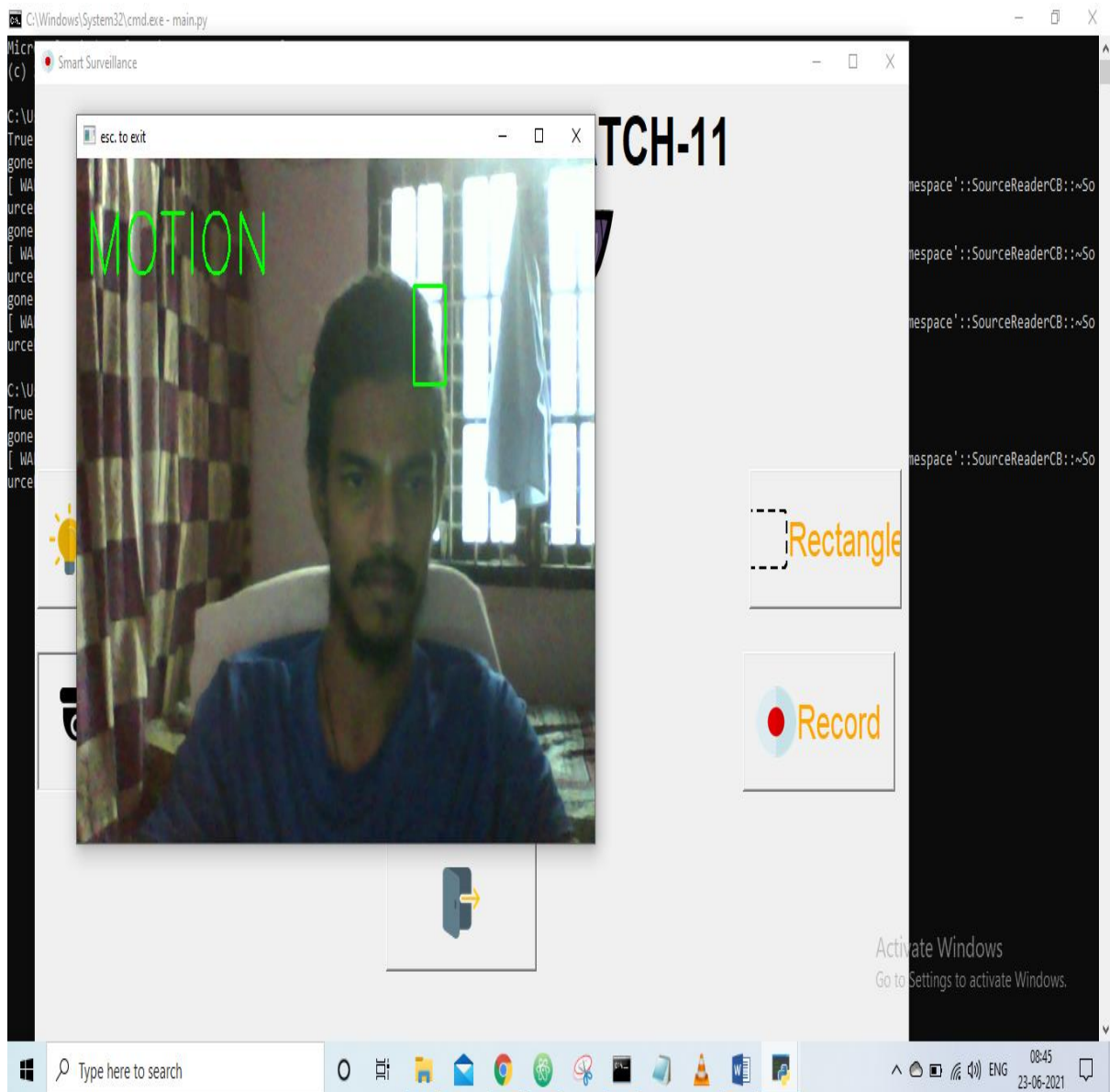
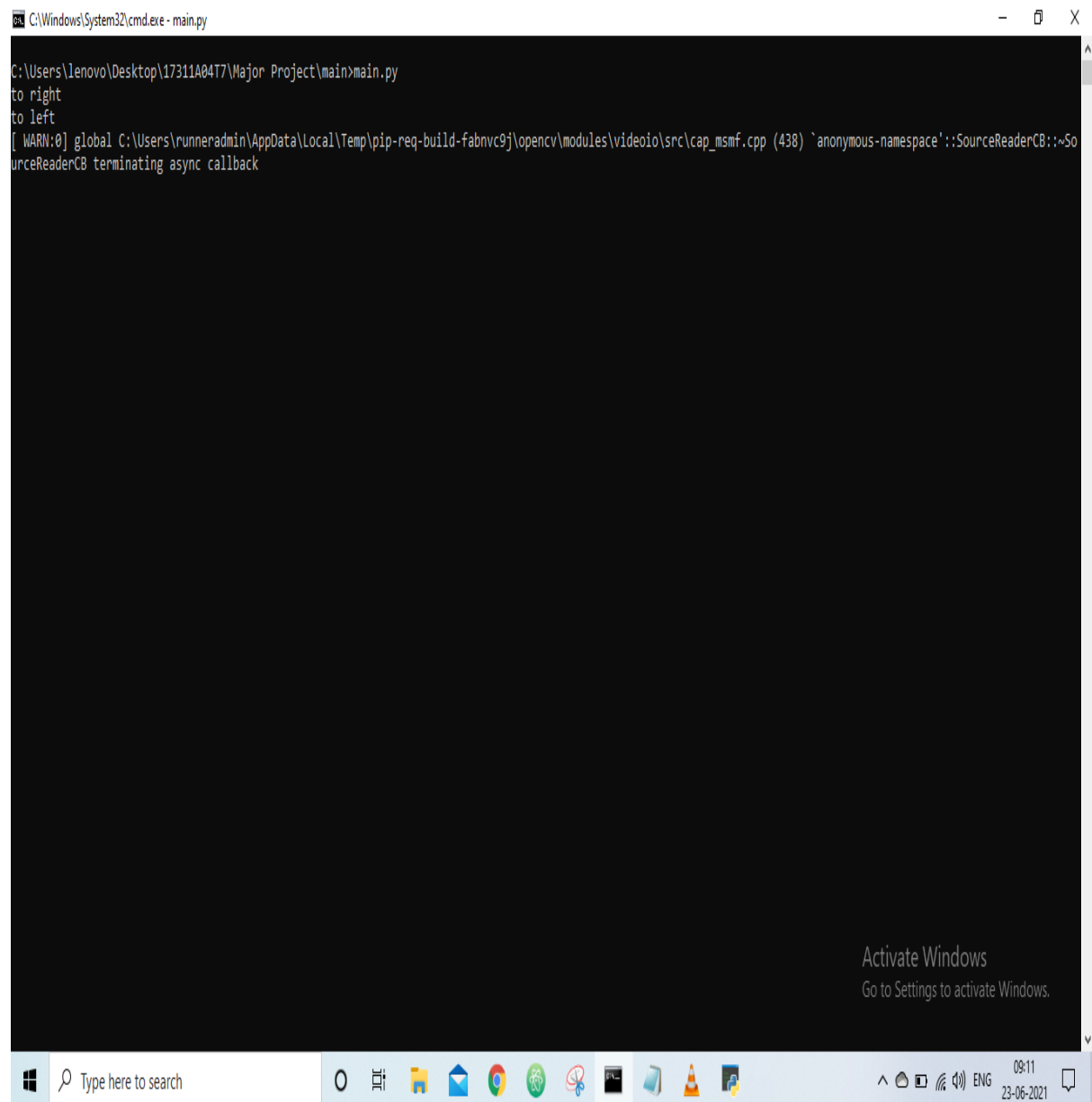


Figure 8.15 Motion is being identified

## 8.5 IN AND OUT

Here we use these commands to detect the motion from right to left or left to right.



```
C:\Windows\System32\cmd.exe - main.py
C:\Users\lenovo\Desktop\17311A04T7\Major Project\main>main.py
to right
to left
[ WARN:0] global C:\Users\runneradmin\AppData\Local\Temp\pip-req-build-fabnvc9j\opencv\modules\videoio\src\cap_msmf.cpp (438) 'anonymous-namespace':::SourceReaderCB::~SourceReaderCB terminating async callback
```

The screenshot shows a Windows command prompt window titled "C:\Windows\System32\cmd.exe - main.py". The user has navigated to the directory "C:\Users\lenovo\Desktop\17311A04T7\Major Project\main" and executed the command "main.py". The output of the script shows "to right" and "to left" on separate lines, followed by a warning message from OpenCV: "[ WARN:0] global C:\Users\runneradmin\AppData\Local\Temp\pip-req-build-fabnvc9j\opencv\modules\videoio\src\cap\_msmf.cpp (438) 'anonymous-namespace':::SourceReaderCB::~SourceReaderCB terminating async callback". The Windows taskbar at the bottom shows the search bar, task view, and several application icons. The system tray on the right indicates the time as 09:11 on 23-06-2021, with the language set to ENG. An "Activate Windows" watermark is visible in the bottom right corner of the command prompt window.

Figure 8.16 Commands for in and out motion



Here the inward motion is being detected as towards left.



Figure 8.17 Person is moving in

Here the outward motion is being detected as towards right.



Figure 8.18 Person is moving out

This is the visitors files path created for in out feature.

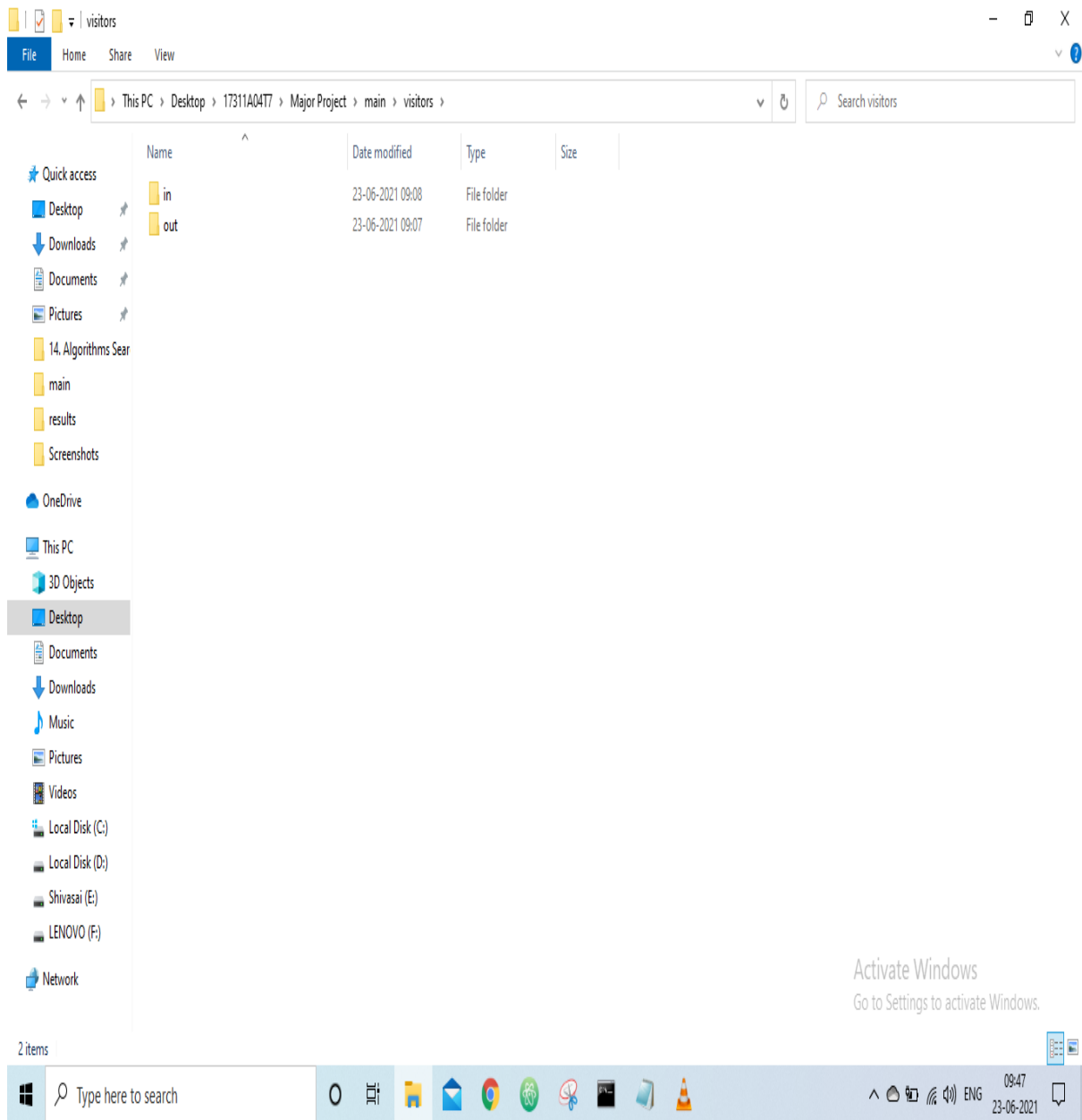


Figure 8.19 Visitors file

This is the overall files that consist of overall database regarding project

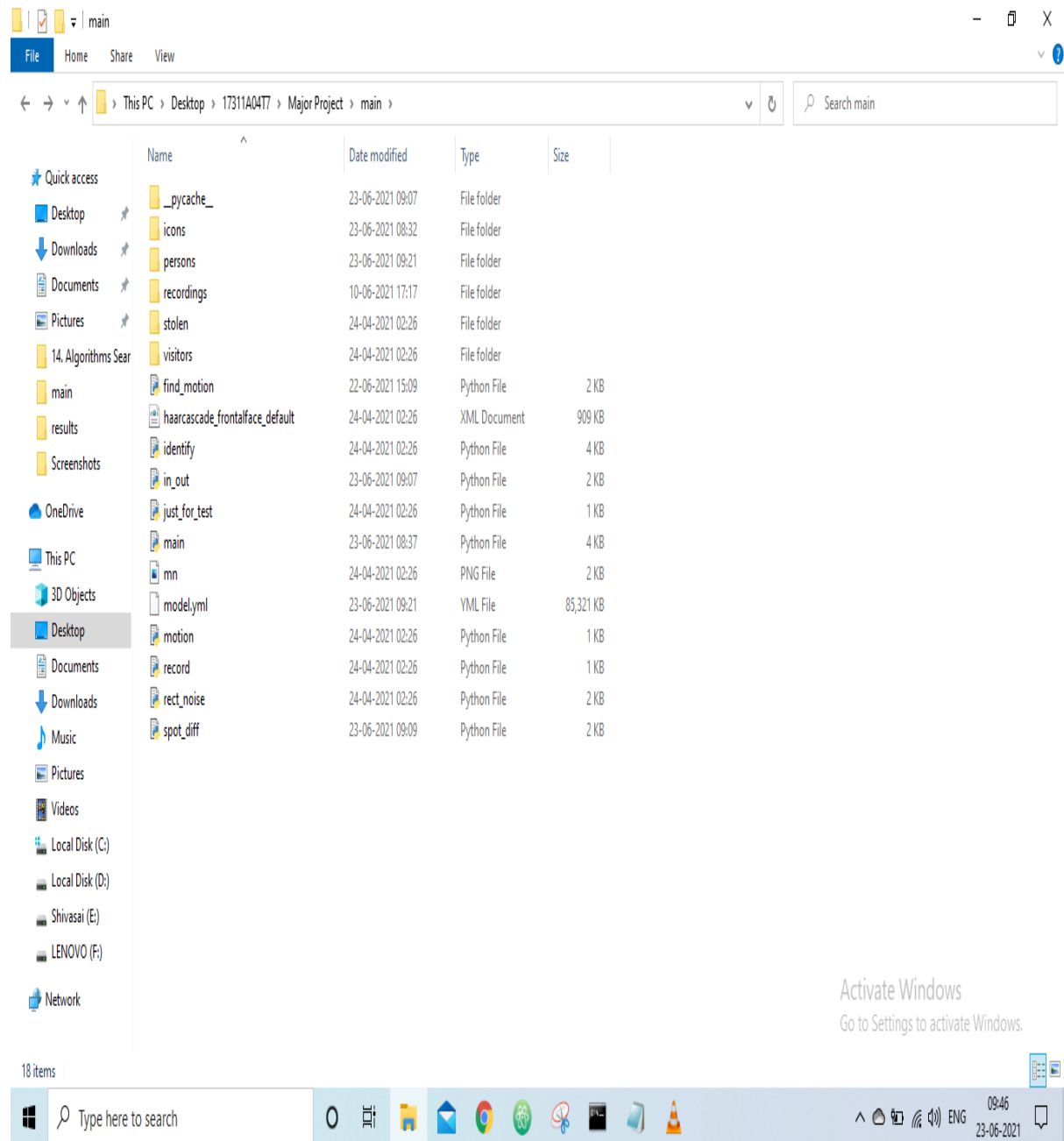


Figure 8.20 Complete Database

Storing of captured images with time and date.

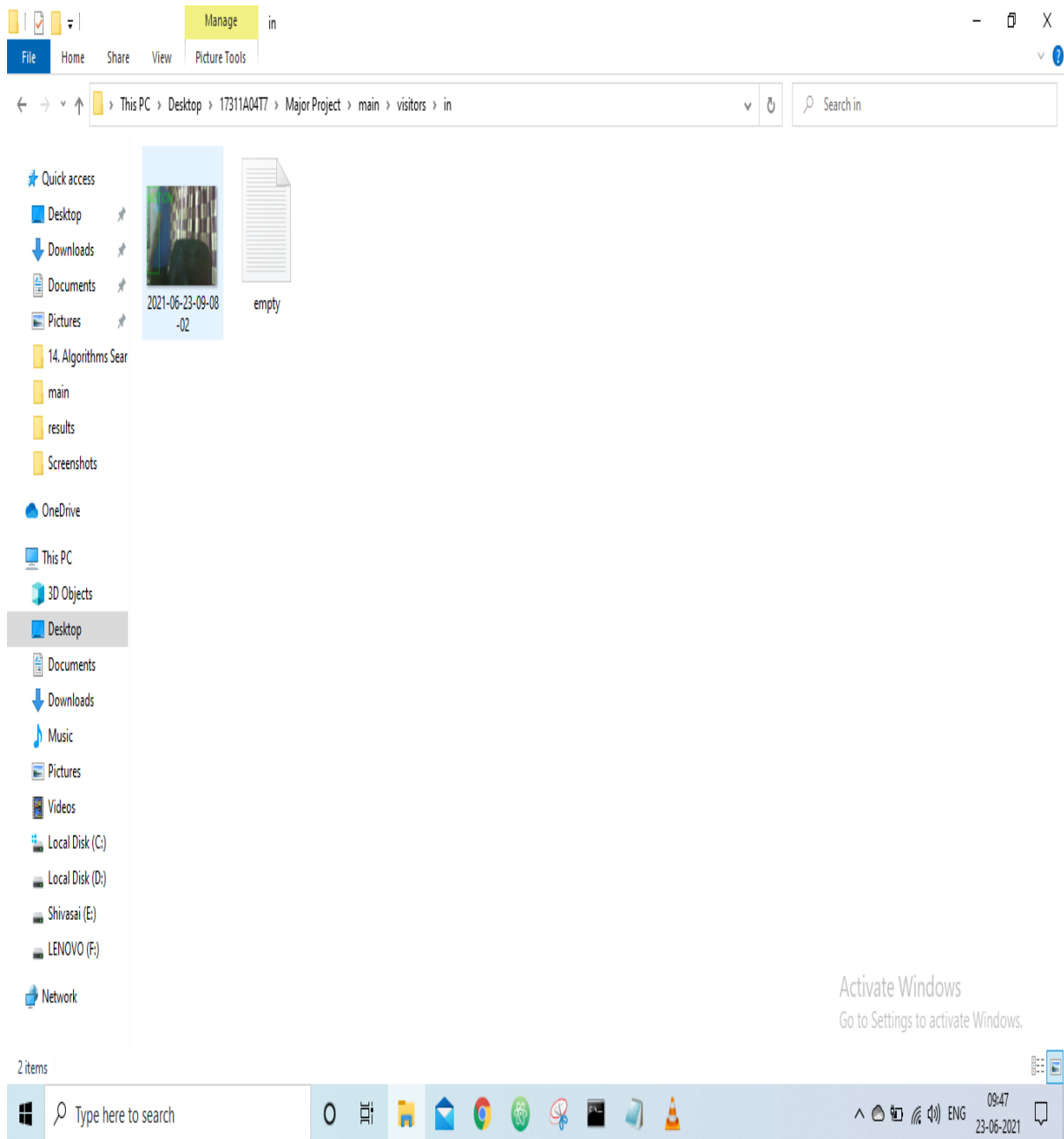


Figure 8.21 Captured images with date and time

## **CHAPTER-9**

### **ANALYSIS**

1. Written the code in such a way that it monitors the surroundings.
2. It identify the person whose is already pre member of it and take the data of new member.
3. It is detecting the motion of an object in a particular area or complete space it observes.
4. It is detecting whether the person is coming or coming out of a room or frame of vision.
5. It is absolutely recording the action going on when want to record it for further use.

## **CHAPTER-10**

### **FUTURE SCOPE**

This implementation can also be used to update the existing surveillance cameras and make them smart by adding computational power by using a Raspberry Pi. By using cameras with high resolution we can make decisions precisely and in a robust way.

This can be made as an IoT device by connecting it through a cloud and can be monitored remotely and get alerts using RPA (Robotic Process Automation) and can also be made to react to the situations using actuators. This can be clubbed with sensors such as IR and Ultrasonic to make it even more precise.

## **CHAPTER-11**

### **REFERENCES**

- [1] Yong-Liang Zhang, Zhi-Qin Zhang, Gang Xiao, Rui-Dong Wang, Xia He, “Perimeter Intrusion Detection based on Intelligent Video Analysis”, 15th International Conference on Control, Automation and Systems (ICCAS), 2015
- [2] Jayathi Ghosh Dastidar, Rana Biswas, “ Tracking Human Intrusion through a CCTV”, International Conference on Computational Intelligence and Communication Networks (CICN), 2015
- [3] Paul L. Rosin, Tim Ellis, “Detecting and Classifying Intruders in Image Sequences”, in Proceedings of the British Machine Vision Conference, 1991
- [4] Hang Chen, Dongfang Chen, Xiaofeng Wang, “Intrusion detection of specific area based on video”, 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISPBMEI), 2016
- [5] Jin-xiang Wang, “Research and implementation of intrusion detection algorithm in video surveillance”, International Conference on Audio, Language and Image Processing (ICALIP), 2016
- [6] Mozammel Chowdhury, Junbin Gao, Rafiqul Islam, “Human detection and localization in secure access control by analysing facial features”, IEEE 11th Conference on Industrial Electronics and Applications (ICIEA), 2016
- [7] Yang Xiao-Qiang , Zhang Xiao-Bing , “ Face Recognition Research Based on the Fusion of Layered LBP Feature”, International Conference on Industrial Informatics – Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII), 2017
- [8] Nikita Bakshi, Vibha Prabhu, “ Face recognition system for access control using principal component analysis”, International Conference on Intelligent Communication and Computational Techniques (ICCT), 2017
- [9] Junying Zeng, Xiaoxiao Zhao, Chuanbo Qin, “Single sample per person face recognition based on deep convolutional neural network”, 3<sup>rd</sup> IEEE International Conference on Computer and Communications (ICCC), 2017



# B11 REPORT

---

## ORIGINALITY REPORT

---

26%

SIMILARITY INDEX

20%

INTERNET SOURCES

10%

PUBLICATIONS

18%

STUDENT PAPERS

---

## PRIMARY SOURCES

---

1

[numpy.org](https://numpy.org)

Internet Source

3%

2

[medium.com](https://medium.com)

Internet Source

2%

3

Akshay Bharadwaj K H, Deepak, V  
Ghanavanth, Harish Bharadwaj R, R Uma,  
Gowranga Krishnamurthy. "Smart CCTV  
Surveillance System for Intrusion Detection  
With Live Streaming", 2018 3rd IEEE  
International Conference on Recent Trends in  
Electronics, Information & Communication  
Technology (RTEICT), 2018

Publication

2%

4

Submitted to University of Maryland,  
University College

Student Paper

1%

5

[python919.wordpress.com](https://python919.wordpress.com)

Internet Source

1%

6

[gist.github.com](https://gist.github.com)

Internet Source

1%

---

7	Submitted to October University for Modern Sciences and Arts (MSA) Student Paper	1 %
8	docplayer.net Internet Source	1 %
9	towardsdatascience.com Internet Source	1 %
10	Submitted to Istanbul Bilgi University Student Paper	1 %
11	www3.hs-albsig.de Internet Source	1 %
12	Submitted to Texas A & M University, Kingville Student Paper	1 %
13	norkandirblog.wordpress.com Internet Source	1 %
14	Submitted to Vel Tech University Student Paper	1 %
15	Submitted to University of Northumbria at Newcastle Student Paper	1 %
16	cheapjewelryus.com Internet Source	1 %
17	maelfabien.github.io Internet Source	<1 %

18	<a href="http://www.programcreek.com">www.programcreek.com</a> Internet Source	<1 %
19	<a href="http://answers.opencv.org">answers.opencv.org</a> Internet Source	<1 %
20	<a href="http://dspace.daffodilvarsity.edu.bd:8080">dspace.daffodilvarsity.edu.bd:8080</a> Internet Source	<1 %
21	Submitted to Universiti Kebangsaan Malaysia Student Paper	<1 %
22	Submitted to University of Nottingham Student Paper	<1 %
23	<a href="http://repositorium.sdum.uminho.pt">repositorium.sdum.uminho.pt</a> Internet Source	<1 %
24	<a href="http://stackoverflow.com">stackoverflow.com</a> Internet Source	<1 %
25	Submitted to University of Greenwich Student Paper	<1 %
26	Submitted to Sunway Education Group Student Paper	<1 %
27	<a href="http://theailearner.com">theailearner.com</a> Internet Source	<1 %
28	Submitted to Baylor University Student Paper	<1 %
29	<a href="http://repozitorij.unizg.hr">repozitorij.unizg.hr</a> Internet Source	<1 %

30	<a href="http://www.itread01.com">www.itread01.com</a> Internet Source	<1 %
31	Submitted to Bilkent University Student Paper	<1 %
32	Submitted to Cardiff University Student Paper	<1 %
33	Submitted to Harrisburg University of Science and Technology Student Paper	<1 %
34	<a href="http://publications.lib.chalmers.se">publications.lib.chalmers.se</a> Internet Source	<1 %
35	Submitted to University of Leeds Student Paper	<1 %
36	Submitted to University of South Australia Student Paper	<1 %
37	<a href="http://www.aiktcdspace.org:8080">www.aiktcdspace.org:8080</a> Internet Source	<1 %
38	<a href="http://www.learnpythonwithrune.org">www.learnpythonwithrune.org</a> Internet Source	<1 %
39	<a href="http://dev.to">dev.to</a> Internet Source	<1 %
40	<a href="http://git.kpi.fei.tuke.sk">git.kpi.fei.tuke.sk</a> Internet Source	<1 %
41	<a href="http://junyelee.blogspot.com">junyelee.blogspot.com</a> Internet Source	<1 %

<1 %

42

[www.squncle.com](http://www.squncle.com)

Internet Source

<1 %

43

Submitted to Barnsley College, South  
Yorkshire

Student Paper

<1 %

44

[acodigo.blogspot.com](http://acodigo.blogspot.com)

Internet Source

<1 %

45

[www.slideshare.net](http://www.slideshare.net)

Internet Source

<1 %

46

H. Nakamura, K. Takagi, K. Okada, Y.  
Matsushita. "Hierarchical group oriented key  
management method HGK", [1990]  
Proceedings of the Sixth Annual Computer  
Security Applications Conference, 1990

Publication

<1 %

47

[opencvjackychi.blogspot.com](http://opencvjackychi.blogspot.com)

Internet Source

<1 %

48

Debopama Ghosh, Suvrodepta  
Bhattacharya, Arkaprabha Lodh, Poulami  
Mitra, Debosama Ghosh, Writam Mallik.  
"Assisted Technological Headset using  
Internet of Things", 2018 IEEE 9th Annual  
Information Technology, Electronics and

<1 %

# Mobile Communication Conference (IEMCON), 2018

Publication

49

Submitted to Loughborough University

Student Paper

<1 %

50

Submitted to Universitas Muhammadiyah  
Yogyakarta

Student Paper

<1 %

51

school35blog.files.wordpress.com

Internet Source

<1 %

Exclude quotes Off

Exclude matches Off

Exclude bibliography Off