



FACE DETECTION AND RECOGNITION FOR CRIMINAL IDENTIFICATION SYSTEM

A PROJECT REPORT

Submitted by

RAHESH S	113320104074
PRAVEEN M	113320104072
MUTHUVEL M K	113320104065

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

VELAMMAL INSTITUTE OF TECHNOLOGY, CHENNAI

ANNA UNIVERSITY:: CHENNAI 600 025

MAY 2023

BONAFIDE CERTIFICATE

Certified that this project report “**FACE DETECTION AND RECOGNITION FOR CRIMINAL IDENTIFICATION SYSTEM**” is the bonafide work of ”**RAHESH S 113320104074, PRAVEEN M 113320104072, MUTHUVEL MK 113320104065**” who carried out the project work under my supervision.

SIGNATURE

Dr.V.P.GLADIS PUSHPARATHI,
M.TECH.,Ph.D.,
HEAD OF THE DEPARTMENT
Computer Science and Engineering.,
Velammal Institute of Technology
Velammal Gardens, Panchetti,
Chennai-601 204.

SIGNATURE

Mr. V. MANICKAVASAGAN,
M.E.,
ASSISTANT PROFESSOR
Computer Science and Engineering.,
Velammal Institute Technology
Velammal Gardens, Panchetti,
Chennai-601 204.

FACE DETECTION AND RECOGNITION FOR CRIMINAL IDENTIFICATION SYSTEM

VIVA-VOCE EXAMINATION

The viva-voce examination of this project work was done as a part of the Bachelor's Degree in Computer Science and Engineering held on

RAHESH .S	113320104074
PRAVEEN .M	113320104072
MUTHUVEL .M K	113320104065

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We are personally indebted to many who had helped us during the course of this project work. Our deepest gratitude to the God Almighty.

We are greatly and profoundly thankful to our beloved Chairman **Thiru.M.V.Muthuramalingam** for facilitating us with this opportunity. Our sincere thanks to our respected Director **Thiru. M. V. M. Sasi Kumar** who took keen interest on us. We are also thankful to our Advisors Prof. **K. Razak**, Shri. **M. Vasu** and our Principal **Dr. N. Balaji** and our Vice Principal **Dr.S.Soundararajan** for their never-ending encouragement which accelerates us towards innovation.

We are extremely thankful to our Head of the Department **Dr.V.P.Gladis Pushparathi**, Project Coordinator **Ms. Pranamita Nanda**, Assistant Professor, Computer Science and Engineering Department for their valuable teachings and suggestions. From the bottom of our heart, we would like to thank our guide **Mr. V .Manickavasagan** who has been the pillar of this project without whom we would not have been able to complete the project successfully.

The Acknowledgment would be incomplete if we would not mention word of thanks to our Parents, Teaching and Non-Teaching Staffs, Administrative Staffs and Friends who had motivated and lend their support throughout the project. Thank you one and all.

ABSTRACT

The process of identifying and spotting a criminal is slow and difficult. Criminals, these days are getting smarter by not leaving any form of biological evidence or fingerprint impressions on the crime scene. A quick and easy solution is using state-of-the-art face identification systems. With the advancement in security technology, CCTV cameras are being installed at most of the buildings and traffic lights for surveillance purposes. The video footage from the camera can be used to identify suspects, criminals, runaways, missing persons etc. This paper explores a way to develop a criminal identification system using ML and deep neural networks. The following method can be used as an elegant way to make law enforcement hassle-free.

We all know that our Face is a unique and crucial part of the human body structure that identifies a person. Therefore, we can use it to trace the identity of a criminal person. With the advancement in technology, we are placed CCTV at many public places to capture the criminal's crime. Using the previously captured faces and criminal's images that are available in the police station, the criminal face recognition system of can be implemented. In this paper, we propose an automatic criminal identification system for Police Department to enhance and upgrade the criminal distinguishing into a more effective and efficient approach. Using technology, this idea will add plus point in the current system while bringing criminals spotting to a whole new level by automating tasks Technology working behind it will be face recognition, from the footage captured by the CCTV cameras, our system will detect the face and recognize the criminal who is coming to that public place. The captured images of the person coming to that public place get compared with the criminal data we have in our database. If any person's face from public place matches, the system will display their image on the system screen and will give the message with their name that the criminal is found and present in this public place. This system matching more than 50% of the captured images with database images

TABLE OF CONTENT

CHAPTER NO	TITLE	PAGE NO
1	INTRODUCTION	2
	1.1 SCOPE OF THE PROJECT	2

2	LITERATURE SURVEY	3
3	SYSTEM ANALYSIS	5
	3.1 EXISTING SYSTEM	5
	3.2 PROBLEM STATEMENT	5
	3.3 PROPOSED SYSTEM	6
4	REQUIREMENT SPECIFICATION	7
	4.1 INTRODUCTION	7
	4.2 HARDWARE REQUIREMENTS	7
	4.3 SOFTWARE REQUIREMENTS	7
5	SYSTEM DESIGN	16
	5.1 BLOCK DIAGRAM	16
	5.2 DATAFLOW DIAGRAM	17
	5.3 STATE DIAGRAM	18
	5.4 USECASE DIAGRAM	19
6	SYSTEM IMPLEMENTATION	20
	6.1 MODULE IMPLEMENTATION	20
	6.2 MODULE DESCRIPTION	20
7	CONCLUSION AND FUTURE ENHANCEMENT	33
	7.1 CONCLUSION	33
	7.2 FUTURE ENHANCEMENT	33
8	APPENDICES	34
	APPENDIX – SAMPLE CODING	34

	APPENDIX – SNAP SHOTS	45
9	REFERENCES	48

CHAPTER 1

INTRODUCTION

Throughout the years, tracking down a criminal has been a difficult process. Earlier, the entire method consisted of leads based on evidence found on the crime scene. Biological evidence can be easily tracked down. However, criminals have evolved and are smarter than ever in terms of covering tracks and not leaving behind any kind of traceable evidence. Face recognition and detection come into play here. The face is significant for human identity and due to its distinguishable nature, every face is unique. Face recognition for criminal identification is one of a kind biometric technique that possesses the merit of high accuracy and low intrusiveness. It is a technique that uses the person's face to automatically detect and verify their identity from video frames or images.

The face identification system presented in this paper is a unique combination of the best techniques available today for face detection, feature extraction and finally classification. The deep learning methods like MTCNN for detection and FaceNet for embeddings have previously been proven to be elegant and state of the art.

1.1 SCOPE OF THE PROJECT:

The objective of face recognition is, from the incoming image, to find a series of data of the same face in a set of training images in a database. The great difficulty is ensuring that this process is carried out in real-time, something that is not available to all biometric face recognition software providers

CHAPTER 2

LITERATURE SURVEY

A literature review is a text of a scholarly paper, which includes the current knowledge including substantive findings, as well as theoretical and methodological contributions to a particular topic. Literature reviews are secondary sources and do not report new or original experimental work.

1.Title : Face Detection and Recognition System using Digital Image Processing

Authors Gurlove Singh and Amit Kumar Goel

Year: 2020

Description:

While recognizing any individual, the most important attribute is face. It serves as an individual identity of everyone and therefore face recognition helps in authenticating any person's identity using his personal characteristics. The whole procedure for authenticating any face data is sub-divided into two phases, in the first phase, the face detection is done quickly except for those cases in which the object is placed quite far, followed by this the second phase is initiated in which the face is recognized as an individual. Then the whole process is repeated thereby helping in developing a face recognition model which is considered to be one of the most extremely deliberated biometric technology. Basically, there are two types of techniques that are currently being followed in face recognition patterns that is, the Eigenface method and the Fisherface method. The Eigenface method basically makes use of the PCA (Principal Component Analysis) to minimize the face dimensional space of the facial features. The area of concern of this paper is using the digital image processing to develop a face recognition system.

2.Title :Face Detection and Recognition Using OpenCV

Authors: Madhura Mahajan, KTV Reddy, Manita Rajput

Year: 2018

Description:

Face detection and picture or video recognition is a popular subject of research on biometrics. Face recognition in a real-time setting has an exciting area and a rapidly growing challenge. Framework for the use of face recognition application authentication. This proposes the PCA (Principal Component Analysis) facial recognition system. The key component analysis (PCA) is a statistical method under the broad heading of factor analysis. The aim of the PCA is to reduce the large amount of data storage to the size of the feature space that is required to represent the data economically. The wide 1-D pixel vector made of the 2-D face picture in compact main elements of the space function is designed for facial recognition by the PCA. This is called a projection of self-space. The proper space is determined with the identification of the covariance matrix's own vectors, which are centered on a collection of fingerprint images. I build a camera-based real-time face recognition system and set an algorithm by developing programming on OpenCV, Haar Cascade, Eigenface, Fisher Face, LBPH, and Python.

CHAPTER 3

SYSTEM ANALYSIS

System analysis is the act, process, or profession of studying an activity (as a procedure, a business, or a physiological function) typically by mathematical means in order to define its goals or purposes and to discover operations and procedure for accomplishing them most efficiently. It is a process of collecting and interpreting facts, identifying the problems, and decomposition of a system into its components. System analysis is conducted for the purpose of studying a system or its parts in order to identify its objectives. It is a problem-solving technique that improves the system and ensures that all the components of the system work efficiently to accomplish their purpose.

3.1 EXISTING SYSTEM

- Detection of Crimes and its respective Criminals is slower process and it divides into three phases,
- The discovery that a crime has been committed.
- The identification of a suspect.
- The collection of sufficient evidence to indict the suspect.
- Forensic science plays an important role in the investigation of serious crimes. One of the first significant achievements in the field was the development of techniques for identifying individuals by their fingerprints.

3.2 PROBLEM STATEMENT

In smart cities, it is crucial to ensure the protection of women. To measure our heart rate, we are arriving with the palpitation technique. In order to protect against criminals, the nerve stimulator produces electric shock pulses.

3.3 PROPOSED SYSTEM

In this paper, we are Using CCTV cameras which are continuously working in a public place. In the Implementation of the system, we already saved criminal's images data with their names on photographs in the database. We are processing those images and extracting features from them and in feature extraction; we are taking the face encodings of the present images and saving them into one file using Pickle.

Using open-CV while capturing the footage in CCTV and captured images face encodings taken placed and comparing with our saved face encodings of the criminal database if any match is found then automatically on screen it will display an image of that criminal whose face matches and display the message with his name that criminal found, and his captured image will be saved into special folder police will go and catch him from that public place even if he once captured in the CCTV footage

CHAPTER 4

REQUIREMENT SPECIFICATIONS

4.1 INTRODUCTION

Requirement analysis determines the requirements of a new system. This project analyses on product and resource requirement, which is required for this successful system. The product requirement includes input and output requirements it gives the wants in term of input to produce the required output. The resource requirements give in brief about the software and hardware that are needed to achieve the required functionality.

4.2 HARDWARE REQUIREMENTS

- 4GB RAM (Minimum).
- 80GB HDD.
- Dual Core Processor.
- Camera (Webcam/CCTV).
- VGA (Video Graphics Array) Resolution monitor.

4.3 SOFTWARE REQUIREMENTS

- Microsoft Windows 10 (or) Higher.
- Python 3.5+.
- OpenCV 4.4.0.40.
- SQLite Studio.

Hardware Description :

Video Graphics Array :

VGA, which stands for video graphics array, is currently the most popular standard for PC screen display equipment. Technically, a VGA is a type of video adapter (circuitry in the screen). IBM developed the VGA for its PS/2 line of computers (the name Video Graphics Array is an IBM trademark), but loads of other manufacturers make VGA add-in boards (that plug into a slot in the pc) and VGA chips (in some pcs, these VGA chips are built right into the main part of the computer, the motherboard). A VGA monitor is a monitor that works with a VGA adapter.

A standard VGA system displays up to 640x 480 pixels (little dots) on the screen, with up to 16 different colors at a time. In lower resolution, 320x 200 pixels, the screen can show up to 256 colors at once. These specifications are much better than the older video adapter standards, the CGA and EGA, but they're not good enough for many people. If you're buying a new system or replacing an older video adapter, make sure you get a 'Super VGA' adapter which can handle higher resolutions (800x 600 or higher) and many more colors. Remember though, that the higher the resolution and the more colors you have to work with, the slower the display will function, and the more memory you'll need on the card.

Components of VGA:

The VGA consists of a couple of connectors with a VGA cable in between. The connectors facilitate the information exchange by connecting the destination and source. The VGA connector has a total of 15 pins in it to connect the VGA cable to the source device. The pins enable the transport of analog signals. The VGA connectors with pins are referred to as the “male” VGA connectors while the “female” connectors have holes in place of pins. The VGA connector also houses a couple of screws on both sides of the connector head, which houses the pins (holes in the case of female VGA). These screws are used to connect the cable to the source device.

The most commonly used VGA for most purposes is the typical 15 pin VGA connector, referred to as the DE-15. However, for smaller factor devices, VGA is available in the form of a lesser pinned connector called the DE-9. As the name suggests, the connector consists of only 9 pins

Additional VGA specifications include:

- 256 KB video random access memory (VRAM)
- 262,144 total colours
- 16-color and 256-color modes
- Master clock operating at 25.175 MHz or 28.322 MHz
- Planar mode
- Packed-pixel mode
- Up to 800 horizontal pixels
- Up to 600 lines
- Split screen support
- Refresh rates with a maximum of 70 Hz
- Support for smooth hardware scroll

Software Description :

Python :

Python, one of the most popular programming languages in the world, has created everything from Netflix's recommendation algorithm to the software that controls self-driving cars. Python is a general-purpose language, which means it's designed to be used in a range of applications, including data science, software and web development, automation, and generally getting stuff done. The block diagram representation of embedded system programming

Python is a computer programming language often used to build websites and software, automate tasks, and conduct data analysis. Python is a general-purpose language, meaning it can be used to create a variety of different programs and isn't specialized for any specific problems. This

versatility, along with its beginner-friendliness, has made it one of the most-used programming languages today.

Stack Overflow's 2022 Developer Survey revealed that Python is the fourth most popular programming language, with respondents saying that they use Python almost 50 percent of the time in their development work. Survey results also showed that Python is tied with Rust as the most-wanted technology, with 18% percent of developers who aren't using it already saying that they are interested in learning Python

What is Python used for?

Python is commonly used for developing websites and software, task automation, data analysis, and data visualization. Since it's relatively easy to learn, Python has been adopted by many non-programmers such as accountants and scientists, for a variety of everyday tasks, like organizing finances.

“Writing programs is a very creative and rewarding activity,” says University of Michigan and Coursera instructor Charles R Severance in his book *Python for Everybody*. “You can write programs for many reasons, ranging from making your living to solving a difficult data analysis problem to having fun to helping someone else solve a problem.”

What can you do with python? Some things include:

- Data analysis and machine learning
- Web development
- Automation or scripting
- Software testing and prototyping
- Everyday tasks

Here's a closer look at some of these common ways Python is used.

Data analysis and machine learning :

Python has become a staple in data science, allowing [data analysts](#) and other professionals to use the language to conduct complex statistical calculations, create data visualizations, build machine learning algorithms, manipulate and analyze data, and complete other data-related tasks.

Python can build a wide range of different data visualizations, like line and bar graphs, pie charts, histograms, and 3D plots. Python also has a number of libraries that enable coders to write programs for data analysis and machine learning more quickly and efficiently, like TensorFlow and Keras.

Web development :

Python is often used to develop the back end of a website or application—the parts that a user doesn't see. Python's role in web development can include sending data to and from servers, processing data and communicating with databases, URL routing, and ensuring security. Python offers several frameworks for web development. Commonly used ones include Django and Flask.

Some web development jobs that use Python include back-end engineers, full stack engineers, Python developers, software engineers, and DevOps engineers.

Automation or scripting :

If you find yourself performing a task repeatedly, you could work more efficiently by automating it with Python. Writing code used to build these automated processes is called scripting. In the coding world, automation can be used to check for errors across multiple files, convert files, execute simple math, and remove duplicates in data.

Python can even be used by relative beginners to automate simple tasks on the computer—such as renaming files, finding and downloading online content or sending emails or texts at desired intervals.

Software testing and prototyping :

In software development, Python can aid in tasks like build control, bug tracking, and testing. With Python, software developers can automate testing for new products or features. Some Python tools used for software testing include Green and Requestium.

Everyday tasks :

Python isn't only for programmers and data scientists. Learning Python can open new possibilities for those in less data-heavy professions, like journalists, small business owners, or social media marketers. Python can also enable non-programmers to simplify certain tasks in their lives. Here are just a few of the tasks you could automate with Python:

- Keep track of stock market or crypto prices
- Send yourself a text reminder to carry an umbrella anytime it's raining
- Update your grocery shopping list
- Renaming large batches of files
- Converting text files to spreadsheets
- Randomly assign chores to family members
- Fill out online forms automatically

OPEN CV :

OpenCV is a Python open-source library, which is used for computer vision in Artificial intelligence, Machine Learning, face recognition, etc. In OpenCV, the CV is an abbreviation form of a computer vision, which is defined as a field of study that helps computers to understand the content of the digital images such as photographs and videos.

The purpose of computer vision is to understand the content of the images. It extracts the description from the pictures, which may be an object, a text description, and three-dimension model, and so on. For example, cars can be facilitated with computer vision, which will be able to identify and different objects around the road, such as traffic lights, pedestrians, traffic signs, and so on, and acts accordingly.

Computer vision allows the computer to perform the same kind of tasks as humans with the same efficiency. There are a two main task which are defined below:

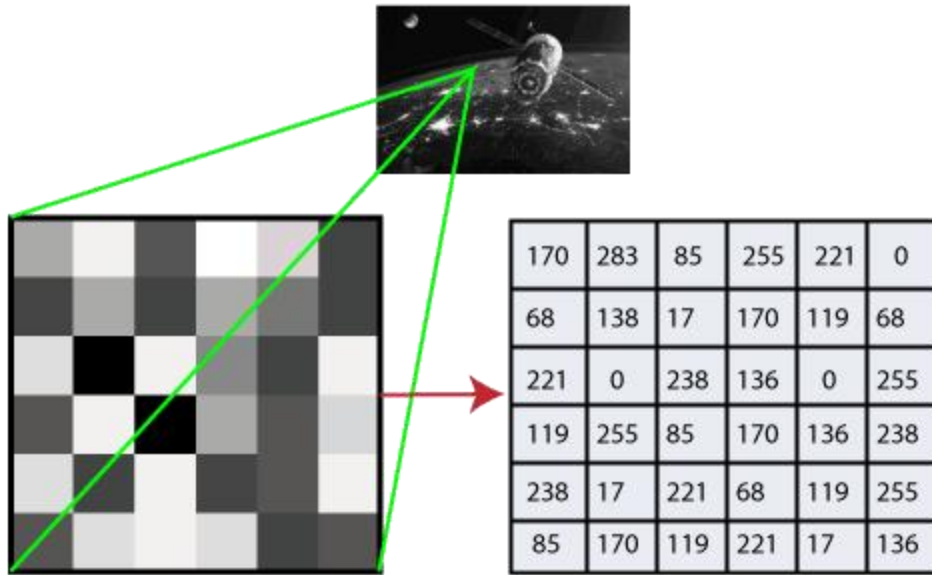
- **Object Classification** - In the object classification, we train a model on a dataset of particular objects, and the model classifies new objects as belonging to one or more of your training categories
- **Object Identification** - In the object identification, our model will identify a particular instance of an object

How OpenCV Works :

In this tutorial, we will learn how computers perform image recognition.

How does computer recognize the image?

Human eyes provide lots of information based on what they see. Machines are facilitated with seeing everything, convert the vision into numbers and store in the memory. Here the question arises how computer convert images into numbers. So the answer is that the pixel value is used to convert images into numbers. A pixel is the smallest unit of a digital image or graphics that can be displayed and represented on a digital display device.



The picture intensity at the particular location is represented by the numbers. In the above image, we have shown the pixel values for a grayscale image consist of only one value, the intensity of the black color at that location.

There are two common ways to identify the images:

1. Grayscale

Grayscale images are those images which contain only two colors black and white. The contrast measurement of intensity is black treated as the weakest intensity, and white as the strongest intensity. When we use the grayscale image, the computer assigns each pixel value based on its level of darkness.

2. RGB

An RGB is a combination of the red, green, blue color which together makes a new color. The computer retrieves that value from each pixel and puts the results in an array to be interpreted.

SQLite studio :

SQLiteStudio is desktop application to create, edit, and browse SQLite databases. Some of its features are:

- Encryption
- Scripting with JavaScript, Python, and Tcl
- Importing/exporting for various formats
- Cross-platform—Windows, MacOS, and Linux
- Data population for testing with pre-generated data

CHAPTER 5

SYSTEM DESIGN

5.1 BLOCK DIAGRAM

A block diagram is a visual representation of a system that uses simple, labeled blocks that represent single or multiple items, entities or concepts, connected by lines to show relationships between them.

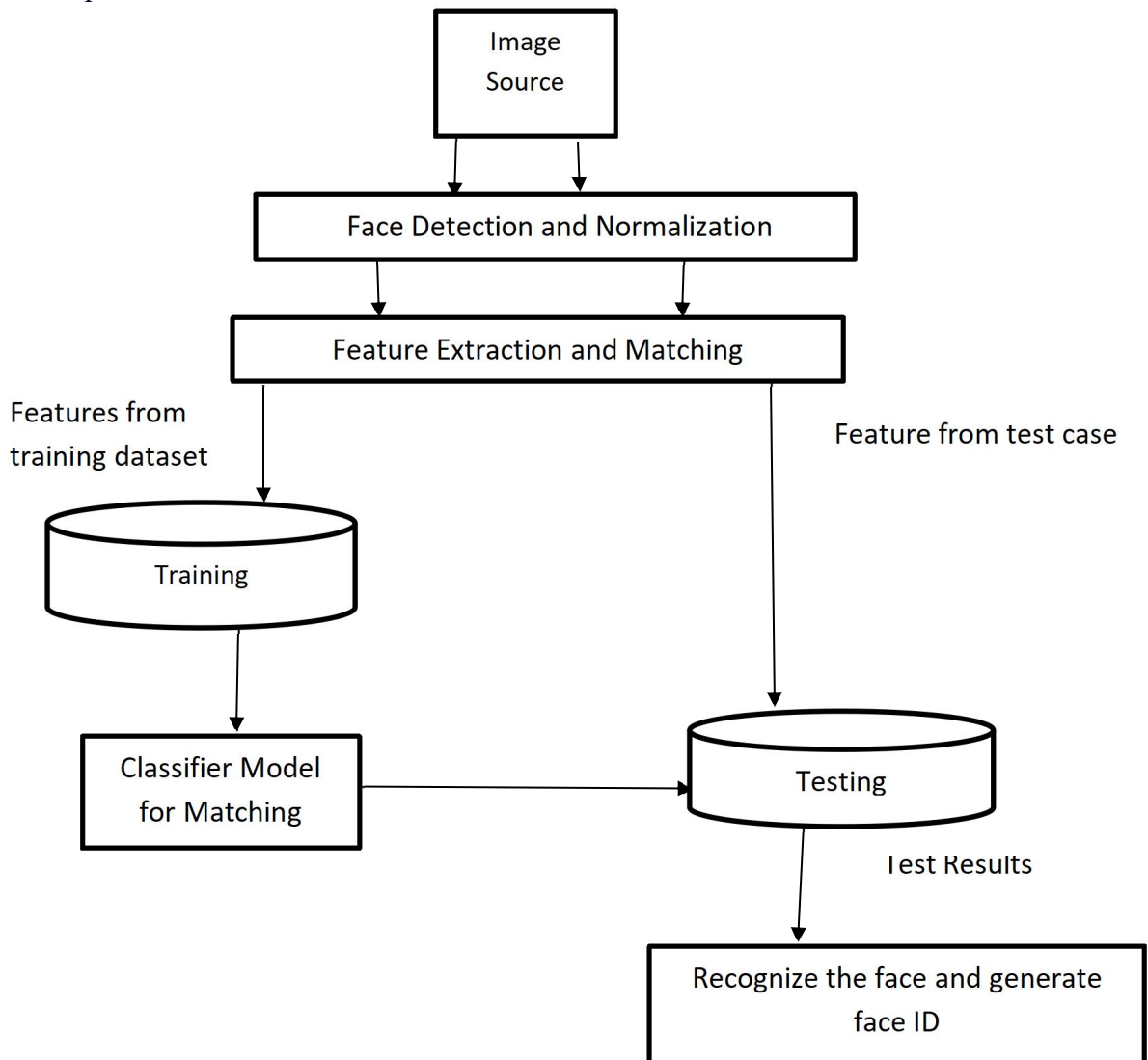


Fig:5.1 Block diagram

5.2 DATA FLOW DIAGRAM

Data flow across a process or system is depicted using a data-flow diagram, which is typically an information system. The DFD additionally gives details about each entity's inputs and outputs as well as the process itself. Data flow (flow, dataflow) depicts the movement of information (and occasionally, physical objects) from one system component to another.

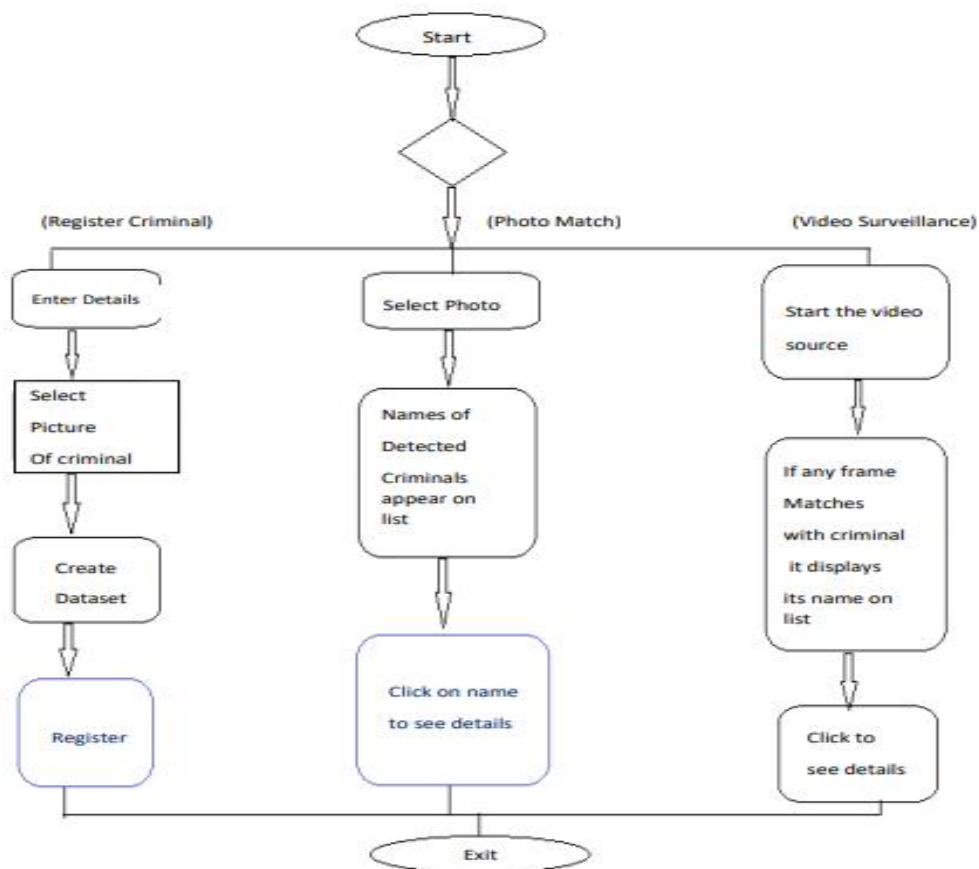


Fig 5.2 Data flow diagram

5.3 STATE DIAGRAM

A state diagram shows the behavior of classes in response to external stimuli. Specifically a state diagram describes the behavior of a single object in response to a series of events in a system. Sometimes it's also known as a Harel state chart or a state machine diagram. This UML diagram models the dynamic flow of control from state to state of a particular object within a system.

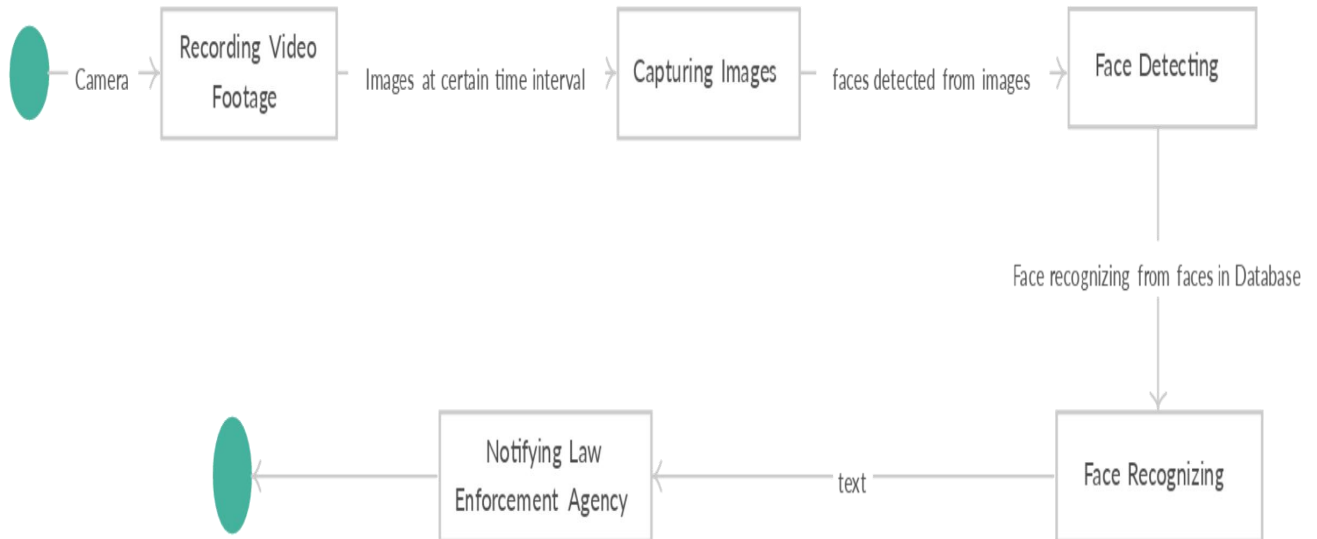


Fig 5.3 state diagram

5.4 USECASE DIAGRAM

Use-case diagrams describe the high-level functions and scope of a system. These diagrams also identify the interactions between the system and its actors. The use cases and actors in use-case diagrams describe what the system does and how the actors use it, but not how the system operates internally.

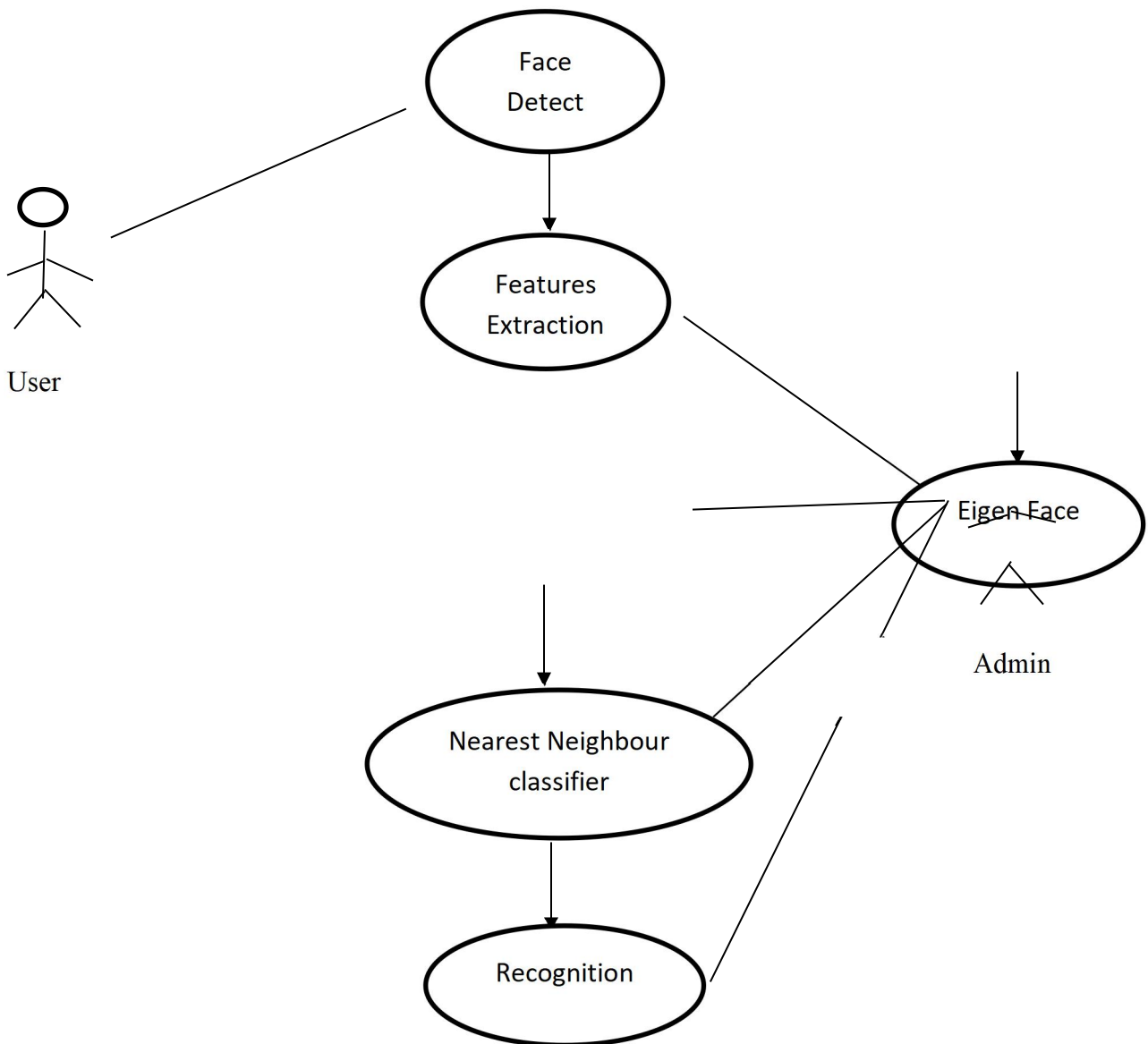


Fig 5.4 Use case diagram

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 MODULES IMPLEMENTATION

A modular design reduces complexity, facilitates change (a critical aspect of software maintainability), and results in easier implementation by encouraging parallel development of different parts of the system. Software with effective modularity is easier to develop because functions may be compartmentalized and interfaces are simplified. Software architecture embodies modularity that is software is divided into separately named and addressable components called modules that are integrated to satisfy problem requirements.

6.2 MODULES DESCRIPTION

The face detection and recognition module for a criminal identification system typically consists of several sub-modules that work together to identify and track individuals based on their facial features. Here are some of the key modules that may be included:

1. Image and video acquisition :

This module is responsible for acquiring and preprocessing images and video feeds from various sources such as cameras, videos, or images. It may also include functionalities like image resizing, rotation, and filtering to improve the quality of the input data.

2. Face detection :

This module is responsible for detecting and localizing faces in the acquired images or video feeds. It typically uses algorithms like Haar cascades or deep learning-based approaches like Convolutional Neural Networks (CNNs) to detect faces.

3. Face alignment:

This module is responsible for aligning the detected faces to a canonical pose. It corrects for factors like facial pose, rotation, and illumination, which can affect the accuracy of face recognition.

4. Feature extraction:

This module is responsible for extracting discriminative features from the detected and aligned faces. Common feature extraction techniques include Principal Component Analysis (PCA), Local Binary Patterns (LBP), and Histogram of Oriented Gradients (HOG).

5. Face recognition:

This module is responsible for comparing the extracted features of the detected faces with a database of known faces to identify potential matches. It typically uses algorithms like Euclidean distance or Support Vector Machines (SVM) to compare the features.

6. Database management:

This module is responsible for managing the database of known faces, including adding, deleting, and updating records.

7. Alert generation:

This module is responsible for generating alerts and notifications when a match is found between a detected face and a known criminal in the database

8. System management:

This module is responsible for managing the overall system, including configuring system settings, monitoring system performance, and handling system errors and exceptions.

These modules work together to create a robust and reliable face detection and recognition system for criminal identification. The accuracy and effectiveness of the system depend on the quality of the data, the design of the algorithms, and the optimization of the system parameters.

CHAPTER - 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

This paper presents an innovative approach to face recognition and how it can be implemented for an important purpose which is Criminal Detection and identification. Face Recognition technologies have a wide range of applications, similar approaches can be used for solving a lot of real-world problems. We believe developing a system as such is an enthusiastic step towards making the process of catching criminals and law enforcement speedy and efficient. This system can be further implemented to criminals in real-time using a dynamic dataset. Application of computer vision can be challenging but create solutions to difficult problems easier.

7.2 FUTURE ENHANCEMENT

- An elegant face identification system like this can be automated to detect criminals through CCTV cameras installed at multiple places.
- This system can also be used to detect missing people at time of disasters and mis-happenings.
- Criminal Identification system can also give details of where the criminals was exactly spotted using location of cameras.

The database can also incorporate more details than the present project of criminals such as age, last spotted, etc. to provide additional details of the criminal

APPENDIX - 1

SAMPLE CODING

```
httpimport tkinter as tk
from tkinter import filedialog
from tkinter import messagebox
from PIL import Image
from PIL import ImageTk
import threading
import shutil
from facerec import *
from register import *
from face_detection import *
# from dbHandler import *
from handler import *
import time
import csv
import numpy as np
import ntpath
import os
```

```

active_page = 0

thread_event = None

left_frame = None

right_frame = None

heading = None

webcam = None

img_label = None

img_read = None

img_list = []

slide_caption = None

slide_control_panel = None

current_slide = -1

root = tk.Tk()

root.geometry("1000x900+200+100")

# create Pages

pages = []

for i in range(5):

    pages.append(tk.Frame(root, bg="#3E3B3C"))

    pages[i].pack(side="top", fill="both", expand=True)

    pages[i].place(x=0, y=0, relwidth=1, relheight=1)

    def goBack():

        global active_page, thread_event, webcam

```

```

if (active_page==4 and not thread_event.is_set()):

    thread_event.set()

    webcam.release()

for widget in pages[active_page].winfo_children():

    widget.destroy()

pages[0].lift()

active_page = 0

def basicPageSetup(pageNo):

    global left_frame, right_frame, heading

    back_img = tk.PhotoImage(file= r"D:\FRD for criminal identification\img\back.png")

    back_button = tk.Button(pages[pageNo], image=back_img, bg="#3E3B3C", bd=0,
highlightthickness=0,

        activebackground="#3E3B3C", command=goBack)

    back_button.image = back_img

    back_button.place(x=10, y=10)

    heading = tk.Label(pages[pageNo], fg="white", bg="#3E3B3C", font="Arial 20 bold",
pady=10)

    heading.pack()

    content = tk.Frame(pages[pageNo], bg="#3E3B3C", pady=20)

    content.pack(expand="true", fill="both")

    left_frame = tk.Frame(content, bg="#3E3B3C")

    left_frame.grid(row=0, column=0, sticky="nsew")

```



```

right_frame = tk.LabelFrame(content, text="Detected Criminals", bg="#3E3B3C",
font="Arial 20 bold", bd=4,

                                foreground="#2ea3ef", labelanchor="n")

right_frame.grid(row=0, column=1, sticky="nsew", padx=20, pady=20)
content.grid_columnconfigure(0, weight=1, uniform="group1")
content.grid_columnconfigure(1, weight=1, uniform="group1")
content.grid_rowconfigure(0, weight=1)

def showImage(frame, img_size):

    global img_label, left_frame

    img = cv2.resize(frame, (img_size, img_size))

    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

    img = Image.fromarray(img)

    img = ImageTk.PhotoImage(img)

    if (img_label == None):

        img_label = tk.Label(left_frame, image=img, bg="#202d42")

        img_label.image = img

        img_label.pack(padx=20)

    else:

        img_label.configure(image=img)

        img_label.image = img

def getNewSlide(control):

    global img_list, current_slide

    if(len(img_list) > 1):

        if(control == "prev"):

```

```

current_slide = (current_slide-1) % len(img_list)

    else:

        current_slide = (current_slide+1) % len(img_list)

    img_size = left_frame.winfo_height() - 200

    showImage(img_list[current_slide], img_size)

    slide_caption.configure(text = "Image {} of {}".format(current_slide+1, len(img_list)))

def selectMultiImage(opt_menu, menu_var):

    global img_list, current_slide, slide_caption, slide_control_panel

    filetype = [("images", "*.jpg *.jpeg *.png")]

    path_list = filedialog.askopenfilenames(title="Choose atleast 5 images", filetypes=filetype)

    if(len(path_list) < 5):

        messagebox.showerror("Error", "Choose atleast 5 images.")

    else:

        img_list = []

        current_slide = -1

        # Resetting slide control panel

        if (slide_control_panel != None):

            slide_control_panel.destroy()

        # Creating Image list

        for path in path_list:

            img_list.append(cv2.imread(path))

        # Creating choices for profile pic menu

        menu_var.set("")

```

```

opt_menu['menu'].delete(0, 'end')

for i in range(len(img_list)):

    ch = "Image " + str(i+1)

    opt_menu['menu'].add_command(label=ch, command= tk._setit(menu_var, ch))

    menu_var.set("Image 1")

# Creating slideshow of images

img_size = left_frame.winfo_height() - 200

current_slide += 1

showImage(img_list[current_slide], img_size)

slide_control_panel = tk.Frame(left_frame, bg="#202d42", pady=20)

slide_control_panel.pack()

back_img = tk.PhotoImage(file="previous.png")

next_img = tk.PhotoImage(file="next.png")

prev_slide = tk.Button(slide_control_panel, image=back_img, bg="#202d42", bd=0,
highlightthickness=0,

                        activebackground="#202d42", command=lambda : getNewSlide("prev"))

prev_slide.image = back_img

prev_slide.grid(row=0, column=0, padx=60)

slide_caption = tk.Label(slide_control_panel, text="Image 1 of {}".format(len(img_list)),
fg="#ff9800",

                        bg="#202d42", font="Arial 15 bold")

slide_caption.grid(row=0, column=1)

next_slide = tk.Button(slide_control_panel, image=next_img, bg="#202d42", bd=0,
highlightthickness=0,

```

```

        activebackground="#202d42", command=lambda : getNewSlide("next"))

    next_slide.image = next_img

    next_slide.grid(row=0, column=2, padx=60)

def register(entries, required, menu_var):

    global img_list

    # Checking if no image selected

    if(len(img_list) == 0):

        messagebox.showerror("Error", "Select Images first.")

        return

    # Fetching data from entries

    entry_data = {}

    for i, entry in enumerate(entries):

        # print(i)

        val = entry[1].get()

        # print(val)

    if (len(val) == 0 and required[i] == 1):

        messagebox.showerror("Field Error", "Required field missing :\n\n%s" % (entry[0]))

        return

    else:

        entry_data[entry[0]] = val.lower()

    # Setting Directory

    path = os.path.join('face_samples', "temp_criminal")

    if not os.path.isdir(path):

```

```

os.mkdir(path)

no_face = []

for i, img in enumerate(img_list):

    # Storing Images in directory

    id = registerCriminal(img, path, i + 1)

    if(id != None):

        no_face.append(id)

# check if any image doesn't contain face

if(len(no_face) > 0):

    no_face_st = ""

    for i in no_face:

        no_face_st += "Image " + str(i) + ", "

    messagebox.showerror("Registration Error", "Registration failed!\n\nFollowing images
doesn't contain"

        " face or Face is too small:\n\n%s"%(no_face_st))

    shutil.rmtree(path, ignore_errors=True)

else:

    # Storing data in database

    insertData(entry_data)

    rowId=1

    if(rowId >= 0):

        messagebox.showinfo("Success", "Criminal Registered Successfully.")

        shutil.move(path, os.path.join('face_samples', entry_data["Name"]))

```

```

# save profile pic

profile_img_num = int(menu_var.get().split(' ')[1]) - 1

if not os.path.isdir("profile_pics"):

    os.mkdir("profile_pics")

    cv2.imwrite("profile_pics/criminal %d.png"%rowId, img_list[profile_img_num])

    goBack()

else:

    shutil.rmtree(path, ignore_errors=True)

    messagebox.showerror("Database Error", "Some error occured while storing data.")

## update scrollregion when all widgets are in canvas

def on_configure(event, canvas, win):

    canvas.configure(scrollregion=canvas.bbox('all'))

    canvas.itemconfig(win, width=event.width)

## Register Page ##

def getPage1():

    global active_page, left_frame, right_frame, heading, img_label

    active_page = 1

    img_label = None

    opt_menu = None

    menu_var = tk.StringVar(root)

    pages[1].lift()

    basicPageSetup(1)

    heading.configure(text="Register Criminal", bg="#3E3B3C")

```

```

right_frame.configure(text="Enter Details", fg="white", bg="#3E3B3C")

btn_grid = tk.Frame(left_frame, bg="#3E3B3C")

btn_grid.pack()

tk.Button(btn_grid, text="Select Images", command=lambda: selectMultiImage(opt_menu,
menu_var), font="Arial 15 bold", bg="#000000",

        fg="white", pady=10, bd=0, highlightthickness=0, activebackground="#3E3B3C",

        activeforeground="white").grid(row=0, column=0, padx=25, pady=25)

# Creating Scrollable Frame

canvas = tk.Canvas(right_frame, bg="#202d42", highlightthickness=0)

canvas.pack(side="left", fill="both", expand="true", padx=30)

scrollbar = tk.Scrollbar(right_frame, command=canvas.yview, width=20,
troughcolor="#3E3B3C", bd=0,

        activebackground="#3E3B3C", bg="#000000", relief="raised")

scrollbar.pack(side="left", fill="y")

scroll_frame = tk.Frame(canvas, bg="#3E3B3C", pady=20)

scroll_win = canvas.create_window((0, 0), window=scroll_frame, anchor='nw')

canvas.configure(yscrollcommand=scrollbar.set)

canvas.bind('<Configure>', lambda event, canvas=canvas, win=scroll_win:
on_configure(event, canvas, win))

tk.Label(scroll_frame, text="* Required Fields", bg="#3E3B3C", fg="yellow", font="Arial
13 bold").pack()

# Adding Input Fields

input_fields = ("Name", "Father's Name", "Gender", "DOB(yyyy-mm-dd)", "Crimes Done",
"Profile Image")

ip_len = len(input_fields)

```

```

required = [1, 1, 1, 1, 1, 1]

entries = []

for i, field in enumerate(input_fields):

    print()

    row = tk.Frame(scroll_frame, bg="#3E3B3C")

    row.pack(side="top", fill="x", pady=15)

    label = tk.Text(row, width=20, height=1, bg="#3E3B3C", fg="#ffffff", font="Arial 13",
highlightthickness=0, bd=0)

    label.insert("insert", field)

    label.pack(side="left")

    if(required[i] == 1):

        label.tag_configure("star", foreground="yellow", font="Arial 13 bold")

        label.insert("end", " *", "star")

        label.configure(state="disabled")

    if(i != ip_len-1):

        ent = tk.Entry(row, font="Arial 13", selectbackground="#90ceff")

        ent.pack(side="right", expand="true", fill="x", padx=10)

        entries.append((field, ent))

    else:

        menu_var.set("Image 1")

        choices = ["Image 1"]

        opt_menu = tk.OptionMenu(row, menu_var, *choices)

        opt_menu.pack(side="right", fill="x", expand="true", padx=10)

```



```
    opt_menu.configure(font="Arial 13", bg="#000000", fg="white", bd=0,
highlightthickness=0, activebackground="#3E3B3C")

    menu = opt_menu.nametowidget(opt_menu.menuname)

    menu.configure(font="Arial 13", bg="white", activebackground="#90ceff", bd=0)

# print(entries)

tk.Button(scroll_frame, text="Register", command=lambda: register(entries, required,
menu_var), font="Arial 15 bold",
```

APPENDIX 2

SNAP SHOTS

Login Page

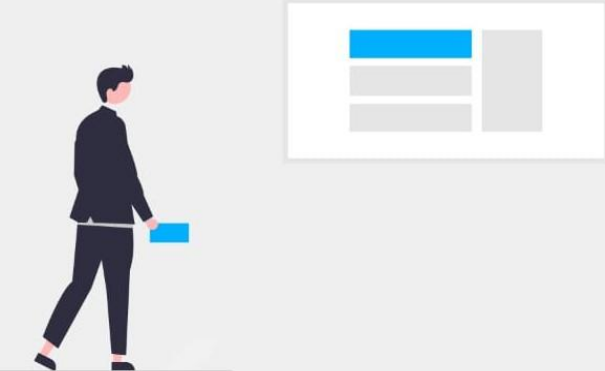


Login ID

Password 

Fig 9.1 Login page

Register Page



Name

Rank

Police ID


Date Of Birth

State

District

Branch

Gender ☒ Male ☐ Female

Password 

Re-Password

Phone Number

Email ID

Fig 9.2 Register page

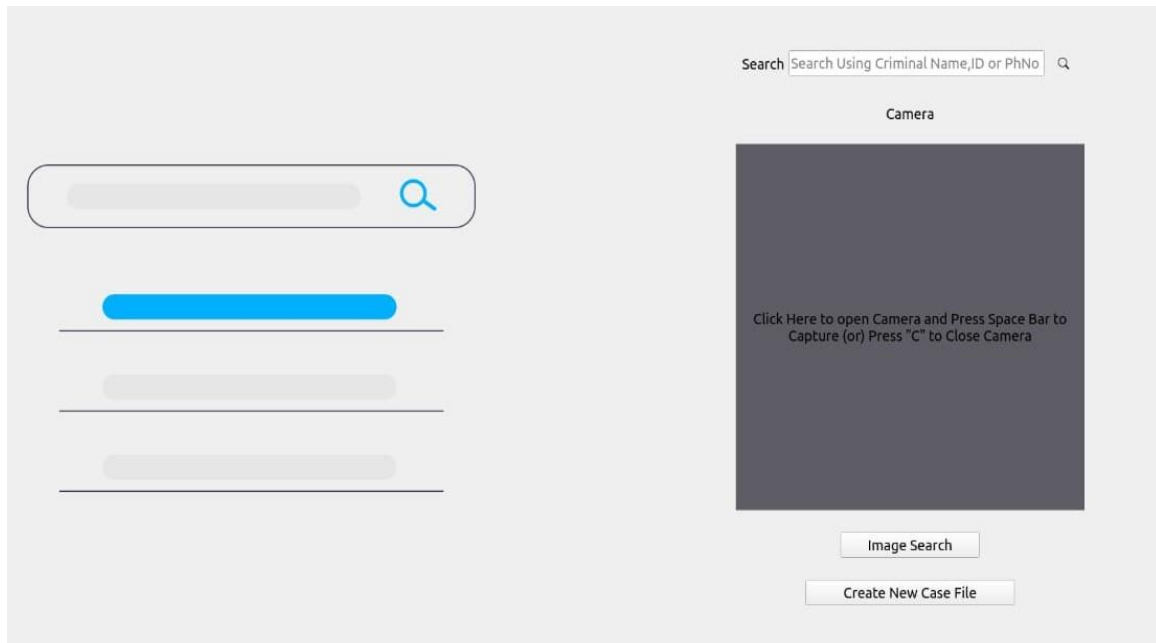


Fig 9.3 Search page

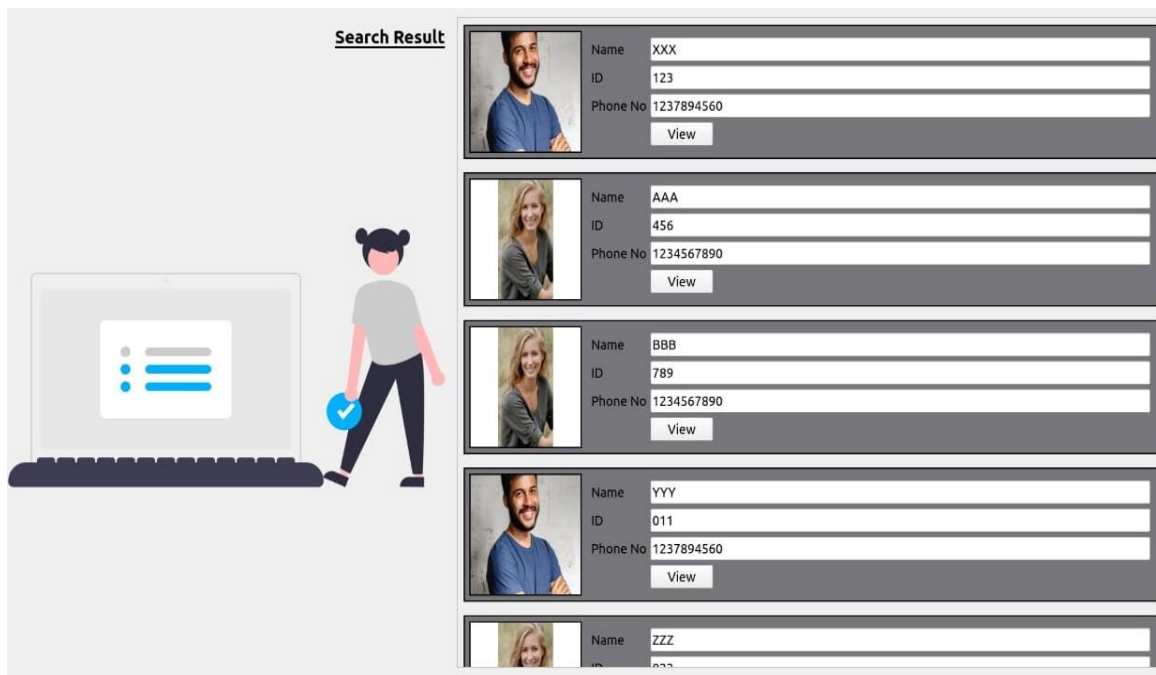


Fig 9.4 Search result


Criminal Name	<input type="text" value="XXX"/>	<div>Camera</div>  <div>Delete</div> <div>Edit</div>
Criminal ID	<input type="text" value="123"/>	
Age	<input type="text" value="20"/>	
Gender	<input checked="" type="checkbox"/> Male <input type="checkbox"/> Female <input type="checkbox"/> Others	
Phone No.	<input type="text" value="1237894560"/>	
Address	<input type="text" value="Bangalore"/>	
No Of Cases	<input type="text" value="1"/>	
No Of Years Prison	<input type="text" value="0.5"/>	
Case History	<input type="text" value="Bike Theft"/>	

Fig 9.5 Criminal details page

Criminal Name	<input type="text"/>	<div>Camera</div> <div> Click Here to open Camera and Press Space Bar to Capture (or) Press "C" to Close Camera </div> <div>Delete</div> <div>Submit</div>
Criminal ID	<input type="text"/>	
Age	<input type="text" value="18"/>	
Gender	<input checked="" type="checkbox"/> Male <input type="checkbox"/> Female <input type="checkbox"/> Others	
Phone No.	<input type="text"/>	
Address	<input type="text"/>	
No Of Cases	<input type="text" value="1"/>	
No Of Years Prison	<input type="text" value="0.0"/>	
Case History	<input type="text"/>	

Fig 9.6 Create new criminal case page

REFERENCES

- [1] P. M. Corcoran and C. Iancu, "Automatic face recognition system for hidden markov model techniques," *New Approaches to Characterization and Recognition of Faces*, pp. 3-28, 2011.
- [2] Bledsoe, "Manual measurements", 1960.
- [3] A.J. Goldstein, L.D. Harmon and A.B. Lesk, "Identification of human faces, "in *proceedings of the IEEE*, vol 59, pp. 748-760, May 1971.
- [4] LSirovich and M.Kirby, "Low dimensional procedure for the characterisation of human faces," in *Journal of the Optical Society of America A*, vol 4, pp. 519-524, 1987
- [5] M. Turk and A. Pentland, "Eigenfaces for Recognition," in *Journal of cognitive neuroscience*, vol 3, pp. 71-86, Jan 1991.
- [6] N. A. Abdullah, Md. J. Saidi, N. H. A. Rahman, C. C. Wen, and L. R. A. Hamid, "Face recognition for criminal Identification: An implementation of principal component analysis for face recognition," *AIP Conference Proceedings* 1891:1, Oct 2017.
- [7] P. Kakkar and V. Sharma, "Criminal identification system using face detection and recognition," in *International Journal of Advanced Research in Computer and Communication Engineering*, vol 7, pp. 238-243, March 2018
- [8] P.Apoorva, H.C. Impana, S.L. Siri., M.R.Varshitha and B.Ramesh, "Automated criminal identification by face recognition using open computer vision classifiers," in *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, pp. 775-778, 2019
- [9] P. Chhoriya, "Automated criminal identification system using facedetection and recognition", in *International Research Journal of Engineering and Technology (IRJET)*, vol 6. pp. 910-914, Oct 2019.
- [10] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features,"

