

**FACE RECOGNITION ATTENDANCE SYSTEM**

**PROJECT SYNOPSIS**

of Summer Training Project

## BACHELOR OF TECHNOLOGY

COMPUTER SCIENCE AND ENGINEERING

**Submitted To**

# Dr. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY LUCKNOW

SUBMITTED BY

Name : Pavan Singh &

Mayank Srivastava

**UNDER THE SUPERVISION OF**

Dr . Vijay Kumar Dwivedi

Assistant Professor

# UNITED COLLEGE OF ENGINEERING AND RESEARCH , PRAYAGRAJ

# ACKNOWLEDGEMENT

# We would like to express our deep gratitude to our project guide Mr. Vijay Kumar Dwivedi , Assistant Professor, Department of Computer Science and Engineering, for his/her guidance with unsurpassed knowledge and immense encouragement. We are grateful to Dr. Vijay Kumar Dwivedi , Head of the Department, Computer Science and Engineering, for providing us with the required facilities for the completion of the project work.

# We are very much thankful to the Principal and Management , for their encouragement and cooperation to carry out this work.

# We express our thanks to all teaching faculty of Department of CSE , whose suggestions during reviews helped us in accomplishment of our project. We would like to thank all non-teaching staff of the Department of CSE , for providing great assistance in accomplishment of our project.

# We would like to thank our parents, friends, and classmates for their encouragement throughout our project period. At last but not the least, we thank everyone for supporting us directly or indirectly in completing this project successfully.

# 

# PROJECT STUDENTS

# Mayank Srivastava ( 2103420100056 )

# Pavan Singh ( 2103420100063 )

# Table Of Contents

# 1. Introduction\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 

# Problem Definition\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 1.2 Objective\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 1.3 Block Diagram of the General Framework\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 1.4 Flow Chart\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 1.5 Definitions, Acronyms, and Abbreviations\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 1.6 Deep Learning for Face Recognition\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 1.7 Limitation of existing system or Research gap\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 2. Overall Description\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 

# 2.1 Product Functions\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 2.1.1 UseCase Model Survey \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 2.2 Interfaces\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 2.2.1 User Interfaces\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 2.2.2 Hardware Interfaces\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 2.2.3 Software Interfaces\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 2.2.4 Communications Interfaces\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 3. Code\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 4. Conclusion\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 5. Future Recommendations\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 1. Introduction

# This document contains the system requirements for Project. It is prepared by the team. This specification document includes descriptions of the functions and the specifications of the project. In this section(Section 1), a review of the entire document is provided. The reader would get familiarized with the contents before the further details are described

# 1.1 Problem Definition

# To develop an automated attendance system using face recognition. Concept In a classroom with large number of students, it is a very tedious and time-consuming task to take the attendance manually. Therefore, we can implement an effective system which will mark the attendance of students automatically by recognizing their faces.

# ● Save Users Time as this refers no physical interaction

# ● Supplying the user a smooth and clear interface

# ● Configuring a fast replying server system

# 1.2 Objective

# To identify the student faces accurately. To mark the attendance automatically. To reduce the time and the efforts required for manual attendance to provide a valuable attentive system for both teacher and students. It provides flexibility and reduces the time loss. There will be no chance for a proxy.

The objective of this project is to develop face recognition based automated student attendance system. Expected achievements in order to fulfill the objectives are:

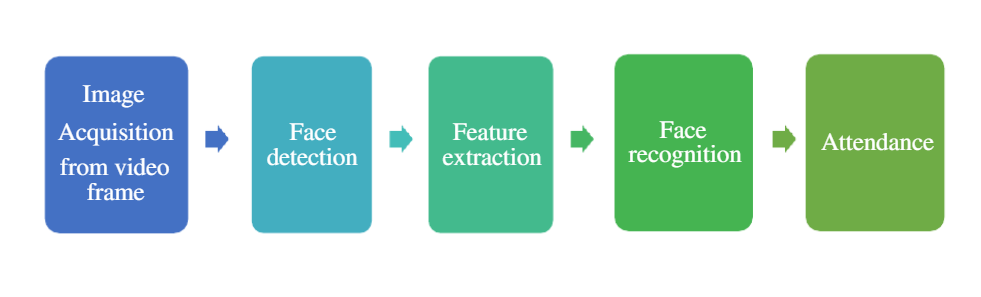
• To detect the face segment from the video frame.

• To extract the useful features from the face detected.

• To classify the features in order to recognize the face detected.

• To record the attendance of the identified student.

**1.3 Block Diagram of the General Framework**



# 1.4 Flow Chart

# A diagram of a face recognition process Description automatically generated

# 1.5 Definitions, Acronyms, and Abbreviations

# The document contains words and abbreviations related to computer science. The terms and abbreviations are the following ones:

# Server: A program that awaits and fulfills requests from client programs in the same or other computers.

# Deep Learning: A machine learning method that stimulates the neural network in human brain

# Python: One of the most commonly used programming languages.

# Kernel: The lowest level of easily replaceable software that interfaces with the hardware in your compute

# NLP: Natural Language Processing

# API: Application Program Interface

## 1.6 Deep Learning for Face Recognition

Some of the widely used Deep Learning-based face recognition systems are:

* + - * DeepFace
      * **DeepID** series of systems
      * VGGFace
      * FaceNet

Face recognizers generally take face images and find the important points such as the corner of the mouth, an eyebrow, eyes, nose, lips, etc. The coordinates of these points are called facial feature points. There are 66 such points. In this way, a different technique for finding feature  
points give different results.

# 

# 1.7 Limitation of existing system

# 

# 2. Overall Description

# 

# This section is about the requirements, constraints and the interfaces included in the project. A map of functions are also supplied. The document follows the IEEE standards, yet some of the sections are discarded as they are not compatible for this project.

# 2.1 Product Functions

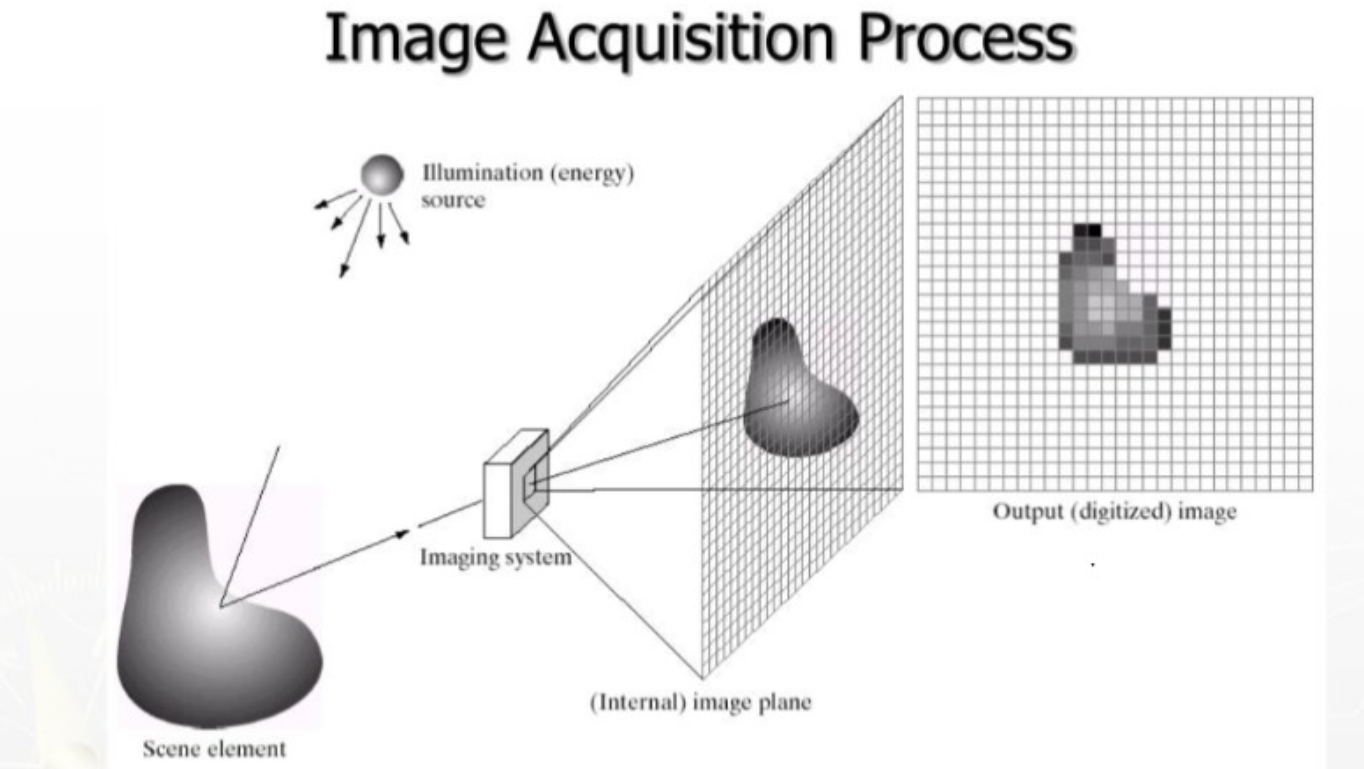
# Major functions of the product and brief descriptions of these functions can be found in this section. Also detailed diagrams and descriptions can be found in subsections of this section.

|  |  |  |
| --- | --- | --- |
| **ID** | **Function** | **Description** |
| 1. | Image Acquisition from video | obtain a two-dimensional image using a single sensor |
| 2. | Face detection | Detect the face among noise |
| 3. | **Feature Extraction** | Extract features of faces that will be used for training and recognition tasks |
| 4. | Face Recognition | Match the face from database |
| 5. | Mark Attendance | Marking Present And save in a .csv file |

# 2.1.1 Use Case Model Survey

# This section includes use case diagrams and their detailed descriptions of the functions that mentioned in section 2.1.

* **Image Acquisition from video**

 This involves capturing an image using

digital camera or scanner, or

importing a existing image

into a computer. Image is captured for

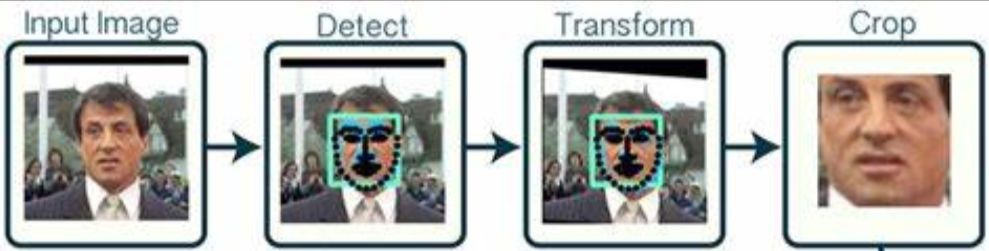
further processor like matching

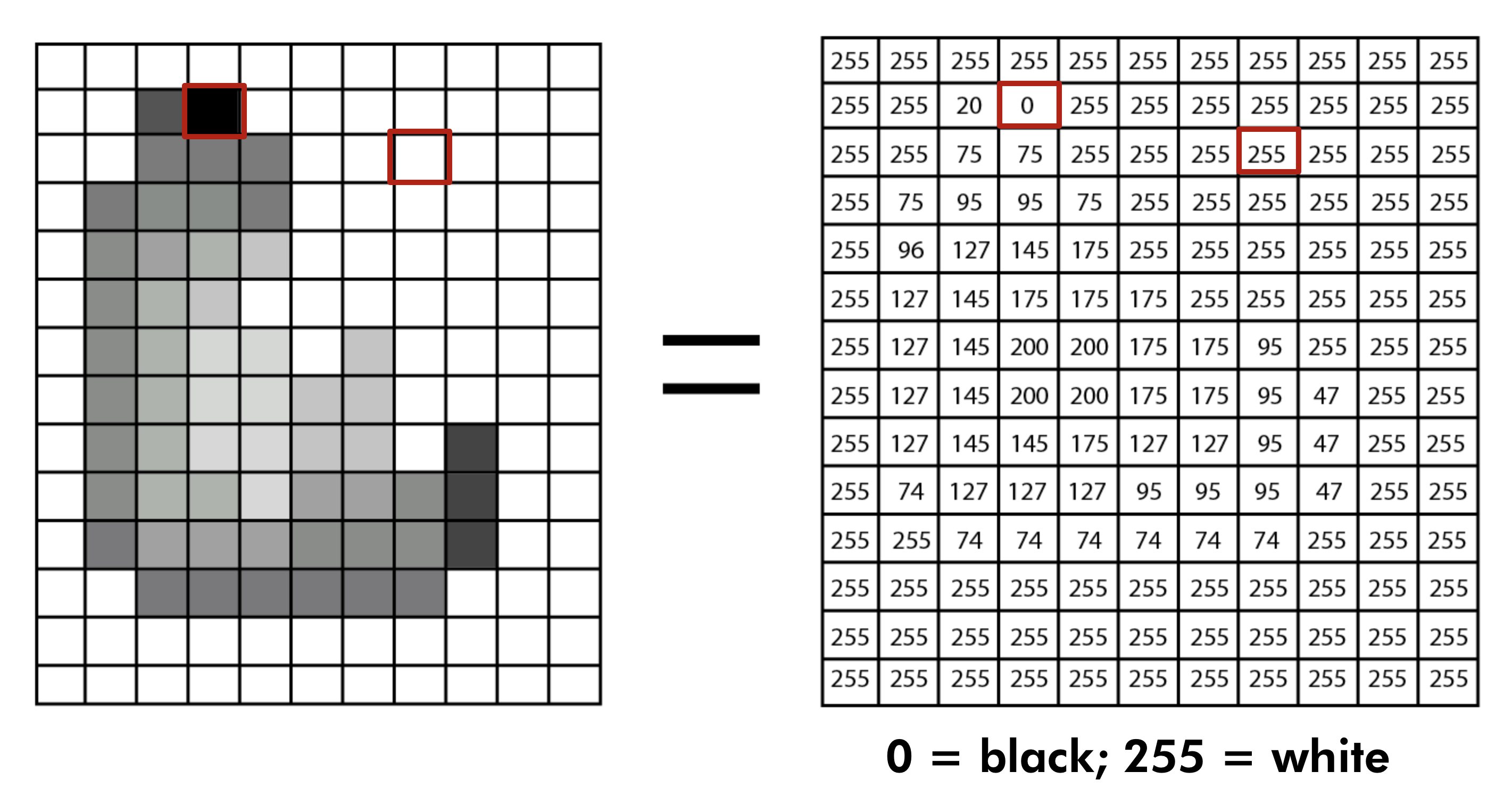
image’s existence in database .

|  |  |
| --- | --- |
| Use Case Id | 1 |
| Use Case | Image Acquisition from video |
| Description | This involves capturing an image using digital camera or scanner |
| Actor | user |
| Trigger | Automatic |
| Primary Scenario | * User must come in front of camera * Machine capture the user photo |
| Exceptional Scenario | None |

* **Face detection**

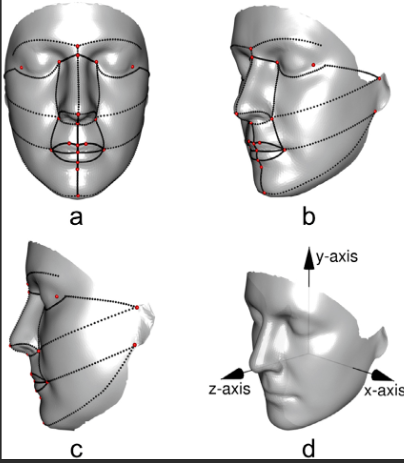
Face detectionis a computer technology being used in a variety of applications that identifies human faces in digital images **.**

****

****

|  |  |
| --- | --- |
| Use Case Id | 2 |
| Use Case | Face detection |
| Description | Detects the human face among noise |
| Actor | Machine |
| Trigger | Automatic |
| Primary Scenario | * Detects the face and convert it into RGB * RGB to digital |
| Exceptional Scenario | Image is not clear |

* **Feature Extraction**



Facial feature extraction is the process of extracting face

component features like eyes, nose, mouth, etc from human face

image. Facial feature extraction is very much important for

the initialization of processing techniqueslike face tracking,

facial expression recognition or face recognition.

|  |  |
| --- | --- |
| Use Case Id | 3 |
| Use Case | **Feature Extraction** |
| Description | extracting face component features like eyes, nose, mouth, etc from human face |
| Actor | Machine |
| Trigger | Automatic |
| Primary Scenario | Detect perfect dimensions |
| Exceptional Scenario | None |

* **Face Recognition**

Face recognizers generally take face images and find the important points such

as the corner of the mouth, an eyebrow, eyes, nose, lips, etc. The coordinates of these

points are called facial feature points. There are 66 such points. In this way, a

different technique for finding feature points give different results.

|  |  |
| --- | --- |
| Use Case Id | 4 |
| Use Case | Face Recognition |
| Description | Recognise the face and match from train data |
| Actor | Machine |
| Trigger | Automatic |
| Primary Scenario | Detect and match correct face |
| Exceptional Scenario | None |

* **Mark Attendance**

# After all the process have done then it will automatically mark the attendance as present, and save the name, time and date in a csv file for future use.

|  |  |
| --- | --- |
| Use Case Id | 5 |
| Use Case | Mark Attendance |
| Description | mark the attendance as present |
| Actor | Machine |
| Trigger | Automatic |
| Primary Scenario | Write Attendance of user |
| Exceptional Scenario | DO Nothing |

# 2.2 Interfaces

# 2.2.1 User Interfaces

# 

# The user interfaces will be an GUI window . the window will appear after executing the program and used whenever it wants . The prototype interface is the following :

# A screenshot of a face recognition system Description automatically generated

# 2.2.2 Hardware Interfaces

# Not applicable.

# 2.2.3 Software Interfaces

# The system will use **High **-** definitionCamera** for scanning a Face .

# 2.2.4 Communications Interfaces

# The only communication is between the program and the server and database .

# 3. Code

import dlib

import sys

import cv2

import face\_recognition

import os

import numpy as np

from datetime import datetime

import pickle

def AttenCall():

    path = 'C:\\Users\mayan\Project\dada\_set'

    images = []

    classNames = []

    mylist  = os.listdir(path)

    for cl in mylist:

        curImg = cv2.imread(f'{path}/{cl}')

        images.append(curImg)

        classNames.append(os.path.splitext(cl)[0])

    def findEncodings(images):

        encodeList = []

        for img in images:

            img = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

            encoded\_face = face\_recognition.face\_encodings(img)[0]

            encodeList.append(encoded\_face)

        return encodeList

    encoded\_face\_train = findEncodings(images)

    def markAttendance(name):

        with open('Attendance.csv','r+') as f:

            myDataList = f.readlines()

            nameList = []

            for line in myDataList:

                entry = line.split(',')

                nameList.append(entry[0])

            if name not in nameList:

                now = datetime.now()

                time = now.strftime('%I:%M:%S:%p')

                date = now.strftime('%d-%B-%Y')

                f.writelines(f'{name}, {time}, {date},\n')

    # take pictures from webcam

    cap  = cv2.VideoCapture(0)

    while (True):

        success, img = cap.read()

        imgS = cv2.resize(img, (0,0), None, 0.25,0.25)

        imgS = cv2.cvtColor(imgS, cv2.COLOR\_BGR2RGB)

        faces\_in\_frame = face\_recognition.face\_locations(imgS)

        encoded\_faces = face\_recognition.face\_encodings(imgS, faces\_in\_frame)

        for encode\_face, faceloc in zip(encoded\_faces,faces\_in\_frame):

            matches = face\_recognition.compare\_faces(encoded\_face\_train, encode\_face)

            faceDist = face\_recognition.face\_distance(encoded\_face\_train, encode\_face)

            matchIndex = np.argmin(faceDist)

            print(matchIndex)

            if matches[matchIndex]:

                name = classNames[matchIndex].upper().lower()

                y1,x2,y2,x1 = faceloc

                # since we scaled down by 4 times

                y1, x2,y2,x1 = y1\*4,x2\*4,y2\*4,x1\*4

                cv2.rectangle(img,(x1,y1),(x2,y2),(0,255,0),2)

                cv2.rectangle(img, (x1,y2-35),(x2,y2), (0,255,0), cv2.FILLED)

                cv2.putText(img,name, (x1+6,y2-5), cv2.FONT\_HERSHEY\_COMPLEX,1,(255,255,255),2)

                markAttendance(name)

        cv2.imshow('webcam', img)

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

            break

    cap.release()

    cv2.destroyAllWindows()

import pandas as pd

df=pd.read\_csv("Attendance.csv")

df

row\_count = df.shape[0]

print(row\_count)

print(df)

path = 'C:\\Users\mayan\Project\dada\_set'

images = []

classNames = []

mylist  = os.listdir(path)

for cl in mylist:

    curImg = cv2.imread(f'{path}/{cl}')

    images.append(curImg)

    classNames.append(os.path.splitext(cl)[0])

from tkinter import \*

from tkinter.ttk import \*

import tkinter as tk

import tkinter.font as tkFont

import csv

filepath= 'C:\\Users\mayan\Project\Attendance.csv'

File = open(filepath)

Reader = csv.reader(File)

Data = list(Reader)

class App:

    def \_\_init\_\_(self, root):

        #setting title

        root.title("Face Recognition Attendance System")

        #setting window size

        width=600

        height=500

        screenwidth = root.winfo\_screenwidth()

        screenheight = root.winfo\_screenheight()

        alignstr = '%dx%d+%d+%d' % (width, height, (screenwidth - width) / 2, (screenheight - height) / 2)

        root.geometry(alignstr)

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

        GLabel\_72=tk.Label(root)

        ft = tkFont.Font(family='Times',size=23)

        GLabel\_72["font"] = ft

        GLabel\_72["fg"] = "#333333"

        GLabel\_72["justify"] = "center"

        GLabel\_72["text"] = "Face Recognition Attendance System"

        GLabel\_72["relief"] = "ridge"

        GLabel\_72.place(x=0,y=0,width=601,height=85)

        GButton\_343=tk.Button(root)

        GButton\_343["bg"] = "#ffffff"

        ft = tkFont.Font(family='Times',size=16)

        GButton\_343["font"] = ft

        GButton\_343["fg"] = "#000000"

        GButton\_343["justify"] = "center"

        GButton\_343["text"] = "Start Attendance"

        GButton\_343.place(x=140,y=120,width=315,height=126)

        GButton\_343["command"] = self.GButton\_343\_command

        GButton\_303=tk.Button(root)

        GButton\_303["bg"] = "#fbfbfb"

        ft = tkFont.Font(family='Times',size=16)

        GButton\_303["font"] = ft

        GButton\_303["fg"] = "#000000"

        GButton\_303["justify"] = "center"

        GButton\_303["text"] = "Check Attendance"

        GButton\_303.place(x=140,y=300,width=313,height=127)

        GButton\_303["command"] = CheckAttenWindow

    def GButton\_343\_command(self):

        AttenCall()

def CheckAttenWindow():

    top = Toplevel()

    top.title("Attendance")

    list\_of\_entries = []

    for x in list(range(0,len(Data))):

        list\_of\_entries.append(Data[x][0])

    listbox1 = Listbox(top)

    for x,y in enumerate(list\_of\_entries):

        listbox1.insert(x,y)

    listbox1.grid(row=0,column=0)

if \_\_name\_\_ == "\_\_main\_\_":

    root = tk.Tk()

    app = App(root)

    root.mainloop()

# 4. CONCLUSION

# Face recognition systems are part of facial image processing applications and their significance as a research area are increasing recently. Implementations of system are crime prevention, video surveillance, person verification, and similar security activities. The face recognition system implementation can be part of Universities. Face Recognition Based Attendance System has been envisioned for the purpose of reducing the errors that occur in the traditional (manual) attendance taking system. The aim is to automate and make a system that is useful to the organization such as an institute. The efficient and accurate method of attendance in the office environment that can replace the old manual methods. This method is secure enough, reliable and available for use. Proposed algorithm is capable of detect multiple faces, and performance of system has acceptable good results

# 5. Future Recommendations

# The system can be made more flexible and scalable using these recommendations. Please note that the system implemented here is just a prototype of idea presented via this project. The recommendations are as follows:

# 

# • The system can be extended to more number of students with freedom to change list of students according to class changes.

# • The system can be made more flexible to allow updating of templates in case student incurs significant amount of change in his facial features.

# • The system can also be extended to allow better face recognition algorithm in which even rotational features of face can be detected efficiently.