Student attendance Management System based Facial Recognition

A Project Report Submitted in partial fulfillment of the requirement for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

Submitted by

B.GANGADHAR	18T91A0510
G.SUSHMITHA	18T91A0517
M L N V KRISHNA TEJA	18T91A0529
S NEELESH KUMAR	18T91A0544

Under the Esteemed Guidance of

Dr. SHEIK MEERA SHARIF

Professor & HOD

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

GIET ENGINEERING COLLEGE

[Affiliated to JNTUK, Kakinada | Accredited by NACC]

NH-16 CHAITANYA KNOWLEDGE CITY, RAJAMAHENDRAVARAM (A.P)

2018-2022

GIET ENGINEERING COLLEGE

[Affiliated to JNTUK, Kakinada | Accredited by NACC]

NH-16 CHAITANYA LNOWLEDGE CITY, RAJAMAHENDRAVARAM (A.P)

2018-2022

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that

B GANGADHAR	18T91A0510
G SUSHMITHA	18T91A0517
M L N V KRISHNA TEJA	18T91A0529
S NEELESH KUMAR	18T91A0544

Studying IV B.Tech II Semester of Computer Science and Engineering branch have submitted their project "**Student Attendance Management System based facial recognition**" during the academic year 2018-2022 in partial fulfillment of the requirements for the award of degree in **Bachelor of Technology, JNTUK, Kakinada**. The result embodied in this project has not been submitted to any other University or Institute for the award of degree.

Project Guide	**Head of the Department**
Dr. SHEIK MEERA SHARIF	Dr. SHEIK MEERA SHARIF
Professor & HOD	Professor & HOD
Department of CSE	Department of CSE

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

It is a privilege for us to have undertaken the project "**Student Attendance Management System based Facial Recognition**" in GIET ENGINEERING COLLEGE, RAJAMAHENDRAVARAM.

We avail this opportunity to express our deep sense of gratitude and heart-full thanks to Sri K.SASI KIRAN VARMA, Vice Chairman of GIET ENGINEERING COLLEGE, RAJAMAHENDRAVARAM.

We are thankful to our Principal Dr. T. VAMSEE KIRAN for encouraging us to do this project. We are deeply indebted to Dr. SHEIK MEERA SHARIF, Professor and HOD, Department of Computer Science and Engineering, GIET ENGINEERING COLLEGE, whose motivation in the field of software development made us overcome all the hardships during the course study. We are heartily thankful to our coordinator and internal guide Dr.SHEIK MEERA SHARIEF, Professor, GIET ENGINEERING COLLEGE, for his moral support which was always there to comfort and solace during tough times. Finaly we would like to thank our TEACHING AND NON-TEACHING STAFF whose blessing and encouragement were always there as a source of strength and inspiration.

Although the title "Acknowledgement" cannot represent our true feelings for all these persons, we feel very much thankful to all of them and also to our PARENTS and FRIENDS for encouraging and giving us all the moral support required for making this endeavor a reality.

B GANGADHAR	18T91A0510
G SUSHMITHA	18T91A0517
M L N V KRISHNA TEJA	18T91A0529
S NEELESH KUMAR	18T91A0544

DECLARATION

We hereby declare that this project "**Student Attendance Management System based Facial Recognition**" submitted to the Department of CSE, GIET ENGINEERING COLLEGE, affiliated to JNTUK, Kakinada, as partial/complete fulfilment for the award of Bachelor of Technology degree is entirely the original work done by us and has not been submitted to any other organization.

PROJECT MEMBERS	PIN NO	SIGNATURE
B GANGADHAR	18T91A0510	
G SUSHMITHA	18T91A0517	
M L N V KRISHNA TEJA	18T91A0529	
S NEELESH KUMAR	18T91A0544	

ABSTRACT

Face recognition technology has made many improvements in a changing world. Smart attendance using real-time face recognition is a real-world solution that comes with a method to handle the daily activities of student attendance systems. Attendance management can be a great burden for the teacher if it is done manually. To solve the problem, the smart attendance management has been used. Our attendance system takes the attendance automatically using the face recognition. In our method we using, continuous observation technique which improves the performance for the estimation of the attendance based on face recognition. Then, we introduced our systema structure and plan. Finally, our attendance management system are implemented to provide as evidence to support the plan. The result shows that continuous observation improved the performance for the estimation of the attendance. The Attendance Management system is evident to fulfill all the Facilities that it is designed for. In addition to the time-consuming issue, such method is also at higher risk of having students cheating about their attendance, especially in a large classroom. The updated attendance list is then uploaded to an online database and can also be saved as a file to be transferred to a PC later on. This system will help to eliminate the current problems, while also promoting a paperless environment at the same time.

TABLE OF CONTENTS

INTRODUCTION

1.1 Introduction to Face Recognition:

Face recognition is one of the many wonders that AI research has brought forward to the world. It is a subject of curiosity for many techies — who would like to have a basic understanding of how things work. Let us take a dip into the subject, to see how things work.

1.2 Intuition:

To understand how a machine can recognize faces, we can start with asking ourselves — how do we recognize a face? Most images of human faces have two eyes, a nose, lips, forehead, chin, ears, hair… That rarely changes. Yet, faces are different from each other. What makes them different? At the same time, face of the same person changes with emotion, expression, age… In fact just change in orientation creates a different image. How do we identify a person in spite of all that?

On a gross level, one can say that there are some components of a face are related to age, emotion and orientation. While there are some other components that are stick to the person irrespective of the age, emotion, etc. Further, we can say that these components are not orthogonal or independent. We have all seen people who look alike sideways, but very different otherwise. Or, a kid in the family reminds people of his parents at that age, etc.

So, it is not so easy to logically identify these individual components. But, one can say that there are several overlapping components of the face — which are individually responsible for the perception of emotion, age and the person himself.

Essentially, we know that there is “some relation that is too complex for logic” — that is where machine learning shows up

1.3 Training the Model:

To train our neural network, we need a huge amount of data. Now, what kind of data do we use when we train a face recognition model? The model we generate should be able to measure similarity between two faces. It should be able to map an image of a face to a vector space such that images of the same person are closer to each other — irrespective of the other parameters. So our data set has to include several images of each person.

Like most other AI problems, these concepts are pretty old. But, the progress on face recognition was held back because of lack of data and processing power. With the boom in social media, we have obtained a huge amount of data — with a decent amount of labeling. Many researchers proposed different ways of going about this problem. Some started with the abstract — using unsupervised learning to start with just segregating and identifying different types of faces. Some took the down to earth approach with hard training the model based on each individual feature of the face. After working on various different approaches on a variety of data sets, we now have several solutions that offers accuracy better than human limits.

1.4 Face recognition:

When working on problems like this, it is best not to reinvent the wheel — we can’t! It is best to follow up with models that researchers have provided us. A lot of it is available in the open source as well. One such Python library is face recognition. It works in a few steps:

- Identify a face in a given image
- Identify specific features in the face
- Generate a face encoding vector of 128 values

Based on this encoding, we can measure the similarity between two face images — that can tell us if they belong to the same person.

1.5 Face Recognition Algorithm:

Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, “Rapid Object Detection using a Boosted Cascade of Simple Features” in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images.

Here we will work with face detection. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. For this, haar features shown in below image are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under white rectangle from sum of pixels under black rectangle.

Now all possible sizes and locations of each kernel is used to calculate plenty of features. (Just imagine how much computation it needs? Even a 24x24 window results over 160000 features). For each feature calculation, we need to find sum of pixels under white and black rectangles. To solve this, they introduced the integral images. It simplifies calculation of sum of pixels, how large may be the number of pixels, to an operation involving just four pixels.

But among all these features we calculated, most of them are irrelevant.

1.6 How Facial Recognition work:

Any facial recognition algorithm uses biometrics to map out facial features captured in a video still or a photograph. That information is then compared to a database of faces. There are four general steps in the process, which we'll explain further.

STEP 1: FACE DETECTION

First, a camera will detect and recognize a human's face – one that can either be in a crowd or alone. It is most easily detected when the person is looking straight at the camera. However, modern technological advances allow face recognition software to still work if the person's face is angled slightly.

STEP 2: FACE ANALYSIS

After detection and recognition, a photo will capture the face and will then be analyzed. The majority of face recognition technology use 2D images instead of 3D. This is because 2D photos are more readily correlated with public photos or pictures in a database (these are typically 2D as well). During analysis, the face will be separated into distinguishable landmarks – we can call these nodal points. A human face has eight nodal points. Face recognition technology will analyze each of these points – for example, the distance between your eyebrows.

STEP 3: CONVERTING AN IMAGE INTO DATA

After analysis, each nodal point becomes a number in the application database. The entire numerical code is referred to as a faceprint. Just like how everybody has a unique thumbprint, everyone also has a unique faceprint.

STEP 4: MATCHING

The final step of the process is finding a match. Your faceprint is compared to a database of other facial codes. The number of faces that are compared depends on the database and how many databases the software has access to. For instance, the FBA has access to 21 state databases, with 641 million photos across them. The facial recognition technology then identifies a match for your exact facial features – it returns the user with the found match and other relevant information, such as an address and a name.

LITERATURE REVIEW

2.1 A Counterpart Approach to Attendance System using Machine Learning Techniques:

In this paper, the idea of two technologies namely Student Attendance has been implemented with a machine learning approach. This system automatically detects the student performance and maintains the student's records like attendance and their attendance records. Therefore the attendance of the student can be made available by recognizing the face. On recognizing, the attendance details and details about the detailed report of the student is obtained as feedback.

2.2 Automated Attendance System Using Face Recognition:

Automated Attendance System using Face Recognition proposes that the system is based on face detection and recognition algorithms, which is used to automatically detects the student face when he/she enters the class and the system is capable to marks the attendance by recognizing him. Haar-cascade Algorithm has been used for face detection which detect human face using cascade classifier for feature selection and SVM for classification. When it is compared to traditional attendance marking this system saves the time and also helps to monitor the students.

2.3 Student Attendance System Using Iris Detection:

In this proposed system the student is requested to stand Face Recognition Based Attendance Publication in front of the camera to detect and recognize the iris, for the system to mark attendance for the student. Some algorithms like Gray Scale Conversion, Six Segment Rectangular Filter, Skin Pixel Detection is being used to detect the iris. It helps in preventing the proxy issues and it maintains the attendance of the student in an effective manner, but in one of the time-consuming process for a student or a staff to wait until the completion of the previous members.

2.4 Face Recognition-based Lecture Attendance System:

This paper proposes that the system takes the attendance automatically recognition obtained by continuous observation. Continuous observation helps in estimating and improving the performance of the attendance. To obtain the attendance, positions and face images of the students present in the class room are captured. Through continuous observation and recording the system estimates seating position and location of each student for attendance marking. The work is focused on the method to obtain the different weights of each focused seat according to its location. The effectiveness of the picture is also being discussed to enable the faster recognition of the image.

3.1 EXISTING SYSTEM

- The Existing System is a Manual Entry for the students. here the attendance will be carried out in the hand written registers. the retrieval of the information is not easy as the records are maintained in the hand written registers.
- this application requires correct feed on input into the respective field.

Disadvantages of Existing system

1.Manual time entry is very time-consuming

2.Are ineffective and outdated

3.Too much paperwork

4.There is a risk of human error

5.The Human effort is more here

6.The retrieval of the information is not as easy as the records are maintained in the handwritten registers

3.2 PROPOSED SYSTEM

- To overcome the drawbacks of the existing system, the proposed system has been evolved.
- This project aims to reduce the paper work and saving time to generate accurate results from the student’s attendance.
- The system provides with the best user interface. The efficient reports can be generated by using this proposed system.

Advantages of Proposed System

1.It is trouble-free to use.

2.It is a relatively fast approach to enter attendance.

3.Is highly reliable, approximate result from user.

4.Efficient reports

3.3 HARDWARE REQUIREMENTS

1.Minimum RAM :- 4GB

2.Hard Disk :- 120GB

3. Processor :- Intel 5

3.4 SOFTWARE REQUIREMENTS

1. Text Editor (IDE’s): Microsoft Visual Studio

2.Programming Language: Python.

3.Library:Numpy,Pandas,Matplotlib,Face-Recognition,Open-CV,Mysql connector

4.Database:Mysql

FACE DETECTION USING HAAR CASCADE METHOD

4.1 INTRODUCTION :

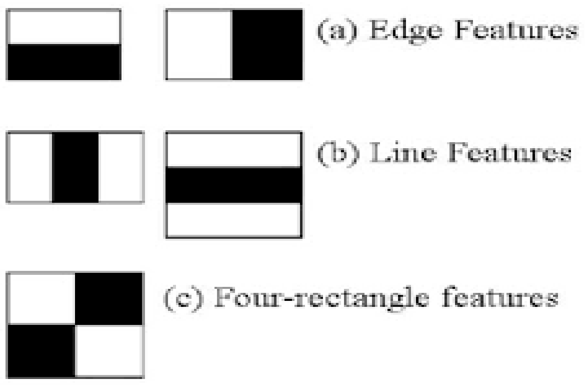
Face detection is a type of application classified under “computer vision” technology. It is the process in which algorithms are developed and trained to properly locate faces or objects (in object detection, a related system), in images. These can be in real time from a video camera or from photographs. An example where this technology is used are in airport security systems. In order to recognize a face, the camera software must first detect it and identify the features before making an identification.

Likewise, when Facebook makes tagging suggestions to identify people in photos it must first locate the face. On social media apps like Snapchat, face detection is required to augment reality which allows users to virtually wear dog face masks using fancy filters. Another use of face detection is in smartphone face ID security. In this project, I implemented a system for locating faces in digital images.

These are in JPEG format only. Before we continue, we must differentiate between face recognition and face detection. They are not the same, but one depends on the other. In this case face recognition needs face detection for making an identification to “recognize” a face. I will only cover face detection. Face detection uses classifiers, which are algorithms that detects what is either a face (1) or not a face (0) in an image. Classifiers have been trained to detect faces using thousands to millions of images in order to get more accuracy.

4.2 HAAR CASCADES :

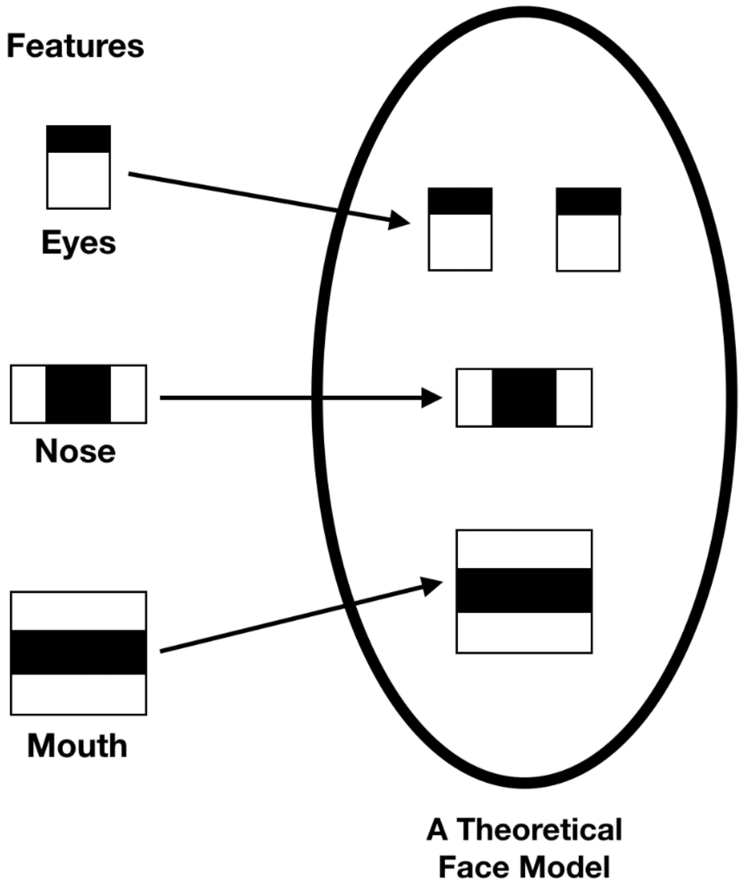
Haar Cascade classifier is based on the Haar Wavelet technique to analyse pixels in the image into squares by function. This uses "integral image" concepts to compute the "features" detected. Haar Cascades use the Ada-boost learning algorithm which selects a small number of important features from a large set to give an efficient result of classifiers then use cascading techniques to detect face in a image. Here are some Haar-Features.



4.2(a) Haar features

4.3 FEATURE EXTRACTION

As I mentioned earlier, Haar Cascades use machine learning techniques in which a function is trained from a lot of positive and negative images. This process in the algorithm is feature extraction.



4.3(a) Face Detection kernels

Haar features are similar to these convolution kernels which are used to detect the presence of that feature in the given image. The first two are "edge features", used to detect edges. The third is a "line feature", while the fourth is a "four rectangle feature", most likely used to detected a slanted line.

4.4 HAAR FEATURES APPLIED TO IMAGE

Each feature results in a single value which is calculated by subtracting the sum of pixels under white rectangle from the sum of pixels under black rectangle. Every haar feature have some sort of resemblance to identify a part of face. Value = ∑ (pixels in white area) – ∑ (pixels in black area).

If we consider all possible parameters of the haar features like position, scale and type we end up calculating about 160,000+ features. So we need to evaluate huge set of features for every 24*24 PX. So to avoid this we have a idea to avoid redundant features and pick only those features which are very useful for us.

SYSTEM DESIGN

5.1 INTRODUCTION

Uml diagram:

The uml diagram shows the structure of how the application works in the form of a flowchart.these flowcharts are used to understand the functionality of basic schema.

The schema funtions are described by the flowchart.one can analyze the overall working just based alone on the flowcharts.there are different types of uml diagrams used in the following applications.

Sequence diagram:

A sequence diagram is a type of interaction diagram because it describes how—and in what order—a group of objects works together . first the system application starts the camera .then the image is processed in the image database .then this data is again stored in the database.

Use case diagram:

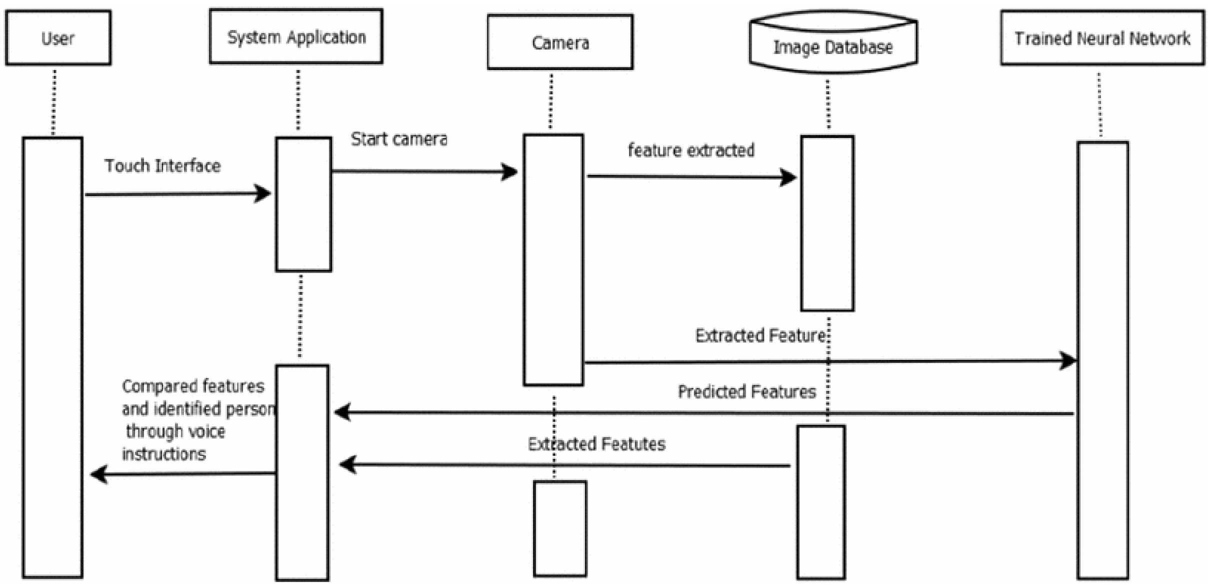
a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system . the student can only either check the attendance report after login into the website. the teacher can even check the list of absentee along with the students who are present.

Block diagram:

The block diagram shows the basic flowchart of the entire system.

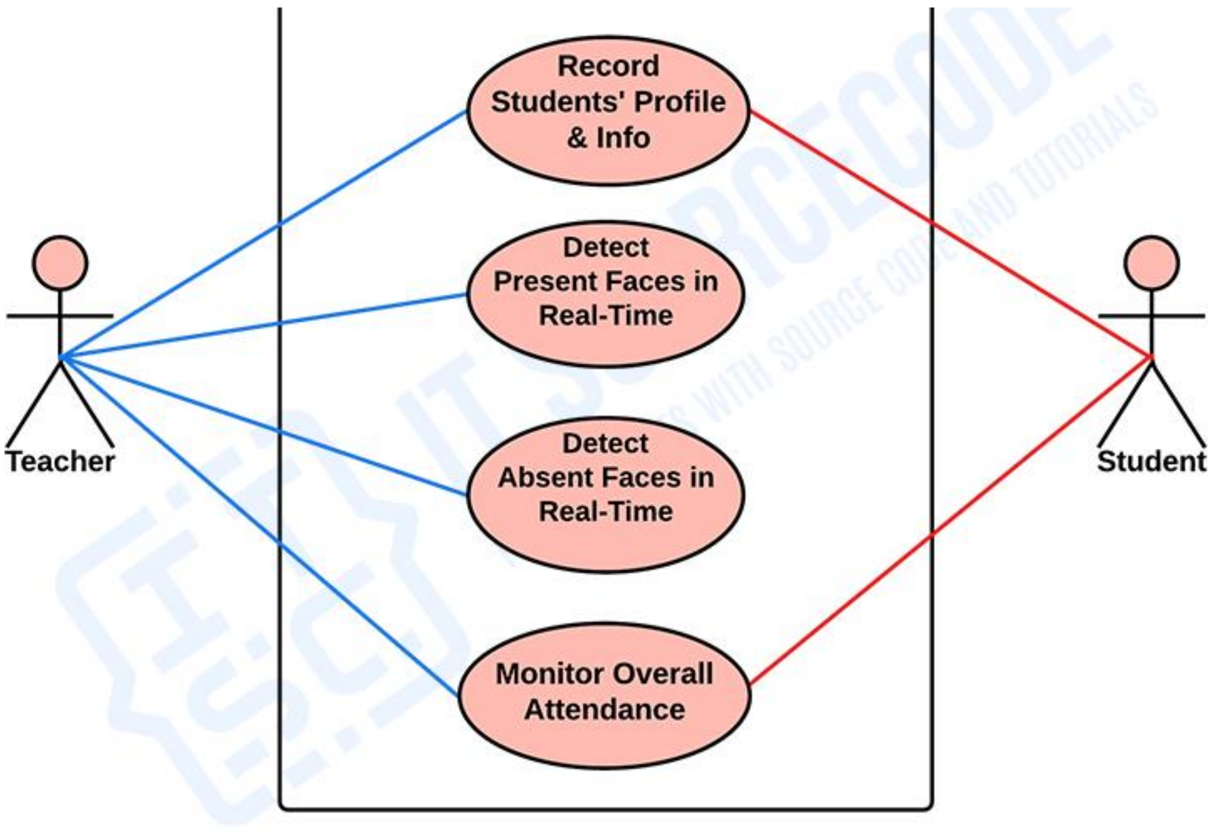
It shows how the entire software works .the image taken by the camera is processed and the students whose face is not detected are marked absent.

5.2 SEQUENCE DIAGRAM FOR FACE RECOGNITION



5.2 (a) Sequence diagram

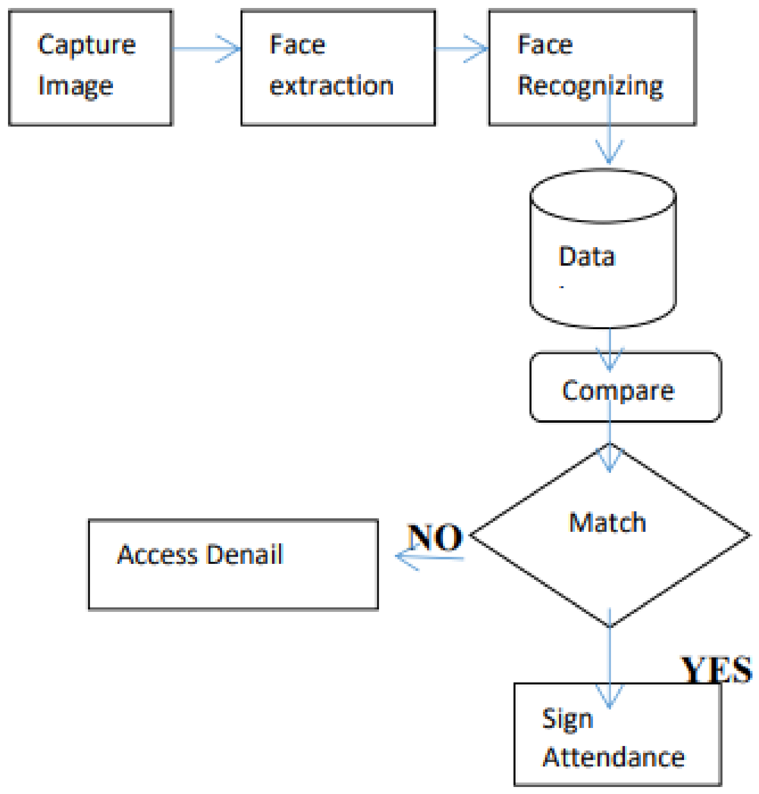
5.3 Use case Diagram



Record Students' Profile & Info
Detect Present Faces in Real-Time
Detect Absent Faces in Real-Time
Monitor Overall Attendance
Teacher
Student

5.3 (a) use case diagram

5.4 BLOCK DIAGRAM



5.4 (a) block diagram

SYSTEM IMPLEMENTATION

6.1 INTRODUCTION

The goal of the coding or programming phase is to translate the design of system produced during the design phase into code in a given programming language,which can be executed by a computer and that performs the computation specified by the design.

The coding phase affects both testing and maintenance.the goal of coding is not to reduce the implementation cost but the goal should be to reduce the cost of later phases.in other words the goal is not to simply the job of programmer.

6.2 SOURCE CODE:

Login.py:

```
from tkinter import*
from tkinter import ttk
from PIL import Image,ImageTk
from tkinter import messagebox
from register import Register
import mysql.connector
# -------------------------
from train import Train
from student import Student
from train import Train
from face_recognition import Face_Recognition
from attendance import Attendance
from developer import Developer
from helpsupport import Helpsupport
import os


class Login:
    def __init__(self,root):
        self.root=root
        self.root.title("Login")
        self.root.geometry("1366x768+0+0")


        # variables
        self.var_ssq=StringVar()
        self.var_sa=StringVar()
        self.var_pwd=StringVar()


        self.bg=ImageTk.PhotoImage(file=r"C:\Python-FYP-Face-Recognition-
Attendence-System-master\Images_GUI\loginBg1.jpg")


        lb1_bg=Label(self.root,image=self.bg)
        lb1_bg.place(x=0,y=0, relwidth=1,relheight=1)


        frame1= Frame(self.root,bg="#002B53")
        frame1.place(x=560,y=170,width=340,height=450)


        img1=Image.open(r"C:\Python-FYP-Face-Recognition-Attendence-
System-master\Images_GUI\log1.png")
```

```
img1=img1.resize((100,100),Image.ANTIALIAS)

self.photoimage1=ImageTk.PhotoImage(img1)

lb1img1 = Label(image=self.photoimage1,bg="#002B53")

lb1img1.place(x=690,y=175, width=100,height=100)


get_str = Label(frame1,text="Login",font=("times new roman",20,"bold"),fg="white",bg="#002B53")

get_str.place(x=140,y=100)


#label1

username =lb1= Label(frame1,text="Username:",font=("times new roman",15,"bold"),fg="white",bg="#002B53")

username.place(x=30,y=160)


#entry1

self.txtuser=ttk.Entry(frame1,font=("times new roman",15,"bold"))

self.txtuser.place(x=33,y=190,width=270)


#label2

pwd =lb1= Label(frame1,text="Password:",font=("times new roman",15,"bold"),fg="white",bg="#002B53")

pwd.place(x=30,y=230)


#entry2

self.txtpwd=ttk.Entry(frame1,font=("times new roman",15,"bold"))
```

```
        self.txtpwd.place(x=33,y=260,width=270)


        # Creating Button Login

        loginbtn=Button(frame1,command=self.login,text="Login",font=("times new roman",15,"bold"),bd=0,relief=RIDGE,fg="#002B53",bg="white",activeforeground="white",activebackground="#007ACC")

        loginbtn.place(x=33,y=320,width=270,height=35)


        # Creating Button Registration

        loginbtn=Button(frame1,command=self.reg,text="Register",font=("times new roman",10,"bold"),bd=0,relief=RIDGE,fg="white",bg="#002B53",activeforeground="orange",activebackground="#002B53")

        loginbtn.place(x=33,y=370,width=50,height=20)


        # Creating Button Forget


loginbtn=Button(frame1,command=self.forget_pwd,text="Forget",font=("times new roman",10,"bold"),bd=0,relief=RIDGE,fg="white",bg="#002B53",activeforeground="orange",activebackground="#002B53")

        loginbtn.place(x=90,y=370,width=50,height=20)


    #  THis function is for open register window

    def reg(self):

        self.new_window=Toplevel(self.root)

        self.app=Register(self.new_window)
    def login(self):
        if (self.txtuser.get()=="" or self.txtpwd.get()==""):
            messagebox.showerror("Error","All Field Required!")
        elif(self.txtuser.get()=="admin" and self.txtpwd.get()=="admin"):
            messagebox.showinfo("Sussessfully","Welcome to Attendance Managment System Using Facial Recognition")
        else:
            # messagebox.showerror("Error","Please Check Username or Password !")
            conn = mysql.connector.connect(username='root', password='Ganga@932',host='localhost',database='face_recognition',port=3306)
            mycursor = conn.cursor()
            mycursor.execute("select * from regteach where email=%s and pwd=%s",(
                self.txtuser.get(),
                self.txtpwd.get()
            ))
            row=mycursor.fetchone()
            if row==None:
                messagebox.showerror("Error","Invalid Username and Password!")
            else:
                open_min=messagebox.askyesno("YesNo","Access only Admin")
                if open_min>0:
                    self.new_window=Toplevel(self.root)
                self.app=Face_Recognition_System(self.new_window)
            else:
                if not open_min:
                    return
        conn.commit()
        conn.close()
#=======================Reset Passowrd Function============================
    def reset_pass(self):
        if self.var_ssq.get()=="Select":
            messagebox.showerror("Error","Select the Security Question!",parent=self.root2)
        elif(self.var_sa.get()==""):
            messagebox.showerror("Error","Please Enter the Answer!",parent=self.root2)
        elif(self.var_pwd.get()==""):
            messagebox.showerror("Error","Please Enter the New Password!",parent=self.root2)
        else:
            conn = mysql.connector.connect(username='root', password='Ganga@932',host='localhost',database='face_recognition',port=3306)
            mycursor = conn.cursor()
            query=("select * from regteach where email=%s and ss_que=%s and s_ans=%s")
            value=(self.txtuser.get(),self.var_ssq.get(),self.var_sa.get())
            mycursor.execute(query,value)
            row=mycursor.fetchone()

            if row==None:

               messagebox.showerror("Error","Please Enter the Correct Answer!",parent=self.root2)

            else:

               query=("update regteach set pwd=%s where email=%s")

               value=(self.var_pwd.get(),self.txtuser.get())

               mycursor.execute(query,value)



               conn.commit()

               conn.close()

               messagebox.showinfo("Info","Successfully Your password has been rest, Please login with new Password!",parent=self.root2)





# =====================Forget window=========================================

   def forget_pwd(self):

         if self.txtuser.get()=="":

            messagebox.showerror("Error","Please Enter the Email ID to reset Password!")

         else:

            conn = mysql.connector.connect(username='root', password='Ganga@932',host='localhost',database='face_recognition',port=3306)

            mycursor = conn.cursor()
```

```
query=("select * from regteach where email=%s")

value=(self.txtuser.get(),)

mycursor.execute(query,value)

row=mycursor.fetchone()

# print(row)


if row==None:

   messagebox.showerror("Error","Please Enter the Valid Email ID!")

else:

   conn.close()

   self.root2=Toplevel()

   self.root2.title("Forget Password")

   self.root2.geometry("400x400+610+170")

   l=Label(self.root2,text="Forget Password",font=("times new roman",30,"bold"),fg="#002B53",bg="#fff")

   l.place(x=0,y=10,relwidth=1)

   # -------------------fields-------------------

   #label1

   ssq =lb1= Label(self.root2,text="Select Security Question:",font=("times new roman",15,"bold"),fg="#002B53",bg="#F2F2F2")

   ssq.place(x=70,y=80)


   #Combo Box1

   self.combo_security = ttk.Combobox(self.root2,textvariable=self.var_ssq,font=("times new roman",15,"bold"),state="readonly")
```

```
        self.combo_security["values"]=("Select","Your Date of Birth","Your Nick Name","Your Favorite Book")

        self.combo_security.current(0)

        self.combo_security.place(x=70,y=110,width=270)


        #label2

        sa =lb1= Label(self.root2,text="Security Answer:",font=("times new roman",15,"bold"),fg="#002B53",bg="#F2F2F2")

        sa.place(x=70,y=150)


        #entry2

        self.txtpwd=ttk.Entry(self.root2,textvariable=self.var_sa,font=("times new roman",15,"bold"))

        self.txtpwd.place(x=70,y=180,width=270)


        #label2

        new_pwd =lb1= Label(self.root2,text="New Password:",font=("times new roman",15,"bold"),fg="#002B53",bg="#F2F2F2")

        new_pwd.place(x=70,y=220)


        #entry2


self.new_pwd=ttk.Entry(self.root2,textvariable=self.var_pwd,font=("times new roman",15,"bold"))

        self.new_pwd.place(x=70,y=250,width=270)


        # Creating Button New Password
        loginbtn=Button(self.root2,command=self.reset_pass,text="Reset Password",font=("times new roman",15,"bold"),bd=0,relief=RIDGE,fg="#fff",bg="#002B53",activeforeground="white",activebackground="#007ACC")

        loginbtn.place(x=70,y=300,width=270,height=35)




# =====================main program Face deteion system====================


class Face_Recognition_System:

  def __init__(self,root):

    self.root=root

    self.root.geometry("1366x768+0+0")

    self.root.title("Face_Recogonition_System")


# This part is image labels setting start

    # first header image

    img=Image.open(r"C:\Python-FYP-Face-Recognition-Attendence-System-master\Images_GUI\banner.jpg")

    img=img.resize((1366,130),Image.ANTIALIAS)

    self.photoimg=ImageTk.PhotoImage(img)


    # set image as lable

    f_lb1 = Label(self.root,image=self.photoimg)
```

```
f_lb1.place(x=0,y=0,width=1366,height=130)


# backgorund image
bg1=Image.open(r"C:\Python-FYP-Face-Recognition-Attendence-System-master\Images_GUI\bg3.jpg")
bg1=bg1.resize((1366,768),Image.ANTIALIAS)
self.photobg1=ImageTk.PhotoImage(bg1)


# set image as lable
bg_img = Label(self.root,image=self.photobg1)
bg_img.place(x=0,y=130,width=1366,height=768)


#title section
title_lb1 = Label(bg_img,text="Attendance Managment System Using Facial Recognition",font=("verdana",30,"bold"),bg="white",fg="navyblue")
title_lb1.place(x=0,y=0,width=1366,height=45)


# Create buttons below the section
# -------------------------------------------------------------------------------------------------------------
# student button 1
std_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-Attendence-System-master\Images_GUI\std1.jpg")
std_img_btn=std_img_btn.resize((180,180),Image.ANTIALIAS)
self.std_img1=ImageTk.PhotoImage(std_img_btn)
```

```
        std_b1                                                                                   =
Button(bg_img,command=self.student_pannels,image=self.std_img1,cursor="h
and2")

        std_b1.place(x=250,y=100,width=180,height=180)


        std_b1_1  =  Button(bg_img,command=self.student_pannels,text="Student
Pannel",cursor="hand2",font=("tahoma",15,"bold"),bg="white",fg="navyblue")

        std_b1_1.place(x=250,y=280,width=180,height=45)


        # Detect Face  button 2

        det_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-
Attendence-System-master\Images_GUI\det1.jpg")

        det_img_btn=det_img_btn.resize((180,180),Image.ANTIALIAS)

        self.det_img1=ImageTk.PhotoImage(det_img_btn)


        det_b1                                                                                   =
Button(bg_img,command=self.face_rec,image=self.det_img1,cursor="hand2",)

        det_b1.place(x=480,y=100,width=180,height=180)


        det_b1_1          =          Button(bg_img,command=self.face_rec,text="Face
Detector",cursor="hand2",font=("tahoma",15,"bold"),bg="white",fg="navyblue
")

        det_b1_1.place(x=480,y=280,width=180,height=45)


         # Attendance System  button 3

        att_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-Attendence-
System-master\Images_GUI\att.jpg")
```

```
        att_img_btn=att_img_btn.resize((180,180),Image.ANTIALIAS)

        self.att_img1=ImageTk.PhotoImage(att_img_btn)


        att_b1                                                                    =
Button(bg_img,command=self.attendance_pannel,image=self.att_img1,cursor="
hand2",)

        att_b1.place(x=710,y=100,width=180,height=180)


        att_b1_1                                                                  =
Button(bg_img,command=self.attendance_pannel,text="Attendance",cursor="h
and2",font=("tahoma",15,"bold"),bg="white",fg="navyblue")

        att_b1_1.place(x=710,y=280,width=180,height=45)


         # Help  Support  button 4

        hlp_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-
Attendence-System-master\Images_GUI\hlp.jpg")

        hlp_img_btn=hlp_img_btn.resize((180,180),Image.ANTIALIAS)

        self.hlp_img1=ImageTk.PhotoImage(hlp_img_btn)


        hlp_b1 = Button(bg_img,image=self.hlp_img1,cursor="hand2",)

        hlp_b1.place(x=940,y=100,width=180,height=180)


        hlp_b1_1                    =                    Button(bg_img,text="Help
Support",cursor="hand2",font=("tahoma",15,"bold"),bg="white",fg="navyblue"
)

        hlp_b1_1.place(x=940,y=280,width=180,height=45)
```

```
# Top 4 buttons end.......

# -----------------------------------------------------------------------------------------------------------------------

# Start below buttons.........

 # Train   button 5

tra_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-Attendence-System-master\Images_GUI\tra1.jpg")

tra_img_btn=tra_img_btn.resize((180,180),Image.ANTIALIAS)

self.tra_img1=ImageTk.PhotoImage(tra_img_btn)


tra_b1 = Button(bg_img,command=self.train_pannels,image=self.tra_img1,cursor="hand2",)

tra_b1.place(x=250,y=330,width=180,height=180)


tra_b1_1 = Button(bg_img,command=self.train_pannels,text="Data Train",cursor="hand2",font=("tahoma",15,"bold"),bg="white",fg="navyblue")

tra_b1_1.place(x=250,y=510,width=180,height=45)


# Photo   button 6

pho_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-Attendence-System-master\Images_GUI\qr1.png")

pho_img_btn=pho_img_btn.resize((180,180),Image.ANTIALIAS)

self.pho_img1=ImageTk.PhotoImage(pho_img_btn)
```

```
        pho_b1                                                                    =
Button(bg_img,command=self.open_img,image=self.pho_img1,cursor="hand2"
,)

        pho_b1.place(x=480,y=330,width=180,height=180)


        pho_b1_1        =        Button(bg_img,command=self.open_img,text="QR-
Codes",cursor="hand2",font=("tahoma",15,"bold"),bg="white",fg="navyblue")

        pho_b1_1.place(x=480,y=510,width=180,height=45)


        # Developers   button 7

        dev_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-
Attendence-System-master\Images_GUI\dev.jpg")

        dev_img_btn=dev_img_btn.resize((180,180),Image.ANTIALIAS)

        self.dev_img1=ImageTk.PhotoImage(dev_img_btn)


        dev_b1                                                                    =
Button(bg_img,command=self.developr,image=self.dev_img1,cursor="hand2",)

        dev_b1.place(x=710,y=330,width=180,height=180)


        dev_b1_1                                                                  =
Button(bg_img,command=self.developr,text="Developers",cursor="hand2",font
=("tahoma",15,"bold"),bg="white",fg="navyblue")

        dev_b1_1.place(x=710,y=510,width=180,height=45)


        # exit   button 8

        exi_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-
Attendence-System-master\Images_GUI\exi.jpg")
```

```
        exi_img_btn=exi_img_btn.resize((180,180),Image.ANTIALIAS)
        self.exi_img1=ImageTk.PhotoImage(exi_img_btn)

        exi_b1 = Button(bg_img,image=self.exi_img1,cursor="hand2",)
        exi_b1.place(x=940,y=330,width=180,height=180)

        exi_b1_1 = Button(bg_img,text="Exit",cursor="hand2",font=("tahoma",15,"bold"),bg="white",fg="navyblue")
        exi_b1_1.place(x=940,y=510,width=180,height=45)

# ==================Funtion for Open Images Folder==================
    def open_img(self):
        os.startfile("data_img")
# ==================Functions Buttons=====================
    def student_pannels(self):
        self.new_window=Toplevel(self.root)
        self.app=Student(self.new_window)

    def train_pannels(self):
        self.new_window=Toplevel(self.root)
        self.app=Train(self.new_window)

    def face_rec(self):
        self.new_window=Toplevel(self.root)
        self.app=Face_Recognition(self.new_window)

    def attendance_pannel(self):
        self.new_window=Toplevel(self.root)
        self.app=Attendance(self.new_window)

    def developr(self):
        self.new_window=Toplevel(self.root)
        self.app=Developer(self.new_window)

    def open_img(self):
        os.startfile("dataset")
if __name__ == "__main__":
    root=Tk()
    app=Login(root)
    root.mainloop()
```

Main.py:

```
from tkinter import*
from tkinter import ttk
from train import Train
from PIL import Image,ImageTk
from student import Student
from train import Train
from face_recognition import Face_Recognition
from attendance import Attendance
from developer import Developer
import os
from helpsupport import Helpsupport

class Face_Recognition_System:
    def __init__(self,root):
        self.root=root
        self.root.geometry("1366x768+0+0")
        self.root.title("Face_Recogonition_System")

# This part is image labels setting start
        # first header image
        img=Image.open(r"C:\Python-FYP-Face-Recognition-Attendence-
System-master\Images_GUI\banner.jpg")
        img=img.resize((1366,130),Image.ANTIALIAS)
        self.photoimg=ImageTk.PhotoImage(img)

        # set image as lable
        f_lb1 = Label(self.root,image=self.photoimg)
        f_lb1.place(x=0,y=0,width=1366,height=130)

        # backgorund image
        bg1=Image.open(r"C:\Python-FYP-Face-Recognition-Attendence-
System-master\Images_GUI\bg3.jpg")

        bg1=bg1.resize((1366,768),Image.ANTIALIAS)

        self.photobg1=ImageTk.PhotoImage(bg1)


        # set image as lable

        bg_img = Label(self.root,image=self.photobg1)

        bg_img.place(x=0,y=130,width=1366,height=768)


        #title section

        title_lb1 = Label(bg_img,text="Attendance Managment System
Using Facial
Recognition",font=("verdana",30,"bold"),bg="white",fg="navyblue")

        title_lb1.place(x=0,y=0,width=1366,height=45)


        # Create buttons below the section

        # ------------------------------------------------------------------------
-------------------------------------

        # student button 1

        std_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-
Attendence-System-master\Images_GUI\std1.jpg")

        std_img_btn=std_img_btn.resize((180,180),Image.ANTIALIAS)

        self.std_img1=ImageTk.PhotoImage(std_img_btn)


        std_b1 =
Button(bg_img,command=self.student_pannels,image=self.std_img1,cu
rsor="hand2")
```

```
        std_b1.place(x=250,y=100,width=180,height=180)


        std_b1_1 =
Button(bg_img,command=self.student_pannels,text="Student
Pannel",cursor="hand2",font=("tahoma",15,"bold"),bg="white",fg="nav
yblue")

        std_b1_1.place(x=250,y=280,width=180,height=45)


        # Detect Face  button 2

        det_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-
Attendence-System-master\Images_GUI\det1.jpg")

        det_img_btn=det_img_btn.resize((180,180),Image.ANTIALIAS)

        self.det_img1=ImageTk.PhotoImage(det_img_btn)


        det_b1 =
Button(bg_img,command=self.face_rec,image=self.det_img1,cursor="h
and2",)

        det_b1.place(x=480,y=100,width=180,height=180)


        det_b1_1 = Button(bg_img,command=self.face_rec,text="Face
Detector",cursor="hand2",font=("tahoma",15,"bold"),bg="white",fg="na
vyblue")

        det_b1_1.place(x=480,y=280,width=180,height=45)


         # Attendance System  button 3

        att_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-
Attendence-System-master\Images_GUI\att.jpg")

        att_img_btn=att_img_btn.resize((180,180),Image.ANTIALIAS)
        self.att_img1=ImageTk.PhotoImage(att_img_btn)


        att_b1 =
Button(bg_img,command=self.attendance_pannel,image=self.att_img1,
cursor="hand2",)

        att_b1.place(x=710,y=100,width=180,height=180)


        att_b1_1 =
Button(bg_img,command=self.attendance_pannel,text="Attendance",c
ursor="hand2",font=("tahoma",15,"bold"),bg="white",fg="navyblue")

        att_b1_1.place(x=710,y=280,width=180,height=45)


         # Help  Support  button 4

        hlp_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-
Attendence-System-master\Images_GUI\hlp.jpg")

        hlp_img_btn=hlp_img_btn.resize((180,180),Image.ANTIALIAS)

        self.hlp_img1=ImageTk.PhotoImage(hlp_img_btn)


        hlp_b1 =
Button(bg_img,command=self.helpSupport,image=self.hlp_img1,cursor
="hand2",)

        hlp_b1.place(x=940,y=100,width=180,height=180)


        hlp_b1_1 = Button(bg_img,command=self.helpSupport,text="Help
Support",cursor="hand2",font=("tahoma",15,"bold"),bg="white",fg="na
vyblue")

        hlp_b1_1.place(x=940,y=280,width=180,height=45)
        # Top 4 buttons end.......

        # --------------------------------------------------------------------------
------------------------------------------

        # Start below buttons.........

         # Train   button 5

        tra_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-
Attendence-System-master\Images_GUI\tra1.jpg")

        tra_img_btn=tra_img_btn.resize((180,180),Image.ANTIALIAS)

        self.tra_img1=ImageTk.PhotoImage(tra_img_btn)


        tra_b1 =
Button(bg_img,command=self.train_pannels,image=self.tra_img1,curs
or="hand2",)

        tra_b1.place(x=250,y=330,width=180,height=180)


        tra_b1_1 =
Button(bg_img,command=self.train_pannels,text="Data
Train",cursor="hand2",font=("tahoma",15,"bold"),bg="white",fg="navyb
lue")

        tra_b1_1.place(x=250,y=510,width=180,height=45)


        # Photo   button 6

        pho_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-
Attendence-System-master\Images_GUI\qr1.png")

        pho_img_btn=pho_img_btn.resize((180,180),Image.ANTIALIAS)

        self.pho_img1=ImageTk.PhotoImage(pho_img_btn)
        pho_b1 =
Button(bg_img,command=self.open_img,image=self.pho_img1,cursor=
"hand2",)

        pho_b1.place(x=480,y=330,width=180,height=180)


        pho_b1_1 = Button(bg_img,command=self.open_img,text="QR-
Codes",cursor="hand2",font=("tahoma",15,"bold"),bg="white",fg="navy
blue")

        pho_b1_1.place(x=480,y=510,width=180,height=45)


        # Developers   button 7

        dev_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-
Attendence-System-master\Images_GUI\dev.jpg")

        dev_img_btn=dev_img_btn.resize((180,180),Image.ANTIALIAS)

        self.dev_img1=ImageTk.PhotoImage(dev_img_btn)


        dev_b1 =
Button(bg_img,command=self.developr,image=self.dev_img1,cursor="h
and2",)

        dev_b1.place(x=710,y=330,width=180,height=180)


        dev_b1_1 =
Button(bg_img,command=self.developr,text="Developers",cursor="han
d2",font=("tahoma",15,"bold"),bg="white",fg="navyblue")

        dev_b1_1.place(x=710,y=510,width=180,height=45)


        # exit   button 8
        exi_img_btn=Image.open(r"C:\Python-FYP-Face-Recognition-
Attendence-System-master\Images_GUI\exi.jpg")

        exi_img_btn=exi_img_btn.resize((180,180),Image.ANTIALIAS)

        self.exi_img1=ImageTk.PhotoImage(exi_img_btn)


        exi_b1 =
Button(bg_img,command=self.Close,image=self.exi_img1,cursor="hand
2",)

        exi_b1.place(x=940,y=330,width=180,height=180)


        exi_b1_1 =
Button(bg_img,command=self.Close,text="Exit",cursor="hand2",font=(
"tahoma",15,"bold"),bg="white",fg="navyblue")

        exi_b1_1.place(x=940,y=510,width=180,height=45)


# ==================Funtion for Open Images
Folder==================

    def open_img(self):

        os.startfile("dataset")

# ==================Functions
Buttons=====================

    def student_pannels(self):

        self.new_window=Toplevel(self.root)

        self.app=Student(self.new_window)


    def train_pannels(self):

        self.new_window=Toplevel(self.root)
        self.app=Train(self.new_window)

    def face_rec(self):
        self.new_window=Toplevel(self.root)
        self.app=Face_Recognition(self.new_window)

    def attendance_pannel(self):
        self.new_window=Toplevel(self.root)
        self.app=Attendance(self.new_window)

    def developr(self):
        self.new_window=Toplevel(self.root)
        self.app=Developer(self.new_window)

    def helpSupport(self):
        self.new_window=Toplevel(self.root)
        self.app=Helpsupport(self.new_window)

    def Close(self):
        root.destroy()

if __name__ == "__main__":
    root=Tk()
    obj=Face_Recognition_System(root)
    root.mainloop()
```

student.py:

```
from tkinter import*

from tkinter import ttk

from PIL import Image,ImageTk

from tkinter import messagebox

import mysql.connector

import cv2

# Testing Connection

"""

conn = mysql.connector.connect(username='root',
password='Ganga@932',host='localhost',database='face_recognition',port=3306
)

cursor = conn.cursor()



cursor.execute("show databases")



data = cursor.fetchall()



print(data)



conn.close()
```

```
"""
class Student:
    def __init__(self,root):
        self.root=root
        self.root.geometry("1366x768+0+0")
        self.root.title("Student Pannel")

        #-----------Variables-------------------
        self.var_dep=StringVar()
        self.var_course=StringVar()
        self.var_year=StringVar()
        self.var_semester=StringVar()
        self.var_std_id=StringVar()
        self.var_std_name=StringVar()
        self.var_div=StringVar()
        self.var_roll=StringVar()
        self.var_gender=StringVar()
        self.var_dob=StringVar()
        self.var_email=StringVar()
        self.var_mob=StringVar()
        self.var_address=StringVar()
        self.var_teacher=StringVar()

    # This part is image labels setting start
```

```
        # first header image

        img=Image.open(r"C:\Python-FYP-Face-Recognition-Attendence-System-
master\Images_GUI\banner.jpg")

        img=img.resize((1366,130),Image.ANTIALIAS)

        self.photoimg=ImageTk.PhotoImage(img)


        # set image as lable

        f_lb1 = Label(self.root,image=self.photoimg)

        f_lb1.place(x=0,y=0,width=1366,height=130)


         # backgorund image

        bg1=Image.open(r"C:\Python-FYP-Face-Recognition-Attendence-System-
master\Images_GUI\bg3.jpg")

        bg1=bg1.resize((1366,768),Image.ANTIALIAS)

        self.photobg1=ImageTk.PhotoImage(bg1)


        # set image as lable

        bg_img = Label(self.root,image=self.photobg1)

        bg_img.place(x=0,y=130,width=1366,height=768)


        #title section

        title_lb1 = Label(bg_img,text="Welcome to Student
Pannel",font=("verdana",30,"bold"),bg="white",fg="navyblue")

        title_lb1.place(x=0,y=0,width=1366,height=45)
        # Creating Frame
        main_frame = Frame(bg_img,bd=2,bg="white") #bd mean border
        main_frame.place(x=5,y=55,width=1355,height=510)

        # Left Label Frame
        left_frame =
LabelFrame(main_frame,bd=2,bg="white",relief=RIDGE,text="Student
Details",font=("verdana",12,"bold"),fg="navyblue")
        left_frame.place(x=10,y=10,width=660,height=480)

        # Current Course
        current_course_frame =
LabelFrame(left_frame,bd=2,bg="white",relief=RIDGE,text="Current
Course",font=("verdana",12,"bold"),fg="navyblue")
        current_course_frame.place(x=10,y=5,width=635,height=150)

        #label Department

dep_label=Label(current_course_frame,text="Department",font=("verdana",12,
"bold"),bg="white",fg="navyblue")
        dep_label.grid(row=0,column=0,padx=5,pady=15)

        #combo box

dep_combo=ttk.Combobox(current_course_frame,textvariable=self.var_dep,wi
dth=15,font=("verdana",12,"bold"),state="readonly")
        dep_combo["values"]=("Select
Department","BSCS","BSIT","BSENG","BSPHY","BSMATH")
```

```
        dep_combo.current(0)
        dep_combo.grid(row=0,column=1,padx=5,pady=15,sticky=W)

        # ---------------------------------------------------

        #label Course

cou_label=Label(current_course_frame,text="Course",font=("verdana",12,"bold
"),bg="white",fg="navyblue")
        cou_label.grid(row=0,column=2,padx=5,pady=15)

        #combo box

cou_combo=ttk.Combobox(current_course_frame,textvariable=self.var_course,
width=15,font=("verdana",12,"bold"),state="readonly")
        cou_combo["values"]=("Select Course","SE","FE","TE","BE","MS")
        cou_combo.current(0)
        cou_combo.grid(row=0,column=3,padx=5,pady=15,sticky=W)

        #----------------------------------------------------------

        #label Year

year_label=Label(current_course_frame,text="Year",font=("verdana",12,"bold"
),bg="white",fg="navyblue")
        year_label.grid(row=1,column=0,padx=5,sticky=W)
        #combo box

year_combo=ttk.Combobox(current_course_frame,textvariable=self.var_year,w
idth=15,font=("verdana",12,"bold"),state="readonly")
        year_combo["values"]=("Select Year","2017-21","2018-22","2019-
23","2020-24","2021-25")
        year_combo.current(0)
        year_combo.grid(row=1,column=1,padx=5,pady=15,sticky=W)


        #-------------------------------------------------------------


        #label Semester

year_label=Label(current_course_frame,text="Semester",font=("verdana",12,"b
old"),bg="white",fg="navyblue")
        year_label.grid(row=1,column=2,padx=5,sticky=W)


        #combo box

year_combo=ttk.Combobox(current_course_frame,textvariable=self.var_semest
er,width=15,font=("verdana",12,"bold"),state="readonly")
        year_combo["values"]=("Select Semester","Semester-1","Semester-
2","Semester-3","Semester-4","Semester-5","Semester-6","Semester-
7","Semester-8")
        year_combo.current(0)
        year_combo.grid(row=1,column=3,padx=5,pady=15,sticky=W)


        #Class Student Information
        class_Student_frame =
LabelFrame(left_frame,bd=2,bg="white",relief=RIDGE,text="Class Student
Information",font=("verdana",12,"bold"),fg="navyblue")

        class_Student_frame.place(x=10,y=160,width=635,height=230)


        #Student id

        studentId_label = Label(class_Student_frame,text="Std-
ID:",font=("verdana",12,"bold"),fg="navyblue",bg="white")

        studentId_label.grid(row=0,column=0,padx=5,pady=5,sticky=W)


        studentId_entry =
ttk.Entry(class_Student_frame,textvariable=self.var_std_id,width=15,font=("ver
dana",12,"bold"))

        studentId_entry.grid(row=0,column=1,padx=5,pady=5,sticky=W)


        #Student name

        student_name_label = Label(class_Student_frame,text="Std-
Name:",font=("verdana",12,"bold"),fg="navyblue",bg="white")

        student_name_label.grid(row=0,column=2,padx=5,pady=5,sticky=W)


        student_name_entry =
ttk.Entry(class_Student_frame,textvariable=self.var_std_name,width=15,font=(
"verdana",12,"bold"))

        student_name_entry.grid(row=0,column=3,padx=5,pady=5,sticky=W)


        #Class Didvision

        student_div_label = Label(class_Student_frame,text="Class
Division:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
```

```
        student_div_label.grid(row=1,column=0,padx=5,pady=5,sticky=W)


div_combo=ttk.Combobox(class_Student_frame,textvariable=self.var_div,widt
h=13,font=("verdana",12,"bold"),state="readonly")
        div_combo["values"]=("Morning","Evening")
        div_combo.current(0)
        div_combo.grid(row=1,column=1,padx=5,pady=5,sticky=W)

        #Roll No
        student_roll_label = Label(class_Student_frame,text="Roll-
No:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
        student_roll_label.grid(row=1,column=2,padx=5,pady=5,sticky=W)

        student_roll_entry =
ttk.Entry(class_Student_frame,textvariable=self.var_roll,width=15,font=("verda
na",12,"bold"))
        student_roll_entry.grid(row=1,column=3,padx=5,pady=5,sticky=W)

        #Gender
        student_gender_label =
Label(class_Student_frame,text="Gender:",font=("verdana",12,"bold"),fg="nav
yblue",bg="white")
        student_gender_label.grid(row=2,column=0,padx=5,pady=5,sticky=W)

        #combo box
```

```
gender_combo=ttk.Combobox(class_Student_frame,textvariable=self.var_gend
er,width=13,font=("verdana",12,"bold"),state="readonly")
        gender_combo["values"]=("Male","Female","Others")
        gender_combo.current(0)
        gender_combo.grid(row=2,column=1,padx=5,pady=5,sticky=W)


        #Date of Birth
        student_dob_label =
Label(class_Student_frame,text="DOB:",font=("verdana",12,"bold"),fg="navyb
lue",bg="white")
        student_dob_label.grid(row=2,column=2,padx=5,pady=5,sticky=W)


        student_dob_entry =
ttk.Entry(class_Student_frame,textvariable=self.var_dob,width=15,font=("verda
na",12,"bold"))
        student_dob_entry.grid(row=2,column=3,padx=5,pady=5,sticky=W)


        #Email
        student_email_label =
Label(class_Student_frame,text="Email:",font=("verdana",12,"bold"),fg="navy
blue",bg="white")
        student_email_label.grid(row=3,column=0,padx=5,pady=5,sticky=W)


        student_email_entry =
ttk.Entry(class_Student_frame,textvariable=self.var_email,width=15,font=("ver
dana",12,"bold"))
        student_email_entry.grid(row=3,column=1,padx=5,pady=5,sticky=W)
        #Phone Number

        student_mob_label = Label(class_Student_frame,text="Mob-
No:",font=("verdana",12,"bold"),fg="navyblue",bg="white")

        student_mob_label.grid(row=3,column=2,padx=5,pady=5,sticky=W)


        student_mob_entry =
ttk.Entry(class_Student_frame,textvariable=self.var_mob,width=15,font=("verd
ana",12,"bold"))

        student_mob_entry.grid(row=3,column=3,padx=5,pady=5,sticky=W)


        #Address

        student_address_label =
Label(class_Student_frame,text="Address:",font=("verdana",12,"bold"),fg="na
vyblue",bg="white")

        student_address_label.grid(row=4,column=0,padx=5,pady=5,sticky=W)


        student_address_entry =
ttk.Entry(class_Student_frame,textvariable=self.var_address,width=15,font=("v
erdana",12,"bold"))

        student_address_entry.grid(row=4,column=1,padx=5,pady=5,sticky=W)


        #Teacher Name

        student_tutor_label = Label(class_Student_frame,text="Tutor
Name:",font=("verdana",12,"bold"),fg="navyblue",bg="white")

        student_tutor_label.grid(row=4,column=2,padx=5,pady=5,sticky=W)
        student_tutor_entry = 
ttk.Entry(class_Student_frame,textvariable=self.var_teacher,width=15,font=("v
erdana",12,"bold"))

        student_tutor_entry.grid(row=4,column=3,padx=5,pady=5,sticky=W)


        #Radio Buttons

        self.var_radio1=StringVar()

        radiobtn1=ttk.Radiobutton(class_Student_frame,text="Take Photo 
Sample",variable=self.var_radio1,value="Yes")

        radiobtn1.grid(row=5,column=0,padx=5,pady=5,sticky=W)


        radiobtn1=ttk.Radiobutton(class_Student_frame,text="No Photo 
Sample",variable=self.var_radio1,value="No")

        radiobtn1.grid(row=5,column=1,padx=5,pady=5,sticky=W)


        #Button Frame

        btn_frame = Frame(left_frame,bd=2,bg="white",relief=RIDGE)

        btn_frame.place(x=10,y=390,width=635,height=60)


        #save button


save_btn=Button(btn_frame,command=self.add_data,text="Save",width=7,font
=("verdana",12,"bold"),fg="white",bg="navyblue")

        save_btn.grid(row=0,column=0,padx=5,pady=10,sticky=W)


        #update button
```

```
update_btn=Button(btn_frame,command=self.update_data,text="Update",width
=7,font=("verdana",12,"bold"),fg="white",bg="navyblue")
        update_btn.grid(row=0,column=1,padx=5,pady=8,sticky=W)


        #delete button

del_btn=Button(btn_frame,command=self.delete_data,text="Delete",width=7,fo
nt=("verdana",12,"bold"),fg="white",bg="navyblue")
        del_btn.grid(row=0,column=2,padx=5,pady=10,sticky=W)


        #reset button

reset_btn=Button(btn_frame,command=self.reset_data,text="Reset",width=7,fo
nt=("verdana",12,"bold"),fg="white",bg="navyblue")
        reset_btn.grid(row=0,column=3,padx=5,pady=10,sticky=W)


        #take photo button

take_photo_btn=Button(btn_frame,command=self.generate_dataset,text="Take
Pic",width=9,font=("verdana",12,"bold"),fg="white",bg="navyblue")
        take_photo_btn.grid(row=0,column=4,padx=5,pady=10,sticky=W)


        #update photo button
        update_photo_btn=Button(btn_frame,text="Update
Pic",width=9,font=("verdana",12,"bold"),fg="white",bg="navyblue")
        update_photo_btn.grid(row=0,column=5,padx=5,pady=10,sticky=W)
        #-----------------------------------------------------------------------

        # Right Label Frame

        right_frame = 
LabelFrame(main_frame,bd=2,bg="white",relief=RIDGE,text="Student 
Details",font=("verdana",12,"bold"),fg="navyblue")

        right_frame.place(x=680,y=10,width=660,height=480)


        #Searching System in Right Label Frame

        search_frame = 
LabelFrame(right_frame,bd=2,bg="white",relief=RIDGE,text="Search 
System",font=("verdana",12,"bold"),fg="navyblue")

        search_frame.place(x=10,y=5,width=635,height=80)


        #Phone Number

        search_label = 
Label(search_frame,text="Search:",font=("verdana",12,"bold"),fg="navyblue",b
g="white")

        search_label.grid(row=0,column=0,padx=5,pady=5,sticky=W)

        self.var_searchTX=StringVar()

        #combo box


search_combo=ttk.Combobox(search_frame,textvariable=self.var_searchTX,wi
dth=12,font=("verdana",12,"bold"),state="readonly")

        search_combo["values"]=("Select","Roll-No")
        search_combo.current(0)

        search_combo.grid(row=0,column=1,padx=5,pady=15,sticky=W)


        self.var_search=StringVar()

        search_entry =
ttk.Entry(search_frame,textvariable=self.var_search,width=12,font=("verdana",
12,"bold"))

        search_entry.grid(row=0,column=2,padx=5,pady=5,sticky=W)



search_btn=Button(search_frame,command=self.search_data,text="Search",wid
th=9,font=("verdana",12,"bold"),fg="white",bg="navyblue")

        search_btn.grid(row=0,column=3,padx=5,pady=10,sticky=W)


        showAll_btn=Button(search_frame,command=self.fetch_data,text="Show
All",width=8,font=("verdana",12,"bold"),fg="white",bg="navyblue")

        showAll_btn.grid(row=0,column=4,padx=5,pady=10,sticky=W)


        # ----------------------------Table Frame------------------------------------------
-----

        #Table Frame

        #Searching System in Right Label Frame

        table_frame = Frame(right_frame,bd=2,bg="white",relief=RIDGE)

        table_frame.place(x=10,y=90,width=635,height=360)


        #scroll bar

        scroll_x = ttk.Scrollbar(table_frame,orient=HORIZONTAL)
        scroll_y = ttk.Scrollbar(table_frame,orient=VERTICAL)


        #create table

        self.student_table = 
ttk.Treeview(table_frame,column=("ID","Name","Dep","Course","Year","Sem"
,"Div","Gender","DOB","Mob-No","Address","Roll-
No","Email","Teacher","Photo"),xscrollcommand=scroll_x.set,yscrollcommand
=scroll_y.set)


        scroll_x.pack(side=BOTTOM,fill=X)

        scroll_y.pack(side=RIGHT,fill=Y)

        scroll_x.config(command=self.student_table.xview)

        scroll_y.config(command=self.student_table.yview)


        self.student_table.heading("ID",text="StudentID")

        self.student_table.heading("Name",text="Name")

        self.student_table.heading("Dep",text="Department")

        self.student_table.heading("Course",text="Course")

        self.student_table.heading("Year",text="Year")

        self.student_table.heading("Sem",text="Semester")

        self.student_table.heading("Div",text="Division")

        self.student_table.heading("Gender",text="Gender")

        self.student_table.heading("DOB",text="DOB")

        self.student_table.heading("Mob-No",text="Mob-No")

        self.student_table.heading("Address",text="Address")

        self.student_table.heading("Roll-No",text="Roll-No")
        self.student_table.heading("Email",text="Email")
        self.student_table.heading("Teacher",text="Teacher")
        self.student_table.heading("Photo",text="PhotoSample")
        self.student_table["show"]="headings"

        # Set Width of Colums
        self.student_table.column("ID",width=100)
        self.student_table.column("Name",width=100)
        self.student_table.column("Dep",width=100)
        self.student_table.column("Course",width=100)
        self.student_table.column("Year",width=100)
        self.student_table.column("Sem",width=100)
        self.student_table.column("Div",width=100)
        self.student_table.column("Gender",width=100)
        self.student_table.column("DOB",width=100)
        self.student_table.column("Mob-No",width=100)
        self.student_table.column("Address",width=100)
        self.student_table.column("Roll-No",width=100)
        self.student_table.column("Email",width=100)
        self.student_table.column("Teacher",width=100)
        self.student_table.column("Photo",width=100)

        self.student_table.pack(fill=BOTH,expand=1)
        self.student_table.bind("<ButtonRelease>",self.get_cursor)
```

```
        self.fetch_data()

# ==================Function
Decleration===============================

    def add_data(self):

        if self.var_dep.get()=="Select Department" or self.var_course.get=="Select
Course" or self.var_year.get()=="Select Year" or
self.var_semester.get()=="Select Semester" or self.var_std_id.get()=="" or
self.var_std_name.get()=="" or self.var_div.get()=="" or self.var_roll.get()==""
or self.var_gender.get()=="" or self.var_dob.get()=="" or
self.var_email.get()=="" or self.var_mob.get()=="" or
self.var_address.get()=="" or self.var_teacher.get()=="":

            messagebox.showerror("Error","Please Fill All Fields are
Required!",parent=self.root)

        else:

            try:

                conn = mysql.connector.connect(username='root',
password='Ganga@932',host='localhost',database='face_recognition',port=3306
)

                mycursor = conn.cursor()

                mycursor.execute("insert into student
values(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s)",(

                self.var_std_id.get(),

                self.var_std_name.get(),

                self.var_dep.get(),

                self.var_course.get(),

                self.var_year.get(),

                self.var_semester.get(),

                self.var_div.get(),
                self.var_gender.get(),

                self.var_dob.get(),

                self.var_mob.get(),

                self.var_address.get(),

                self.var_roll.get(),

                self.var_email.get(),

                self.var_teacher.get(),

                self.var_radio1.get()

                ))


                conn.commit()

                self.fetch_data()

                conn.close()

                messagebox.showinfo("Success","All Records are 
Saved!",parent=self.root)

            except Exception as es:

                messagebox.showerror("Error",f"Due to: {str(es)}",parent=self.root)


       # ==========================Fetch data form database to table 
================================



    def fetch_data(self):

        conn = mysql.connector.connect(username='root', 
password='Ganga@932',host='localhost',database='face_recognition',port=3306
)

        mycursor = conn.cursor()
```

```
        mycursor.execute("select * from student")
        data=mycursor.fetchall()

        if len(data)!= 0:
            self.student_table.delete(*self.student_table.get_children())
            for i in data:
                self.student_table.insert("",END,values=i)
            conn.commit()
        conn.close()

    #===============================get cursor
function======================

    def get_cursor(self,event=""):
        cursor_focus = self.student_table.focus()
        content = self.student_table.item(cursor_focus)
        data = content["values"]

        self.var_std_id.set(data[0]),
        self.var_std_name.set(data[1]),
        self.var_dep.set(data[2]),
        self.var_course.set(data[3]),
        self.var_year.set(data[4]),
        self.var_semester.set(data[5]),
        self.var_div.set(data[6]),

        self.var_gender.set(data[7]),

        self.var_dob.set(data[8]),

        self.var_mob.set(data[9]),

        self.var_address.set(data[10]),

        self.var_roll.set(data[11]),

        self.var_email.set(data[12]),

        self.var_teacher.set(data[13]),

        self.var_radio1.set(data[14])

    # ============================================Update
Function=========================

    def update_data(self):

        if self.var_dep.get()=="Select Department" or self.var_course.get=="Select
Course" or self.var_year.get()=="Select Year" or
self.var_semester.get()=="Select Semester" or self.var_std_id.get()=="" or
self.var_std_name.get()=="" or self.var_div.get()=="" or self.var_roll.get()==""
or self.var_gender.get()=="" or self.var_dob.get()=="" or
self.var_email.get()=="" or self.var_mob.get()=="" or
self.var_address.get()=="" or self.var_teacher.get()=="":

            messagebox.showerror("Error","Please Fill All Fields are
Required!",parent=self.root)

        else:

            try:

                Update=messagebox.askyesno("Update","Do you want to Update this
Student Details!",parent=self.root)

                if Update > 0:
```

```
                conn = mysql.connector.connect(username='root',
password='Ganga@932',host='localhost',database='face_recognition',port=3306
)

                mycursor = conn.cursor()

                mycursor.execute("update student set
Name=%s,Department=%s,Course=%s,Year=%s,Semester=%s,Division=%s,G
ender=%s,DOB=%s,Mobile_No=%s,Address=%s,Roll_No=%s,Email=%s,Teac
her_Name=%s,PhotoSample=%s where Student_ID=%s",(

                self.var_std_name.get(),

                self.var_dep.get(),

                self.var_course.get(),

                self.var_year.get(),

                self.var_semester.get(),

                self.var_div.get(),

                self.var_gender.get(),

                self.var_dob.get(),

                self.var_mob.get(),

                self.var_address.get(),

                self.var_roll.get(),

                self.var_email.get(),

                self.var_teacher.get(),

                self.var_radio1.get(),

                self.var_std_id.get()

                ))

            else:

                if not Update:
                return
            messagebox.showinfo("Success","Successfully
Updated!",parent=self.root)
            conn.commit()
            self.fetch_data()
            conn.close()
        except Exception as es:
            messagebox.showerror("Error",f"Due to: {str(es)}",parent=self.root)


#==============================Delete
Function=========================================
def delete_data(self):
    if self.var_std_id.get()=="":
        messagebox.showerror("Error","Student Id Must be
Required!",parent=self.root)
    else:
        try:
            delete=messagebox.askyesno("Delete","Do you want to
Delete?",parent=self.root)
            if delete>0:
                conn = mysql.connector.connect(username='root',
password='Ganga@932',host='localhost',database='face_recognition',port=3306
)
                mycursor = conn.cursor()
                sql="delete from student where Student_ID=%s"
                val=(self.var_std_id.get(),)
                mycursor.execute(sql,val)
            else:
                if not delete:
                return

            conn.commit()
            self.fetch_data()
            conn.close()
            messagebox.showinfo("Delete","Successfully
Deleted!",parent=self.root)
        except Exception as es:
            messagebox.showerror("Error",f"Due to: {str(es)}",parent=self.root)

# Reset Function
def reset_data(self):
    self.var_std_id.set(""),
    self.var_std_name.set(""),
    self.var_dep.set("Select Department"),
    self.var_course.set("Select Course"),
    self.var_year.set("Select Year"),
    self.var_semester.set("Select Semester"),
    self.var_div.set("Morning"),
    self.var_gender.set("Male"),
    self.var_dob.set(""),
    self.var_mob.set(""),
    self.var_address.set(""),
    self.var_roll.set(""),

    self.var_email.set(""),

    self.var_teacher.set(""),

    self.var_radio1.set("")


# ===========================Search
Data===================

def search_data(self):

    if self.var_search.get()=="" or self.var_searchTX.get()=="Select":

        messagebox.showerror("Error","Select Combo option and enter entry
box",parent=self.root)

    else:

        try:

            conn = mysql.connector.connect(username='root',
password='Ganga@932',host='localhost',database='face_recognition',port=3306
)

            my_cursor = conn.cursor()

            sql = "SELECT
Student_ID,Name,Department,Course,Year,Semester,Division,Gender,DOB,M
obile_No,Address,Roll_No,Email,Teacher_Name,PhotoSample FROM student
where Roll_No='" +str(self.var_search.get()) + "'"

            my_cursor.execute(sql)

            # my_cursor.execute("select * from student where Roll_No= "
+str(self.var_search.get())+" "+str(self.var_searchTX.get())+"")

            rows=my_cursor.fetchall()

            if len(rows)!=0:

                self.student_table.delete(*self.student_table.get_children())
                for i in rows:
                    self.student_table.insert("",END,values=i)
                if rows==None:
                    messagebox.showerror("Error","Data Not
Found",parent=self.root)
                    conn.commit()
                conn.close()
            except Exception as es:
                messagebox.showerror("Error",f"Due To :{str(es)}",parent=self.root)


#====================This part is related to Opencv Camera
part======================
# ================================Generate Data set take
image========================
    def generate_dataset(self):
        if self.var_dep.get()=="Select Department" or self.var_course.get=="Select
Course" or self.var_year.get()=="Select Year" or
self.var_semester.get()=="Select Semester" or self.var_std_id.get()=="" or
self.var_std_name.get()=="" or self.var_div.get()=="" or self.var_roll.get()==""
or self.var_gender.get()=="" or self.var_dob.get()=="" or
self.var_email.get()=="" or self.var_mob.get()=="" or
self.var_address.get()=="" or self.var_teacher.get()=="":
            messagebox.showerror("Error","Please Fill All Fields are
Required!",parent=self.root)
        else:
            try:
```

```
        conn = mysql.connector.connect(username='root',
password='Ganga@932',host='localhost',database='face_recognition',port=3306
)

        mycursor = conn.cursor()

        mycursor.execute("select * from student")

        myreslut = mycursor.fetchall()

        id=0

        for x in myreslut:

            id+=1


        mycursor.execute("update student set
Name=%s,Department=%s,Course=%s,Year=%s,Semester=%s,Division=%s,G
ender=%s,DOB=%s,Mobile_No=%s,Address=%s,Roll_No=%s,Email=%s,Teac
her_Name=%s,PhotoSample=%s where Student_ID=%s",(

            self.var_std_name.get(),

            self.var_dep.get(),

            self.var_course.get(),

            self.var_year.get(),

            self.var_semester.get(),

            self.var_div.get(),

            self.var_gender.get(),

            self.var_dob.get(),

            self.var_mob.get(),

            self.var_address.get(),

            self.var_roll.get(),

            self.var_email.get(),
            self.var_teacher.get(),
            self.var_radio1.get(),
            self.var_std_id.get()==id+1
            ))
        conn.commit()
        self.fetch_data()
        self.reset_data()
        conn.close()

        # ====================part of
opencv=====================

        face_classifier =
cv2.CascadeClassifier("haarcascade_frontalface_default.xml")

        def face_croped(img):
            # conver gary sacle
            gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
            faces = face_classifier.detectMultiScale(gray,1.3,5)
            #Scaling factor 1.3
            # Minimum naber 5
            for (x,y,w,h) in faces:
                face_croped=img[y:y+h,x:x+w]
                return face_croped
        cap=cv2.VideoCapture(0)
```

```
            img_id=0
            while True:
                ret,my_frame=cap.read()
                if face_croped(my_frame) is not None:
                    img_id+=1
                    face=cv2.resize(face_croped(my_frame),(200,200))
                    face=cv2.cvtColor(face,cv2.COLOR_BGR2GRAY)
                    file_path="data_img/student."+str(id)+"."+str(img_id)+".jpg"
                    cv2.imwrite(file_path,face)

cv2.putText(face,str(img_id),(50,50),cv2.FONT_HERSHEY_COMPLEX,2,(0,
255,0),2)
                    cv2.imshow("Capture Images",face)


                if cv2.waitKey(1)==13 or int(img_id)==100:
                    break
            cap.release()
            cv2.destroyAllWindows()
            messagebox.showinfo("Result","Generating dataset
completed!",parent=self.root)
        except Exception as es:
            messagebox.showerror("Error",f"Due to: {str(es)}",parent=self.root)


# main class object
if __name__ == "__main__":
    root=Tk()
    obj=Student(root)
    root.mainloop()
```

face recognition.py:

```
# import re
from sys import path
from tkinter import*
from tkinter import ttk
from PIL import Image,ImageTk
import os
import mysql.connector
import cv2
import numpy as np
from tkinter import messagebox
from time import strftime
from datetime import datetime
class Face_Recognition:

    def __init__(self,root):
        self.root=root
        self.root.geometry("1366x768+0+0")
        self.root.title("Face Recognition Pannel")


        # This part is image labels setting start
        # first header image
        img=Image.open(r"C:\Python-FYP-Face-Recognition-Attendence-System-master\Images_GUI\banner.jpg")
        img=img.resize((1366,130),Image.ANTIALIAS)
        self.photoimg=ImageTk.PhotoImage(img)


        # set image as lable
        f_lb1 = Label(self.root,image=self.photoimg)
        f_lb1.place(x=0,y=0,width=1366,height=130)


        # backgorund image
        bg1=Image.open(r"Images_GUI\bg2.jpg")
        bg1=bg1.resize((1366,768),Image.ANTIALIAS)
        self.photobg1=ImageTk.PhotoImage(bg1)


        # set image as lable
        bg_img = Label(self.root,image=self.photobg1)
        bg_img.place(x=0,y=130,width=1366,height=768)


        #title section
        title_lb1 = Label(bg_img,text="Welcome to Face Recognition Pannel",font=("verdana",30,"bold"),bg="white",fg="navyblue")
        title_lb1.place(x=0,y=0,width=1366,height=45)


        # Create buttons below the section

        # --------------------------------------------------------------------------
------------------------------------

        # Training button 1

        std_img_btn=Image.open(r"Images_GUI\f_det.jpg")

        std_img_btn=std_img_btn.resize((180,180),Image.ANTIALIAS)

        self.std_img1=ImageTk.PhotoImage(std_img_btn)


        std_b1 =
Button(bg_img,command=self.face_recog,image=self.std_img1,cursor=
"hand2")

        std_b1.place(x=600,y=170,width=180,height=180)


        std_b1_1 = Button(bg_img,command=self.face_recog,text="Face
Detector",cursor="hand2",font=("tahoma",15,"bold"),bg="white",fg="na
vyblue")

        std_b1_1.place(x=600,y=350,width=180,height=45)

#=====================Attendance=====================


    def mark_attendance(self,i,r,n):

        with open("attendance.csv","r+",newline="\n") as f:

            myDatalist=f.readlines()

            name_list=[]

            for line in myDatalist:
```

```
            entry=line.split((","))
            name_list.append(entry[0])

        if((i not in name_list)) and ((r not in name_list)) and ((n not in 
name_list)):
            now=datetime.now()
            d1=now.strftime("%d/%m/%Y")
            dtString=now.strftime("%H:%M:%S")
            f.writelines(f"\n{i}, {r}, {n}, {dtString}, {d1}, Present")

  #================face recognition==================
  def face_recog(self):
      def 
draw_boundray(img,classifier,scaleFactor,minNeighbors,color,text,clf):
        gray_image=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

featuers=classifier.detectMultiScale(gray_image,scaleFactor,minNeighb
ors)

        coord=[]

        for (x,y,w,h) in featuers:
            cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),3)
            id,predict=clf.predict(gray_image[y:y+h,x:x+w])

            confidence=int((100*(1-predict/300)))
            conn = mysql.connector.connect(username='root',
password='Ganga@932',host='localhost',database='face_recognition',po
rt=3306)

            cursor = conn.cursor()


            cursor.execute("select Name from student where
Student_ID="+str(id))

            n=cursor.fetchone()

            n="+".join(n)


            cursor.execute("select Roll_No from student where
Student_ID="+str(id))

            r=cursor.fetchone()

            r="+".join(r)


            cursor.execute("select Student_ID from student where
Student_ID="+str(id))

            i=cursor.fetchone()

            i="+".join(i)


            if confidence > 77:

               cv2.putText(img,f"Student_ID:{i}",(x,y-
80),cv2.FONT_HERSHEY_COMPLEX,0.8,(64,15,223),2)

               cv2.putText(img,f"Name:{n}",(x,y-
55),cv2.FONT_HERSHEY_COMPLEX,0.8,(64,15,223),2)
```

```
              cv2.putText(img,f"Roll-No:{r}",(x,y-
30),cv2.FONT_HERSHEY_COMPLEX,0.8,(64,15,223),2)

              self.mark_attendance(i,r,n)

           else:

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

              cv2.putText(img,"Unknown Face",(x,y-
5),cv2.FONT_HERSHEY_COMPLEX,0.8,(255,255,0),3)


           coord=[x,y,w,y]


        return coord


     #==========

     def recognize(img,clf,faceCascade):

coord=draw_boundray(img,faceCascade,1.1,10,(255,25,255),"Face",clf)
        return img



faceCascade=cv2.CascadeClassifier("haarcascade_frontalface_default.x
ml")

     clf=cv2.face.LBPHFaceRecognizer_create()

     clf.read("clf.xml")


     videoCap=cv2.VideoCapture(0)
```

```
        while True:
            ret,img=videoCap.read()
            img=recognize(img,clf,faceCascade)
            cv2.imshow("Face Detector",img)

            if cv2.waitKey(1) == 13:
                break
        videoCap.release()
        cv2.destroyAllWindows()




if __name__ == "__main__":
    root=Tk()
    obj=Face_Recognition(root)
    root.mainloop()
```

Attendence.py:

```
# import re
import re
from sys import path
from tkinter import*
from tkinter import ttk
from PIL import Image,ImageTk
import os
import mysql.connector
import cv2
import numpy as np
from tkinter import messagebox
from time import strftime
from datetime import datetime
import csv
from tkinter import filedialog

#Global variable for importCsv Function
mydata=[]
class Attendance:

    def __init__(self,root):
        self.root=root
        self.root.geometry("1366x768+0+0")
        self.root.title("Attendance Pannel")

        #-----------Variables-------------------
        self.var_id=StringVar()
        self.var_roll=StringVar()
        self.var_name=StringVar()
        self.var_dep=StringVar()
        self.var_time=StringVar()
        self.var_date=StringVar()
        self.var_attend=StringVar()
        # This part is image labels setting start
        # first header image
        img=Image.open(r"C:\Python-FYP-Face-Recognition-Attendence-
System-master\Images_GUI\banner.jpg")
        img=img.resize((1366,130),Image.ANTIALIAS)
        self.photoimg=ImageTk.PhotoImage(img)

        # set image as lable
        f_lb1 = Label(self.root,image=self.photoimg)
        f_lb1.place(x=0,y=0,width=1366,height=130)

        # backgorund image
        bg1=Image.open(r"Images_GUI\bg2.jpg")
        bg1=bg1.resize((1366,768),Image.ANTIALIAS)
        self.photobg1=ImageTk.PhotoImage(bg1)

        # set image as lable
        bg_img = Label(self.root,image=self.photobg1)
        bg_img.place(x=0,y=130,width=1366,height=768)

        #title section
        title_lb1 = Label(bg_img,text="Welcome to Attendance
Pannel",font=("verdana",30,"bold"),bg="white",fg="navyblue")
        title_lb1.place(x=0,y=0,width=1366,height=45)
        #========================Section
Creating=====================================

        # Creating Frame

        main_frame = Frame(bg_img,bd=2,bg="white") #bd mean border

        main_frame.place(x=5,y=55,width=1355,height=510)

        # Left Label Frame

        left_frame =
LabelFrame(main_frame,bd=2,bg="white",relief=RIDGE,text="Student
Details",font=("verdana",12,"bold"),fg="navyblue")

        left_frame.place(x=10,y=10,width=660,height=480)

        # ==================================Text boxes and
Combo Boxes====================

        #Student id

        studentId_label = Label(left_frame,text="Std-
ID:",font=("verdana",12,"bold"),fg="navyblue",bg="white")

        studentId_label.grid(row=0,column=0,padx=5,pady=5,sticky=W)

        studentId_entry =
ttk.Entry(left_frame,textvariable=self.var_id,width=15,font=("verdana",
12,"bold"))
```

```
        studentId_entry.grid(row=0,column=1,padx=5,pady=5,sticky=W)


        #Student Roll

        student_roll_label =
Label(left_frame,text="Roll.No:",font=("verdana",12,"bold"),fg="navybl
ue",bg="white")

student_roll_label.grid(row=0,column=2,padx=5,pady=5,sticky=W)


        student_roll_entry =
ttk.Entry(left_frame,textvariable=self.var_roll,width=15,font=("verdana
",12,"bold"))

student_roll_entry.grid(row=0,column=3,padx=5,pady=5,sticky=W)


        #Studnet Name

        student_name_label = Label(left_frame,text="Std-
Name:",font=("verdana",12,"bold"),fg="navyblue",bg="white")

student_name_label.grid(row=1,column=0,padx=5,pady=5,sticky=W)


        student_name_entry =
ttk.Entry(left_frame,textvariable=self.var_name,width=15,font=("verda
na",12,"bold"))

student_name_entry.grid(row=1,column=1,padx=5,pady=5,sticky=W)


        #Department
        # dep_label = 
Label(left_frame,text="Department:",font=("verdana",12,"bold"),fg="na
vyblue",bg="white")

        # dep_label.grid(row=1,column=2,padx=5,pady=5,sticky=W)


        # dep_entry = 
ttk.Entry(left_frame,textvariable=self.var_dep,width=15,font=("verdana
",12,"bold"))

        # dep_entry.grid(row=1,column=3,padx=5,pady=5,sticky=W)


        #time

        time_label = 
Label(left_frame,text="Time:",font=("verdana",12,"bold"),fg="navyblue"
,bg="white")

        time_label.grid(row=1,column=2,padx=5,pady=5,sticky=W)


        time_entry = 
ttk.Entry(left_frame,textvariable=self.var_time,width=15,font=("verdan
a",12,"bold"))

        time_entry.grid(row=1,column=3,padx=5,pady=5,sticky=W)


        #Date

        date_label = 
Label(left_frame,text="Date:",font=("verdana",12,"bold"),fg="navyblue"
,bg="white")

        date_label.grid(row=2,column=0,padx=5,pady=5,sticky=W)
        date_entry =
ttk.Entry(left_frame,textvariable=self.var_date,width=15,font=("verdan
a",12,"bold"))

        date_entry.grid(row=2,column=1,padx=5,pady=5,sticky=W)


        #Attendance

        student_attend_label = Label(left_frame,text="Attend-
status:",font=("verdana",12,"bold"),fg="navyblue",bg="white")


student_attend_label.grid(row=2,column=2,padx=5,pady=5,sticky=W)



attend_combo=ttk.Combobox(left_frame,textvariable=self.var_attend,
width=13,font=("verdana",12,"bold"),state="readonly")

        attend_combo["values"]=("Status","Present","Absent")

        attend_combo.current(0)

        attend_combo.grid(row=2,column=3,padx=5,pady=5,sticky=W)


        # ==================================Table Sql Data
View==========================

        table_frame = Frame(left_frame,bd=2,bg="white",relief=RIDGE)

        table_frame.place(x=10,y=100,width=635,height=310)


        #scroll bar

        scroll_x = ttk.Scrollbar(table_frame,orient=HORIZONTAL)

        scroll_y = ttk.Scrollbar(table_frame,orient=VERTICAL)
        #create table

        self.attendanceReport_left = 
ttk.Treeview(table_frame,column=("ID","Roll_No","Name","Time","Dat
e","Attend"),xscrollcommand=scroll_x.set,yscrollcommand=scroll_y.set
)


        scroll_x.pack(side=BOTTOM,fill=X)

        scroll_y.pack(side=RIGHT,fill=Y)

        scroll_x.config(command=self.attendanceReport_left.xview)

        scroll_y.config(command=self.attendanceReport_left.yview)


        self.attendanceReport_left.heading("ID",text="Std-ID")

        self.attendanceReport_left.heading("Roll_No",text="Roll.No")

        self.attendanceReport_left.heading("Name",text="Std-Name")

        self.attendanceReport_left.heading("Time",text="Time")

        self.attendanceReport_left.heading("Date",text="Date")

        self.attendanceReport_left.heading("Attend",text="Attend-status")

        self.attendanceReport_left["show"]="headings"


        # Set Width of Colums

        self.attendanceReport_left.column("ID",width=100)

        self.attendanceReport_left.column("Roll_No",width=100)

        self.attendanceReport_left.column("Name",width=100)

        self.attendanceReport_left.column("Time",width=100)

        self.attendanceReport_left.column("Date",width=100)

        self.attendanceReport_left.column("Attend",width=100)
        self.attendanceReport_left.pack(fill=BOTH,expand=1)

self.attendanceReport_left.bind("<ButtonRelease>",self.get_cursor_left
)



        # ==========================button
section==========================


        #Button Frame
        btn_frame = Frame(left_frame,bd=2,bg="white",relief=RIDGE)
        btn_frame.place(x=10,y=390,width=635,height=60)


        #Improt button

save_btn=Button(btn_frame,command=self.importCsv,text="Import
CSV",width=12,font=("verdana",12,"bold"),fg="white",bg="navyblue")
        save_btn.grid(row=0,column=0,padx=6,pady=10,sticky=W)


        #Exprot button

update_btn=Button(btn_frame,command=self.exportCsv,text="Export
CSV",width=12,font=("verdana",12,"bold"),fg="white",bg="navyblue")
        update_btn.grid(row=0,column=1,padx=6,pady=8,sticky=W)


        #Update button
```

```
del_btn=Button(btn_frame,command=self.action,text="Update",width
=12,font=("verdana",12,"bold"),fg="white",bg="navyblue")
        del_btn.grid(row=0,column=2,padx=6,pady=10,sticky=W)


        #reset button

reset_btn=Button(btn_frame,command=self.reset_data,text="Reset",wi
dth=12,font=("verdana",12,"bold"),fg="white",bg="navyblue")
        reset_btn.grid(row=0,column=3,padx=6,pady=10,sticky=W)



        # Right
section==================================================
==========


        # Right Label Frame
        right_frame =
LabelFrame(main_frame,bd=2,bg="white",relief=RIDGE,text="Student
Details",font=("verdana",12,"bold"),fg="navyblue")
        right_frame.place(x=680,y=10,width=660,height=480)


        # ----------------------------Table Frame---------------------------------
---------------
        #Table Frame
        #Searching System in Right Label Frame
        table_frame = Frame(right_frame,bd=2,bg="white",relief=RIDGE)
        table_frame.place(x=10,y=90,width=635,height=360)
        #scroll bar
        scroll_x = ttk.Scrollbar(table_frame,orient=HORIZONTAL)
        scroll_y = ttk.Scrollbar(table_frame,orient=VERTICAL)

        #create table
        self.attendanceReport =
ttk.Treeview(table_frame,column=("ID","Roll_No","Name","Time","Dat
e","Attend"),xscrollcommand=scroll_x.set,yscrollcommand=scroll_y.set
)

        scroll_x.pack(side=BOTTOM,fill=X)
        scroll_y.pack(side=RIGHT,fill=Y)
        scroll_x.config(command=self.attendanceReport.xview)
        scroll_y.config(command=self.attendanceReport.yview)

        self.attendanceReport.heading("ID",text="Std-ID")
        self.attendanceReport.heading("Roll_No",text="Roll.No")
        self.attendanceReport.heading("Name",text="Std-Name")
        self.attendanceReport.heading("Time",text="Time")
        self.attendanceReport.heading("Date",text="Date")
        self.attendanceReport.heading("Attend",text="Attend-status")
        self.attendanceReport["show"]="headings"

        # Set Width of Colums
        self.attendanceReport.column("ID",width=100)
        self.attendanceReport.column("Roll_No",width=100)
        self.attendanceReport.column("Name",width=100)
        self.attendanceReport.column("Time",width=100)
        self.attendanceReport.column("Date",width=100)
        self.attendanceReport.column("Attend",width=100)

        self.attendanceReport.pack(fill=BOTH,expand=1)

self.attendanceReport.bind("<ButtonRelease>",self.get_cursor_right)
        self.fetch_data()
    # ==================================update for mysql
button================
    #Update button

del_btn=Button(right_frame,command=self.update_data,text="Update
",width=12,font=("verdana",12,"bold"),fg="white",bg="navyblue")
        del_btn.grid(row=0,column=1,padx=6,pady=10,sticky=W)
    #Update button

del_btn=Button(right_frame,command=self.delete_data,text="Delete",
width=12,font=("verdana",12,"bold"),fg="white",bg="navyblue")
        del_btn.grid(row=0,column=2,padx=6,pady=10,sticky=W)
    # ===============================update function for
mysql database=================
    def update_data(self):
        if self.var_id.get()=="" or self.var_roll.get=="" or
self.var_name.get()=="" or self.var_time.get()=="" or
self.var_date.get()=="" or self.var_attend.get()=="Status":
```

```
         messagebox.showerror("Error","Please Fill All Fields are
Required!",parent=self.root)

      else:

         try:

            Update=messagebox.askyesno("Update","Do you want to
Update this Student Attendance!",parent=self.root)

            if Update > 0:

               conn = mysql.connector.connect(username='root',
password='Ganga@932',host='localhost',database='face_recognition',po
rt=3306)

               mycursor = conn.cursor()

               mycursor.execute("update stdattendance set
std_id=%s,std_roll_no=%s,std_name=%s,std_time=%s,std_date=%s,st
d_attendance=%s where std_id=%s",(

               self.var_id.get(),

               self.var_roll.get(),

               self.var_name.get(),

               self.var_time.get(),

               self.var_date.get(),

               self.var_attend.get(),

               self.var_id.get()

               ))

            else:

               if not Update:

                  return

            messagebox.showinfo("Success","Successfully
Updated!",parent=self.root)
```

```
                conn.commit()
                self.fetch_data()
                conn.close()
            except Exception as es:
                messagebox.showerror("Error",f"Due to:
{str(es)}",parent=self.root)
    # ===============================Delete Attendance form
my sql==============================
    def delete_data(self):
        if self.var_id.get()=="":
            messagebox.showerror("Error","Student Id Must be
Required!",parent=self.root)
        else:
            try:
                delete=messagebox.askyesno("Delete","Do you want to
Delete?",parent=self.root)
                if delete>0:
                    conn = mysql.connector.connect(username='root',
password='Ganga@932',host='localhost',database='face_recognition',po
rt=3306)
                    mycursor = conn.cursor()
                    sql="delete from stdattendance where std_id=%s"
                    val=(self.var_id.get(),)
                    mycursor.execute(sql,val)
                else:
                    if not delete:
                        return
            conn.commit()

            self.fetch_data()

            conn.close()

            messagebox.showinfo("Delete","Successfully
Deleted!",parent=self.root)

         except Exception as es:

            messagebox.showerror("Error",f"Due to:
{str(es)}",parent=self.root)

   # ==========================fatch data form mysql
attendance===========


   def fetch_data(self):

      conn = mysql.connector.connect(username='root',
password='Ganga@932',host='localhost',database='face_recognition',po
rt=3306)

      mycursor = conn.cursor()


      mycursor.execute("select * from stdattendance")

      data=mycursor.fetchall()


      if len(data)!= 0:

self.attendanceReport.delete(*self.attendanceReport.get_children())

         for i in data:

            self.attendanceReport.insert("",END,values=i)

         conn.commit()
```

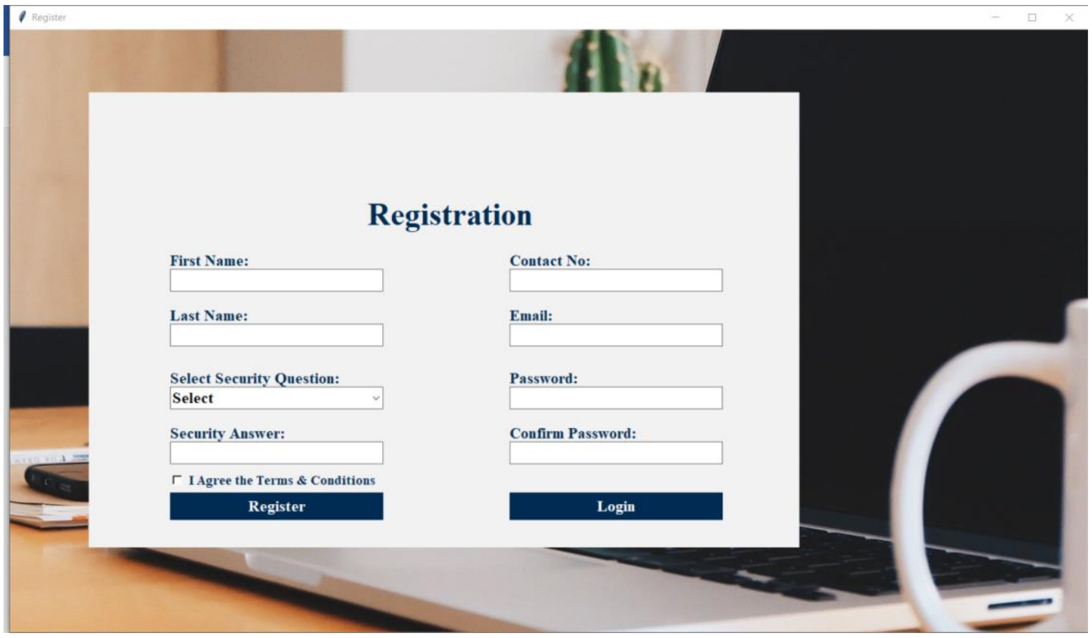
```
        conn.close()


    #==============================Reset
Data=======================
    def reset_data(self):
        self.var_id.set("")
        self.var_roll.set("")
        self.var_name.set("")
        self.var_time.set("")
        self.var_date.set("")
        self.var_attend.set("Status")


    # =========================Fetch Data Import data
===============


    def fetchData(self,rows):
        global mydata
        mydata = rows

self.attendanceReport_left.delete(*self.attendanceReport_left.get_childr
en())
        for i in rows:
            self.attendanceReport_left.insert("",END,values=i)
            print(i)
    def importCsv(self):
        mydata.clear()
        fln=filedialog.askopenfilename(initialdir=os.getcwd(),title="Open
CSV",filetypes=(("CSV File","*.csv"),("All File","*.*")),parent=self.root)
        with open(fln) as myfile:
            csvread=csv.reader(myfile,delimiter=",")
            for i in csvread:
                mydata.append(i)
        self.fetchData(mydata)



    #==================Experot CSV=============
    def exportCsv(self):
        try:
            if len(mydata)<1:
                messagebox.showerror("Error","No Data
Found!",parent=self.root)
                return False

fln=filedialog.asksaveasfilename(initialdir=os.getcwd(),title="Open
CSV",filetypes=(("CSV File","*.csv"),("All File","*.*")),parent=self.root)
            with open(fln,mode="w",newline="") as myfile:
                exp_write=csv.writer(myfile,delimiter=",")
                for i in mydata:
                    exp_write.writerow(i)
            messagebox.showinfo("Successfuly","Export Data
Successfully!")
        except Exception as es:
            messagebox.showerror("Error",f"Due to:
{str(es)}",parent=self.root)


    #=============Cursur Function for
CSV========================


    def get_cursor_left(self,event=""):
        cursor_focus = self.attendanceReport_left.focus()
        content = self.attendanceReport_left.item(cursor_focus)
        data = content["values"]


        self.var_id.set(data[0]),
        self.var_roll.set(data[1]),
        self.var_name.set(data[2]),
        self.var_time.set(data[3]),
        self.var_date.set(data[4]),
        self.var_attend.set(data[5])


    #=============Cursur Function for
mysql========================


    def get_cursor_right(self,event=""):
        cursor_focus = self.attendanceReport.focus()
        content = self.attendanceReport.item(cursor_focus)

        data = content["values"]



        self.var_id.set(data[0]),

        self.var_roll.set(data[1]),

        self.var_name.set(data[2]),

        self.var_time.set(data[3]),

        self.var_date.set(data[4]),

        self.var_attend.set(data[5])

    #===========================================Update
CSV=============================



    # export upadte

    def action(self):

        if self.var_id.get()=="" or self.var_roll.get=="" or
self.var_name.get()=="" or self.var_time.get()=="" or
self.var_date.get()=="" or self.var_attend.get()=="Status":

            messagebox.showerror("Error","Please Fill All Fields are
Required!",parent=self.root)

        else:

            try:

                conn = mysql.connector.connect(username='root',
password='Ganga@932',host='localhost',database='face_recognition',po
rt=3306)

                mycursor = conn.cursor()

                mycursor.execute("insert into stdattendance
values(%s,%s,%s,%s,%s,%s)",(
                self.var_id.get(),
                self.var_roll.get(),
                self.var_name.get(),
                self.var_time.get(),
                self.var_date.get(),
                self.var_attend.get()
                ))

                conn.commit()
                self.fetch_data()
                conn.close()
                messagebox.showinfo("Success","All Records are Saved in
Database!",parent=self.root)
            except Exception as es:
                messagebox.showerror("Error",f"Due to:
{str(es)}",parent=self.root)






    #     conn = mysql.connector.connect(username='root',
password='Ganga@932',host='localhost',database='face_recognition',po
rt=3306)
    #     mycursor = conn.cursor()
```

```
    #     if messagebox.askyesno("Confirmation","Are you sure you want to save attendance on database?"):
    #         for i in mydata:
    #             uid = i[0]
    #             uroll = i[1]
    #             uname = i[2]
    #             utime = i[3]
    #             udate = i[4]
    #             uattend = i[5]
    #             qury = "INSERT INTO stdattendance(std_id, std_roll_no, std_name, std_time, std_date, std_attendance) VALUES(%s,%s,%s,%s,%s,%s)"
    # mycursor.execute(qury,(uid,uroll,uname,utime,udate,uattend))
    #         conn.commit()
    #         conn.close()
    #         messagebox.showinfo("Success","Successfully Updated!",parent=self.root)
    #     else:
    #         return False



if __name__ == "__main__":
    root=Tk()
    obj=Attendance(root)
    root.mainloop()
```

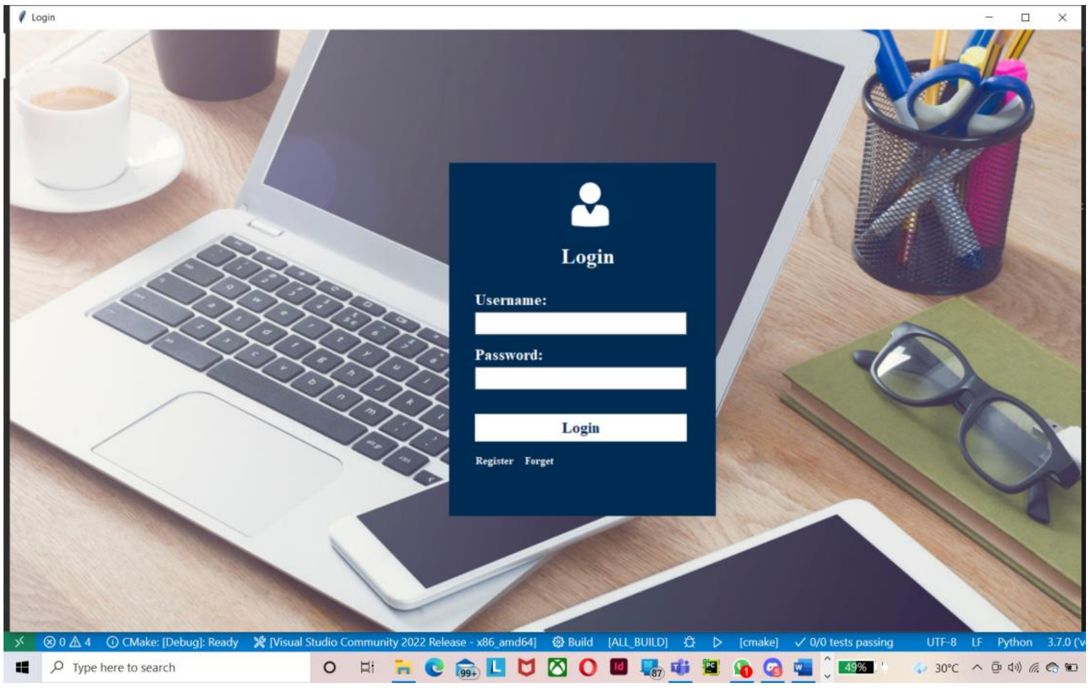
Database.py:

```
import mysql.connector

conn=mysql.connector.connect(user='krishna',
password='krishna',host='localhost',database='face_recognitio
n',port=3306,auth_plugin='mysql_native_password')

cursor = conn.cursor()


cursor.execute("show databases")


data = cursor.fetchall()


print(data)


conn.close()
```

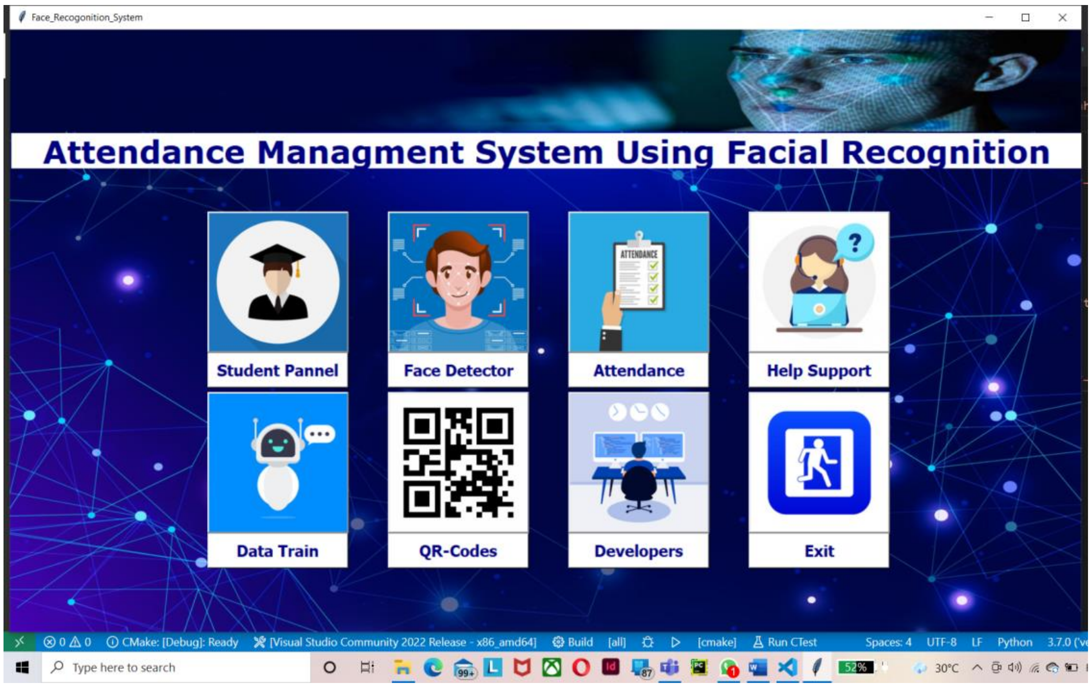
6.3 OUTPUT SCREENSHOTS



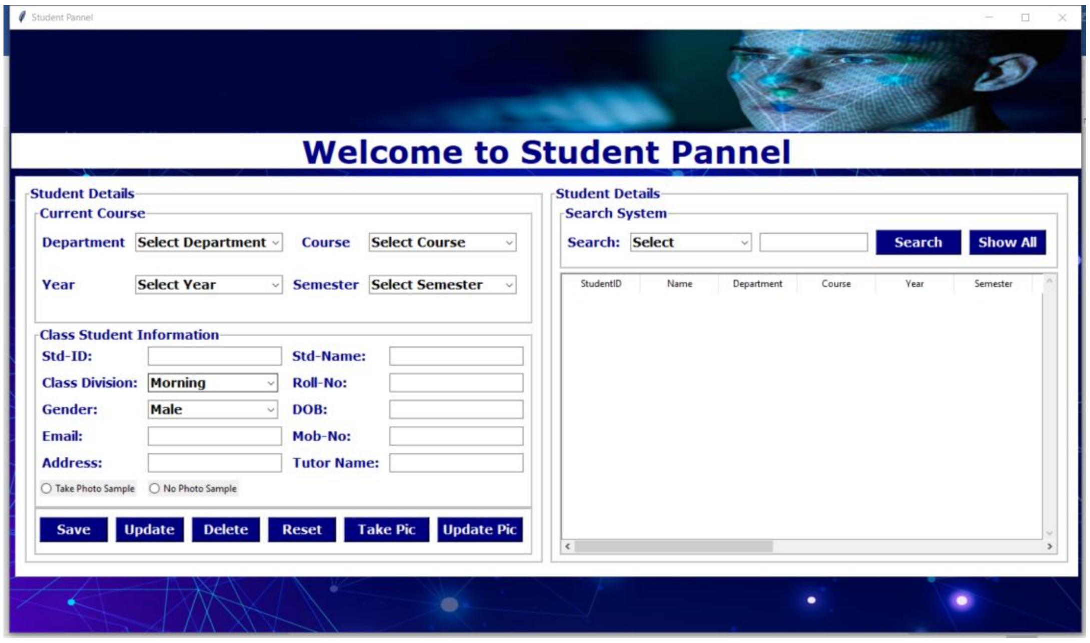
6.3 (a) – Registration page



6.3 (b) – Login page



6.3(c)-Home Page



Student Details

Current Course

Department	Select Department	Course	Select Course
Year	Select Year	Semester	Select Semester

Class Student Information

Std-ID:		Std-Name:	
Class Division:	Morning	Roll-No:	
Gender:	Male	DOB:	
Email:		Mob-No:	
Address:		Tutor Name:	

○ Take Photo Sample ○ No Photo Sample

Save Update Delete Reset Take Pic Update Pic

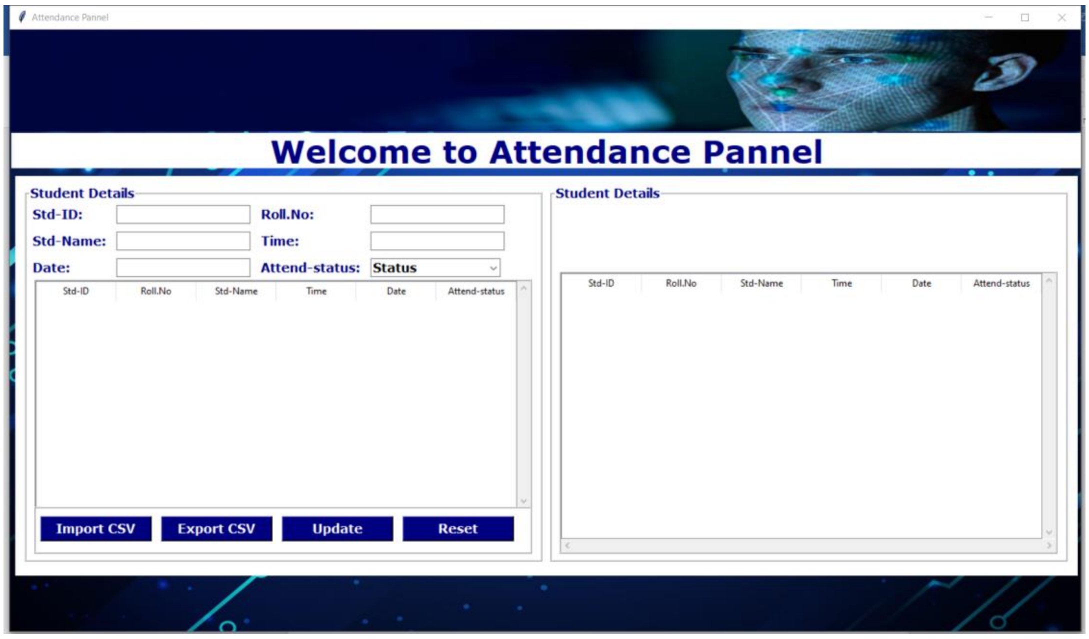
Student Details

Search System

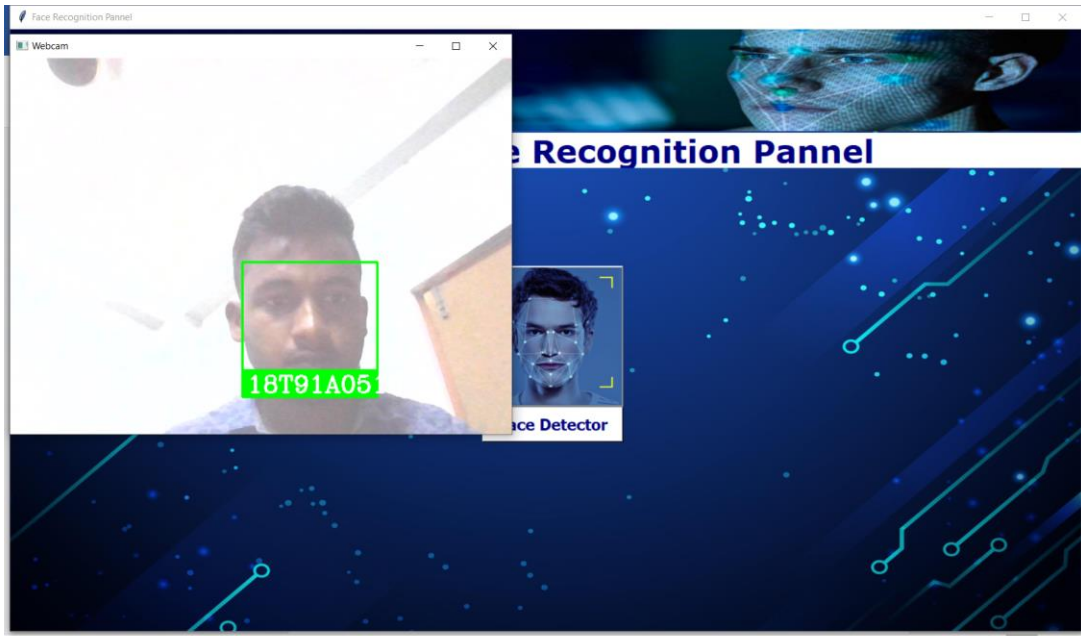
Search: Select Search Show All

StudentID	Name	Department	Course	Year	Semester

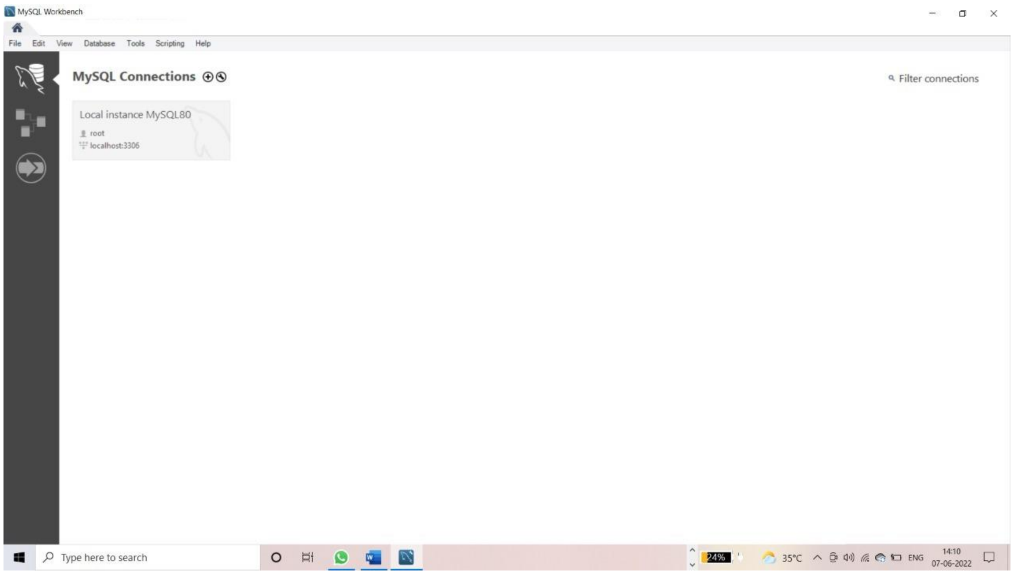
6.3 (d) – Form to fill Student Details



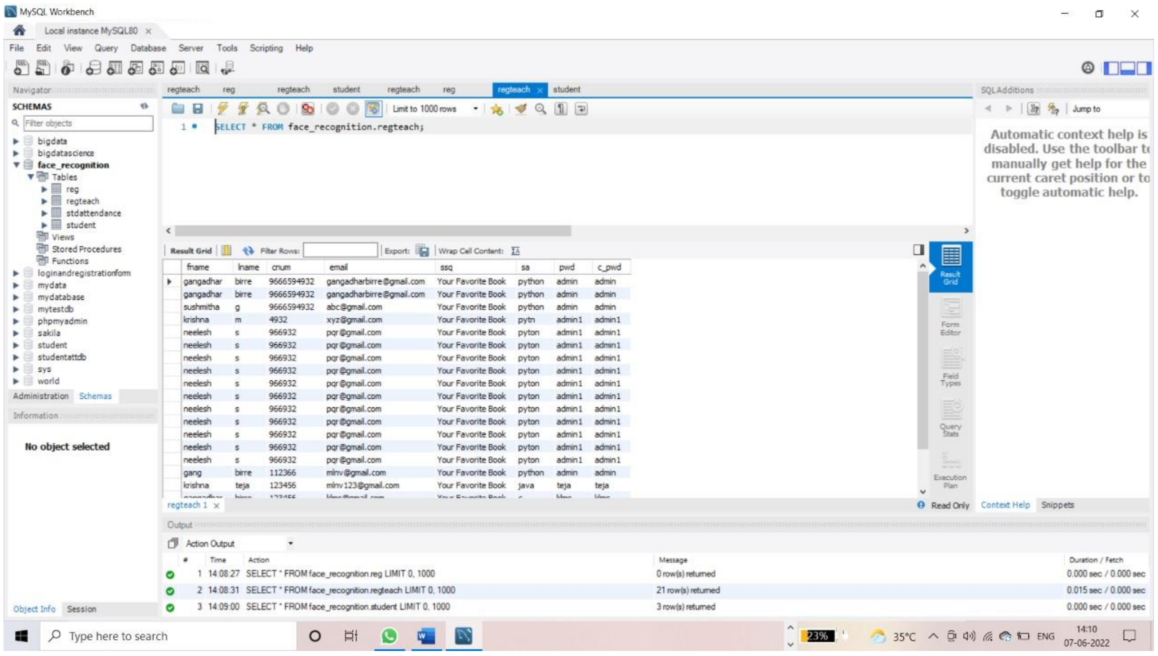
6.3 (e) – Form for Students To mark Attendance



6.3 (f) -Face Capturing to Mark Attendance



6.3 (g) -MYSQL Workbench

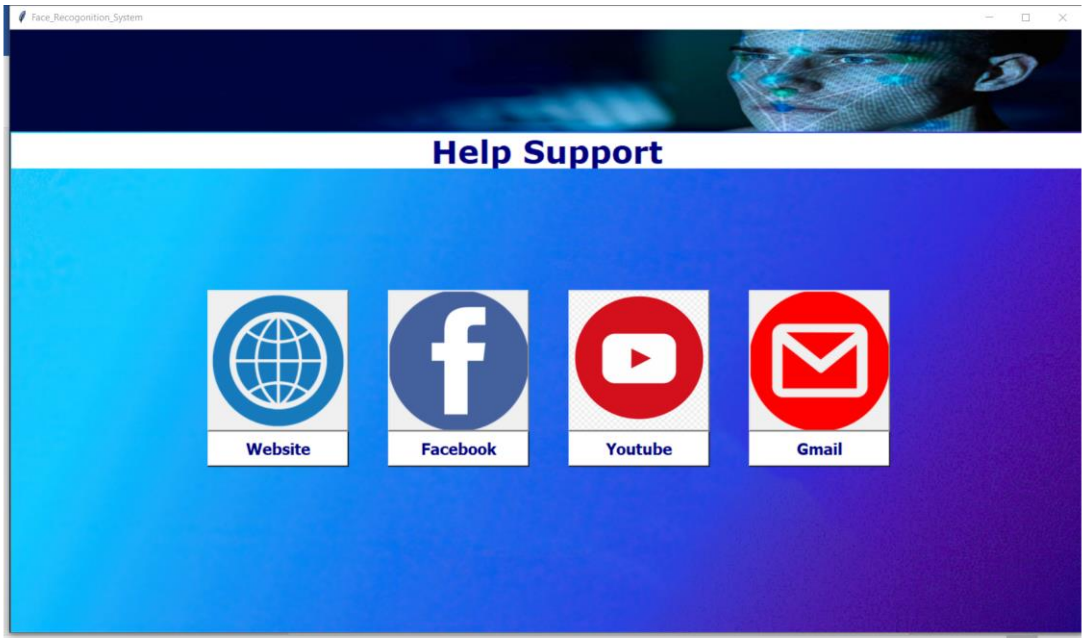


```
SELECT * FROM face_recognition.regteach;
```

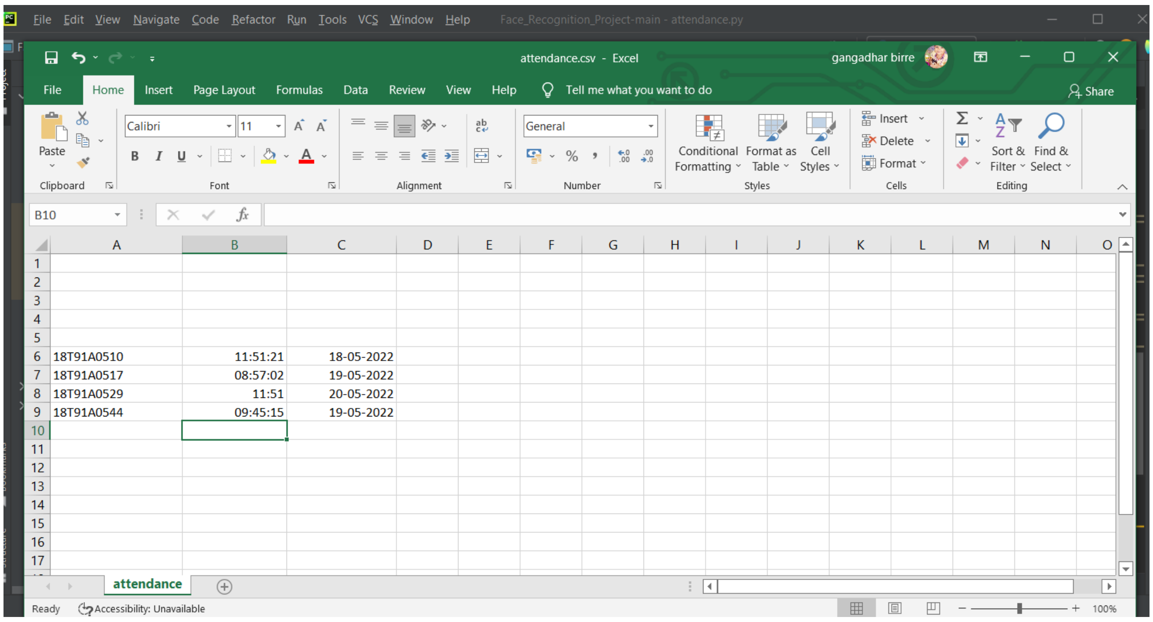
fname	lname	cnum	email	ssq	sa	pwd	c_pwd
gangadhar	birre	9666594932	gangadharbirre@gmail.com	Your Favorite Book	python	admin	admin
gangadhar	birre	9666594932	gangadharbirre@gmail.com	Your Favorite Book	python	admin	admin
sushmitha	g	9666594932	abc@gmail.com	Your Favorite Book	python	admin	admin
krishna	m	4932	xyz@gmail.com	Your Favorite Book	pytn	admin1	admin1
neelesh	s	966932	pqr@gmail.com	Your Favorite Book	pyton	admin1	admin1
neelesh	s	966932	pqr@gmail.com	Your Favorite Book	pyton	admin1	admin1
neelesh	s	966932	pqr@gmail.com	Your Favorite Book	pyton	admin1	admin1
neelesh	s	966932	pqr@gmail.com	Your Favorite Book	pyton	admin1	admin1
neelesh	s	966932	pqr@gmail.com	Your Favorite Book	pyton	admin1	admin1
neelesh	s	966932	pqr@gmail.com	Your Favorite Book	pyton	admin1	admin1
neelesh	s	966932	pqr@gmail.com	Your Favorite Book	pyton	admin1	admin1
neelesh	s	966932	pqr@gmail.com	Your Favorite Book	pyton	admin1	admin1
neelesh	s	966932	pqr@gmail.com	Your Favorite Book	pyton	admin1	admin1
neelesh	s	966932	pqr@gmail.com	Your Favorite Book	pyton	admin1	admin1
neelesh	s	966932	pqr@gmail.com	Your Favorite Book	pyton	admin1	admin1
gang	birre	112366	mlnv@gmail.com	Your Favorite Book	python	admin	admin
krishna	teja	123456	mlnv123@gmail.com	Your Favorite Book	java	teja	teja

#	Time	Action	Message	Duration / Fetch
1	14:08:27	SELECT * FROM face_recognition.reg LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
2	14:08:31	SELECT * FROM face_recognition.regteach LIMIT 0, 1000	21 row(s) returned	0.015 sec / 0.000 sec
3	14:09:00	SELECT * FROM face_recognition.student LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec

6.3 (h)-MYSQL Workbench



6.3 (i)-Help & Support



	A	B	C
6	18T91A0510	11:51:21	18-05-2022
7	18T91A0517	08:57:02	19-05-2022
8	18T91A0529	11:51	20-05-2022
9	18T91A0544	09:45:15	19-05-2022

6.3 (j)-Excel sheet For Attendance

CONCLUSION

In this approach, a face recognition based automated student attendance system is thoroughly described. The proposed approach provides a method to identify the individuals by comparing their input image obtained from recording video frame with respect to train image. This proposed approach able to detect and localize face from an input facial image, which is obtained from the recording video frame. Besides, it provides a method in pre-processing stage to enhance the image contrast and reduce the illumination effect.

The Attendance Management System is developed using Machine Learning meets the objectives of the system which it has been developed. The system has reached a steady state where all bugs have been eliminated. The system is operated at a high level of efficiency. The system solves the problem. It was intended to solve as requirement specification.

The system can recognize and identify the face well with an accuracy of 85 %, at a face distance 40 cm from the camera with adequate lighting.

REFERENCES:

1. https://www.researchgate.net/publication/326261079_Face_detection_system_for_attendance_of_class_students

2. Hapani, Smit, et al. "Automated Attendance System Using Image Processing." 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA). IEEE, 2018.

3. Akbar, Md Sajid, et al. "Face Recognition and RFID Verified Attendance System." 2018 International Conference on Computing, Electronics & Communications Engineering (iCCECE). IEEE, 2018.

4. Okokpujie, Kennedy O., et al. "Design and implementation of a student attendance system using iris biometric recognition." 2017 International Conference on Computational Science and Computational Intelligence (CSCI). IEEE, 2017.

5. Rathod, Hemantkumar, et al. "Automated attendance system using machine learning approach." 2017 International Conference on Nascent Technologies in Engineering (ICNTE). IEEE, 2017.

6. Siswanto, Adrian Rhesa Septian, Anto Satriyo Nugroho, and Maulahikmah Galinium. "Implementation of face recognition algorithm for biometrics based time attendance system." 2014 International Conference on ICT For Smart Society (ICISS). IEEE, 2014.

7. Lukas, Samuel, et al. "Student attendance system in classroom using face recognition technique." 2016 International Conference on Information and Communication Technology Convergence (ICTC). IEEE, 2016.

8. https://becominghuman.ai/face-detection-using-opencv-with-haar-cascade-classifiers-941dbb25177

9. https://www.superdatascience.com/blogs/opencv-face-recognition

10. Salim, Omar Abdul Rhman, Rashidah Funke Olanrewaju, and Wasiu Adebayo Balogun. "Class attendance management system using face recognition." 2018 7th International Conference on Computer and Communication Engineering (ICCCE). IEEE, 2018.