A Project Report

on

ATTENDLY

Face Recognition Attendance System

Submitted in partial
fulfillment of the
requirement for the
award of the degree of

BTECH IN CSE

GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

Under The Supervision of

MR. RAJAKUMAR P.

Submitted By

SAMEER VERMA (21SCSE1011328)
RITIK KUMAR (21SCSE1011297)
GAURAV KUMAR (21SCSE1011532)

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA
INDIA

DEC, 2023

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA

We hereby certify that the work which is being presented in the project, entitled **"Attendly – Face Recognition Attendence System".** in partial fulfillment of the requirements for the award of the BTECH IN CSE submitted in the School of Computing Science and Engineering of Galgotias University.

Greater Noida, is an original work carried out during the period of September 2023 to December 2023, under the supervision of **MR. RAJAKUMAR P.**, Department of Computer Science and Engineering of School of Computing Science and Engineering, Galgotias University, Greater Noida.

The matter presented in the project has not been submitted by us for the award of any other degree of this or any other places.

SAMEER VERMA (21SCSE1011328)

RITIK KUMAR (21SCSE1011297)

GAURAV KUMAR (21SCSE1011532)

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

MR. RAJAKUMAR P.

CERTIFICATE

The Final Project Viva-Voce examination of SAMEER VERMA – (21SCSE1011328) has been held on 21_____/12/2022 and his/her work is recommended for the award of BTECH IN

CSE

Signature of Examiner(s) **Signature of Supervisor(s)**

Signature of Program Chair **Signature of Dean**

Date: JUNE, 2022

Place: Greater Noida

Abstract

The Face Recognition Attendance System aims to address several critical issues in traditional attendance tracking. Firstly, it eliminates the need for manual processes, reducing errors and saving valuable time for both educators and employers. This technology enhances security by ensuring that only authorized individuals can mark their attendance, thus preventing proxy attendance fraud. It also provides real-time data and analytics, allowing institutions and organizations to make informed decisions. Additionally, it minimizes the environmental footprint by reducing paper-based processes. Overall, this system enhances efficiency, security, and transparency, making attendance tracking more reliable and efficient for educational institutions and businesses. The Face Recognition Attendance System seeks to bridge critical gaps in the existing attendance tracking methods. Traditional methods, such as manual roll calls or card swiping systems, are prone to errors and proxy attendance, leading to inaccuracies. Our system leverages advanced facial recognition technology to provide a robust, accurate, and automated solution, thereby eliminating these gaps. It ensures real-time data accuracy, improves security by uniquely identifying individuals, and enhances administrative efficiency. Moreover, it offers data analytics and insights, enabling institutions and organizations to make data-driven decisions, which traditional methods lack. This innovation aims to modernize and optimize attendance management, revolutionizing the way we track attendance. Using Facial attendance System Teachers and students Time save.

Table of Contents

Title		Page No.
Candidates Declaration		**I**
Acknowledgement		**II**
Abstract		**III**
Contents		**IV**
List of Images		**V**
Acronyms		**VI**
Chapter 1	**Introduction**	
	1.1 Introduction	**8**
	1.2 Tool and Technology Used	**10**
Chapter 2	**Literature Survey/Project Design**	**12**
Chapter 3	**Functionality/Working of Project and CODE**	**15**
	3.1. Main Code	15
	3.2.Student Management Code	17
	3.3. Login Code	20
	3.4. Show Data code	22
	3.5. Delete Data Code	22
Chapter 4	**Results**	**23**
Chapter 5	**Conclusion and Future Scope**	**25**
	Reference	**26**

List of Figures/Images

S.No.	Title	Page No.
1	Attendly Logo	14
2	Attendly FlowChart	14
3	Attendly Main Window	23
4	Attendly Student Management window	23
5	Attendly Login Window	24

Acronyms

MUI	Multilingual User Interface
JSX	**JavaScript XML**
JS	JavaScript
GUI	Graphical user interface
ID	Identification

CHAPTER-1

1.1. Introduction

Attendance is prime important for both the teacher and student of an educational organization. So it is very important to keep record of the attendance. The problem arises when we think about the traditional process of taking attendance in class room.

Calling name or roll number of the student for attendance is not only a problem of time consumption but also it needs energy. So an automatic attendance system can solve all above problems.

There are some automatic attendances making system which are currently used by much institution. One of such system is biometric technique and RFID system. Although it is automatic and a step ahead of traditional method it fails to meet the time constraint. The student has to wait in queue for giving attendance, which is time taking.

This project introduces an involuntary attendance marking system, devoid of any kind of interference with the normal teaching procedure. The system can be also implemented during exam sessions or in other teaching activities where attendance is highly essential. This system eliminates classical student identification such as calling name of the student, or checking respective identification cards of the student, which can not only interfere with the ongoing teaching process, but also can be stressful for students during examination sessions. In addition, the students have to register in the database to be recognized. The enrolment can be done on the spot through the user friendly interface.

The human face is a sophisticated multidimensional structure that can convey a lot of information about the individual, including expression, feeling, facial features. Effectively and efficiently analyzing the features related to facial information is a challenging task that requires lot of time and efforts. Recently, many facial recognition-based algorithms for automatic attendance management has been proposed, successfully implemented and used as in Refs. [1–4] and also new algorithms developed or some existing algorithms improved or combined with other methods, techniques, or algorithms to build facial recognition systems or applications as in Refs. [5–8].

1.1.1. The project features are as follows:

- College/School: Accurate identification of students for attendance tracking.
- Security: Enhanced access control by recognizing authorized personnel.
- Office: Secure entry by identifying employees for attendance and security purposes.

- Facial recognition restricts access to hospital medicine storage, permitting only authorized medical staff entry for securing sensitive supplies.

- At a concert venue, facial recognition is used to facilitate swift entry for ticket holders, reducing queues and enhancing crowd management.

- In an airport, boarding gates use facial recognition to verify passengers' identities, expediting the boarding process and enhancing security measures

1.1.2. Formulation of Problem

1.1.2.1 Merits:

- Easy to Use
- Contact-Less Attenance
- User friendly interface
- Security and Data privacy.

1.1.2.2. Demerits:

- Need Proper Light to Detect the Face More Accurately.
- Costly as required a Good Camera Setup.

1.2. Tool and Technology Used

1.2.1. Python:

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its advanced document creation process combined with dynamic typing and dynamic binding makes it very attractive for rapid application development and text or word generation to link existing products. Python's simple and easy-to-learn syntax emphasizes readability, reducing maintenance cost. Python supports modules and packages that support program modularization and code reuse. The Python interpreter and public libraries are freely available in source or binary form for all major platforms and can be freely distributed. [9]

Python uses OpenCV and dlib libraries using the LBPH (native binary mode histogram) algorithm for the real face. Using pad for graphics and Tkinter and ttk for GUI, developers can create facial recognition using algorithms, graphics, and user-friendly interface to improve performance and user experience.

1.2.2. Tkinter

Tkinter **Tutorial** provides **a** basic and advanced **overview** of Python Tkinter. Our Tkinter **tu torials are** designed for beginners and **experts.**

Python provides the **Tkinter** standard library for creating user interface **graphics** for desktop **a pplication.**

Developing desktop applications **using Python** Tkinter is not a **difficult** task. **You can use the following steps to create an** empty Tkinter top-level **window.**

- **Import** Tkinter **modules.**
- **Create a** main application **window.**
- **Add new** labels, buttons, **frames and other widgets.** to the **window.**
- **Calls** the main event loop **to be displayed** on the user's computer screen.

1.2.3. PILLOW

The Python Visualization Library (an extension of PIL) is the de facto visualization package for the Python language. It has lightweight photo manipulation tools that help you edit, create and save photos. Support for the Python graphics library was discontinued in 2011, but a project called Pillow forked the original PIL project and added Python 3.x support to it. It was announced that the pillow would replace the PIL for future use. The pad supports many image file formats, including BMP, PNG, JPEG and TIFF. The support library provides support for new books in the library by creating new cutouts.

CHAPTER-2
Literature Survey

In our Attendly, it is divided into two main sections The first section concentrated substantially on perfecting the face recognition algorithm while the alternate section concentrated on the attendance operation system grounded on the honored mortal faces. In the first section, a digital live camera will be used at the entrance to capture images of Pupil/ staffs entering an office Classroom or a structure, which some advanced image processing ways, similar as discrepancy adaptation, noise reduction using bilateral sludge, image histogram equalization, are applied to the captured images to ameliorate their quality, also the Haar Algorithm will be applied to the captured images to descry individual faces, which will be used as an input to the Face Recognition System. And also the same advanced image processing ways over, plus Image Blending fashion will be applied, a previous, to the training/ tem plate face images, also the bettered input images will be compared with the advanced training images using the LBP algorithm, to yield an advanced LBP canons to fete faces, therefore the facial recognition delicacy will be bettered compared to the traditional LBP canons without our system. In the alternate section, the metadata of the honored facial images similar as date and time are automatically uprooted to automatically mark attendance of each existent.

RELATED WORK

Kohonen Ref.[12] is one of the early pioneers of the most famous face recognition system, which employed a simple neural net using network of Eigen faces by approaching eigenvectors through face images bus correlation matrix. Although, the system wasn't veritably successful to be virtually enforced in a real- life terrain due to associated high demand for normalization and positioning when run in a large database with numerous types of face conditions.

In employing and improving the work of Kohonen, Kirby and Sirovich in 1990 as in Ref.[13], directly calculated the Eigen faces using algebraic manipulation with smaller than 100 faces to apply facial recognition, which was further bettered by Turk and Pentland in1991 as in Ref. [14] by determining the exact position and scales of faces and also the use of rendering residual error began from Eigen faces, but in a minimally constrained terrain. More and more new approaches than Kohonen approach for facial recognition using; star element Analysis(PCA), Fisher faces and the traditional Original double Patterns(LBP) were proposed, particularly the LBP, because it has a simple proposition with computational simplicity, steady with respect to any monotonic metamorphosis of argentine scale, has important gyration-steady analysis with a invariant pattern and dis criminates excellently between different colorful kinds of texture as in Ref.[15], but It's known that the LBP isn't as robust as the viola- jones and other algorithms for face discovery as stressed in Refs.[16-18], because of issues similar as noise, illumination variation,

background, disguise, scale and occlusion etc.

In addressing the issue of illumination variation, Ref.[19] eased illumination variation in facial recognition by combining the strengths of robust illumination normalization, original texture- grounded face representations, distance transfigure grounded matching and kernel- grounded point birth and multiple point emulsion, but the result addressed only illumination issues. To address issues of noise in facial recognition, Ref.[20] employed shearlets and LBP for dealing with heavy noise in face recognition, by taking advantage of robust features and edge discovery capabilities of shearlets in the presence of high position of noise. In this system, each face is divided into blocks, individual classifier is used for each block and also combine the similarity scores from all the blocks for better performance, but the result is limited to noise only.

Lately, different styles, ways and algorithms were com bined with either the traditional LBP or modified LBP to achieve facial recognition and enhance facial recognition delicacy. In Ref.[21] a Real-Time Multiple Face Recognition using Deep literacy on Bedded GPU System was proposed and the system used face discovery grounded on convolutional neural network(CNN) with face shadowing and state of the art deep CNN face recognition algorithm.

In addition, in Ref.[22] a Original Binary Pattern Histogram LBPH)- grounded Enhanced Real- Time Face Recognition was used to achieve real- time face recognition in low and high- position images and Ref.[23] proposes a system of perfecting the Recognition of Faces using LBP and SVM Optimized by PSO fashion, in this system, two point extraction algorithms videlicet star element Analysis(PCA) and Original double Pattern(LBP) ways are used to prize features from images. In the recognition process, it uses Support Vector Machine(SVM) for bracket combined with flyspeck mass Optimization.

In another approach in Ref.[24], facial recognition was achieved using Modified Original Binary Pattern and Random Forest, which the sign and magnitude features are combined for the enhancement of facial texture bracket performance and when compared with the traditional LBP for multiple patch variations on a grueling facial dataset, this system proven to be more accurate.

All these ways, styles, algorithms reviewed doesn't wholly addresses issues affecting facial recognition delicacy similar as illumination variation, noise, scale, sharp, pose in one shot, while our method was suitable to do that and also our system focuses on enhancing features of input and training images, therefore bettered LBP canons and achieved better recognition results.

Why we choose “Attendly” name for Our Face Attendance System?

We Used this name because we are try to built a application which is suitable for the Both student(College/school) and Employee(Office).

Our Application is a Contact free application means no one has to touch and gives any type of fingerprints at all.

Figure 1. Attendly logo

FlowChart Diagram

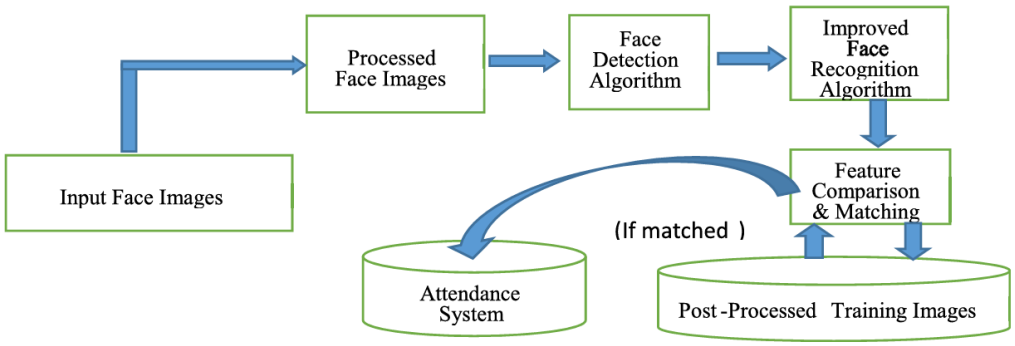


Figure 2. Flowchart diagram

Chapter 3

Functionality/Working of Project and CODE

3.1. Main Code

```
from tkinter import*
from tkinter import ttk
from PIL import Image,ImageTk



class Face_Recognition_System:
    def __init__(self,root):
        self.root=root
        self.root.geometry("1530x790+0+0")
        self.root.title("Attendly- Face Recognition Attendance System")

        # face image 1
        img1=Image.open(r"E:\FaceAttendance System\college_images\Face_attendance 01.jpg")
        img1=img1.resize((500,130),Image.Resampling.LANCZOS)
        self.photoimg1=ImageTk.PhotoImage(img1)

        f_lbl=Label(self.root,image=self.photoimg1)
        f_lbl.place(x=0,y=0,width=500,height=130)

        # face image 2
        # img2=Image.open(r"E:\FaceAttendance System\college_images\Logo.jpeg")
        img2=Image.open(r"E:\FaceAttendance System\college_images\sachin.png")
        img2=img2.resize((500,130),Image.Resampling.LANCZOS)
        self.photoimg2=ImageTk.PhotoImage(img2)

        f_lbl=Label(self.root,image=self.photoimg2)
        f_lbl.place(x=500,y=0,width=520,height=130)

        # face image 3
        img3=Image.open(r"E:\FaceAttendance System\college_images\Face_attendance 01.jpg")
        img3=img3.resize((500,130),Image.Resampling.LANCZOS)
        self.photoimg3=ImageTk.PhotoImage(img3)

        f_lbl=Label(self.root,image=self.photoimg3)
        f_lbl.place(x=1025,y=0,width=520,height=130)

        # background image
        img4=Image.open(r"E:\FaceAttendance System\college_images\bg1.jpg")
        img4=img4.resize((1530,710),Image.Resampling.LANCZOS)
        self.photoimg4=ImageTk.PhotoImage(img4)

        bg_img=Label(self.root,image=self.photoimg4)
        bg_img.place(x=0,y=130,width=1530,height=710)

        # title_lbl=Label(bg_img,text="Attendly- Face Recognition Attendance System",font=("times new roman",35,"bold"),bg="white",fg="black")
        title_lbl=Label(bg_img,text="AttendFace - Automatic Face recogniation Based Attendance",font=("times new roman",35,"bold"),bg="white",fg="black")

        title_lbl.place(x=0,y=0,width=1530,height=55)

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

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

    # Students Details
        img10=Image.open(r"E:\FaceAttendance System\college_images\details.gif")
        img10=img10.resize((220,220),Image.Resampling.LANCZOS)
        self.photoimg10=ImageTk.PhotoImage(img10)


        b1=Label(bg_img,image=self.photoimg10,cursor="hand2")
        b1.place(x=250,y=100,width=220,height=220)

        b1_1=Button(bg_img,text="Student Details",cursor="hand2",font=("times new roman",16,"bold"),bg="darkblue",fg="white")
        b1_1.place(x=250,y=320,width=220,height=30)
# ----------------------------------------------------------------------------------------------
    #
    # Face Detector
        img11=Image.open(r"E:\FaceAttendance System\college_images\FaceDEtector.jpg")
        img11=img11.resize((220,220),Image.Resampling.LANCZOS)
        self.photoimg11=ImageTk.PhotoImage(img11)


        b1=Label(bg_img,image=self.photoimg11,cursor="hand2")
        b1.place(x=550,y=100,width=220,height=220)

        b1_1=Button(bg_img,text="Face Detector",cursor="hand2",font=("times new roman",16,"bold"),bg="darkblue",fg="white")
        b1_1.place(x=550,y=320,width=220,height=30)


# ----------------------------------------------------------------------------------
# Attendance
```

```
        img12=Image.open(r"E:\FaceAttendance System\college_images\attend.jpg")
        img12=img12.resize((220,220),Image.Resampling.LANCZOS)
        self.photoimg12=ImageTk.PhotoImage(img12)


        b1=Label(bg_img,image=self.photoimg12,cursor="hand2")
        b1.place(x=850,y=100,width=220,height=220)

        b1_1=Button(bg_img,text="Attendance",cursor="hand2",font=("times new roman",16,"bold"),bg="darkblue",fg="white")
        b1_1.place(x=850,y=320,width=220,height=30)

# ------------------------------------------------------------------------------------
# Help Desk
        img13=Image.open(r"E:\FaceAttendance System\college_images\Help.jpg")
        img13=img13.resize((220,220),Image.Resampling.LANCZOS)
        self.photoimg13=ImageTk.PhotoImage(img13)

        b1=Label(bg_img,image=self.photoimg13,cursor="hand2")
        b1.place(x=1150,y=100,width=220,height=220)

        b1_1=Button(bg_img,text="Help Desk",cursor="hand2",font=("times new roman",16,"bold"),bg="darkblue",fg="white")
        b1_1.place(x=1150,y=320,width=220,height=30)



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

    # Train Data
        img14=Image.open(r"E:\FaceAttendance System\college_images\traindata.jpg")
        img14=img14.resize((220,220),Image.Resampling.LANCZOS)
        self.photoimg14=ImageTk.PhotoImage(img14)


        b1=Label(bg_img,image=self.photoimg14,cursor="hand2")
        b1.place(x=250,y=380,width=220,height=220)

        b1_1=Button(bg_img,text="Train Data",cursor="hand2",font=("times new roman",16,"bold"),bg="darkblue",fg="white")
        b1_1.place(x=250,y=580,width=220,height=30)

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

    # Photos
        img15=Image.open(r"E:\FaceAttendance System\college_images\photos.jpg")
        img15=img15.resize((220,220),Image.Resampling.LANCZOS)
        self.photoimg15=ImageTk.PhotoImage(img15)


        b1=Label(bg_img,image=self.photoimg15,cursor="hand2")
        b1.place(x=550,y=380,width=220,height=220)

        b1_1=Button(bg_img,text="Photos",cursor="hand2",font=("times new roman",16,"bold"),bg="darkblue",fg="white")
        b1_1.place(x=550,y=580,width=220,height=30)

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

    # Developer
        img16=Image.open(r"E:\FaceAttendance System\college_images\Developer.jpg")
        img16=img16.resize((220,220),Image.Resampling.LANCZOS)
        self.photoimg16=ImageTk.PhotoImage(img16)


        b1=Label(bg_img,image=self.photoimg16,cursor="hand2")
        b1.place(x=850,y=380,width=220,height=220)

        b1_1=Button(bg_img,text="Developer",cursor="hand2",font=("times new roman",16,"bold"),bg="darkblue",fg="white")
        b1_1.place(x=850,y=580,width=220,height=30)

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

    # Exitn
        img17=Image.open(r"E:\FaceAttendance System\college_images\exit.jpg")
        img17=img17.resize((220,220),Image.Resampling.LANCZOS)
        self.photoimg17=ImageTk.PhotoImage(img17)


        b1=Label(bg_img,image=self.photoimg17,cursor="hand2")
        b1.place(x=1150,y=380,width=220,height=220)

        b1_1=Button(bg_img,text="Exit",cursor="hand2",font=("times new roman",16,"bold"),bg="darkblue",fg="white")
        b1_1.place(x=1150,y=580,width=220,height=30)




if __name__ == "__main__":
    root=Tk()
    obj=Face_Recognition_System(root)  #esme Root bracket ke andar dalna tha lekin kaam nahi kar raha hai root see  no 1 24:21
    root.mainloop()
    # ms.after(0, update, 0)
    # ms.mainloop()
```

3.2. Student Code

```
from tkinter import*
from tkinter import ttk
from PIL import Image,ImageTk




class student:
    def __init__(self,root):
        self.root=root
        self.root.geometry("1530x790+0+0")
        self.root.title("Attendly- Face Recognition Attendance System")

        # face image 1
        img1=Image.open(r"E:\FaceAttendance System\college_images\student.jpg")
        img1=img1.resize((500,130),Image.Resampling.LANCZOS)
        self.photoimg1=ImageTk.PhotoImage(img1)

        f_lbl=Label(self.root,image=self.photoimg1)
        f_lbl.place(x=0,y=0,width=500,height=130)

        # face image 2
        img2=Image.open(r"E:\FaceAttendance System\college_images\Attendly2.0.jpg")
        img2=img2.resize((500,130),Image.Resampling.LANCZOS)
        self.photoimg2=ImageTk.PhotoImage(img2)

        f_lbl=Label(self.root,image=self.photoimg2)
        f_lbl.place(x=500,y=0,width=520,height=130)

        # face image 3
        img3=Image.open(r"E:\FaceAttendance System\college_images\Face_attendance 01.jpg")
        img3=img3.resize((500,130),Image.Resampling.LANCZOS)
        self.photoimg3=ImageTk.PhotoImage(img3)

        f_lbl=Label(self.root,image=self.photoimg3)
        f_lbl.place(x=1025,y=0,width=520,height=130)

        # background image
        img4=Image.open(r"E:\FaceAttendance System\college_images\bg1.jpg")
        img4=img4.resize((1530,710),Image.Resampling.LANCZOS)
        self.photoimg4=ImageTk.PhotoImage(img4)

        bg_img=Label(self.root,image=self.photoimg4)
        bg_img.place(x=0,y=130,width=1530,height=710)

        title_lbl=Label(bg_img,text="Student Mangagement System",font=("times new roman",35,"bold"),bg="black",fg="White")
        title_lbl.place(x=0,y=0,width=1530,height=55)

        main_frame=Frame(bg_img,bd=2, bg="white")
        main_frame.place(x=10,y=65,width=1500,height=600)

 # --*-*-*-**-*-*-*-*-*-**--*-**-**-**-*-**----------------------------------------------------------------------------
 # --*-*-*-**-*-*-*-*-*-**--*-**-**-**-*-**----------------------------------------------------------------------------
 # --*-*-*-**-*-*-*-*-*-**--*-**-**-**-*-**----------------------------------------------------------------------------

        # left Label frame

        Left_frame=LabelFrame(main_frame,bd=2, bg="white",relief=RIDGE, text="Student Details",font=("times new roman",12,"bold"))
        Left_frame.place(x=30,y=10,width=700,height=580)

        img_left1=Image.open(r"E:\FaceAttendance System\college_images\Face_attendance 01.jpg")
        img_left1=img_left1.resize((680,130),Image.Resampling.LANCZOS)
        self.photoimg_left1=ImageTk.PhotoImage(img_left1)

        f_lbl=Label(Left_frame,image=self.photoimg_left1)
        f_lbl.place(x=5,y=0,width=680,height=130)

# -------Current Course-------------------------------------------------------------------
        CC_frame=LabelFrame(Left_frame,bd=2, bg="white",relief=RIDGE, text="Current Course Details",font=("times new roman",12,"bold"))
        CC_frame.place(x=5,y=135,width=680,height=120)

        # Department
        dep_label=Label(CC_frame,text="Department",font=("times new roman",12,"bold"),bg="white")
        dep_label.grid(row=0,column=0,padx=10,sticky=W)

        dep_combo=ttk.Combobox(CC_frame,font=("times new roman",12,"bold"),state="read only",width=20)
        dep_combo["values"]=("Select Department","Computer Science","IT","BBA","LLB","Agriculture","Civil")
        dep_combo.current(0)
        dep_combo.grid(row=0,column=1,padx=5,pady=10,sticky=W)

        # Course

        course_label=Label(CC_frame,text="Course",font=("times new roman",12,"bold"),bg="white")
        course_label.grid(row=0,column=2,padx=10,sticky=W)

        course_combo=ttk.Combobox(CC_frame,font=("times new roman",12,"bold"),state="read only",width=20)
        course_combo["values"]=("Select Course","BE","TE","FE","SE","Agriculture","Civil")
        course_combo.current(0)
        course_combo.grid(row=0,column=3,padx=5,pady=10,sticky=W)

        # Year

        year_label=Label(CC_frame,text="Year",font=("times new roman",12,"bold"),bg="white")
        year_label.grid(row=1,column=0,padx=10,sticky=W)
```

```
        year_combo=ttk.Combobox(CC_frame,font=("times new roman",12,"bold"),state="read only",width=20)
        year_combo["values"]=("Select Year","2023-2024","2022-2023","2021-2022","2020-2021","2019-2020","2018-2019")
        year_combo.current(0)
        year_combo.grid(row=1,column=1,padx=5,pady=10,sticky=W)

        # semester

        semester_label=Label(CC_frame,text="Semester",font=("times new roman",12,"bold"),bg="white")
        semester_label.grid(row=1,column=2,padx=10,sticky=W)

        semester_combo=ttk.Combobox(CC_frame,font=("times new roman",12,"bold"),state="read only",width=20)
        semester_combo["values"]=("Select Semester","I","II","III","IV","V","VI","VII","VIII")
        semester_combo.current(0)
        semester_combo.grid(row=1,column=3,padx=5,pady=10,sticky=W)

# ------Class Student Information (CS)----------------------------------------------------------------------------

        CS_frame=LabelFrame(Left_frame,bd=2, bg="white",relief=RIDGE, text="Class Student Information",font=("times new roman",12,"bold"))
        CS_frame.place(x=5,y=255,width=680,height=300)

        # Student Id
        studentid_label=Label(CS_frame,text="Student ID: ",font=("times new roman",12,"bold"),bg="white")
        studentid_label.grid(row=0,column=0,padx=10,pady=5,sticky=W)

        studentid_entry=ttk.Entry(CS_frame,width=20,font=("times new roman",12,"bold"))
        studentid_entry.grid(row=0,column=1,padx=10,pady=5,sticky=W)

        # Student Name

        studentName_label=Label(CS_frame,text="Student Name: ",font=("times new roman",12,"bold"),bg="white")
        studentName_label.grid(row=0,column=2,padx=10,pady=5,sticky=W)

        studentName_entry=ttk.Entry(CS_frame,width=20,font=("times new roman",12,"bold"))
        studentName_entry.grid(row=0,column=3,padx=10,pady=5,sticky=W)

        # Class Division

        division_label=Label(CS_frame,text="Class Division: ",font=("times new roman",12,"bold"),bg="white")
        division_label.grid(row=1,column=0,padx=10,pady=5,sticky=W)

        division_entry=ttk.Entry(CS_frame,width=20,font=("times new roman",12,"bold"))
        division_entry.grid(row=1,column=1,padx=10,pady=5,sticky=W)

        # Roll No or Enrollment no

        roll_label=Label(CS_frame,text="Roll No: ",font=("times new roman",12,"bold"),bg="white")
        roll_label.grid(row=1,column=2,padx=10,pady=5,sticky=W)

        roll_entry=ttk.Entry(CS_frame,width=20,font=("times new roman",12,"bold"))
        roll_entry.grid(row=1,column=3,padx=10,pady=5,sticky=W)

        # Gender

        gender_label=Label(CS_frame,text="Gender: ",font=("times new roman",12,"bold"),bg="white")
        gender_label.grid(row=2,column=0,padx=10,pady=5,sticky=W)

        gender_entry=ttk.Entry(CS_frame,width=20,font=("times new roman",12,"bold"))
        gender_entry.grid(row=2,column=1,padx=10,pady=5,sticky=W)

        # DOB

        DOB_label=Label(CS_frame,text="DOB: ",font=("times new roman",12,"bold"),bg="white")
        DOB_label.grid(row=2,column=2,padx=10,pady=5,sticky=W)

        DOB_entry=ttk.Entry(CS_frame,width=20,font=("times new roman",12,"bold"))
        DOB_entry.grid(row=2,column=3,padx=10,pady=5,sticky=W)

        #  Email

        email_label=Label(CS_frame,text="Email ID: ",font=("times new roman",12,"bold"),bg="white")
        email_label.grid(row=3,column=0,padx=10,pady=5,sticky=W)

        email_entry=ttk.Entry(CS_frame,width=20,font=("times new roman",12,"bold"))
        email_entry.grid(row=3,column=1,padx=10,pady=5,sticky=W)

        # Phone Numbers /contact details

        phone_label=Label(CS_frame,text="Phone No: ",font=("times new roman",12,"bold"),bg="white")
        phone_label.grid(row=3,column=2,padx=10,pady=5,sticky=W)

        phone_entry=ttk.Entry(CS_frame,width=20,font=("times new roman",12,"bold"))
        phone_entry.grid(row=3,column=3,padx=10,pady=5,sticky=W)


        # adress Details

        adress_label=Label(CS_frame,text="Address : ",font=("times new roman",12,"bold"),bg="white")
        adress_label.grid(row=4,column=0,padx=10,pady=5,sticky=W)

        adress_entry=ttk.Entry(CS_frame,width=20,font=("times new roman",12,"bold"))
        adress_entry.grid(row=4,column=1,padx=10,pady=5,sticky=W)

        # Teacher Name

        teacher_label=Label(CS_frame,text="Roll No: ",font=("times new roman",12,"bold"),bg="white")
        teacher_label.grid(row=4,column=2,padx=10,pady=5,sticky=W)

        teacher_entry=ttk.Entry(CS_frame,width=20,font=("times new roman",12,"bold"))
        teacher_entry.grid(row=4,column=3,padx=10,pady=5,sticky=W)

        # radio buttons
```

```
        radioBt1=ttk.Radiobutton(CS_frame,text="Take A Photos Sample",value="Yes")
        radioBt1.grid(row=6,column=0)

        radioBt2=ttk.Radiobutton(CS_frame,text="No Photos Sample",value="Yes")
        radioBt2.grid(row=6,column=1)

# ------- Buttons Frame ------------------------------------------------------------
        btn_frame=Frame(CS_frame,bd=2,relief=RIDGE,bg="white")
        btn_frame.place(x=5,y=200,width=650,height=35)

        save_btn=Button(btn_frame,text="Save",width=17,font=("times new roman",12,"bold"),bg="blue",fg="white")
        save_btn.grid(row=0,column=0)

        update_btn=Button(btn_frame,text="Update",width=17,font=("times new roman",12,"bold"),bg="blue",fg="white")
        update_btn.grid(row=0,column=1)

        delete_btn=Button(btn_frame,text="Delete",width=17,font=("times new roman",12,"bold"),bg="blue",fg="white")
        delete_btn.grid(row=0,column=2)

        reset_btn=Button(btn_frame,text="Reset",width=17,font=("times new roman",12,"bold"),bg="blue",fg="white")
        reset_btn.grid(row=0,column=3)

# -------update and take photos buttoms

        btn_frame1=Frame(CS_frame,bd=2,relief=RIDGE,bg="white")
        btn_frame1.place(x=5,y=240,width=650,height=35)

        TakePhotos_btn=Button(btn_frame1,text="Take Photo Sample",width=35,font=("times new roman",12,"bold"),bg="magenta",fg="black")
        TakePhotos_btn.grid(row=0,column=0)

        UpadtePhotos_btn=Button(btn_frame1,text="Upadte Photo Sample",width=35,font=("times new roman",12,"bold"),bg="magenta",fg="black")
        UpadtePhotos_btn.grid(row=0,column=1)




# --*-*-*-**-*-*-*-*-*-**--*-**-**-**-*-**-------------------------------------------------------------------------------
# --*-*-*-**-*-*-*-*-*-**--*-**-**-**-*-**-------------------------------------------------------------------------------
 # --*-*-*-**-*-*-*-*-*-**--*-**-**-**-*-**-------------------------------------------------------------------------------

        # Right Label frame

        Right_frame=LabelFrame(main_frame,bd=2, bg="white",relief=RIDGE, text="Student Details",font=("times new roman",12,"bold"))
        Right_frame.place(x=750,y=10,width=700,height=580)

        img_Right1=Image.open(r"E:\FaceAttendance System\college_images\Right.jpg")
        img_Right1=img_Right1.resize((680,130),Image.Resampling.LANCZOS)
        self.photoimg_Right1=ImageTk.PhotoImage(img_Right1)

        f_lbl=Label(Right_frame,image=self.photoimg_Right1)
        f_lbl.place(x=5,y=0,width=680,height=130)

        # ========== Search system ============================================================================

        search_frame=LabelFrame(Right_frame,bd=2, bg="white",relief=RIDGE, text="Search System",font=("times new roman",12,"bold"))
        search_frame.place(x=5,y=135,width=680,height=70)

        search_label=Label(search_frame,text="Search By",font=("times new roman",12,"bold"),bg="white")
        search_label.grid(row=0,column=0,padx=10,pady=5,sticky=W)

        search_combo=ttk.Combobox(search_frame,font=("times new roman",12,"bold"),state="read only",width=15)
        search_combo["values"]=("Select ","Roll No","Admission no","Phone no")
        search_combo.current(0)
        search_combo.grid(row=0,column=1,padx=5,pady=10,sticky=W)

        phone_entry=ttk.Entry(search_frame,width=15,font=("times new roman",12,"bold"))
        phone_entry.grid(row=0,column=2,padx=10,pady=5,sticky=W)

        search_btn=Button(search_frame,text="Search",width=12,font=("times new roman",12,"bold"),bg="blue",fg="white")
        search_btn.grid(row=0,column=3,padx=4)

        showall_btn=Button(search_frame,text="Showall",width=12,font=("times new roman",12,"bold"),bg="blue",fg="white")
        showall_btn.grid(row=0,column=4,padx=4)

# ======== table frame ========================
        table_frame=Frame(Right_frame,bd=2, bg="white",relief=RIDGE)
        table_frame.place(x=5,y=210,width=680,height=250)

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


self.student_table=ttk.Treeview(table_frame,column=("dep","course","year","sem","id","name","div","dob","email","phone","address","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("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("id",text="Student ID")
        self.student_table.heading("name",text="Name")
        self.student_table.heading("div",text="Division")
        self.student_table.heading("dob",text="DOB")
        self.student_table.heading("email",text="Email")
        self.student_table.heading("phone",text="Phone")
        self.student_table.heading("address",text="Address")
        self.student_table.heading("teacher",text="Teacher")
```

```
        self.student_table.heading("photo",text="PhotoSampleStatus")
        self.student_table["show"]="headings"

        self.student_table.pack(fill=BOTH,expand=1)
        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("id",width=100)
        self.student_table.column("name",width=100)
        self.student_table.column("div",width=100)
        self.student_table.column("dob",width=100)
        self.student_table.column("email",width=100)
        self.student_table.column("phone",width=100)
        self.student_table.column("address",width=100)
        self.student_table.column("teacher",width=100)
        self.student_table.column("photo",width=150)









if __name__ == "__main__":
    root=Tk()
    obj=student(root)  #esme Root bracket ke andar dalna tha lekin kaam nahi kar raha hai root see  no 1 24:21
    root.mainloop()
    # ms.after(0, update, 0)
    # ms.mainloop()
```

3.3. Login Code

```
import tkinter as tk
from PIL import ImageTk, Image
import sqlite3

# Database setup
conn = sqlite3.connect('user_database.db')
cursor = conn.cursor()

cursor.execute('''
    CREATE TABLE IF NOT EXISTS users (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        username TEXT UNIQUE,
        password TEXT,
        email TEXT
    )
''')

conn.commit()
conn.close()

class LoginPage(tk.Tk):
    def __init__(self):
        tk.Tk.__init__(self)
        self.title("Login Page")

        # Load and display the background image
        bg_image = Image.open(r"C:\Users\Ritik\OneDrive\Desktop\c++ DSA program\minor_project\photo.jpg")
        bg_image = bg_image.resize((1500, 1000), Image.ANTIALIAS)
        self.background_image = ImageTk.PhotoImage(bg_image)
        self.background_label = tk.Label(self, image=self.background_image)
        self.background_label.place(x=0, y=0, relwidth=1, relheight=1)

        # Create a frame to organize entry fields
        entry_frame = tk.Frame(self, bg="white")
        entry_frame.place(relx=0.5, rely=0.5, anchor=tk.CENTER)

        # Username
        self.username_label = tk.Label(entry_frame, text="Username:")
        self.username_label.grid(row=0, column=0, padx=5, pady=5)
        self.username_entry = tk.Entry(entry_frame)
        self.username_entry.grid(row=0, column=1, padx=5, pady=5)

        # Password
```

```
        self.password_label = tk.Label(entry_frame, text="Password:")
        self.password_label.grid(row=1, column=0, padx=5, pady=5)
        self.password_entry = tk.Entry(entry_frame, show="*")
        self.password_entry.grid(row=1, column=1, padx=5, pady=5)

        # Confirm Password
        self.confirm_password_label = tk.Label(entry_frame, text="Confirm Password:")
        self.confirm_password_label.grid(row=2, column=0, padx=5, pady=5)
        self.confirm_password_entry = tk.Entry(entry_frame, show="*")
        self.confirm_password_entry.grid(row=2, column=1, padx=5, pady=5)

        # Email
        self.email_label = tk.Label(entry_frame, text="Email:")
        self.email_label.grid(row=3, column=0, padx=5, pady=5)
        self.email_entry = tk.Entry(entry_frame)
        self.email_entry.grid(row=3, column=1, padx=5, pady=5)

        # Login, Signup, and Forgot Password buttons
        self.login_button = tk.Button(entry_frame, text="Login", command=self.login)
        self.login_button.grid(row=4, column=0, columnspan=2, padx=5, pady=5, sticky="we")

        self.signup_button = tk.Button(entry_frame, text="Sign Up", command=self.signup)
        self.signup_button.grid(row=5, column=0, columnspan=2, padx=5, pady=5, sticky="we")

        self.forgot_password_button = tk.Button(entry_frame, text="Forgot Password", command=self.forgot_password)
        self.forgot_password_button.grid(row=6, column=0, columnspan=2, padx=5, pady=5, sticky="we")

        self.status_label = tk.Label(entry_frame, text="", bg="white")
        self.status_label.grid(row=7, column=0, columnspan=2, padx=5, pady=5)

    def signup(self):
        username = self.username_entry.get()
        password = self.password_entry.get()
        confirm_password = self.confirm_password_entry.get()
        email = self.email_entry.get()

        if not all((username, password, confirm_password, email)):
            self.status_label.config(text="Please fill in all fields")
            return

        if password != confirm_password:
            self.status_label.config(text="Passwords do not match")
            return

            conn = sqlite3.connect('user_database.db')
            cursor = conn.cursor()

        try:
            cursor.execute("INSERT INTO users (username, password, email) VALUES (?, ?, ?)", (username, password, email))
            conn.commit()
            self.status_label.config(text="Signup successful")
        except sqlite3.IntegrityError:
            self.status_label.config(text="Username already exists")

        conn.close()
    # Existing signup method remains the same

        # Existing signup method remains the same

    def login(self):
        username = self.username_entry.get()
        password = self.password_entry.get()

        if not username or not password:
            self.status_label.config(text="Please enter both username and password")
            return

        conn = sqlite3.connect('user_database.db')
        cursor = conn.cursor()

        cursor.execute("SELECT * FROM users WHERE username=? AND password=?", (username, password))
        user = cursor.fetchone()

        if user:
            self.status_label.config(text=f"Welcome, {username}!")
            self.show_after_login_page()
        else:
            self.status_label.config(text="Invalid username or password")

            conn.close()
```

```
    # Existing login method remains the same

        # Existing login method remains the same

    def show_after_login_page(self):
        self.username_label.destroy()
        self.username_entry.destroy()
        self.password_label.destroy()
        self.password_entry.destroy()
        self.confirm_password_label.destroy()
        self.confirm_password_entry.destroy()
        self.email_label.destroy()
        self.email_entry.destroy()
        self.login_button.destroy()
        self.signup_button.destroy()
        self.status_label.destroy()

        # Create elements for the new page after login/signup
        after_login_frame = tk.Frame(self, bg="white")
        after_login_frame.place(relx=0.5, rely=0.5, anchor=tk.CENTER)

        label_after_login = tk.Label(after_login_frame, text="Welcome! You have successfully logged in.", font=("Arial", 20))
        label_after_login.pack()
        # Existing show_after_login_page method remains the same

    def forgot_password(self):
        username = self.username_entry.get()
        email = self.email_entry.get()

        conn = sqlite3.connect('user_database.db')
        cursor = conn.cursor()

        cursor.execute("SELECT * FROM users WHERE username=? AND email=?", (username, email))
        user = cursor.fetchone()

        if user:
            password = user[2]  # Fetch the password from the database
            self.status_label.config(text=f"Your password is: {password}")
        else:
            self.status_label.config(text="Invalid username or email")

        conn.close()

if __name__ == "__main__":
    app = LoginPage()
    app.mainloop()
```

3.4. Show Data

```
import sqlite3

conn = sqlite3.connect('user_database.db')
cursor = conn.cursor()

cursor.execute("SELECT * FROM users")
rows = cursor.fetchall()

for row in rows:
    print(row)

conn.close()
```

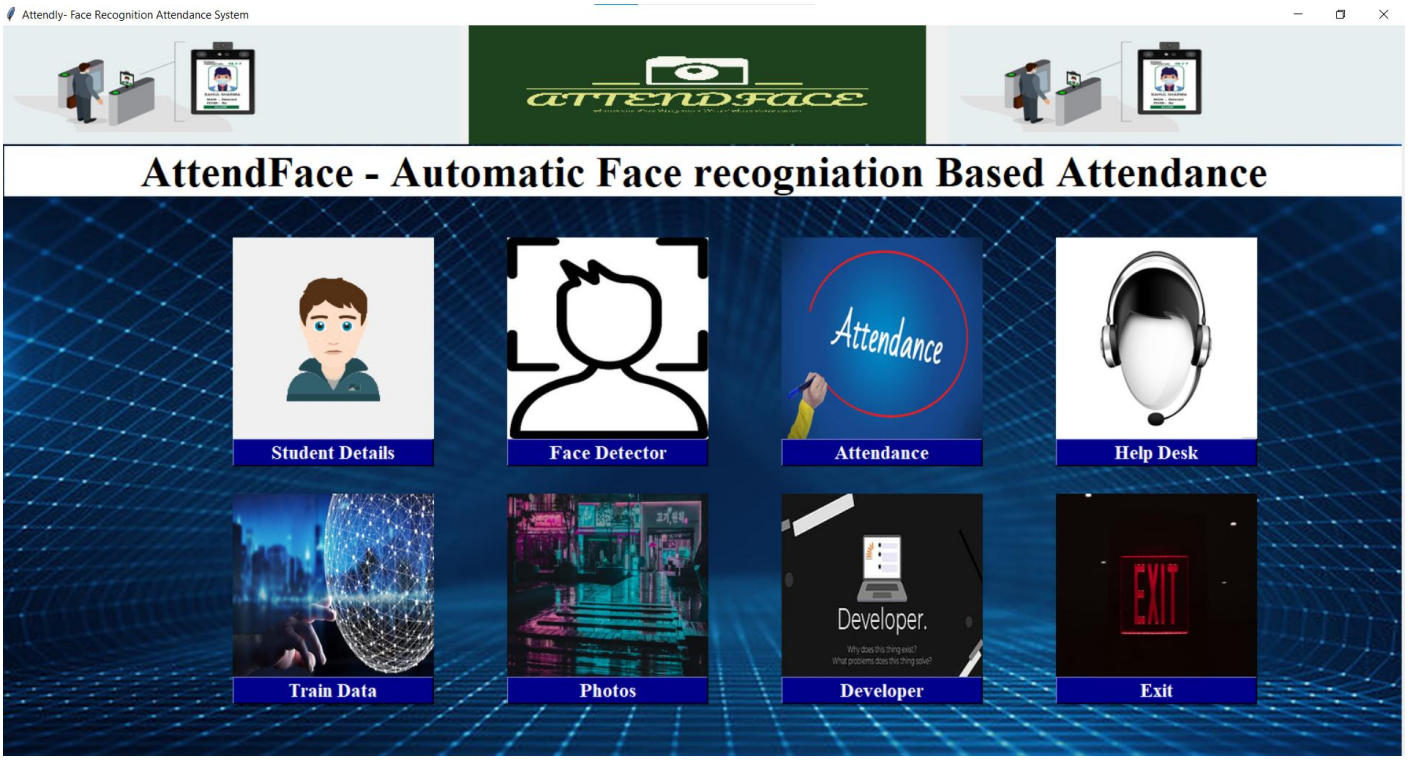
3.5. Delete Data

```
import sqlite3

conn = sqlite3.connect('user_database.db')
cursor = conn.cursor()

# Execute the DELETE SQL statement to clear all rows from the table
cursor.execute("DELETE FROM users")
conn.commit()

conn.close()
```

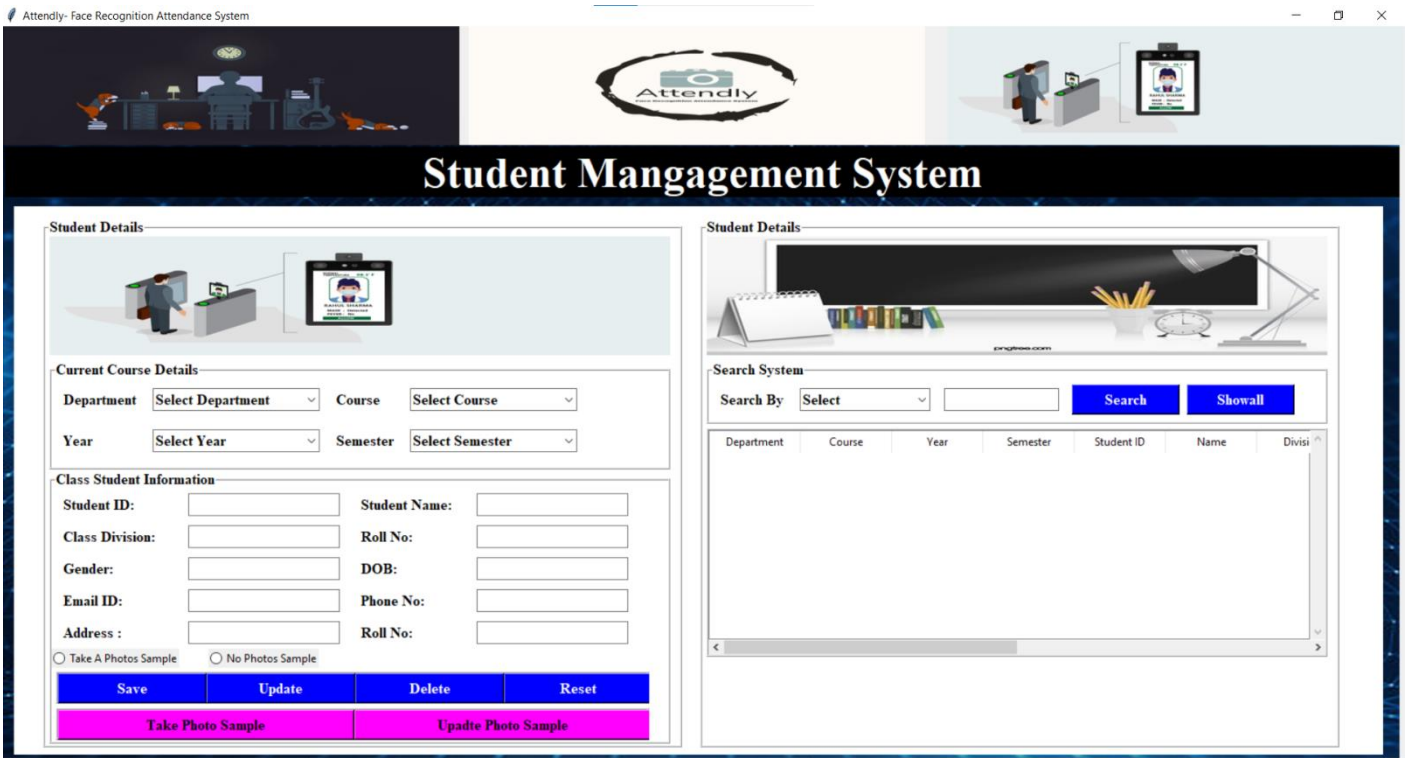
Chapter 4

Results

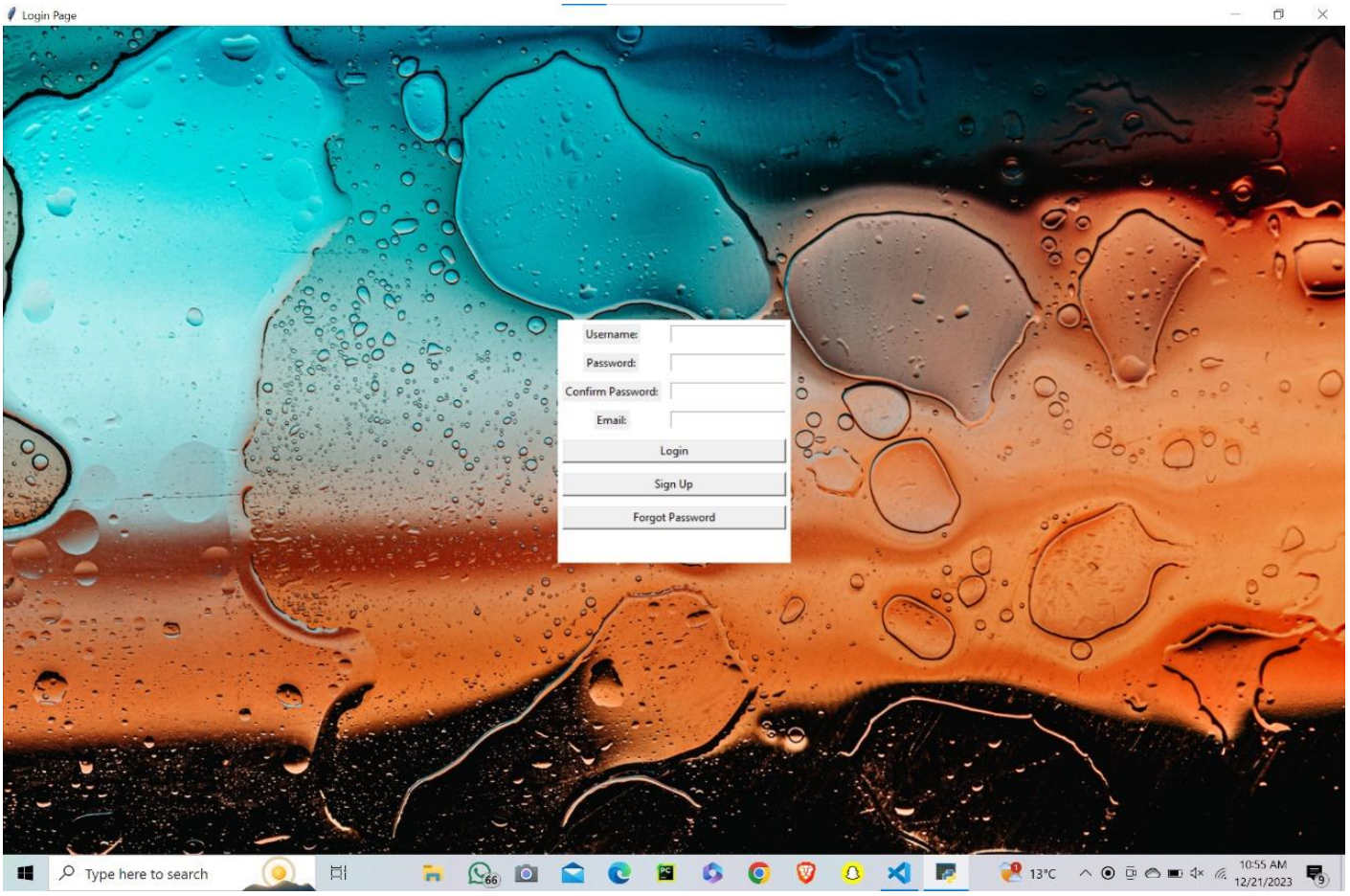
4.1. Main window



4.2. Attendly Student Management Window



4.3. Attendly Login Window



Chapter 5

Conclusion and Future Scope

Conclusion

- We have successful Completed the Front and Student mangament System.It is use to Store the Student Details And Photos.It is in working conditions. Also We Have Successfully Make Login and Sign-up. We have also Added Details View of Items. Also our site is more responsive.
- Here we have completed 60 % of our site.
- Our next goal before is to Combine all the modules and Test the all the modules

Future Scope

- We Will add more Features
- Users will be Able to get notification about their Attendance.
- We will make the Application more fast and accurate.

.

Reference

[1]. Jadhav Akshara, Jadhav Akshay, Ladhe Tushar, Yeolekar Krishna. Automated attendance system using face recognition. Inter Res J Eng Technol (IRJET) January 2017. Volume: 04 Issue: 01, ISSN: 2395-0072.

[2]. Punjani Ali Akbar, Obaid Chowdhary, Yasir Choudhary. Automated attendance management system using face recognition. Inter J Advan Res Computer Eng Technol (IJARCET) August 2017;6(8). ISSN 2278 – 7798.

[3]. Chaudhari Chetan, Raj Rahul, Shirnath Swajey, Sali Mrs Tanuja. Automatic attendance monitoring system using face recognition techniques. Inter J Innov Eng Technol (IJIET) April 2018;10(1). ISSN: 2319-1058.

[4]. Kowsalya P, Pavithra J, Sowmiya G, Shankar CK. Attendance monitoring system using face detection & face recognition. Inter Res J Eng Technol (IRJET) March 2019. Volume: 06 Issue: 03, ISSN: 2395-0072.

[5]. Mallikarjuna Reddy A, Venkata Krishna V, Sumalatha L. Face recognition based on Cross diagonal complete motif matrix. I.J. Image, Graphics and Signal Processing March 2018;3:59–66.

[6]. KAMENCAY Patrik, BENCO Miroslav, MIZDOS Tomas, RADIL Roman. A new method for face recognition using convolutional neural network. Digital Image Processing and Computer Graphics 2017;15(4):663–72.

[7]. Beli Idelette Laure Kambi, Guo Chunsheng. Enhancing face identification using local binary patterns and K-nearest neighbors. Journal of Imaging 2017;3(37):1–12.

[8]. Gaikwad Ashok T. LBP and PCA based on face recognition system. LBP and PCA based on face recognition system. November 2018. p. 368–73. ISSN 2348– 8034.

[9]. https://www.python.org/doc/essays/blurb

[10]. https://www.javatpoint.com/python-tkinter

[11]. https://www.geeksforgeeks.org/python-pillow-a-fork-of-pil

[12]. Kohonen T. Self-organization and associative memory. third ed. 1989. p. 185–209.

[13]. Kirby M, Sirovich L. Application of the karhunen-loeve procedure for the characterization of human faces. IEEE Pattern Analysis and Machine Intelligence 1990;12(1):103–8.

[14]. Turk M, Pentland A. Eigenfaces for recognition. J Cogn Neurosci 1991;3(1):71–86.

[15]. Huang Di, Shan Caifeng, Ardabilian Mohsen, Wang Yunhong, Chen Liming. Local binary patterns and its application to facial image analysis: a survey. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, Institute of Electrical and Electronics Engineers March 2011;4(41):1–17.

[16]. Chehrehgosha Arezou, Emadi Mehran. Face detection using fusion of LBP and AdaBoost. J Soft Comput Appl 2016;(1):1–10.

[17]. Chanchal Amit Kumar, Dutta Maitreyee. Face detection and recognition using local binary patterns. International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering October 2016;5(10):7923–9.

[18]. Bilaniuk Olexa, Fazl-Ersi Ehsan, Laganiere Robert, Xu Christina, Laroche Daniel, Moulder Craig. Fast LBP face detection on low-power SIMD architectures, computer vision and pattern recognition workshops (CVPRW). In: IEEE conference on; 2014. p. 616–22.

[19]. Kalaiselvi P, Nithya S. Face recognition system under varying lighting conditions. IOSR J Comput Eng October 2013;14(3):79–88. e-ISSN: 2278-0661, p- ISSN: 2278 8727.

[20]. Chen Jie, Patel Vishal M, Liu Li, Kellokumpu Vili, Zhao Guoying, Pietik€ ainen Matti, Chellappa Rama. Robust local features for remote face recognition. Image Vis Comput 2017;64:34–46.

[21]. Saypadith Savath, Aramvith Supavadee. Real-time multiple face recognition using deep learning on embedded GPU system. Asia-pacific signal and information processing association annual summit and conference (APSIPA ASC). IEEE Explore ISSN Information: Electronic ISSN: 2640-0103 Print on Demand (PoD); November 2018. p. 1318–24. ISSN: 2640-009X.

[22]. Deeba Farah, Memon Hira, Ali Dharejo Fayaz, Ahmed Aftab, Ghaffar Abddul. LBPH based enhanced real-time face recognition. Int J Adv Comput Sci Appl 2019;10(5). 2019.

[23]. Nisha Maitreyee Dutta. Improving the recognition of faces using LBP and SVM optimized by PSO technique. Int J Exp Diabetes Res 2017;5(4):297–303. ISSN: 2321-9939.

[24]. O'Connor Brian, Roy Kaushik. Facial recognition using modified local binary pattern and random forest. International Journal of Artificial Intelligence & Applications (IJAIA) November 2013;4(6):25–33.