UNIVERSITI MALAYSIA PAHANG
AL-SULTAN ABDULLAH

ASSIGNMENT BSD2513 ARTIFICIAL INTELLIGENCE

TOPIC: STUDENT ATTENDANCE SYSTEM USING FACE RECOGNITION

LECTURER'S NAME: DR KU MUHAMMAD NAIM

GROUP NAME: VISIONARY

NO	NAME	MATRIC ID
1	NURKHAIRUL IZZATI BINTI MOHD SALLEHAN	SD22005
2	AMIRAH YASMIN BINTI ZAILANEE	SD22021
3	NURUL HIDAYAH BINTI ROSLAN	SD22052
4	NUR SYAZWANA BINTI ROSPI	SD22023
5	HASANAH BINTI SHAFII	SD22022

TABLE OF CONTENTS

1. Executive Summary

1.1. Project Description

Ensuring accurate and honest student attendance should not be ignored in educational instituitions. The current and traditional approach for taking attendance which requires students to sign in their attendance on paper is prone to prone. How ethics and honest students in taking attendance may be questioned. Asking classmates to sign for an absent one is a prevalent practice that comprises the integrity of the attendance system. To address this problem, we suggest a real-time attendance monitoring system that makes use of face recognition technology, leveraging artificial intelligence (AI) methods to solve this problem.

In a number of circumstances, face recognition technology has outcasted itself as something truly unique and useful tool. It shows tremendous promise in a variety of applications including personal identification, security and access control. Educators and administrators can ensure that students maintain the integrity of attending classes by having them present their face-to-face learning evidence and immediately putting a facial recognition system into place. The system automatically removes the chance of proxy attendance by confirming the presence of students to classes.

To overcome this issue, our proposed facial recognition attendance system will take pictures of students as they enter the classroom and then will verify their existence to the class. The system updates the students' attendance records, providing real-time information to the educators and administrators so they may take prompt action.

1.2. Problem to be Solved

In every university, students' attendance is monitored every semester of their studies. Taking attendance is important to be checked by the lecturer at the beginning and ending of the class. The attendance of students will burden the lecturers if it needs to be handed in. It will require students to manually sign the attendance every time they attend the class. That will take more time for them to find their name and sometimes some students may mistakenly sign at another students name. Face recognition has been analyzed to improve the attendance system.

The main implementation steps that are used in this system are face detection and recognizing the detected face. It will give computer system of ability to recognize and find human faces fast. The human brain may automatically detect and recognize multiple faces. However, it is extremely difficult for a computer to do all of the challenging tasks at the level of a human brain. Hence, it need to develop a real time operating student attendance systems which is done in defined time

1.3. Description of Data

Our team has carefully curated a dataset comprising photos to facilitate the development of a Student Attendance System using Face Recognition. In our photo collection, we have captured five distinct images that showcase various scenarios related to student attendance.The system has been trained with 3 different students, for each student, the system capture training images for 50 per person to trained it and converted them into grayscale, and stored them with the student's name and ID. Student 1, 2, and 3 feature individuals who are recognized students, while Student 4 and 5 depict individuals who are not recognized students. These photos serve as essential data points for training the system to enhance its ability to identify students based on their attendance status.

By utilizing our image detection method, which is specifically designed to determine whether individuals in the photos are recognized students or not, we can thoroughly analyze these photos and extract valuable insights. The diversity of the dataset, including scenarios with both recognized and unrecognized students, enables the system to adapt to real-world situations and enhance its accuracy in identifying students based on their attendance status.

To further expand the system's capabilities, we have developed a live streaming feature that allows for real-time monitoring of student attendance. This live streaming system enables us to monitor attendance status in real time, providing valuable insights and facilitating prompt actions based on the system's detections. By leveraging this dataset, which includes a variety of student recognition scenarios and demographic factors, the Student Attendance System using Face Recognition aims to train models that are not only accurate but also adaptable to different contexts and lighting conditions. The dataset's diversity ensures that the system can effectively identify students in photos, enhancing the overall efficiency and reliability of the attendance tracking process.

2. Summary of Project Context and Objectives

2.1. Project Context

To address the challenge of accurately tracking attendance in various settings, our project centers on developing an advanced attendance detection system leveraging facial recognition and artificial intelligence technologies. By utilizing real-time video analysis, the system aims to automate attendance monitoring processes, eliminating the need for manual tracking and streamlining efficiency. The technology analyzes facial features to uniquely identify individuals and record their attendance, providing a seamless and reliable solution for attendance management across different environments.

Previous studies have primarily focused on traditional attendance tracking methods, such as manual sign-ins or card swiping systems, which often suffer from inaccuracies and inefficiencies. Our project seeks to revolutionize attendance management by offering a comprehensive solution that harnesses the power of facial recognition and AI algorithms. By integrating deep learning techniques, the system can adapt to various scenarios, including different lighting conditions, facial expressions, and occlusions, ensuring high accuracy and reliability in attendance detection.

The project's dataset comprises images and videos captured by team members, showcasing diverse scenarios and attendance situations. These include group gatherings, classroom settings, and workplace environments, enabling the system to learn from real-world data and adapt to different contexts effectively. Additionally, the project includes the development of a live streaming feature, allowing for real-time monitoring of attendance compliance and providing actionable insights for organizers and administrators.

Overall, our project aims to address the critical need for an automated attendance detection system that can accurately identify individuals in real-time video and images. By leveraging facial recognition technology and AI algorithms, the system offers a scalable and efficient solution for attendance management, promoting transparency, accountability, and productivity across various domains.

2.2. Objectives

This project objectives are:

1. To improve the accuracy of attendance by minimizing human error and ensuring reliable identification of students.
2. To enable instant attendance reports for up-to-date data and timely interventions.
3. To develop a model that can train the image and record the attendance..

3. Methodology

3.1 Data collection and preparation

In order to develop this student attendance system using face recognition, we collected and trained the image of the student. The dataset includes student detail such as ID number and Name, which enable the system to recognize the faces based on their corresponding ID and Name. We registered only 3 student to test the ability of this system. Then we capture 50 images to trained the recognizer in order to improve the performance in identifying correct student faces. The images was captured using the webcam and stored in the file name training_image based on ID number.

The dataset of the training images, attendance and student details csv file was placed inside the Google Drive to make other member can access it. By making this dataset, our aims is to train models that are accurate and able to detect correct student faces during recognizing their faces.

We prepare Attendance.csv file to stored all the attendance data here and studentDetailss.csv to stored registered student data. Then, TrainingImages folder to save all the training image here. Additional file such as haarcascade_frontalface_default.xml is a pre-trained face detector needed in order to trained the images and the trainer will developed Trainner.yml file for our face detector. Image file such as UMP.png and logo.jpg is needed to make our GUI.

3.2 Coding of project without GUI

Firstly, we will using PyCharm to complete this project and create a coding that can detect student faces correctly and saved the attendance data in the CSV file. There are few libraries that need to be installed into PyCharm before continue writing the coding for each function. Our project will use several libraries which are:

- tkinter: tkinter as GUI library in Python used to interface for the attendance system. Others widgets are included to interact with the user. Users allow to enter information such as student names and ID numbers, view attendance records and also to turn on and turn off the camera.
- pillow: Pillow library for open, edit and save image files. Its also can convert format, resize and crop the image for process by OpenCV.

- csv: Csv will store and retrieve attendance records and saved it in a CSV file. It easy to import and export of the attendance data for analysis and visualization.
- numpy: Numpy is a numerical computing library that can perform operations like filtering and transformations of image arrays. It also convert image data into array to be processed by OpenCV.
- matplotlib: Matplotlib as for plotting in data visualization. It generate graphs or charts that display the attendance data trends over time.
- cv2: This library is Open Source Computer Vision that used as face detection and recognition. It will access the webcam to capture real-time images for registering attendance.
- os: Os will manage the file paths that it will create and access the necessary directories and files.
- pandas: a Python library used for working with data sets.It has functions for analyzing, cleaning, exploring, and manipulating data.
- datetime: supplies classes to work with date and time. These classes provide several functions to deal with dates, times, and time intervals.
- time: reading the current time, formatting time and sleeping for a specified number of seconds and so on.

```
import tkinter as tk
import csv
import cv2
import os
import numpy as np
from PIL import Image, ImageTk
import pandas as pd
import datetime
import time
import matplotlib.pyplot as plt
from tkinter import messagebox
```

This part for library that involves to create a GUI application for manage student attendance system. This all library that involves for this project.

```
window = tk.Tk()
window.title("Student Attendance System")
window.geometry('1000x550')
```

This code creates an instance of the Tk class, which initializes a new Tkinter window. This window will serve as the main window for the GUI application.This line sets the title of the Tkinter window to "Student Attendance System". This title will appear in the title bar of the

window.This line sets the dimensions of the window to 1000 pixels wide and 550 pixels high. The geometry method specifies the size of the window.

Function take image for training.

```
def takeImage():
    name = (std_name.get())
    Id = (std_number.get())
    if name.isalpha():
        cam = cv2.VideoCapture(0)
        harcascadePath = "haarcascade_frontalface_default.xml"
        detector = cv2.CascadeClassifier(harcascadePath)
        sampleNum = 0

        while True:
            ret, img = cam.read()
            gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
            faces = detector.detectMultiScale(gray, scaleFactor: 1.1, minNeighbors: 3)
            for (x, y, w, h) in faces:
                cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
                sampleNum = sampleNum + 1
                # store each student picture with its name and id
                cv2.imwrite("TrainingImages\ " + name + "." + Id + '.' + str(sampleNum) + ".jpg",
                            gray[y:y + h, x:x + h])
                cv2.imshow( winname: 'FACE RECOGNIZER', img)
            if cv2.waitKey(100) & 0xFF == ord('q'):
                break
            # stop the camera when the number of picture exceed 50 pictures for each student
            if sampleNum > 50:
                break
```

This function for capture the face of students that will be used in face recognition system. It will need to enter the name and ID student for it to recognized based on their faces. It will capture the faces continuously by **cam.read( )** and convert them into grayscale. It saved the images of detected faces with their identities as a filename in “TrainingImages” directory. This webcam will stop when press the ‘q’ key.

```
        cam.release()
        cv2.destroyAllWindows()
        # print the student name and id after a successful face capturing
        res = 'Student details saved with: \n Matric number : ' + Id + ' and  Full Name: ' + name

        row = [Id, name]

        with open('studentDetailss.csv', 'a+') as csvFile:
            writer = csv.writer(csvFile)
            writer.writerow(row)
        csvFile.close()
        label4.configure(text=res)
    else:

        if name.isalpha():
            res = "Enter correct Matric Number"
            label4.configure(text=res)
```

cam.release() will stop the webcam and destroy all openCV files that were opened during the process. This function will ensure that the resources available and the users receives the accurate input. It will stores the student's details in CSV file and modify the GUI to show the successful or invalid input. The GUI will show the error message when students enter the wrong input.

Function get Images and Labels from registered student to make a image file

```
def getImagesAndLabels(path):
    imagePaths = [os.path.join(path, f) for f in os.listdir(path)]
    faces = []
    Ids = []
    for imagePath in imagePaths:
        pilImage = Image.open(imagePath).convert('L')
        imageNp = np.array(pilImage, dtype: 'uint8')
        Id = int(os.path.split(imagePath)[-1].split(".")[1])
        faces.append(imageNp)
        Ids.append(Id)
    return faces, Ids
```

This function of **getImagesAndLabels** was developed with the objective of processing photos and collecting the Ids that connected to each one. The resulting **imagePaths** list contains the complete file paths for all images in the directory. It opens the image using **Image.open(imagePath)** and converts it to grayscale using **.convert('L')**. The grayscale image is then converted to a NumPy array with data type **'uint8'**.The processed image array is appended to the faces list, and the ID is appended to the Ids list. Finally, the function returns two lists of faces that containing processed image data and Ids that containing corresponding IDs.

Function training images.

```
def trainImage():
    recognizer = cv2.face.LBPHFaceRecognizer_create()
    harcascadePath = "haarcascade_frontalface_default.xml"
    detector = cv2.CascadeClassifier(harcascadePath)
    faces, Id = getImagesAndLabels("TrainingImages")
    recognizer.train(faces, np.array(Id))
    recognizer.save("Trainner.yml")
    res = "Image Trained"
    label4.configure(text=res)
```

The code snippet defines a function trainImage() used for training a facial recognition system with images stored in a directory named "TrainingImages". It creates an instance of LBPHFaceRecognizer using OpenCV, sets the path to a pre-trained Haar Cascade classifier for detecting faces, and loads the classifier. The getImagesAndLabels function is called to retrieve the training images and their corresponding labels. The recognizer is then trained on these images, and the trained model is saved to a file named "Trainner.yml". Finally, a label is updated to display the message "Image Trained" indicating the successful training of the model.

Function trackImage to track student faces and mark attendance

```python
def trackImage():
    recognizer = cv2.face.LBPHFaceRecognizer_create()
    recognizer.read("Trainner.yml")
    harcascadePath = "haarcascade_frontalface_default.xml"
    faceCascade = cv2.CascadeClassifier(harcascadePath)
    df = pd.read_csv("studentDetailss.csv")
    font = cv2.FONT_HERSHEY_COMPLEX_SMALL
    cam = cv2.VideoCapture(0)
    # create a dataframe to hold the student id,name,date and time
    col_names = ['Id', 'Name', 'Date', 'Time', 'Status']
    attendance = pd.DataFrame(columns=col_names)

    while True:
        ret, img = cam.read()
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        faces = faceCascade.detectMultiScale(gray, scaleFactor: 1.1, minNeighbors: 3)

        recognized_ids = set()

        for (x, y, w, h) in faces:
            cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
            Id, conf = recognizer.predict(gray[y:y + h, x:x + w])
            #  a confidence less than 60 indicates a good face recognition
            if conf < 60:
                ts = time.time()
                date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')
                timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M')
                aa = df.loc[df['ID'] == Id]['NAME'].values
```

The `trackImage` function is a crucial component of a student attendance system employing face recognition. It initializes a face recognizer with a pre-trained model, sets up a Haar Cascade Classifier for face detection, and loads student details from a CSV file into a DataFrame. Utilizing the webcam, it continuously captures frames, converts them to grayscale, and detects faces. For each detected face, it draws a rectangle around it and attempts to predict the ID using the face recognizer. If the confidence level is below a threshold, it records the attendance by retrieving the current timestamp, marking the student as 'Present', and appending the attendance record to a DataFrame. This function efficiently manages real-time face recognition and attendance tracking within a single loop.

```
            tt = str(Id) + "-" + aa
            attendance.loc[len(attendance)] = [Id, aa, date, timeStamp, 'Present']
            row2 = [Id, aa, date, timeStamp, 'Present']
            #   open the attendance file for update
            with open('AttendanceFile.csv', 'a+') as csvFile2:
                writer2 = csv.writer(csvFile2)
                writer2.writerow(row2)
            csvFile2.close()
            # print attendance updated on the notification board of the GUI
            res = 'ATTENDANCE UPDATED WITH DETAILS'
            label4.configure(text=res)
            recognized_ids.add(Id)
        else:
            Id = 'Unknown'
            tt = str(Id)
            #  store the unknown images in the images unknown folder
            noOfFile = len(os.listdir("ImagesUnknown")) + 1
            cv2.imwrite("ImagesUnknown\Image" + str(noOfFile) + ".jpg", img[y:y + h, x:x + w])
            res = 'ID UNKNOWN, ATTENDANCE NOT UPDATED'
            label4.configure(text=res)

        # show the student id and name
        cv2.putText(img, str(tt),  org: (x, y + h - 10), font,  fontScale: 0.8,  color: (255, 255, 255),  thickness: 1)
        cv2.imshow( winname: 'FACE RECOGNIZER', img)
```

This Python script is a face recognition system that identifies individuals using a webcam and marks their attendance in a CSV file. The script uses OpenCV's LBPH (Local Binary Patterns Histograms) face recognizer to identify faces from a live video feed. It reads pre-trained face data (Trainer.yml), a Haar cascade classifier for detecting faces, and student details from studentDetails.csv. The video feed is processed frame-by-frame, converting each frame to grayscale and detecting faces. For each detected face, the recognizer predicts the ID and confidence level. If the confidence is below 60, indicating a high likelihood of correct identification, the script records the current timestamp and marks the individual's attendance. The attendance data is saved in Attendance.csv, and unrecognized faces are saved as images in an "ImagesUnknown" folder. The recognized face is highlighted with a rectangle on the video feed and the ID is displayed.

```
    # Mark students who are registered but not recognized as absent
    registered_ids = set(df['ID'])
    for id in registered_ids - recognized_ids:
        ts = time.time()
        date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')
        timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M')
        aa = df.loc[df['ID'] == id]['NAME'].values
        attendance.loc[len(attendance)] = [id, aa, date, timeStamp, 'Absent']
        row2 = [id, aa, date, timeStamp, 'Absent']
        #   open the attendance file for update
        with open('AttendanceFile.csv', 'a+') as csvFile2:
            writer2 = csv.writer(csvFile2)
            writer2.writerow(row2)
        csvFile2.close()

    if cv2.waitKey(1000) == ord('q'):
        break

cam.release()
cv2.destroyAllWindows()
```

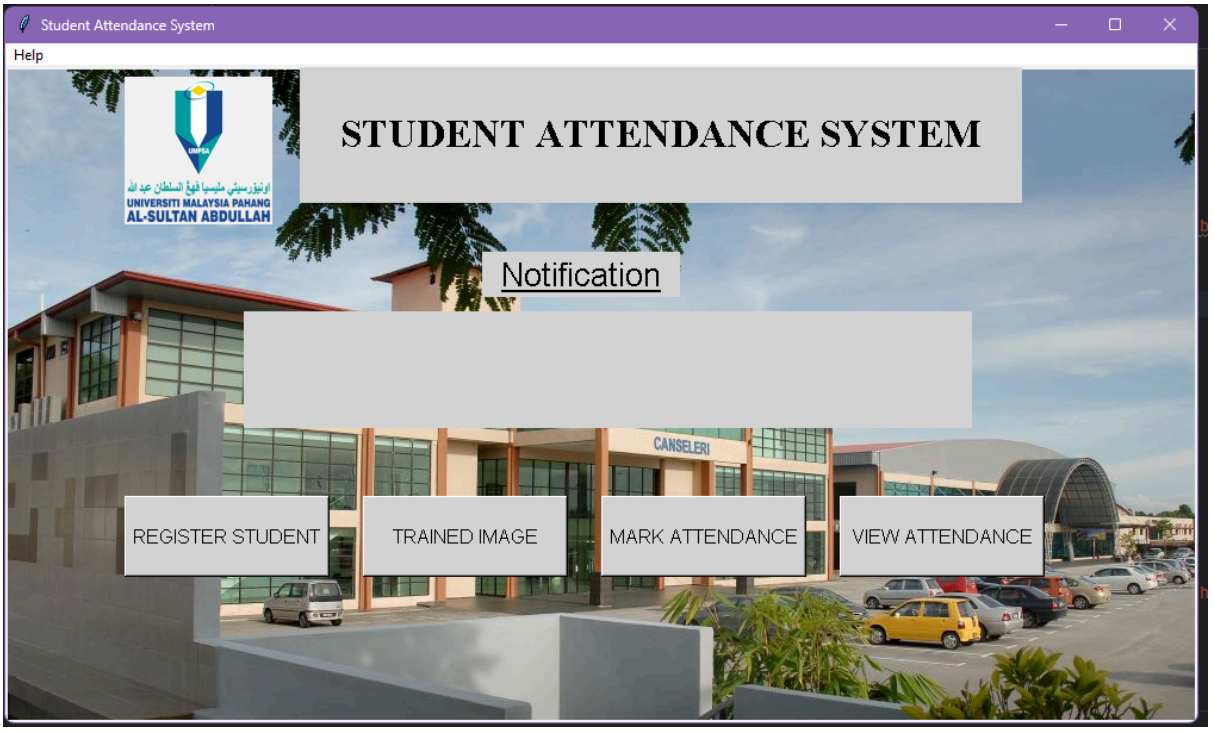
This code show marks of students who are registered but not recognized in the current frame as absent. It starts by creating a set of registered IDs from the DataFrame `**df**`. It then iterates over each ID in the difference between the registered IDs and the recognized IDs. For each unrecognized ID, it captures the current timestamp, formats it into date and time strings, and retrieves the student's name from the DataFrame. The code adds an attendance record for the unrecognized student, marking them as 'Absent', and appends this record to the `attendance` DataFrame. It also writes the **'Absent'** record to **"AttendanceFile.csv"**. The loop continues until the 'q' key is pressed, which breaks the loop. Finally, the webcam is released, and all OpenCV windows are closed.

Function viewAttendance to see the analysis

```
def viewAttendance():
    attendance_df = pd.read_csv("AttendanceFile.csv")

    # Count the number of students present and absent
    presence_counts = attendance_df['STATUS'].value_counts()
    present_count = presence_counts.get('Present', 0)
    absent_count = presence_counts.get('Absent', 0)

    # Visualize attendance
    labels = ['Present', 'Absent']
    sizes = [present_count, absent_count]
    colors = ['#ff9999', '#66b3ff']
    explode = (0.1, 0)  # explode 1st slice

    plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', startangle=140)
    plt.axis('equal')
    plt.title('Attendance Visualization')
    plt.show()
```

This Python function reads attendance data which are presence or absence from a CSV file into a DataFrame. It then extracts the count of present and absent students. Finally, it uses Matplotlib to create a pie chart visualizing the attendance distribution with slices labeled "Present" and "Absent", sized based on student counts, and displayed with a title "Attendance Visualization".

3.3 Adding the GUI to coding




```
label3 = tk.Label(window, background="lightgrey", fg="black", text="Notification", width=10, height=1,
                  font=('Helvetica', 20, 'underline'))
label3.place(x=400, y=155)
label4 = tk.Label(window, background="lightgrey", fg="black", width=55, height=4, font=('Helvetica', 14, 'italic'))
label4.place(x=200, y=205)

regStudentBtn = tk.Button(window, command=registerStudent, background="lightgrey", fg="black", text="REGISTER STUDENT",activebackground="red",
                          width=18, height=3, font=('Helvetica', 12))
regStudentBtn.place(x=100, y=360)


trainImageBtn = tk.Button(window, command=trainImage, background="lightgrey", fg="black", text="TRAINED IMAGE",
                          activebackground="red",
                          width=18, height=3, font=('Helvetica', 12))
trainImageBtn.place(x=300, y=360)
```

```
trackImageBtn = tk.Button(window, command=trackImage, background="lightgrey", fg="black", text="MARK ATTENDANCE", width=18,
                          activebackground="red", height=3, font=('Helvetica', 12))
trackImageBtn.place(x=500, y=360)

viewAttendanceBtn= tk.Button(window, command=viewAttendance, background="lightgrey", fg="black", text="VIEW ATTENDANCE", width=18,
                          activebackground="red", height=3, font=('Helvetica', 12))
viewAttendanceBtn.place(x=700, y=360)

# Menu bar
menu_bar = tk.Menu(window)
help_menu = tk.Menu(menu_bar, tearoff=0)
help_menu.add_command(label="Help", command=show_help)
help_menu.add_command(label="About Us", command=show_about)
menu_bar.add_cascade(label="Help", menu=help_menu)

window.config(menu=menu_bar)
```

Importing and Setting Up

- Library: The tkinter library is used for creating graphical user interfaces in Python.
- Buttons: Buttons are interactive elements that users can click to perform actions.

Code Explanation

1. Register Button
    - regStudentBtn: This creates a button named "REGISTER STUDENTt".
    - window: This is the parent frame in which the button resides.
    - text="REGISTER STUDENT" This sets the text label on the button to "Register Student".
    - command=registerStudent: This associates the button with the register_student function, which will be called when the button is clicked.

2. Train Image
    - trainImageBtn: This creates a button named "TRAINED IMAGE".
    - text="TRAINED IMAGE": This sets the button's label to "TRAINED IMAGE".
    - command=trainImage: This assigns the trainImage function to the button's click event.

3. Mark Attendance
    - trackImageBtn: This creates a button named "MARK ATTENDANCE".
    - text="Mark Attendance": This sets the button's label to "MARK ATTENDANCE".
    - command=TrackImages: This assigns the TrackImages function to the button's click event.

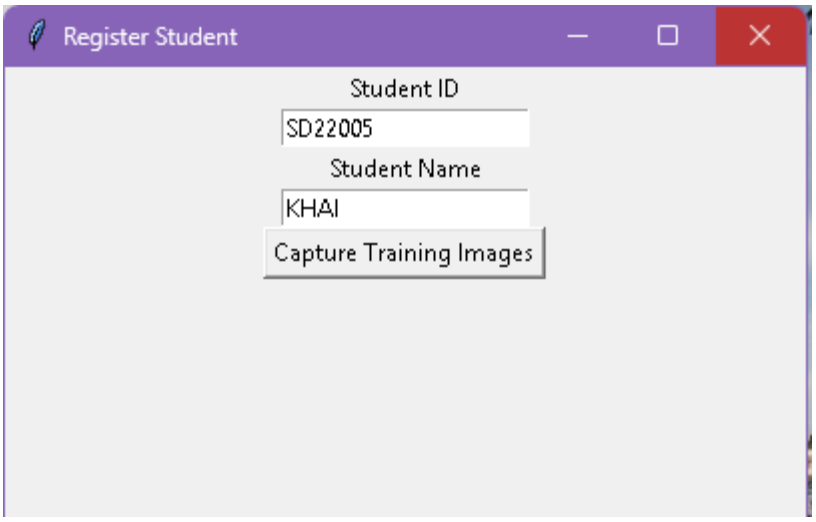
4. View Attendance Button
    - viewAttendanceBtn: This creates a button named "VIEW ATTENDANCE".
    - text="View Attendance": This sets the button's label to "VIEW ATTENDANCE".
    - command=viewAttendance: This assigns the viewAttendance function to the button's click event.

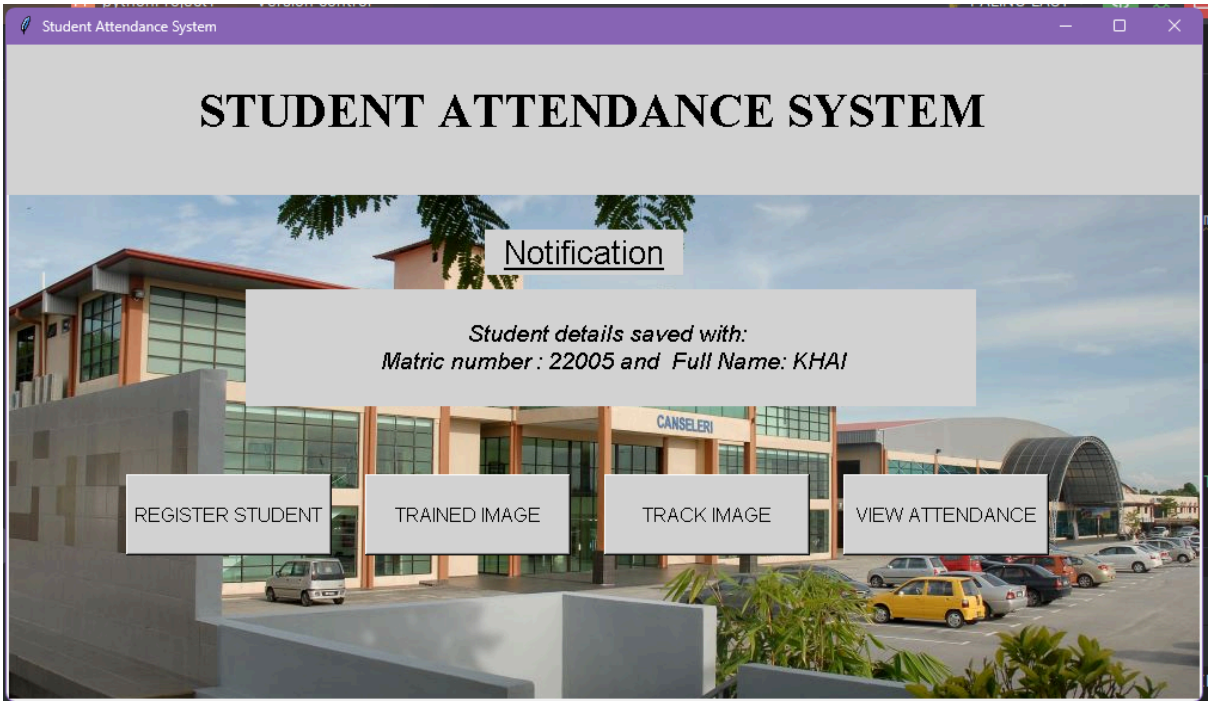
5. Menu Bar

- Help menu bar will have the help and about us menu inside, it will give a detail of this project and credits.

The code sets up a frame containing three buttons within a canvas, each performing a different task related to a student attendance system. The buttons are:

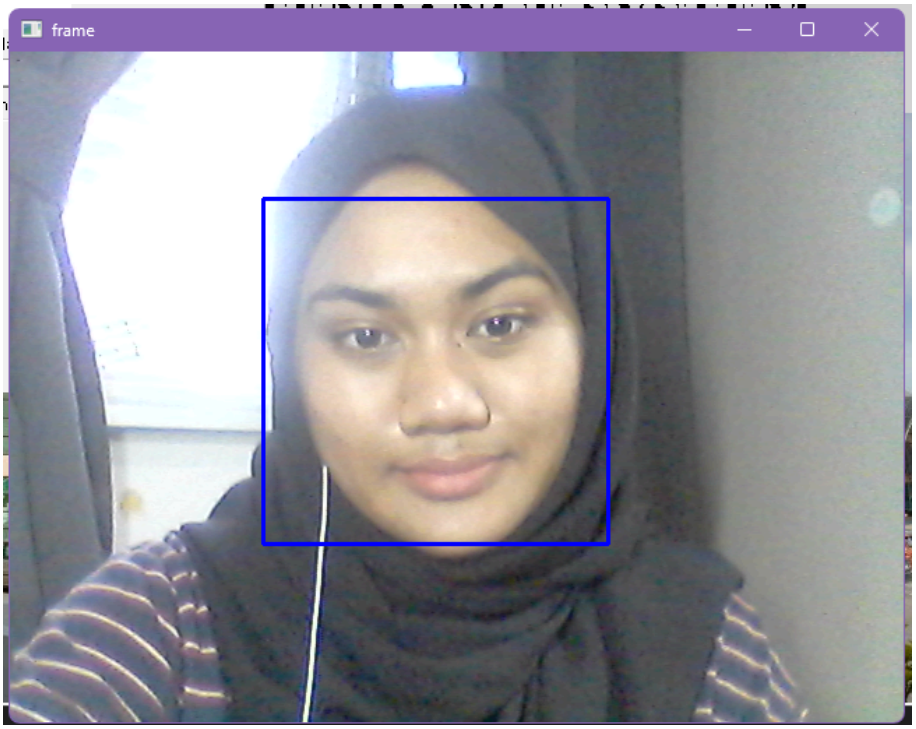
- Register Student: To register a new student.
- Train Image: To train the capture image
- Mark Attendance: To mark attendance for students.
- View Attendance: To view attendance records.





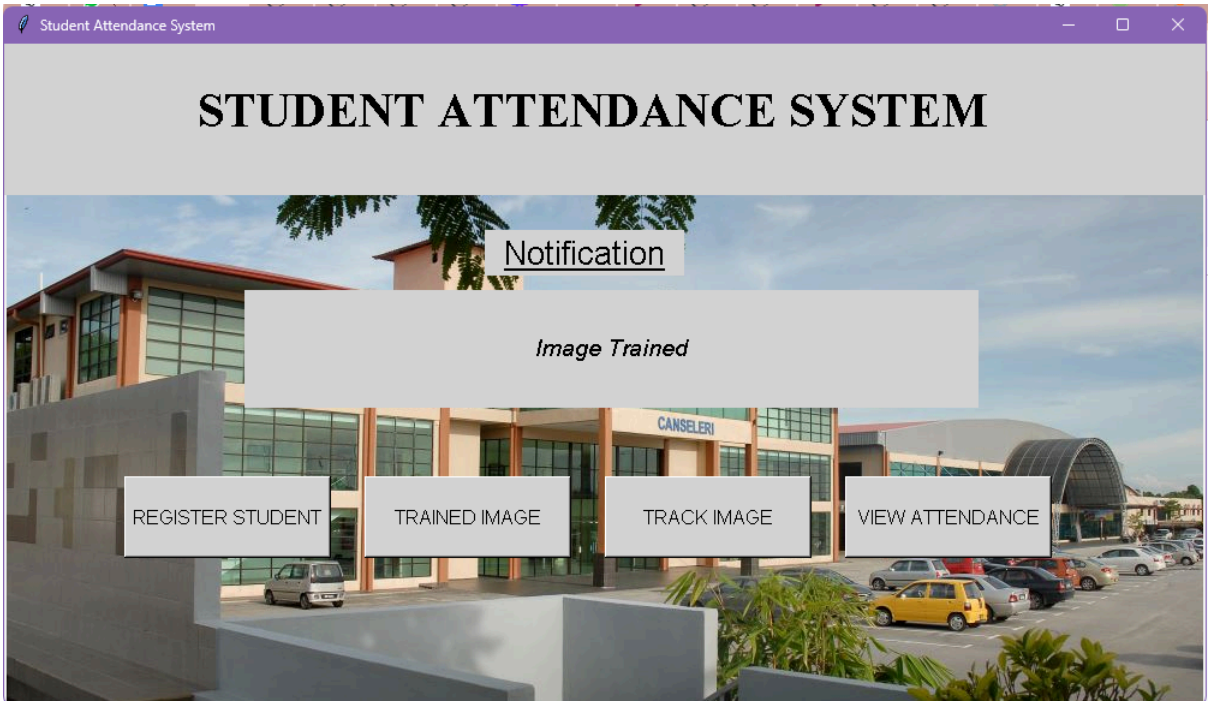
```python
def register_student():
    register_window = tk.Toplevel(window)
    register_window.title("Register Student")
    register_window.geometry("400x300")

    tk.Label(register_window, text="Student ID").pack()
    student_id_entry = tk.Entry(register_window)
    student_id_entry.pack()

    tk.Label(register_window, text="Student Name").pack()
    student_name_entry = tk.Entry(register_window)
    student_name_entry.pack()

    capture_button = tk.Button(register_window, text="Capture Training Images",
                               command=lambda: TakeImages(student_id_entry.get(), student_name_entry.get()))
    capture_button.pack()
```

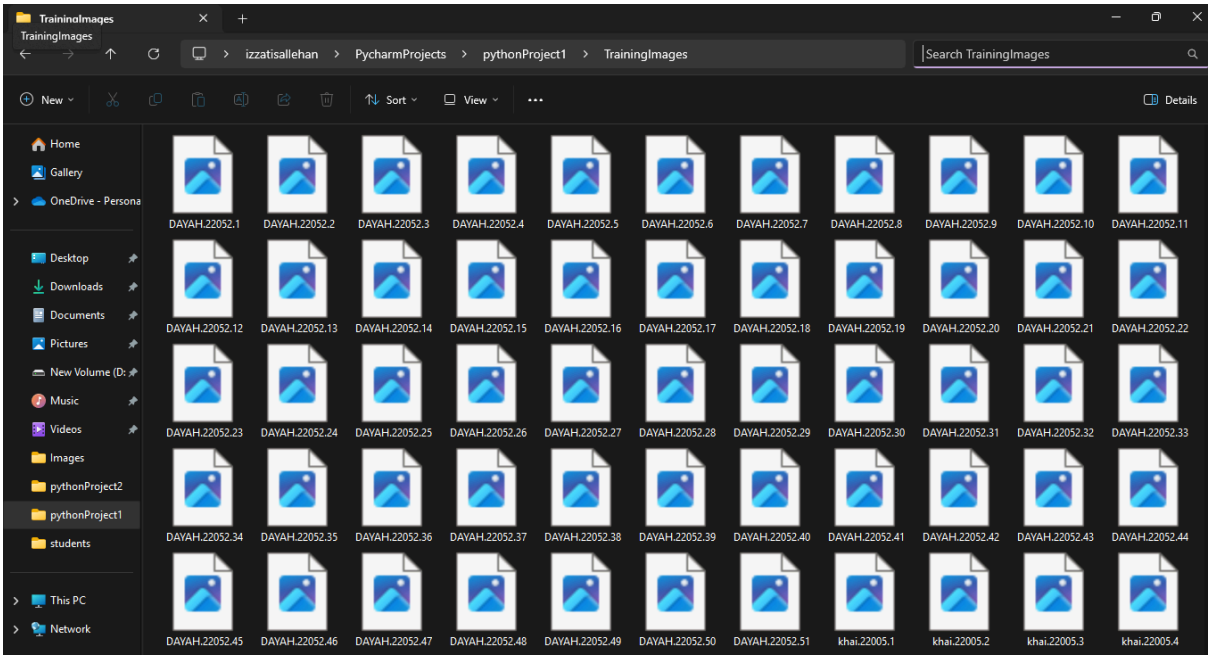
According to the GUI code above,it creates a student registration and attendance system using Tkinter for the interface. The `register_student` function opens a new window for student registration where users can enter the student ID and name. This window includes two input fields for these details and a button labeled "Capture Training Images." When clicked, this button invokes the `TakeImages` function, capturing and saving the student's facial images for training. The main interface, titled "STUDENT ATTENDANCE SYSTEM," features buttons for registering students, training images, tracking images, and viewing attendance. Notifications appear to inform users of successful operations, such as saving student details. This system integrates facial recognition to automate attendance tracking, marking recognized students as present and saving their details in an attendance file, while unidentified faces are stored separately.



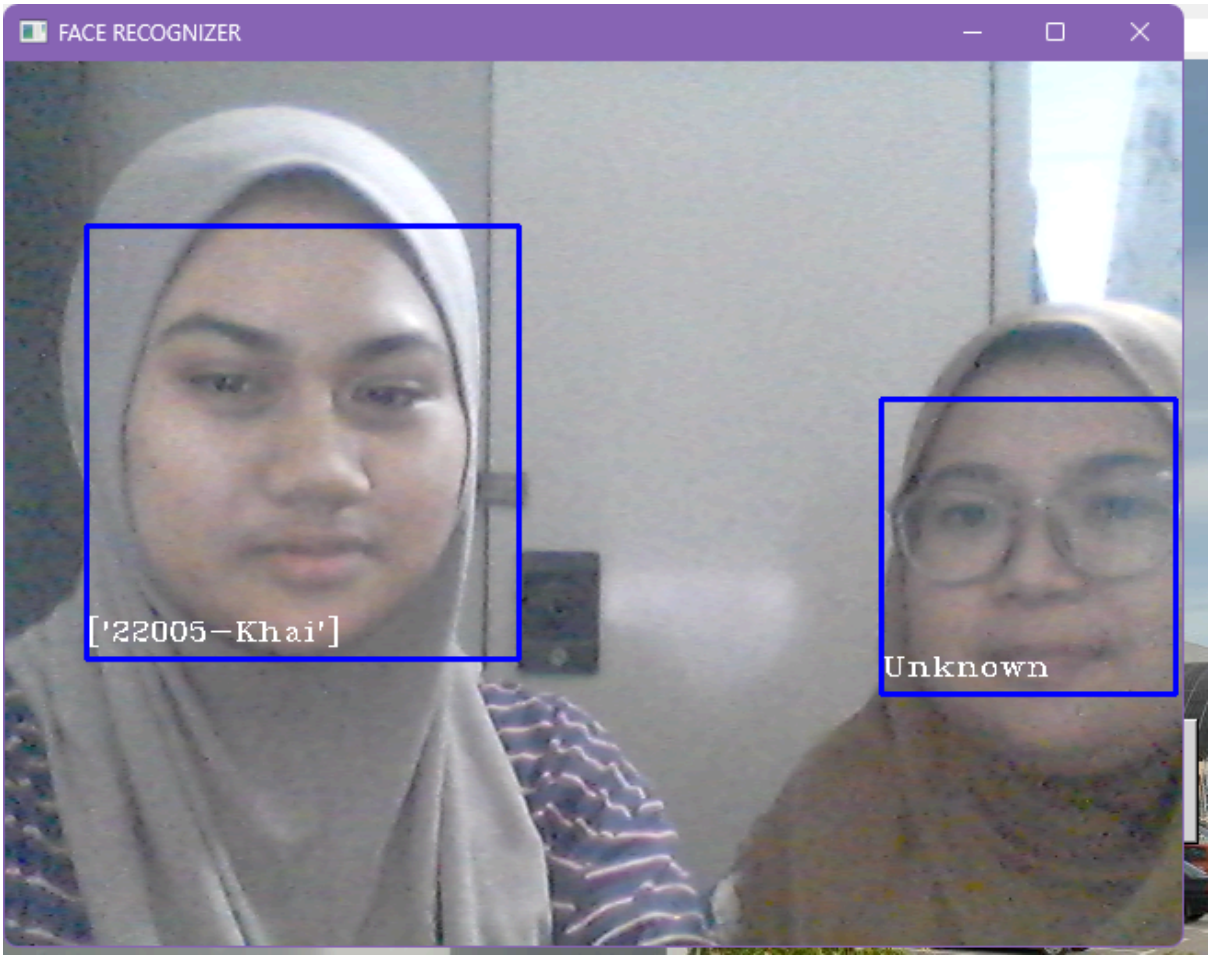
	A	B	C	D
1	ID	NAME		
2	22005	Khai		
3				
4	22023	ZEE		
5				
6	22052	DAYAH		
7				
8				
9				
10				
11				
12				

In the AttendanceFiles.csv, the data registered would go to this path excel file.

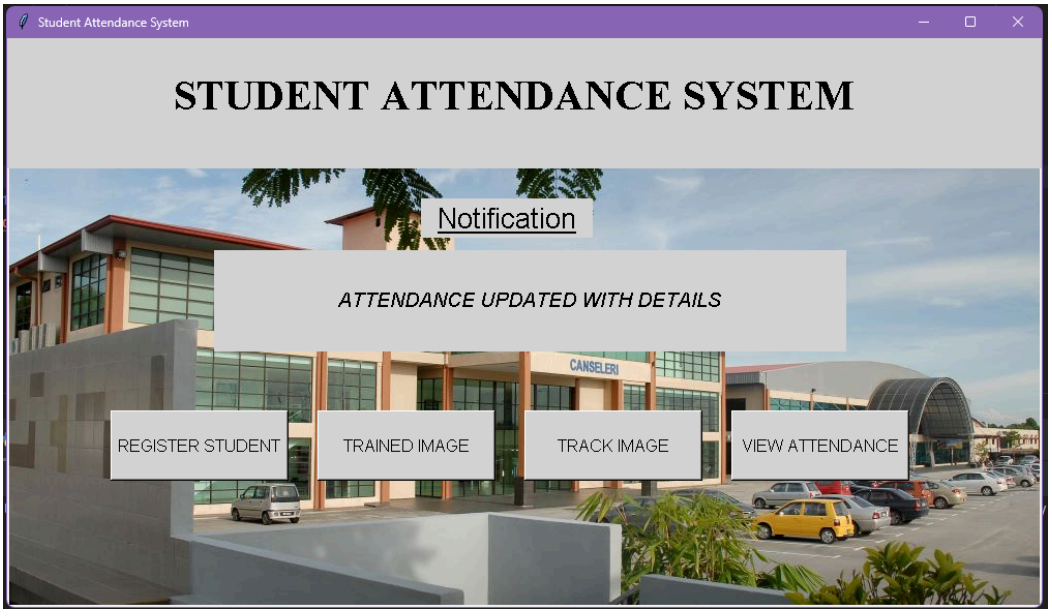




This code snippet defines a register_student function that creates a graphical user interface (GUI) window for registering new students using the Tkinter library in Python. When the function is called, it opens a new top-level window titled "Register Student" with a size of 400x300 pixels. Within this window, two labels and entry fields are created for inputting the student's ID and name. Below these fields, a button labeled "Capture Training Images" is added. When this button is clicked, it triggers the TakeImages function with the entered student ID and name as arguments, allowing the user to capture and store the student's face images for training the face recognition system. Then, the “TRAINED IMAGE” will do training function the images captured.



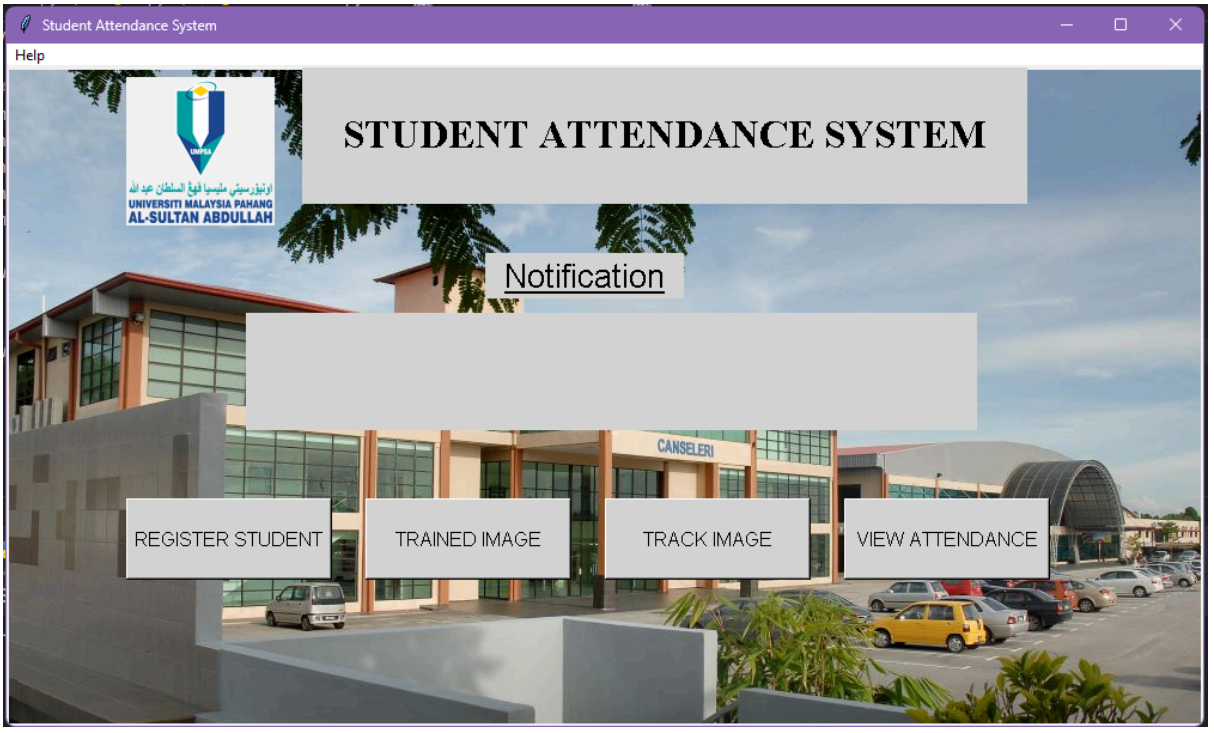
For example, the camera recognized ‘Khai’ because she had registered the system. While the other one, who has not registered yet will be recognized as ‘Unknown’. If recognized, it will display the ID and name.



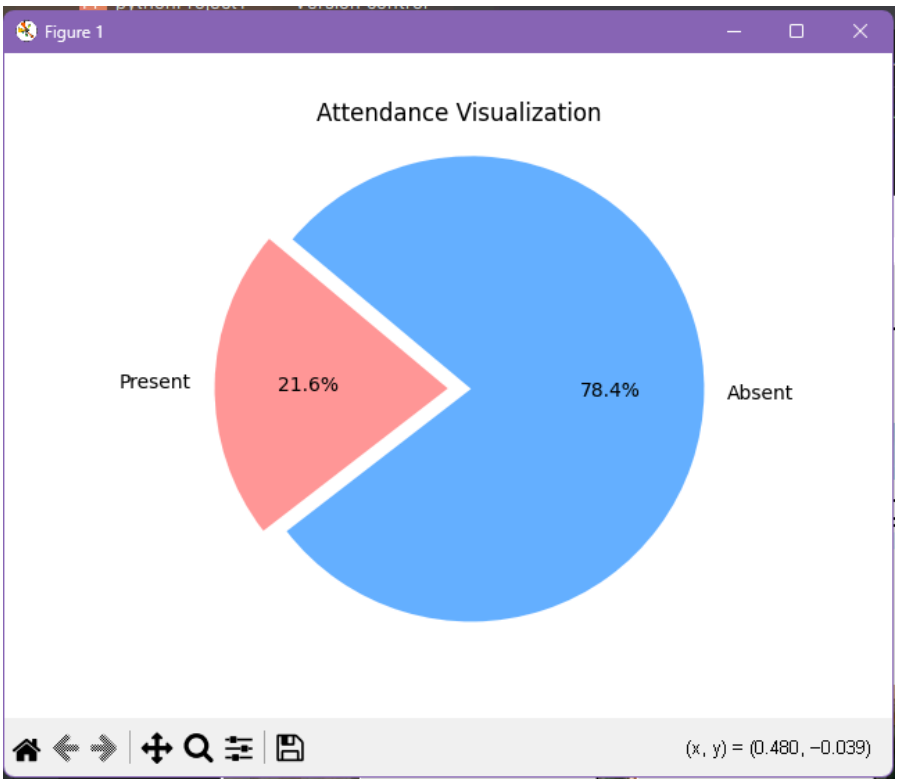
	A	B	C	D	E
1	ID	NAME	DATE	TIME	STATUS
2	22005	['Khai']	9/6/2024	20:40	Present
3	22023	['ZEE']	9/6/2024	20:52	Present
4	22005	['Khai']	9/6/2024	20:52	Present
5	22023	['ZEE']	9/6/2024	20:52	Present
6	22005	['Khai']	9/6/2024	20:52	Present
7	22052	['DAYAH']	9/6/2024	20:54	Present
8	22005	['Khai']	9/6/2024	20:54	Present
9	22052	['DAYAH']	9/6/2024	20:54	Present
10	22052	['DAYAH']	9/6/2024	20:54	Present
11	22005	['Khai']	9/6/2024	20:54	Present
12	22005	['Khai']	9/6/2024	20:56	Present
13	22005	['Khai']	9/6/2024	20:58	Present
14	22052	['DAYAH']	9/6/2024	20:58	Absent
15	22023	['ZEE']	9/6/2024	20:58	Absent
16	22005	['Khai']	9/6/2024	20:58	Present
17	22052	['DAYAH']	9/6/2024	20:58	Absent
18	22023	['ZEE']	9/6/2024	20:58	Absent
19	22005	['Khai']	9/6/2024	20:58	Present
20	22052	['DAYAH']	9/6/2024	20:58	Absent
21	22023	['ZEE']	9/6/2024	20:58	Absent
22	22005	['Khai']	9/6/2024	21:17	Present
23	22052	['DAYAH']	9/6/2024	21:17	Absent

In the AttendanceFiles.csv, the data registered including ID, Name, Date, Time and Status would go to this path excel file.

4. Results and Discussion



When the code was initially started, the user would see the GUI main menu with the title Student Attendance system. The user can select from four buttons in this main menu: Register Student, Trained Image, Track Image and View Attendance.



When choosing the View Attendance, the registered attendance would be calculated and the percentage of absence and presence are visualized in a pie chart.

5. Conclusion

Technology and streamline of the process of recording attendance in universities that have been significantly enhanced by the use of facial recognition in Student Attendance Systems. Applying innovative face recognition technology, the system provides a reliable, effective and trustworthy way to record student attendance. Once it comes to recognizing and identifying student personal identities the facial recognition technology has proved high accuracy. This minimizes the possibility of mistakes that come from applying regular attendance methods including manual countdowns or tracking systems where virtual attendance can be often a problem.

The effort required to mark attendance reduced by applying face recognition. This method just take an amount of seconds as students enter the classroom rather than take too many minutes. This makes it possible to use class time more wisely. The technology uses facial recognition to make sure that only students who have registered are listed as present. Moreover, the safety and accuracy of attendance records are improved, reducing the possibility of duplication and illegal access. Lecturers and students can get real time records of attendance via the system. This makes it easier to reach out right away when there are suspicious attendance records and encourages better options for student management. They will easily use the system's GUI. Students can simply verify their attendance so lecturers can prepare attendance reports and perform data analysis without specialist expertise.

By implementing technologies like learning management system, virtual attendance system that used advanced artificial intelligence algorithms for better recognition accuracy and the system will be improved. In summary, an important technological advancement in the education sector which is the student attendance system with face recognition. It provides an advanced, safe and effective solution that overcomes many of the limitations with manual attendance tracking systems. More improvements and modifications to this system as technology develops will surely produce even more efficient and smoother methods for managing education.

6. References

1. Datta, S. (2021, April 28). *How to get started with Firebase using python*. freeCodeCamp.org. https://www.freecodecamp.org/news/how-to-get-started-with-firebase-using-python/
2. *Face recognition with Real time database: 2 Hour course: Computer vision* (2022) *YouTube*. Available at: https://youtu.be/iBomaK2ARyI?si=iuoZQhLlQ4IYZGH2 (Accessed: 09 June 2024).
3. *Face recognition with real-time database* (2022) *Computer Vision Zone*. Available at: https://www.computervision.zone/courses/face-recognition-with-real-time-database/ (Accessed: 09 June 2024).
4. GeeksforGeeks. (2024, May 8). *Python packages*. https://www.geeksforgeeks.org/python-packages/
5. MeetSuvariya25 (no date) Face-recognition-based-attendance-system/review_demo.py at main · meetsuvariya25/face-recognition-based-attendance-system, GitHub. Available at: https://github.com/MeetSuvariya25/Face-Recognition-Based-Attendance-System/blob/main/review_demo.py (Accessed: 09 June 2024).

7. Appendix

https://drive.google.com/drive/folders/1vnvhYIox_DvWA7IqNOZTf0SzjR4kbWVm?usp=sharing