# A

**Project Report On**

# “FACE RECOGNITION ATTENDANCE SYSTEM USING MACHINE LEARNING”

Submitted in partial fulfilment of the requirements for the Bachelor Degree in

# B.Voc. (Artificial Intelligence and Data Science) Semester-VI

**Session-2023-2024** **To**

# DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

**KANYA MAHA VIDYALAYA, JALANDHAR (AUTONOMOUS)**

# 

**Submitted To:**

**Ms. Simarpreet Kaur**

**Faculty of Vocational Studies**

# Submitted by:

**Name: Khushi**

# Examination Roll No.: 21114000007

## CERTIFICATE OF COMPLETION AND EVALUATION

This is to certify that Khushi, student of B.Voc. (Artificial Intelligence and Data Science) Semester-VI at Kanya Maha Vidyalaya, Jalandhar has successfully completed the Project titled “Face Recognition Attendance System Using Machine Learning” under the Guidance of Ms. Simarpreet Kaur.

Ms. Simarpreet Kaur Dr. Pardeep Arora

Lecturer Head

Department of Artificial Intelligence and Department of Artificial Intelligence and Data Science Data Science

Kanya Maha Vidyalaya Kanya Maha Vidyalaya

The Project/ Assignment was evaluated through a viva examination, and the student demonstrated understanding of the subject matter. The viva examination was conducted by:

Name of the Viva Examiner Name of the Viva Examiner

Date:

# ACKNOWELGEMENT

The accomplishment of the Dissertation entitled: “Face Recognition Attendance System Using Machine Learning” would not have been possible without the encouragement, support and cooperation of number of people.

From the deep core of our heart, we keenly wish to express our sincere thanks to our principle, Prof. (Dr.) Atima Sharma Dwivedi for providing with the necessary facilities required for the project.

It is great pleasure for us to acknowledge Dr. Pardeep Arora, Head, Department of Artificial Intelligence and Data Science for his support. We acknowledge our deep sense of gratitude and reverence to our worthy Project Incharge, Ms. Simarpreet Kaur for his able guidance.

Department for providing us all the facilities required for the completion of this dissertation work. We express our thanks to our family for their support and affection and for believing us always.

Khushi

University Roll No.:21114000007

# DECLARATION

This is to declare that dissertation work entitled: “Face Recognition Attendance System Using Machine Learning” bonafide work done by us. This report is being submitted and in the partial fulfillment of the requirements for the award of the Degree of bachelor of Vocation in Artificial Intelligence and Data Science from Kanya Maha Vidyalaya, Jalandhar.

This report has not been submitted to any other Institutions or University for the award of any other Degree of Diploma.

Khushi University Roll No.:21114000007

# ABSTRACT

The system is a web application developed using Python, Flask, and OpenCV for face recognition. It allows students to register by providing their details and uploading their photo. The photos are securely stored in a database.

When a student wants to mark their attendance, they can click the "Attend" button, and the system will use face recognition to verify their identity. If the student is recognized, their attendance is marked, and it's automatically updated in real-time. The attendance records can be exported to an Excel sheet for easy sharing and analysis.

The system also includes an "Exit" button that allows students to mark their exit time. This feature can help administrators to monitor the duration of attendance.

The system is designed to work with Python, Visual Studio, and Flask.

# 

# TABLE OF CONTENTS

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No** | **Title** | **Page No.** | **Signature** |
| 1. | **INTRODUCTION OF FACE RECOGNITION ATTENDANCE SYSTEM** | 7 |  |
|  | 1.1 Types | 8 |  |
|  | 1.2 Tools &Technologies | 8-9 |  |
|  | 1.3 Process | 9 |  |
|  | 1.4 Applications | 10 |  |
|  | 1.5 Objective | 10 |  |
|  |  |  |  |
| 2. | **WORKING OF FACE RECOGNITION ATTTENDANCE SYSTEM PROJECT** | 11 |  |
|  | 2.1 main.py | 11 |  |
|  | 2.2 student.py | 14 |  |
|  | 2.3 attendance.py | 16-17 |  |
|  | 2.4 face\_recognition.py | 17-18 |  |
|  |  |  |  |
| 3. | **CONCLUSION** | 19 |  |
|  |  |  |  |
| 4. | **REFERENCES** | 20 |  |
|  |  |  |  |

1. **INTRODUCTION**

Face recognition attendance systems are becoming increasingly popular in various industries, including education, healthcare, and corporate environments. These systems use facial recognition technology to identify and verify individuals, making attendance tracking more efficient and accurate. In this project, we will discuss how face recognition attendance systems using Flask, Python, and Visual Studio work.

****

**Figure 1**

Face recognition attendance systems typically consist of three main components: a camera, a face recognition algorithm, and a database to store attendance records. The camera captures images of individuals, and the face recognition algorithm identifies and verifies their identities. The attendance records are then stored in a database for future reference.

To build a face recognition attendance system using Flask, Python, and Visual Studio, you will need to

install several libraries, including OpenCV, face\_recognition, and Flask. OpenCV is popular computer

vision libraries that provides functions for image processing and face detection. The face\_recognition

library is a Python library that uses deep learning to recognize faces. Flask is a lightweight web

Framework that permits you to construct net programs quick and easily.

Here’s a basic introduction to the key components and methods used in Face Recognition Attendance System:

1. Face Detection: This component uses computer vision techniques to detect and locate faces in images or video frames.
2. Face Recognition: Once faces are detected, this component uses machine learning algorithms to perceive and understand people primarily based totally on their facial features.
3. Database Management: The system requires a database to store and manage information about registered individuals, such as their names and unique identifiers.
4. User Interface: Visual Studio can be used to create a user-friendly interface where users can interact with the system.
5. Web Framework (Flask): Flask is a Python web framework that allows you to create a web application for the attendance system. It handles the communication between the user interface and the backend components.
6. Integration and Communication: Python acts as the glue language that integrates the different components of the system and ensures smooth communication between them.
7. Attendance Management: This component keeps track of attendance records for each registered

individual. When a recognized face is detected, the system marks the corresponding individual as present and updates the attendance database.

**1.1 Types of Face Recognition Attendance System:**

There are different types of face recognition attendance systems based on their technology and implementation. Here are a few common types:

1. 2D Face Recognition: This type of system uses 2D images captured by cameras to recognize and verify faces. It analyzes facial features and matches them against a database of stored faces.

2. 3D Face Recognition: 3D face recognition systems use depth-sensing cameras or structured light to capture facial geometry in three dimensions. This technology provides more accurate and robust recognition, even in challenging lighting conditions.

3. Thermal Face Recognition: Thermal face recognition systems use infrared cameras to capture the heat patterns emitted by the face. This technology can work effectively in low-light environments and is less affected by variations in facial appearance.

4. Mobile Face Recognition: Mobile face recognition systems are designed to work on smartphones or tablets. They use the device's front-facing camera to capture and analyze faces for attendance tracking.

5. Cloud-based Face Recognition: Cloud-based face recognition systems store and process face data on remote servers. This allows for scalability, centralized management, and access from multiple locations.

These are just a few examples of the types of face recognition attendance systems available. Each type has its own advantages and may be suitable for different environments and requirements.

# 1.2 Tools &Technologies:

****

**Figure 2**

1. OpenCV: OpenCV is an open-source computer vision library that provides a wide range of functions for image and video processing. It is widely used in face recognition systems for pre-processing images, detecting faces, and extracting features.
2. Face\_recognition: Face\_recognition is a Python library that uses deep learning to recognize faces in images and videos. It provides a simple and easy-to-use API for face recognition, and it can be used with OpenCV to build face recognition systems.
3. Flask: Flask is a lightweight web framework for Python that can be used to build web applications. It can be used to create a web interface for a face recognition system, allowing users to upload images and view the results of the face recognition algorithm.
4. Visual Studio: Visual Studio is an integrated development environment (IDE) for building applications with various programming languages, including Python. It provides a range of tools for debugging, testing, and deploying applications.

**1.3 Process:**

To develop a face recognition attendance system using visual Studio, Python, and Flask, you can follow these general steps:

****

**Figure 3**

1. Set up your development environment: Install Visual Studio, Python, and Flask on your computer.

2. Collect and prepare a dataset: Gather a diverse dataset of images containing faces of the individuals you want to recognize. Ensure the dataset represents different lighting conditions, angles, and facial expressions.

3. Train a face recognition model: Use Python libraries like OpenCV to train a face recognition model using the collected dataset. This involves extracting facial features and creating a unique representation, or embedding, for each face.

4. Build a Flask application: Use Flask, a Python web framework, to create a web application for the attendance system. Set up routes and views to handle requests and responses.

5. Integrate face recognition: Incorporate the trained face recognition model into your Flask application. Use the model to detect and recognize faces in images or video streams.

6. Capture and process images: Implement functionality to capture images from a camera. Process these images by detecting faces and matching them against the stored embeddings.

7. Deployment: Deploy your Flask application on a server or cloud platform to make it accessible to users.

# 

# 1.4 Applications:

A face recognition attendance system has various applications in different industries and settings. Here are a few examples:

1. Educational Institutions: Face recognition attendance systems can be used in schools, colleges, and universities to automate attendance tracking. It eliminates the need for manual roll calls, reduces administrative workload, and provides accurate records.

2. Corporate Offices: Face recognition attendance systems can be implemented in offices to streamline employee attendance management. It ensures accurate time tracking, reduces buddy punching (fraudulent clocking in/out), and simplifies payroll processing.

3. Events and Conferences: Face recognition attendance systems can be used at large events and conferences to manage attendee registration and check-ins. It provides a seamless and efficient entry process, enhances security, and improves crowd management.

4. Government Institutions: Face recognition attendance systems can be utilized in government institutions to monitor employee attendance and ensure accountability. It helps in preventing time theft and maintaining accurate records.

5. Healthcare Facilities: Face recognition attendance systems can be employed in hospitals, clinics, and healthcare facilities to track the attendance of healthcare professionals. It ensures accurate scheduling, improves workflow management, and enhances patient safety.

6. Access Control and Security: Face recognition attendance systems can be integrated with access control systems to grant or deny entry based on recognized faces. It enhances security by preventing unauthorized access and reduces the reliance on traditional access cards or passwords.

These are just a few examples of the applications of face recognition attendance systems. The technology has a wide range of potential uses, and its adoption continues to expand across various industries.

**1.5 Objectives:**

1. Automation: The system aims to automate the attendance process, eliminating the need for manual tracking.
2. Accuracy: By utilizing face recognition technology, the system can accurately identify and authenticate individuals, reducing the chances of errors or fraudulent attendance.
3. Time and Effort Savings: Implementing this system can save time and effort for both students/employees and administrators, as it eliminates the need for manual attendance taking.
4. Visual Studio: Visual Studio provides a powerful development environment for creating applications, making it a suitable choice for building the system.
5. Python: Python is a popular programming language for implementing face recognition algorithms due to its simplicity and extensive libraries.
6. Flask: Flask, a lightweight web framework, can be used to create a user-friendly web interface for the attendance system.

# WORKING OF FACE RECOGNITION ATTENDANCE SYSTEM PROJECT

# main.py

# This code defines a GUI application for a face recognition attendance system using the tkinter library in Python. Here are the steps involved in the code:

# 

**Figure 2.1**

# Import necessary libraries:

# tkinter and ttk for creating the GUI.

# tkinter.messagebox for displaying messages.

# PIL for working with images.

# time and datetime for displaying the current time.

# os for opening files and directories.

# student and Attendance for handling student details and attendance data respectively.

# webbrowser for opening the face recognition web application.

# app for running the Flask server.

# Define the Face\_recognition\_system class:

# Initialize the root window with a specified size and title.

# Load and display images for the GUI using the PIL library.

# Create a label for displaying the current time and update it every second using the time function.

# Create buttons for different functionalities such as student details, face detection, student entry, photos, attendance, and exit.

# Define functions for handling button clicks, such as opening files and directories, displaying messages, and running the Flask server.

# Define functions for handling button clicks:

# open\_img function opens the directory containing student photos.

# open\_attendance function opens the directory containing attendance data.

# iExit function displays a message box asking the user for confirmation before exiting the application.

# Student\_Details function opens a new window for displaying student details.

# attendance\_data function opens a new window for displaying attendance data.

# app\_data function opens the face recognition web application in a browser and runs the Flask server.

# Run the application:

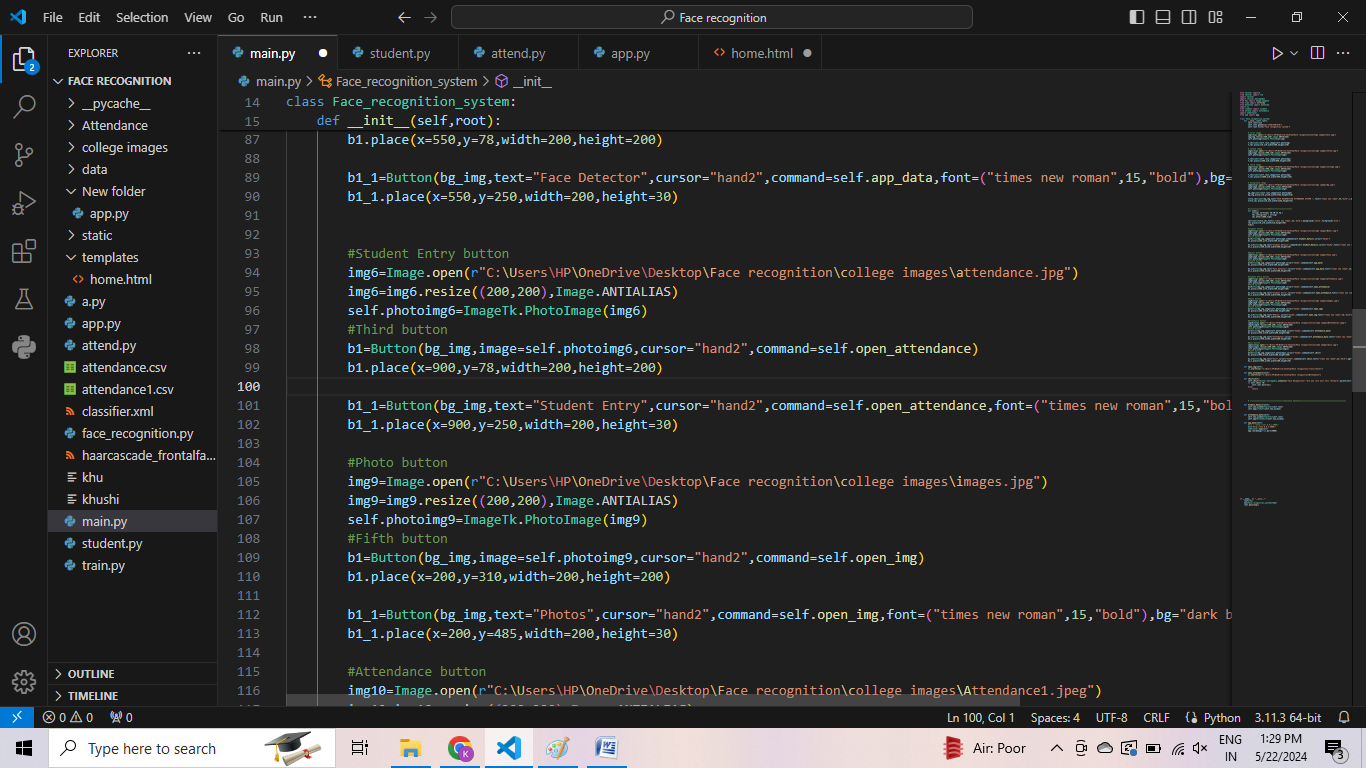
# Create an instance of the Face\_recognition\_system class and start the main event loop using the mainloop() method.

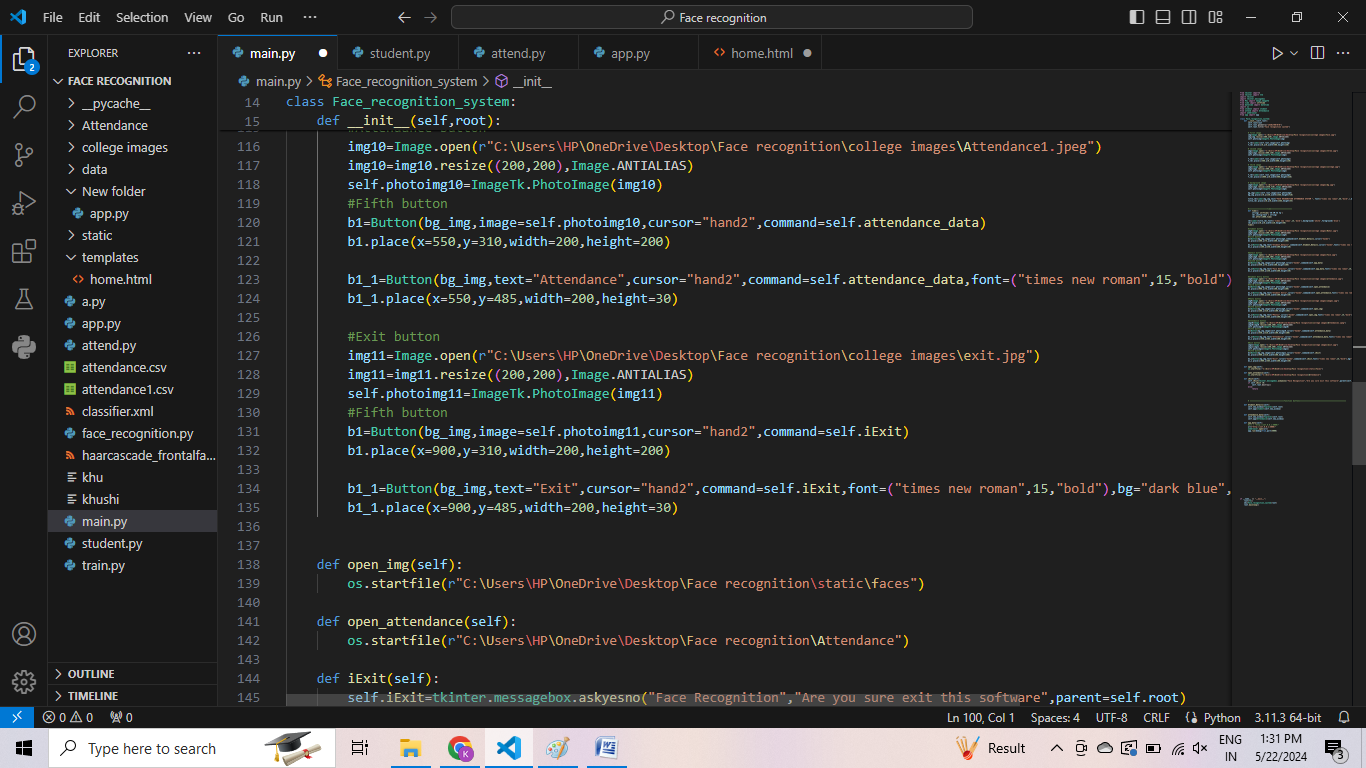
# Overall, this code provides a user-friendly interface for a face recognition attendance system and integrates various functionalities, such as student details, face detection, and attendance records.

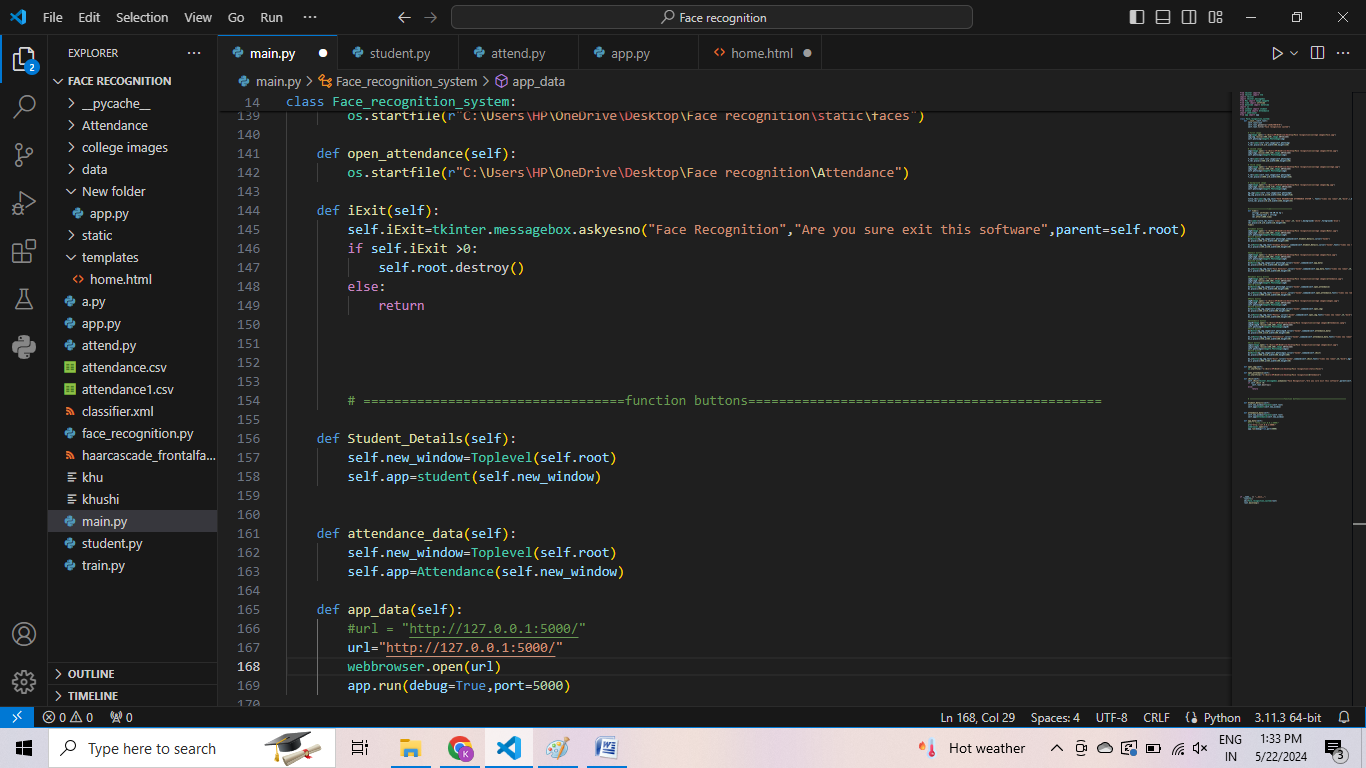
# 2024-05-22.png

# 2024-05-22 (1).png

# 2024-05-22 (3).png

****

****

****

# student.py

# This code defines a class student that creates a graphical user interface (GUI) for managing student information in a school or college using the tkinter library in Python.

# 

**Figure 2.2**

# The \_init\_ method initializes the GUI with a background image and several frames and labels for displaying and entering student information.

# The method loads several images using the PIL library and places them in various frames.

# The method creates several labels and entry fields for entering student information, such as name, father's name, mother's name, gender, phone number, date of birth, email, address, and teacher's name.

# The method creates radio buttons for selecting whether or not to take a photo sample of the student.

# The method creates several buttons for saving, updating, deleting, and resetting student information.

# The method creates a search frame with a dropdown menu and entry field for searching for student information based on a specific attribute.

# The method creates a table frame with a scrollbar and a table for displaying student information.

# The method defines several methods for adding, updating, deleting, resetting, and searching student information using SQL queries.

# The method fetches student information from a MySQL database and displays it in the table.

# The method handles user input and updates the table accordingly.

# Overall, this code provides a user-friendly interface for managing student information and integrates various functionalities, such as searching, adding, updating, and deleting student records.

# 2024-05-22 (26).png

# Attendance.py

# This code creates a GUI-based attendance management system that allows the user to view and manage attendance data in a MySQL database. The system provides options to import and export data to and from CSV files, update attendance records, and reset the form. The system also displays student details in a separate table.

# 2024-05-22 (27).png

**Figure 2.3**

# The code imports several libraries, including tkinter for creating the GUI, mysql.connector for connecting to the MySQL database, csv for working with CSV files, and messagebox for displaying messages to the user.

# The global variable mydata is defined to store the data imported from a CSV file.

# The Attendance class is defined to create the main window of the application and manage the attendance data.

# The \_init\_ method of the Attendance class is called when a new instance of the class is created. It initializes the main window, sets its size and title, and creates several variables to store the attendance data.

# The code loads two images and sets them as the background and header of the main window.

# A label is created to display the title of the application.

# A frame is created to hold the left and right sections of the application.

# A frame is created to hold the attendance details section.

# Several labels and entries are created to allow the user to enter the attendance details.

# A frame is created to hold the attendance report table.

# A scrollbar is created to allow the user to scroll through the attendance report table.

# The attendance report table is created to display the attendance data.

# A frame is created to hold the buttons for importing and exporting CSV files, updating the attendance data, and resetting the form.

# Several buttons are created to allow the user to perform various actions.

# A frame is created to hold the student details section.

# A frame is created to hold the student report table.

# A scrollbar is created to allow the user to scroll through the student report table.

# The student report table is created to display the student data.

# Several methods are defined to handle the actions performed by the user.

# The application is run by creating an instance of the Attendance class and calling its mainloop method.

# 1.png

# face\_recognition.py

# This is a Flask web application for face recognition based attendance system.

# Import necessary libraries such as cv2, os, Flask, request, render\_template, date, datetime, numpy, sklearn, pandas, joblib, and webbrowser.

# Initialize the Flask app and set the number of images to be captured for each user.

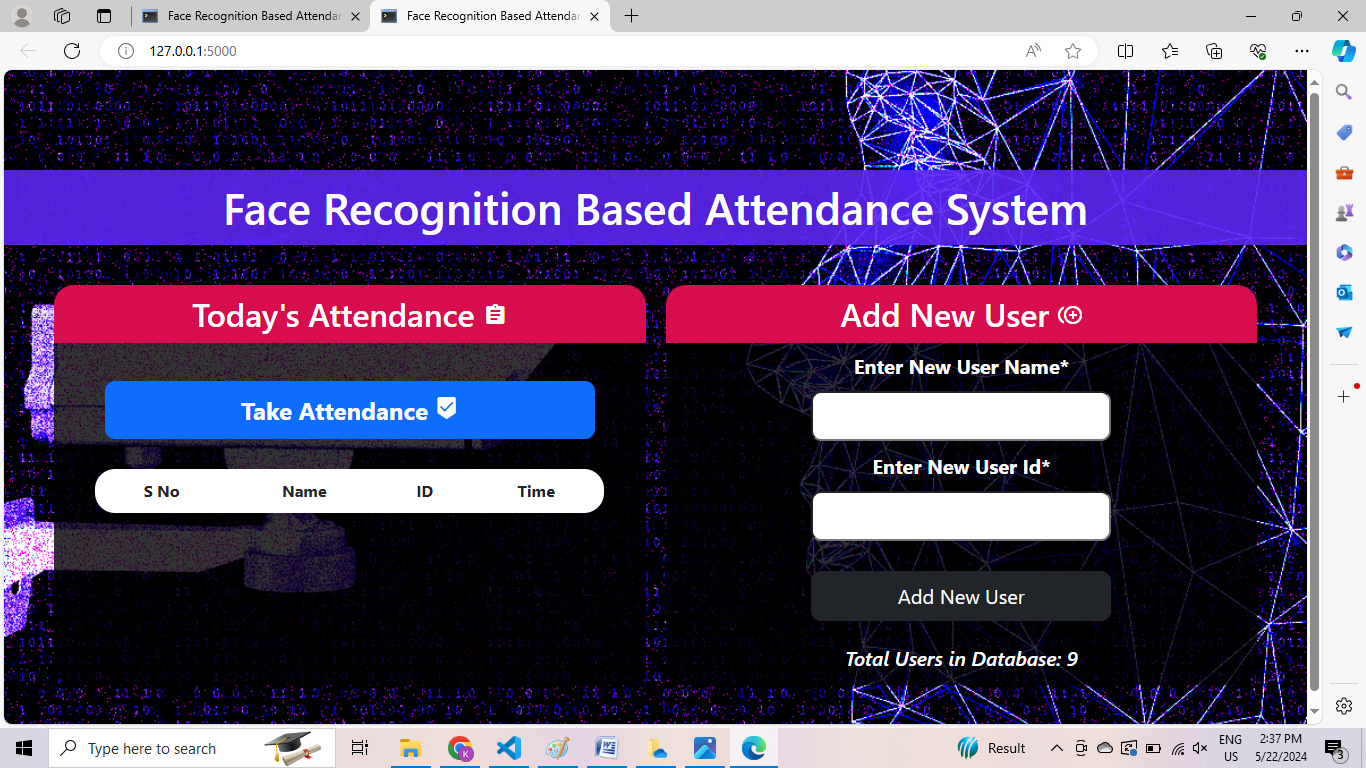
# Load the background image and face detector model.

# Create a directory for attendance and faces if it doesn't exist.

# Check if the attendance file for the current date exists, if not create one.

# Define a function to extract faces from an image.

# Define a function to identify the face using the trained model.

**Figure 2.4**

# Define a function to train the model using the faces in the 'static/faces' directory.

# Define a function to extract attendance data from the CSV file.

# Define a function to add attendance data to the CSV file.

# Define a function to remove attendance data from the CSV file.

# Define a function to get all the users in the 'static/faces' directory.

# Define the home route to render the home page with attendance data.

# Define the start route to capture faces and add attendance data.

# Define the add route to add a new user and capture their faces.

# Run the Flask app and open the URL in the web browser.

# Images: Attendance Record:

# 2024-05-22 (34).png2024-05-22 (33).png

# CONCLUSION

# In conclusion, the Face Recognition Attendance System using Machine Learning is an efficient and accurate way to manage attendance records. The system utilizes Python, Flask, and Visual Studio for development, and OpenCV for face detection and recognition. The system allows students to register by providing their details and uploading their photo, which is securely stored in a database. When a student wants to mark their attendance, the system uses face recognition to verify their identity, and the attendance records are updated in real-time. The system also includes an "Exit" button to monitor the duration of attendance. The Face Recognition Attendance System is a valuable tool for educational institutions, corporate offices, events, government institutions, healthcare facilities, and access control and security. The system automates the attendance process, reduces errors and fraudulent attendance, saves time and effort, and provides accurate records. Overall, the Face Recognition Attendance System using Machine Learning is a reliable and efficient solution for managing attendance records.

# REFERENCES

* <https://www.youtube.com/watch?v=mYVS7QjNjvg&list=PLMnmAHlVrrJJ5bzj07Fr-xGOqEMRHzaGX&index=1>
* <https://www.mysql.com/downloads/>
* <https://code.visualstudio.com/>
* https://youtu.be/\_EDoqeS0etE?si=uGI0qhpxA1PTgfqP