Project Tile
Face Recognition System By

Akash Kumar Soni

REG NO: 221030039

A PROJECT REPORT

Submitted

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF COMPUTER APPLICATIONS

CENTRE FOR OPEN AND DIGITAL EDUCATION

June 2022

HINDUSTAN ONLINE
CENTRE FOR OPEN AND DIGITAL EDUCATION
CODE

HINDUSTAN
INSTITUTE OF TECHNOLOGY & SCIENCE
(DEEMED TO BE UNIVERSITY)

BONAFIDE CERTIFICATE

Certified that this project report entitled "**Face Recognition System"** is the bonafide work of Akash Kumar Soni **(REG. NO. 221030039),** who carried out the project work under my supervision. Certified further that to the best of my knowledge, the work reported here does not from part of any other project or dissertation work on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE **SIGNATURE**

HEAD OF THE DEPARTMENT
Centre for Open and Digital Education
Hindustan Institute of Technology and Science, Padur, Chennai

SUPERVISOR
Department of Computer Applications
Hindustan Institute of Technology and Science, Padur, Chennai

The Project Phase I Viva – Voce Examination is held on _____

INTERNALEXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

First and foremost, praises and thanks to God the Almighty, for the blessings throughout my project work to complete the project successfully.

I would like to express my gratitude to all the authorities who were instrumental in making this project a success.

I am thankful to the Head of the Department, CODE for their support and encouragement to complete my work.

I am highly grateful to my supervisor for stimulating guidance, help, and useful suggestions throughout the present work.

I would like to express my gratitude to the project coordinator for providing valuable comments and hints to improve my project work.

I would like to dedicate this project to my amazingly loving and supportive family, who have always been with me, no matter where I am.

TABLE OF CONTENTS

CHAPTER	TITLE		PAGE NO
	ABSTRACT		I
	LIST OF TABLES		iii
	LIST OF FIGURES		iii
1.	**INTRODUCTION**		
	1.1	Organization Profile	1
	1.2	Project Description	2
2.	**SYSTEM ANALYSIS**		
	2.1	Existing System	
	2.2	Proposed System	
	2.3	Feasibility Study	
3.	**SYSTEM DESIGN**		
	3.1	System Architecture	
	3.2	Data Flow Diagram	
	3.3	ER Diagram	
	3.4	Database Design	
	3.5	Input Design	
	3.6	Output Design	
	3.7	Algorithm Explanation	
4.	**MODULE DESCRIPTION**		
5.	**SYSTEM REQUIREMENTS**		
	5.1	Software and Hardware Requirements	
	5.2	Software Explanation	
6.	**TESTING AND IMPLEMENTATION**		
	6.1	System Testing	
	6.2	Implementation	
7.	**RESULTS AND CONCLUSION**		
	7.1 Results		
	7.2 Conclusion		
8.	**FUTURE ENHANCEMENT**		
9.	**APPENDIX**		

	9.1	Screenshots	
	9.2	Source Code	
	9.3 References		

ABSTRACT

In a world where students still roam around with pen-and-paper attendance registers (yes, in 2025), maintaining accurate attendance manually is… let's just say "not ideal." This project proposes a Face Recognition-Based Attendance System that automates the entire process without requiring teachers to call out names like a school roll-call ceremony.

The system uses **OpenCV**, **LBPH face recognition**, and a clean **SQLite backend** to detect a student's face in real-time and instantly mark attendance with their USN (because names can be duplicated, but USNs rarely betray us). A simple menu-driven interface allows administrators to import students via Excel, capture images, train the model, view attendance, and export reports.

This project not only reduces human effort (and human error) but also gives a modern touch to attendance systems used in colleges. The end result? A fast, accurate, and smart attendance system that makes the traditional register look like a museum artifact.

CHAPTER 1 — INTRODUCTION

1.1 Project Overview

Attendance tracking is one of the most repetitive and time-consuming activities in any educational institution. Traditional methods are inefficient, prone to proxy attendance, and often inaccurate. The proposed system uses **face recognition technology** to automate this process by detecting and identifying a student's face and marking attendance automatically.

1.2 Problem Definition

- Manual attendance wastes classroom time
- Proxy attendance (a.k.a "roll-call fraud") is common
- Registers get lost, damaged, or mismanaged
- Data entry into digital systems is time-consuming

The goal is to overcome these issues through automation while maintaining accuracy and reliability.

1.3 Objectives

- Automate attendance marking using face recognition
- Store student data and attendance records digitally

- Provide easy Excel-based student import
- Offer simple UI for non-technical staff
- Generate attendance reports (daily/weekly/monthly)

1.4 Scope

This system works for classrooms, labs, seminars, and even online sessions (with slight modifications). It is scalable and can handle large datasets as long as the dataset images are clean and training data is adequate.

CHAPTER 2 — SYSTEM ANALYSIS

2.1 Existing System

- Manual attendance marking
- Excel sheets maintained by faculty
- Susceptible to manipulation
- No centralized data
- Very time-consuming

2.2 Proposed System

A face recognition-based attendance system that:

- Detects and identifies student faces
- Marks attendance automatically
- Stores data securely in SQLite
- Provides reports instantly

2.3 Feasibility Study

• Technical Feasibility

Uses Python, OpenCV — both open source. Runs on mid-spec laptops.

• Economic Feasibility

No additional hardware required. Cost-effective.

• Operational Feasibility

Simple UI; can be used by non-technical people easily.

CHAPTER 3 — SYSTEM DESIGN

3.1 System Architecture / Data Flow Diagram

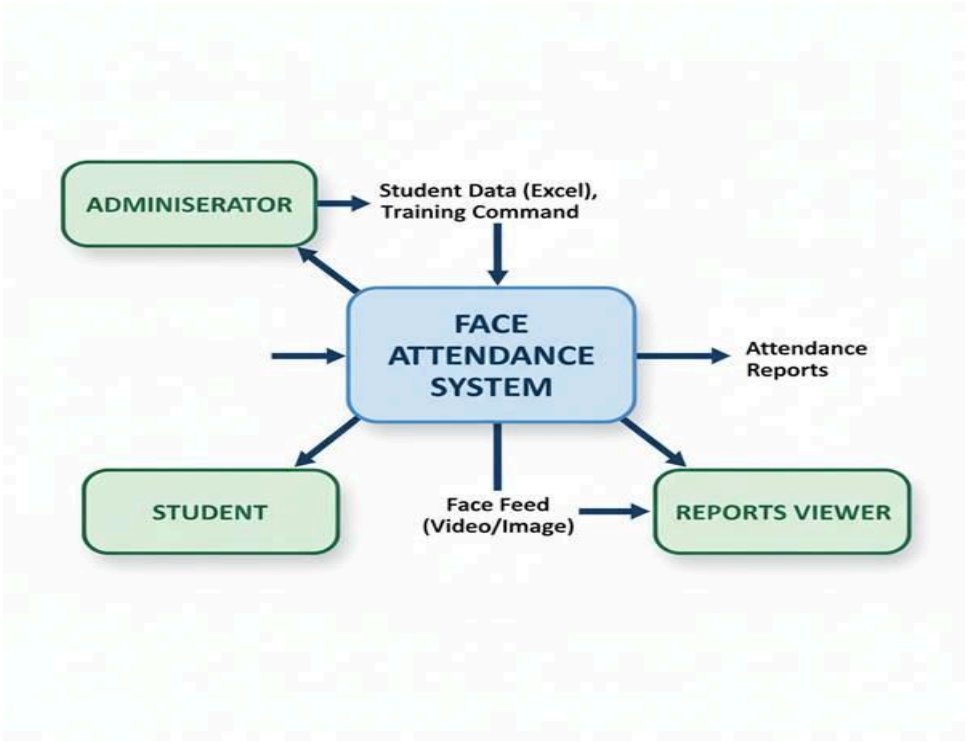
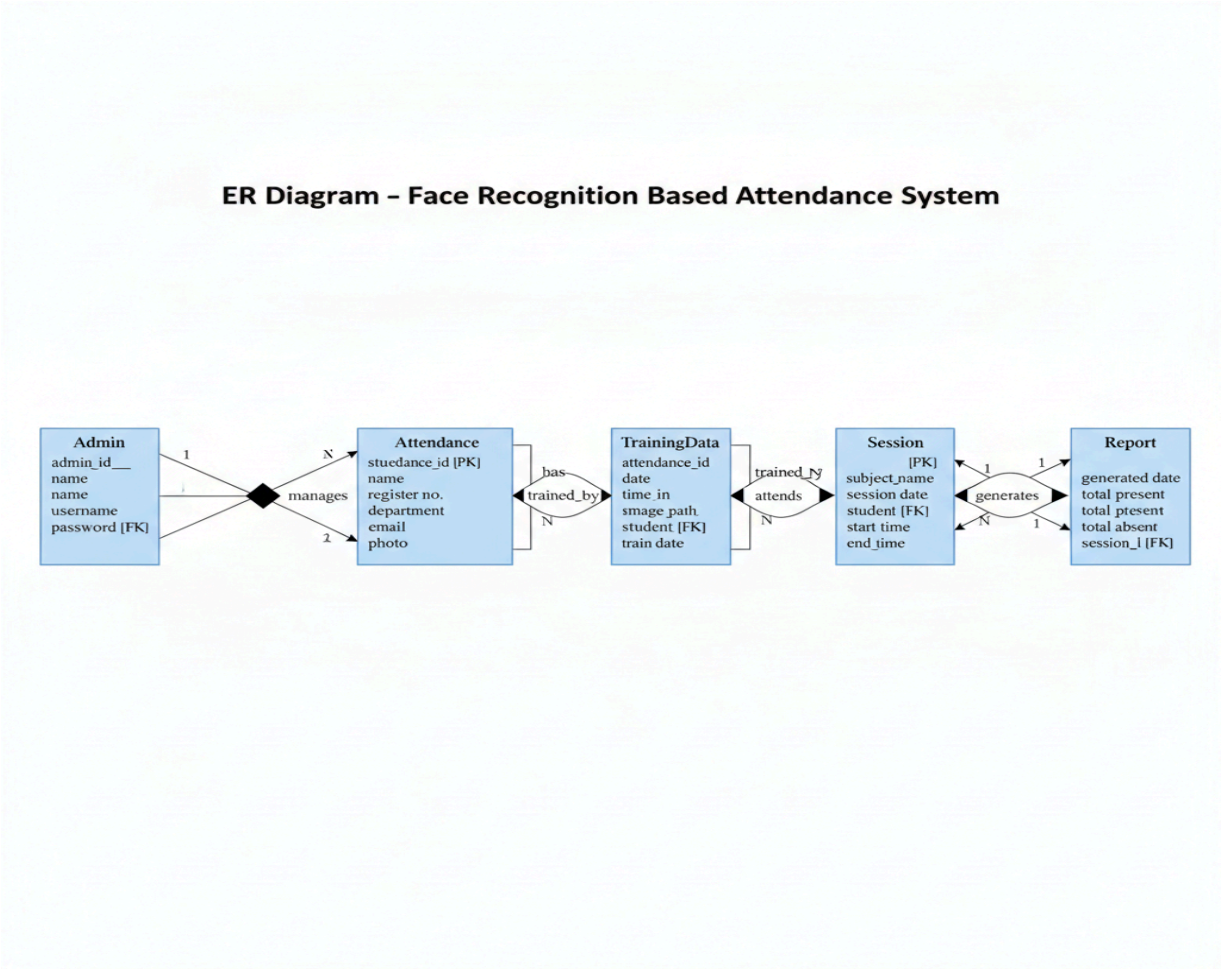


Figure 3.1: Data Flow Diagram of Face Attendance System

3.2 ER Diagram



CHAPTER 4 — MODULE DESCRIPTION

4.1 Database Module

- Uses SQLite
- Creates `students` and `attendance` tables
- Stores student details with USN as primary key
- Handles insertion and retrieval

4.2 Image Collection Module

- Captures 30 images per student
- Stores them in dataset folders
- Ensures dataset uniformity

4.3 Training Module

- Scans dataset folders

- Detects faces
- Trains LBPH model
- Stores labels mapping in `labels.pkl`

4.4 Recognition Module

- Captures live frames
- Detects face using Haar Cascade
- Predicts label using LBPH
- Marks attendance automatically

4.5 Excel Import Module

- Reads `student_details.xlsx`
- Imports name, USN, department, section
- Auto-creates dataset folders

4.6 Attendance Viewer

- View All Attendance
- View by Status (Present/Absent)

4.7 Report Export Module

Exports attendance as:

- Daily
- Weekly
- Monthly
- Custom Range

Formats: Excel or CSV

4.8 Main Menu Module

Your app's brain.
Presents interactive options to use all features easily.

CHAPTER 5 — SYSTEM IMPLEMENTATION

5.1 Technologies Used

- Python
- OpenCV
- LBPH Algorithm
- SQLite
- Pandas (for Excel operations)
- OS Module
- Pickle
- Basic file I/O

5.2 Hardware Requirements

- Laptop with webcam
- 4GB+ RAM recommended

5.3 Software Requirements

- Python 3.x
- OpenCV
- SQLite
- Pandas

CHAPTER 6 — SYSTEM TESTING

6.1 Testing Methods

- Unit Testing
- Functional Testing
- Integration Testing

- User Testing

6.2 Test Cases Example

Test case	Input	Expected	Result
Face Recognition	Student face	Name+usn displayed	**Pass**
Attendance marking	Recognized USN	Attendance Inserted	Pass
Import Excel	.xlsx file	Students added	Pass
Capture Images	Webcam	30 Images saved	Pass

CHAPTER 7 — RESULTS & DISCUSSIONS

- System successfully detects student faces
- Marks attendance automatically
- Good accuracy with clean training images
- LBPH model performs reliably even on mid-range laptops
- Minor challenges include lighting conditions and camera angle

CHAPTER 8 — ADVANTAGES & LIMITATIONS

Advantages

- Fully automated
- Fast & accurate
- No proxy attendance
- Easy database management
- Lightweight and deployable

Limitations

- Dependent on camera quality
- Poor lighting affects detection
- Requires clean training dataset
- Model must be retrained for new students

CHAPTER 9 — FUTURE ENHANCEMENTS

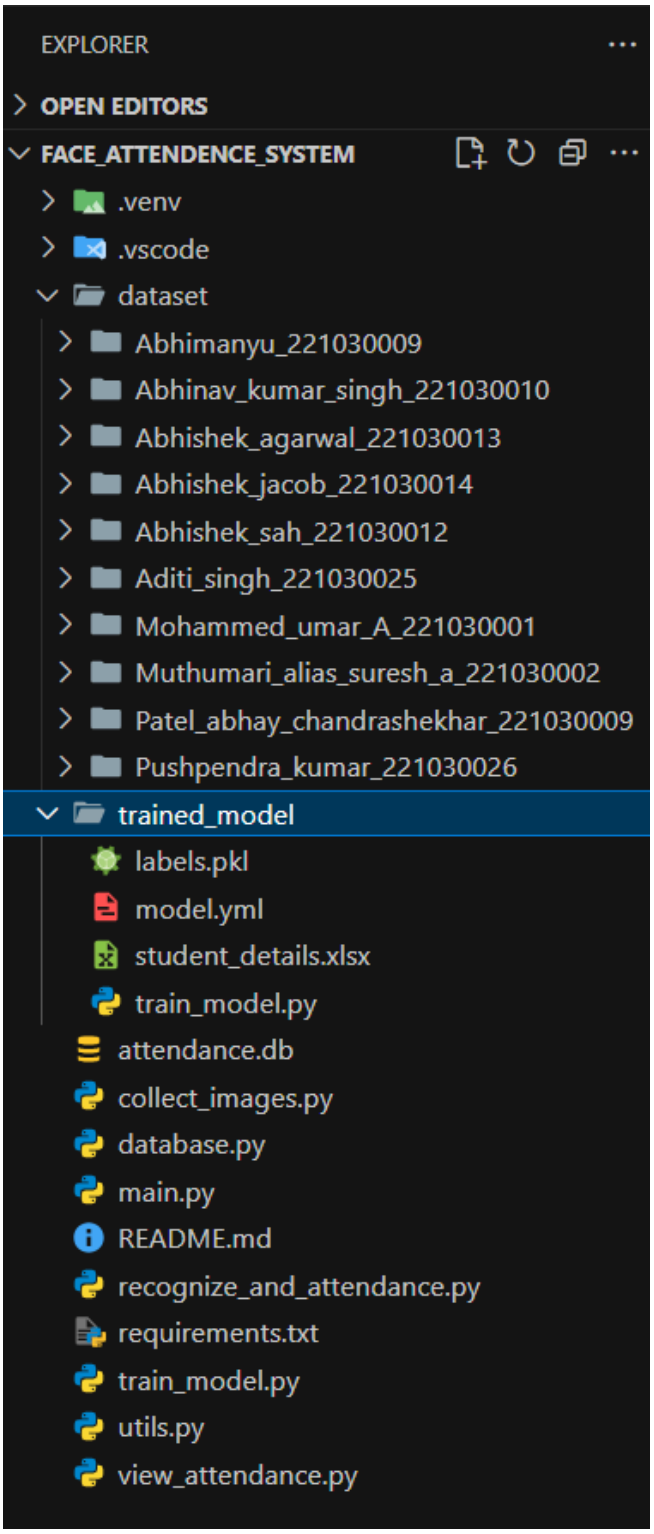
- Mobile app integration
- Multi-camera setup
- Cloud-based face recognition
- Attendance dashboard analytics
- Integration with LMS/ERP
- Masked face recognition

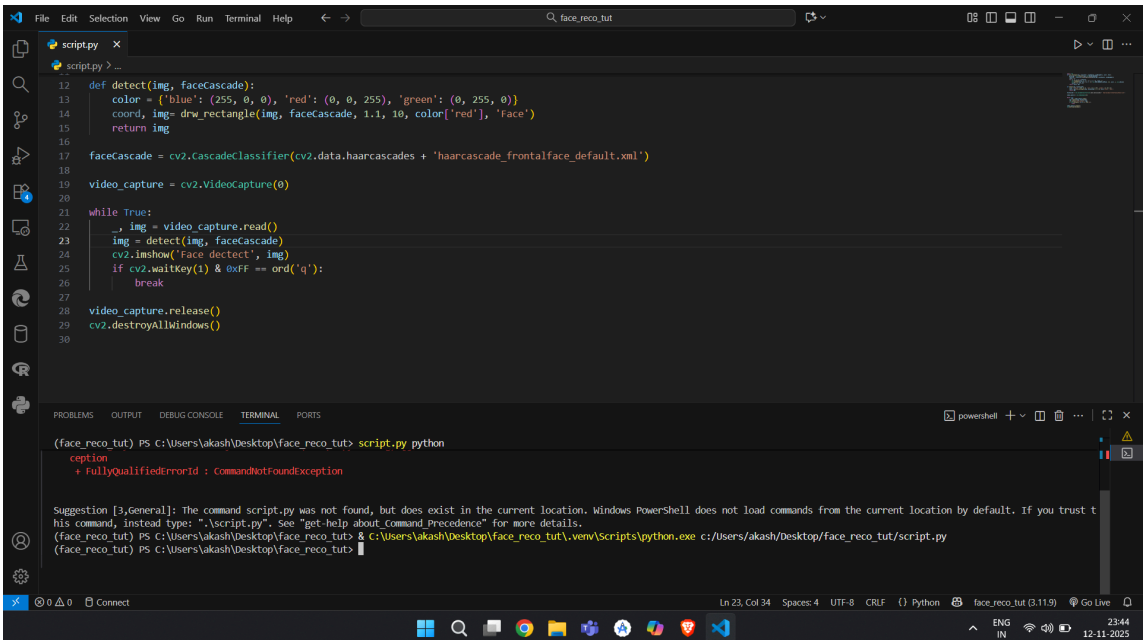
CONCLUSION

The Face Recognition-Based Attendance System provides a modern, efficient, and highly reliable alternative to traditional attendance methods. By incorporating LBPH, SQLite, and a simple interactive interface, the system ensures accuracy, eliminates proxies, and significantly reduces manual workload.

9. APPENDIX

Screenshots of file structure, source code, and attendance sheet by the modal.
You will get the code here in the provided drive link
Source_codes titled-2025-10-31-2142.png
And if you want to see the working of my project, I am attaching another link for your reference.
Watch Video Recording 2025-11-22 123410.mp4


EXPLORER
OPEN EDITORS
FACE_ATTENDENCE_SYSTEM
.venv
.vscode
dataset
Abhimanyu_221030009
Abhinav_kumar_singh_221030010
Abhishek_agarwal_221030013
Abhishek_jacob_221030014
Abhishek_sah_221030012
Aditi_singh_221030025
Mohammed_umar_A_221030001
Muthumari_alias_suresh_a_221030002
Patel_abhay_chandrashekhar_221030009
Pushpendra_kumar_221030026
trained_model
labels.pkl
model.yml
student_details.xlsx
train_model.py
attendance.db
collect_images.py
database.py
main.py
README.md
recognize_and_attendance.py
requirements.txt
train_model.py
utils.py
view_attendance.py


File Edit Selection View Go Run Terminal Help
face_reco_tut
script.py
script.py > ...
12 def detect(img, faceCascade):
13 color = {'blue': (255, 0, 0), 'red': (0, 0, 255), 'green': (0, 255, 0)}
14 coord, img= drw_rectangle(img, faceCascade, 1.1, 10, color['red'], 'Face')
15 return img
16
17 faceCascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
18
19 video_capture = cv2.VideoCapture(0)
20
21 while True:
22 _, img = video_capture.read()
23 img = detect(img, faceCascade)
24 cv2.imshow('Face dectect', img)
25 if cv2.waitKey(1) & 0xFF == ord('q'):
26 break
27
28 video_capture.release()
29 cv2.destroyAllWindows()
30
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
powershell
(face_reco_tut) PS C:\Users\akash\Desktop\face_reco_tut> script.py python
ception
+ FullyQualifiedErrorId : CommandNotFoundException
Suggestion [3,General]: The command script.py was not found, but does exist in the current location. Windows PowerShell does not load commands from the current location by default. If you trust t
his command, instead type: ".\script.py". See "get-help about_Command_Precedence" for more details.
(face_reco_tut) PS C:\Users\akash\Desktop\face_reco_tut> & C:\Users\akash\Desktop\face_reco_tut\.venv\Scripts\python.exe c:/Users/akash/Desktop/face_reco_tut/script.py
(face_reco_tut) PS C:\Users\akash\Desktop\face_reco_tut>
0 0 Connect
Ln 23, Col 34 Spaces: 4 UTF-8 CRLF {} Python face_reco_tut (3.11.9) Go Live
ENG
IN
23:44
12-11-2025