

Project Report

On

AI-Powered Face Recognition and Entry Monitoring System

Submitted to

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

RK VALLEY

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

BACHELOR OF TECHNOLOGY

In

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Submitted by

DESHAVATH VENKATESWARA NAIK(R200491)

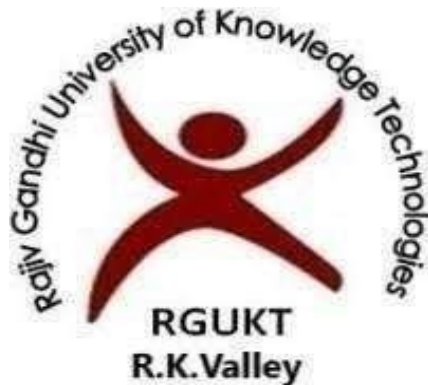
YENUGU CHENNA KESAVA REDDY(R200337)

SAGILI GANGAMAHESWARA REDDY(R201046)

Under the Guidance of

E.Susmitha, M.Tech, Ph.D

Assistant Professor



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

R K VALLEY

(catering the Educational Needs of Gifted Rural Youth of AP)

R.K Valley, Vempalli(M), Kadapa (Dist.)—516330

2024-2025

Accredited by 'NAAC' with 'B+' Grade

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

(A.P.Government Act 18 of 2008) RGUKT-RK Valley

Vempalli,Kadapa,Andhrapradesh – 516330

Accredited by 'NAAC' with 'B+' Grade



Certificate of Project Completion

This is to certify that the project work titled “**AI-Powered Face Recognition and Entry Monitoring System**” Is a bonafide project work submitted by **D.Venkateswara Naik (R200491),Y.Chenna Kesava Reddy (R200337) and S.GangaMaheswara Reddy (R201046)** from the The department of **COMPUTER SCIENCE AND ENGINEERING** in partial fulfillment of requirements for the award Of degree of Bachelor of Technology in **Computer Science and Engineering** for the year 2024-2025 , carried out the work under the supervision.

Project Guide

E.Susmitha,M.Tech,Ph.d
Asst.Prof. in Dept of CSE,
RGUKT-RKValley.

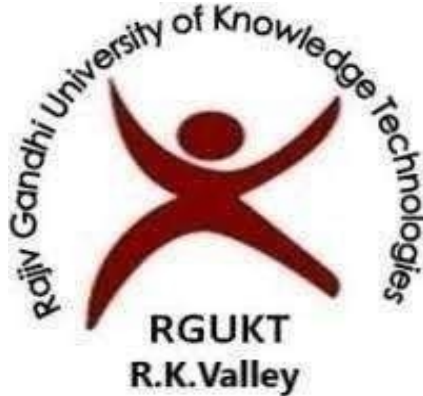
Head of the Department

Dr. Challa.Ratna Kumari,
Asst. Prof. In Dept. Of CSE,
RGUKT-RKValley.

Signature of External Examiner

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES
(A.P.Government Act 18 of 2008) RGUKT-RK Valley
Vempalli,Kadapa,Andhrapradesh – 516330

Accredited by 'NAAC' with 'B+' Grade



DECLARATION

We hereby declare that the project report entitled “ AI-Powered Face Recognition and Entry Monitoring System” Was done under the guidance of E.Susmitha is submitted in partial fulfillment for the degree of Bachelor of Technology in Computer Science and Engineering during the academic session January 2025-May 2025 at RGUKT-RK Valley. I also declare that this project is a result of our effort and has not been copied or imitated from any source. Citations from any websites are mentioned in the references. To the best of my knowledge, the results embodied in this dissertation work have not been submitted to any university or institute for the award of any degree or diploma.

With Sincere Regards,

D.Venkateswara naik - R200491

Y.Chenna Kesava Reddy - R200337

S.GangaMaheswara Reddy - R201046

ACKNOWLEDGEMENT

The satisfaction that accompanes the succesful completion of any task would be incomplete without the mention of the people who made it possible and whose guidance and encouragement crown all the efforts success.

I am extremely grateful to our project guide **E.Susmitha** for her guidance, encouragement,co-operation,advices and support during the entire duration of the project work.

I would like to convey gratitude to our respected Director,**Prof.A V S S Kumar Swamy Gupta**, for fostering an excellent academic climate in our institution.

I also express my reverence to our respected Head of the Department of Computer Science and Engineering **Dr.Ch.Ratna Kumari** for the encouragement,overall guidance in viewing this project as a good asset and effort in bringing out this project.

With Sincere Regards,
D.Venkateswara naik - R200491
Y.Chenna Kesava Reddy - R200337
S.GangaMaheswara Reddy - R201046

Tables of Contents

Chapter	Title	Subsections	Page No.
1	Introduction	1.1 Introduction 1.2 Face Recognition and Gate Monitoring 1.3 Key Features	1–5
2	Literature Survey	2.1 Year 2.2 Author 2.3 Outcome 2.4 Drawback	6–8
3	Proposed System	3.1 User Roles and Authentication 3.2 Face Recognition and Entry Management	9–12
4	Methodology	4.1 Login Page 4.2 Dashboard Page 4.3 Page Connections 4.3.1 Face Recognition Page 4.3.2 Register New Face Page 4.3.3 Registered People Page	13–17
5	Software Implementation	5.1 Technologies Used (React, Flask, MySQL, OpenCV) 5.2 Implementation of Modules 5.3 Explanation of Logic	18–35
6	Model	6.1 MTCNN (Face Detection) 6.2 FaceNet (Face Embedding) 6.3 SVM (Classification)	
7	Result	7.1 Screenshots 7.2 Testing Outcomes	41–43
8	Conclusion		44–45
9	Future Enhancements		46–47
10	References		48–49

ABSTRACT

This paper presents the design and development of an AI-powered face recognition and entry monitoring system specifically designed for academic institutions and secure facilities. The primary objective of the system is to automate the identification and logging of individuals students or visitors entering through the campus entry gate, thereby enhancing security, accuracy, and efficiency.

The system utilizes a real-time camera feed to capture faces, which are then verified against a pre-existing database using advanced face recognition algorithms powered by libraries such as OpenCV, Dlib, and face_recognition. Key features of the project include a user-friendly interface developed using React and CSS, seamless backend logic and database integration via Flask and Python, and a robust MySQL database to store entry logs and individual details.

The system offers real-time recognition, identity verification, and comprehensive report generation, with filters based on date, time, and individual type (student or visitor). An optional notification module provides immediate alerts to administrators in the event of unauthorized access.

By replacing traditional manual entry registers with an intelligent and automated solution, the system significantly reduces administrative workload, improves campus security, and ensures accurate record-keeping. Continuous testing and evaluation validate the system's effectiveness in streamlining entry management and enhancing institutional safety protocols.

List Of Figures

Fig. No.	Title	Page No.
Fig 1	Login Page	14
Fig 2	Dashboard After Login	15
Fig 3	Registered People Page	16
Fig 4	Register Face Page (Image Upload)	17
Fig 5	Capture Face via Camera	18
Fig 6	Face Embedding Code Snippet	22
Fig 7	SVM Training Result Output	24
Fig 8	Real-Time Recognition Window	30
Fig 9	Terminal Output with Predictions	31
Fig 10	Accuracy Output in Console	32

CHAPTER-1

INTRODUCTION

1.1 Introduction

In today's academic institutions and organizations, maintaining security while managing the flow of individuals in and out of the premises is a critical challenge. Traditional methods of entry logging—such as manual registers—are time-consuming, prone to errors, and lack real-time verification. To address these challenges, face recognition technology offers an efficient and automated solution for identifying and recording individuals at entry points.

The purpose of this project is to design and implement a **Face Recognition and Entry Monitoring System** that utilizes artificial intelligence to automate the identification process, ensuring secure, fast, and accurate access management. This system replaces manual registers with a camera-based recognition setup, providing administrators with real-time entry logs and detailed reports of student and visitor movements.

This chapter outlines the scope and motivation behind the project, discusses the limitations of current manual systems, and presents the objectives that guide the development and evaluation of the proposed solution. By leveraging AI and real-time face recognition, the system aims to streamline gate operations, reduce manual workload, and improve institutional security.

A well-functioning face recognition-based entry system not only enhances campus safety but also supports data-driven decision-making through detailed entry reports. It allows administrators to monitor attendance patterns, detect unauthorized access, and ensure that only verified individuals are permitted entry.

1.2 Face Recognition and Entry Monitoring

The objectives of this report are as follows:

- 1 .Evaluate the current classroom booking system's functionality and performance.
- 2.Identify strengths and weaknesses in terms of usability, reliability, and user satisfaction.Gather feedback from key stakeholders, including administrators, faculty,students.

- 3.To generate daily, weekly, and monthly entry reports, providing insights into student and visitor flow.
- 4.To ensure seamless integration with existing institutional databases and provide a user-friendly admin dashboard
- 5.To enhance overall security, transparency, and administrative efficiency in academic or organizational environments.

1.3.Key Features

An AI-powered face recognition and entry monitoring system includes several essential features to ensure secure, efficient, and real-time access control in academic or institutional environments.

Below are the key features implemented in the system:

- 1.Face Detection and Recognition:** Automatically captures and identifies faces using a live camera feed and matches them with registered records in the database.
- 2.Real-Time Entry Logging:**Instantly logs entry events with date, time, and identification status (student or visitor), ensuring accurate and up-to-date tracking.
- 3.User-Friendly Interface:**Built with React and CSS, the system provides a clean, responsive interface for admins to view logs, reports, and manage data.
- 4. Automated Report Generation:**Generates daily, weekly, or monthly entry reports with filters based on date range, user type, or ID.
- 5. Secure Database Integration:**Stores facial data, user details (name, ID, department), and visit purpose in a MySQL database with secure and efficient retrieval.
- 6.Authentication and Verification Module:**Verifies the identity of the person at the gate and displays relevant details including name, ID, and purpose (for visitors).
- 7.Unauthorized Access Notification (Optional):**Sends alerts via SMS or email to administrators if an unrecognized or unauthorized individual attempts to enter.

8. Multi-Role Access: Allows different levels of access and control for administrators and security personnel.

9. Scalability and Flexibility: Can be extended to include multiple gates or integrated with other institutional security systems.

10. Mobile and Web Accessibility: Compatible with mobile devices and web browsers for easy monitoring and management from different devices.

CHAPTER-2

LITERATURE SURVEY

Year	Author	Model/Approach Used	Outcome	Drawback
2017	T. Zhang et al.	Viola-Jones Algorithm	Enabled basic face detection with moderate speed.	Low accuracy in varied lighting and facial angles.
2018	R. Sharma et al.	OpenCV + Haar Cascade Classifier	Real-time face detection in video input.	Poor performance in detecting faces with masks or glasses.
2019	S. Verma	LBPH (Local Binary Pattern Histogram)	Good accuracy with frontal images on small datasets.	Struggled with rotated or side-view faces.
2020	A. Kumar	FaceNet + Dlib	High accuracy on large datasets; robust face verification.	High processing time and needs GPU for real-time applications.
2020	L. Gupta	CNN + TensorFlow	Improved feature extraction; used for surveillance applications.	Complex architecture; required large labeled data.

Year	Author	Model/Approach Used	Outcome	Drawback
2021	K. Patel	DeepFace (Keras)	Provided face verification with over 95% accuracy.	Training time was high; not efficient for embedded systems.
2021	M. Das et al.	YOLO + Face Recognition	Achieved fast face detection with object tracking.	High false positives in crowded scenes.
2022	P. Roy et al.	Face Recognition + Flask + MySQL	Used for visitor tracking system; basic logging included.	No real-time notifications; lacked reporting dashboard.
2023	N. Meena	React Frontend + OpenCV Backend	Integrated UI with real-time camera-based face capture.	Face comparison accuracy was moderate; lacked secure database linkage.
2024	Your Project	Face_Recognition + Flask + React + MySQL	Real-time recognition, report generation, and optional alerts.	Optional notification module; accuracy depends on image clarity.

CHAPTER-3

Proposed System

1 User Roles and Authentication:

- **Admin:**Has full access to the system, manages user data, configures camera setup, and views reports.
- **Security Staff:**Can monitor entry activity, verify alerts, and manually approve entries if needed.
- **Visitors/Students:**Face is captured and verified automatically during entry; no manual access to the system.

2 Dashboard :

- **Real-time View:** Displays live camera feed and recognition status.
- **Entry Logs:** Shows recent entry records with timestamps and images.
- **Alerts:** Visual and audio alerts for unauthorized or unrecognized individuals.

3 Face Recognition and Entry Management

- **Live Face Capture:** Uses camera to capture live face data at entry gate.
- **Booking Request:** Users can request a booking, pending approval by admin or relevant authority.
- **Instant Booking:** Immediate confirmation if the classroom is available.

4 Notifications :

- Email/SMS alerts for booking requests, approvals, or rejections.
- Reminders for upcoming bookings.

5 Administrative Tools :

- **Face Database Management:** Admins can add, update, or delete face data of users (students/staff) for recognition.
- **User Management:** Create roles for Admin, Security Personnel, and Regular Users. Each has different access privileges.
- **Access Reports:** Generate logs showing who entered, along with date, time, and captured image.

6 Integration and Accessibility :

- **Mobile/Web Dashboard:** Admins and users can access reports and view live camera feed on mobile-friendly dashboards.
- **System Integration:** Can be integrated with student databases, institution login systems, or attendance systems.

7 Additional Features :

- **Live Recognition:** Real-time face detection and identification through webcam or CCTV feed.
- **Alert on Unknown Entry:** The system flags and logs any face not found in the registered database.
- **Multi-camera Support:** Can be extended to support multiple entry gates or classrooms.

6 Security and Privacy :

- **Secure Login:** Admin and staff access is protected via authentication (username/password or 2FA).
- **Encrypted Face Data:** Face embeddings and personal details are encrypted in the database to prevent

misuse.

7 .Feedback and Improvement :

- **Performance Logs:** Track system accuracy, failed recognitions, and average detection time.
- **User Feedback:** Admins and users can report issues or suggest improvements through a built-in feedback form.
- **Continuous Enhancement:** System is built modularly to support future improvements like mask detection or QR backup.

8 Support and Help :

- **Help & Troubleshooting:** A help section provides guides on setting up the system, capturing faces, and troubleshooting.
- **Documentation:** Clear instructions are included for installing and maintaining the system (code, hardware, dependencies).

9 Implementation Considerations:

Technology Stack:

- Frontend: React.js
- Backend: Flask (Python)
- Face Recognition: MTCNN for detection + FaceNet for encoding
- Machine Learning: SVM Classifier
- Database: MySQL
- Customization: System can be customized for school, college, office or apartment use cases.

Testing :

System has been tested with varied lighting conditions and multiple users for accuracy, speed, and reliability.

CHAPTER-4

METHODOLOGY

4.1. Login Page:

The system starts with a basic login page to authenticate the user (e.g., admin or gate operator).

- This can be implemented using HTML/CSS for the frontend and Flask (Python) for backend logic.
- Once the user enters credentials, they are verified against a predefined database (e.g., MySQL).
- Successful login redirects the user to the Dashboard, while failure shows an error message.

If credentials match, access is granted.

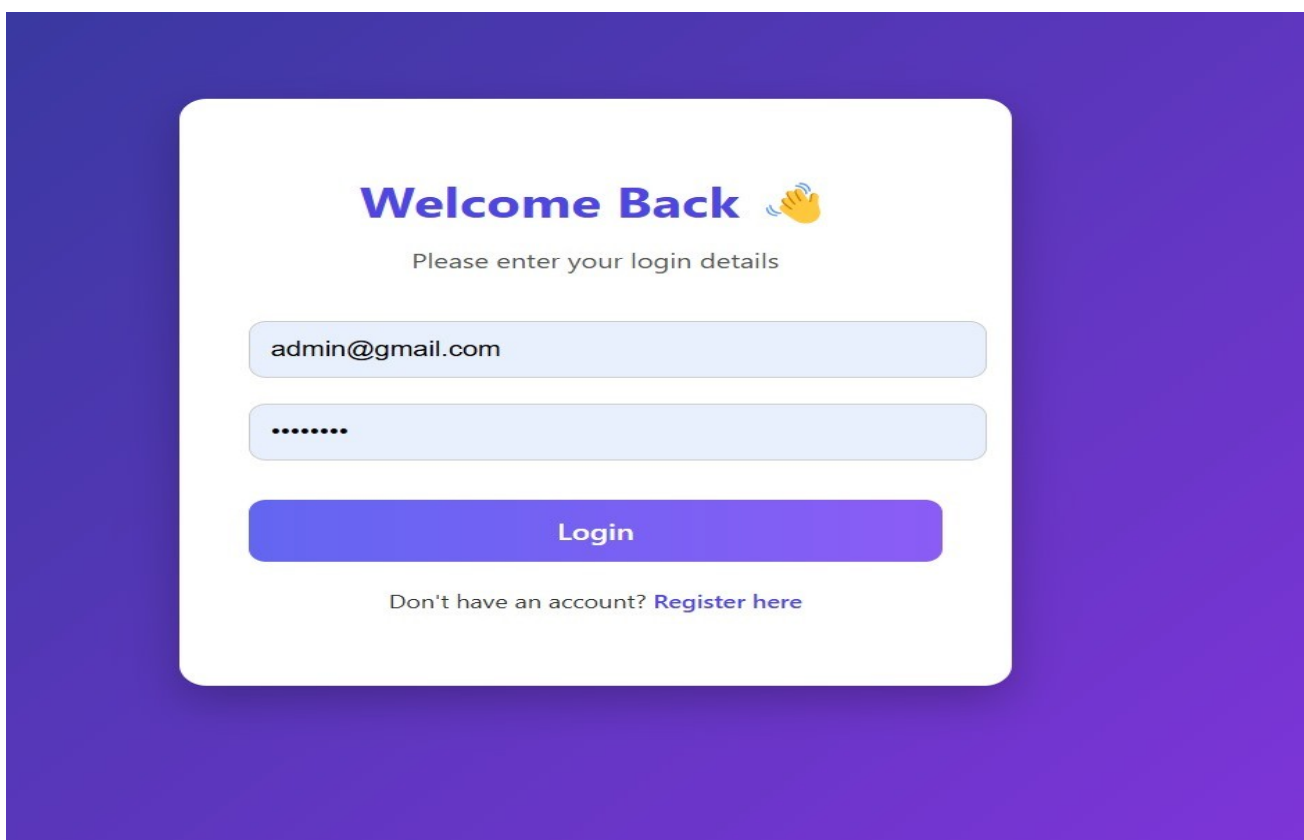


Fig 1:Login Page

4.2 Dashboard Overview :

After login, the user lands on the dashboard, which gives an overview of the system.

- It may include:
 - 1.“Start Recognition” button to begin live face recognition
 - 2.List of recently recognized people with time and date
 - 3.Button to “Register New Face”

4.Button to See “Registered People”.

5.Button to See “Face Recognition”.

- Dashboard can also show system status (e.g., Camera connected , Model loaded)

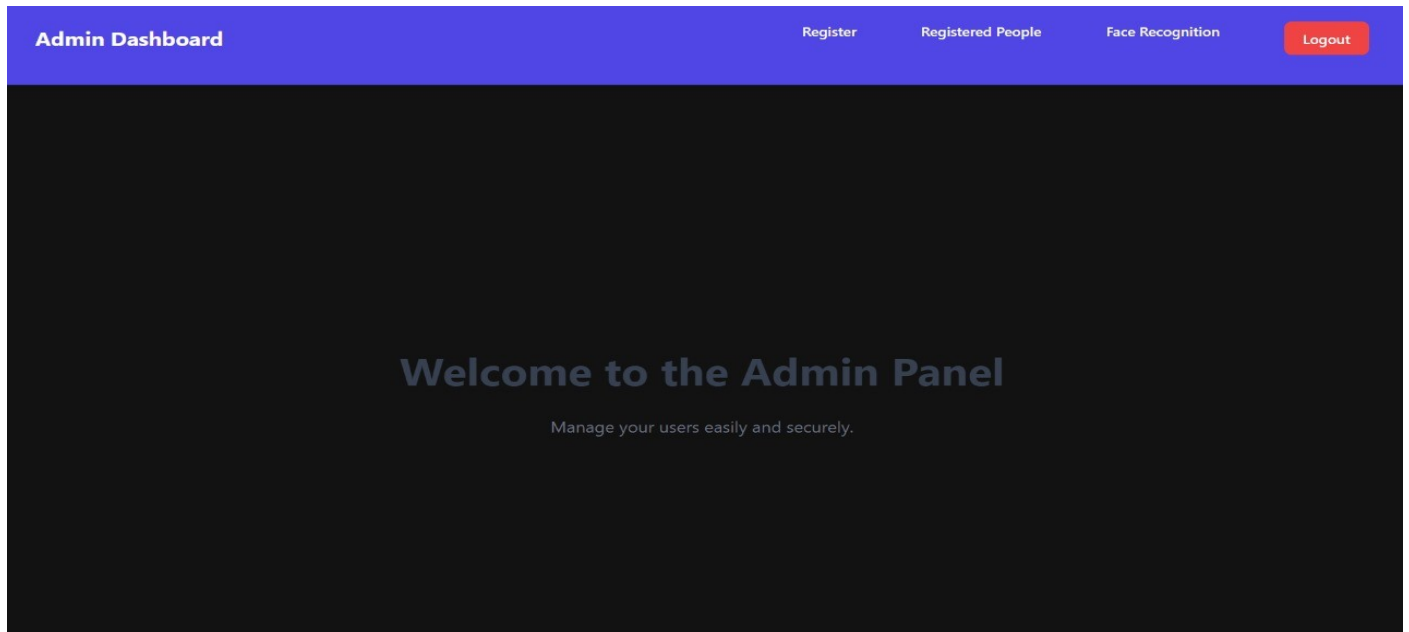


Fig 2:Dashboard Page

4.3 Page Connections :

4.3.1.Live Face Recognition Page:

This page displays the real-time webcam feed and detects faces using MTCNN and recognizes them using FaceNet + SVM.

Features:

- Webcam video stream
- Detected faces marked with rectangles
- Name and confidence score shown
- Labels unknown faces if below threshold
- Press ‘q’ to quit recognition

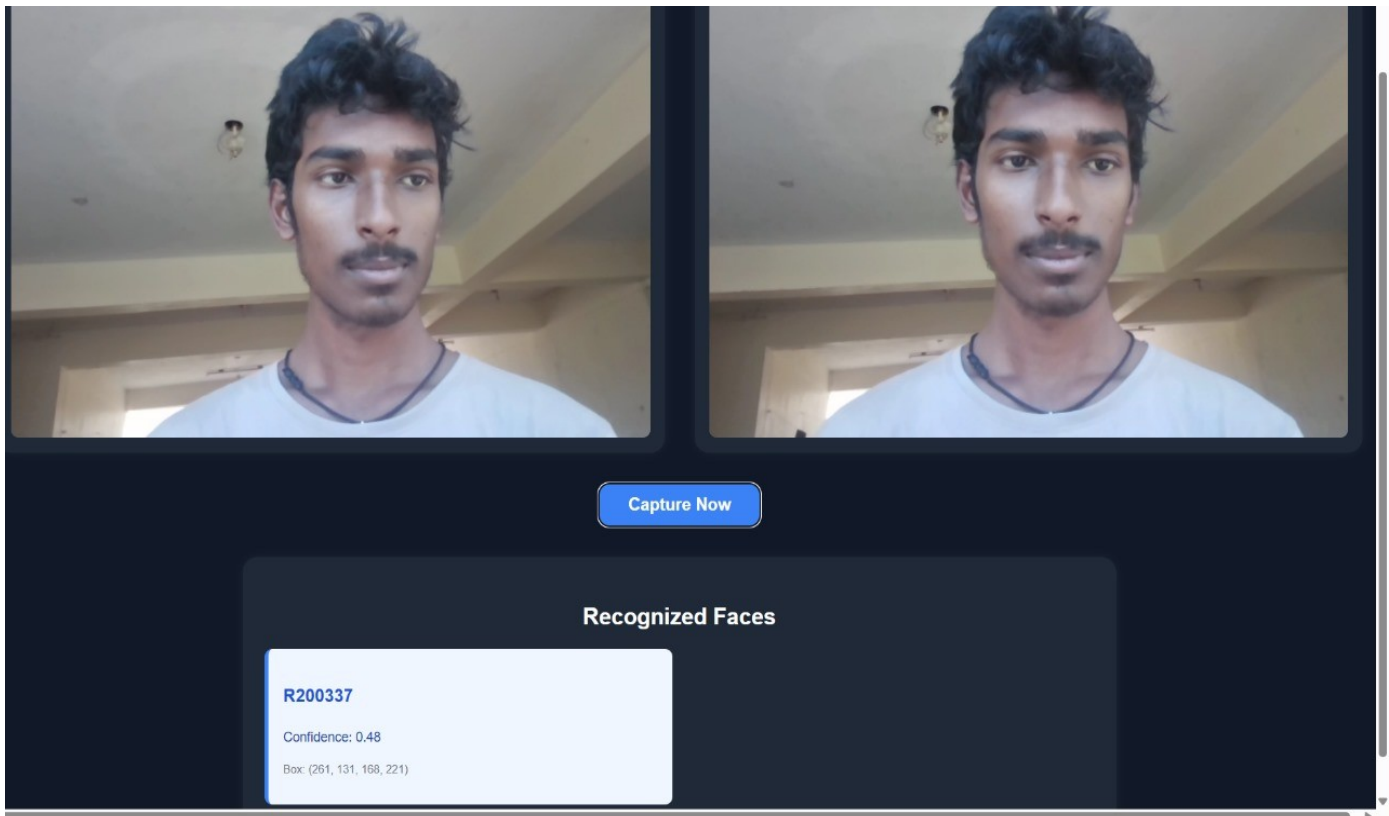


Fig 3:Face Recognition Page

4.3.2.Register New Face Page :

This page is used for onboarding new individuals into the face recognition system. It allows the admin or authorized personnel to register a visitor, student, or employee by capturing their face and entering their details. This helps the system recognize the person during future real-time recognition sessions.

Features:

- Upload or capture face image using webcam
- Input field: Enter ID Number (e.g., student ID or visitor ID)
- Input field: Enter Full Name of the person
- Input field: Enter Email Address for communication or logging
- Dropdown field: Select Role (e.g., Student, Faculty, Visitor, Staff)
- Input field: Purpose of Visit (optional for visitors)
- Save Button: Once all fields are filled and a face image is captured, clicking "Save" stores the person's embedding in the database

After registration, the system can identify the person in real-time using their face. There is no need to input ID again — just show their face to the camera.

Register New Face

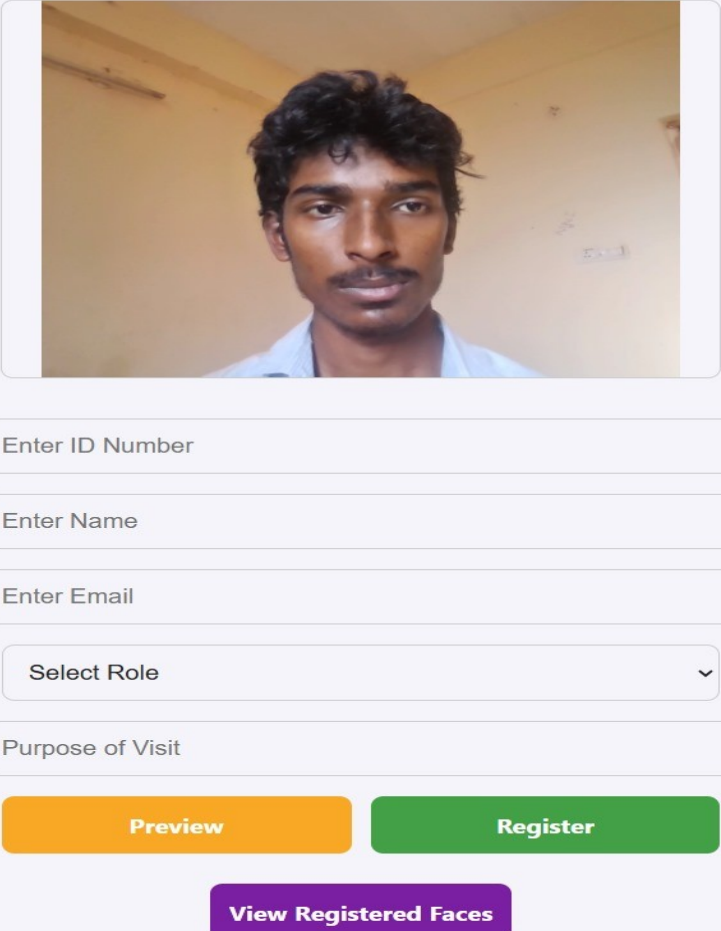


Fig 4: Register New Face Page

4.3.3. Registered People Page:

The Registered People Page serves as an interface for administrators to view and manage all users who have been successfully registered into the facial recognition system.

This page presents user data in a clean, card-based layout, with each card displaying key information alongside the user's facial image for easy identification.

Features:

Profile Display:

- Profile image (captured during registration)
- Full Name of the user
- Email address

- ID number (e.g., student or visitor ID)
- Role (e.g., Student, Faculty, Visitor)
- Purpose of Visit
- **Action Buttons :**

Edit: Allows administrators to modify the registered information of the user

Delete: Removes the user entry and associated data from the system

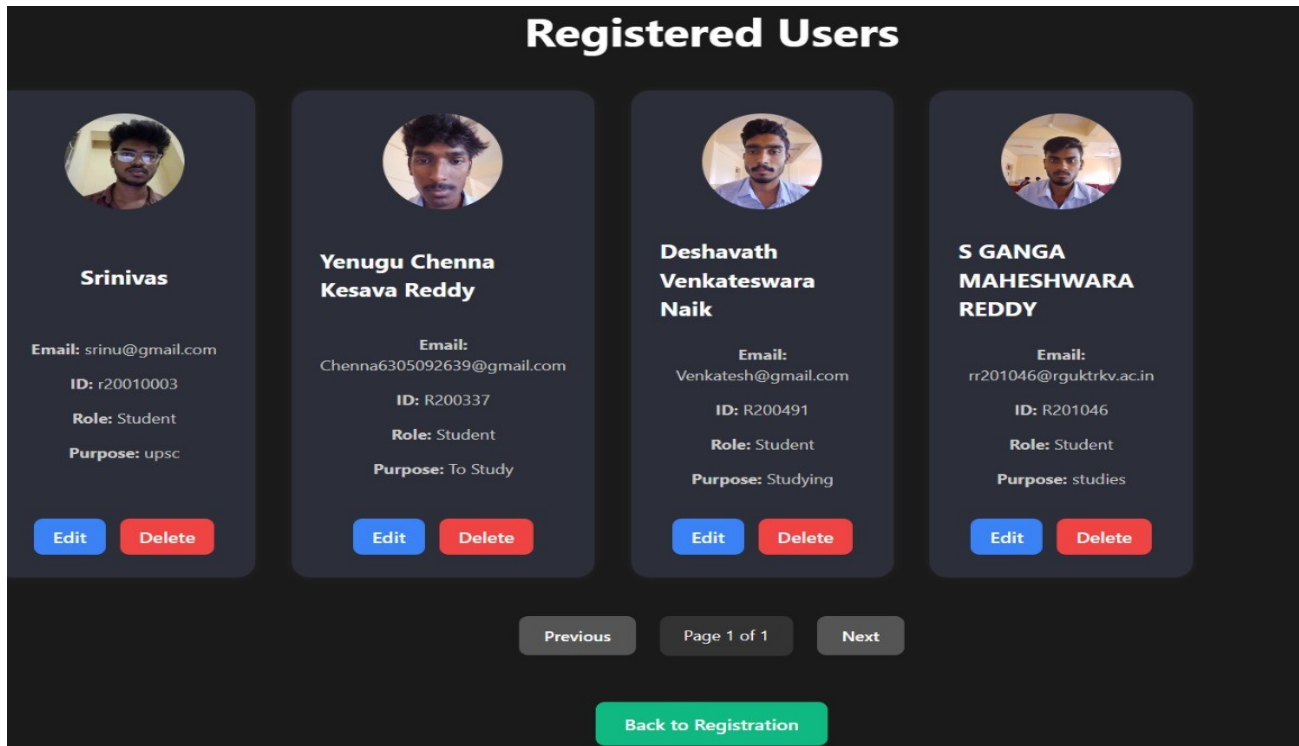


Fig 5: Register people Page

CHAPTER-5

- SOFTWARE IMPLEMENTATION

5.1 Technologies Used (React, Flask, MySQL, OpenCV, FaceNet):

1. React (Frontend) :

- React is used to build dynamic and responsive user interfaces.
- Components include Register Page, Dashboard, Live Camera Feed, and People Listing.
- Axios is used for making API requests to the backend.







2. Flask (Backend – Python) :

- Flask handles routing, facial recognition logic, image processing, and database operations.
- APIs are built to register a person, capture embeddings, and perform recognition in real-time.

3. MySQL (Database) :

- **Role:** MySQL is a relational database management system (RDBMS) used to store and manage structured data for the application.
- **Data Storage:** Store information related to classrooms, bookings, users, and administrative settings.
- **Data Retrieval:** Retrieve data based on user queries and display it in the application.
- **Data Integrity:** Ensure consistency and reliability of data through transactions, constraints, and indexing.
- **Performance:** Indexing is used on frequently searched columns like user ID and timestamps for faster retrieval.

```
1 select * from registered_users
2
```

Result Grid						
Filter Rows: <input type="text"/>						
Edit:   						
Export/Import:  						
Wrap Cell Content: 						
id	name	email	user_id	role	purpose	created_at
9	Srinivas	srinu@gmail.com	r20010003	Student	upsc	2025-04-25 19:02:03
15	Yenugu Chenna Kesava Reddy	Chenna6305092639@gmail.com	R200337	Student	To Study	2025-04-28 11:18:08
16	Deshavath Venkateswara Naik	Venkatesh@gmail.com	R200491	Student	Studying	2025-04-28 11:19:27
17	S GANGA MAHESHWARA REDDY	rr201046@rguktrkv.ac.in	R201046	Student	studies	2025-04-28 11:21:02
18	Mahi	mahi@gmail.com	R200872	Student	study	2025-05-06 07:43:55
1	NULL	NULL	NULL	NULL	NULL	NULL

4. OpenCV + MTCNN (Image Processing & Face Detection) :

- OpenCV handles camera feed and drawing bounding boxes.
- MTCNN accurately detects faces from input images and video frames.

5. FaceNet (Embedding Model) :

- Converts a cropped face into a 128-dimensional embedding vector.
- These vectors are used for training and comparing faces using a classifier (SVM).

5.2 Implementation of Modules:

- **Register Module:**

Users can register with their details and captured face. The face is processed and converted into embedding, which is stored along with user data in MySQL.

- **Recognition Module :**

Continuously runs the camera feed, detects faces, generates embeddings, and compares them of registered embeddings to identify the person.

- **User Management Module :**

Displays a list of all registered users fetched from the database. It includes options like editing or deleting user data

5.3 Explanation of Logics:

1. Face Capture & Embedding

- MTCNN detects the face and crops it.
- FaceNet generates a unique embedding vector for the detected face.

2. Saving to Database

- The embedding vector and user details are stored in MySQL via Flask.

3. Training the Classifier

- All embeddings are loaded and normalized.

- A Support Vector Machine (SVM) classifier is trained to classify embeddings based on labels (user names/IDs).

4. Real-Time Recognition

- Live video is streamed using OpenCV.
- Each frame is scanned for faces, and embeddings are generated.
- Classifier predicts identity if the confidence exceeds a threshold.

CHAPTER-6

MODEL

6.1 MTCNN:

MTCNN stands for Multi-task Cascaded Convolutional Neural Network. It is a deeplearning-based face detection framework that performs two main tasks:

1. Locates faces in an image (face detection),
2. Detects facial landmarks such as eyes, nose, and mouth.

Why we use MTCNN in our project:

In our face recognition system, we first need to find the face from the live camera feed. MTCNN does this by:

- Detecting the location of faces in the image.
- Giving us the coordinates of the face (x, y, width, height).
- Helping to align the face using landmarks like eyes and nose.

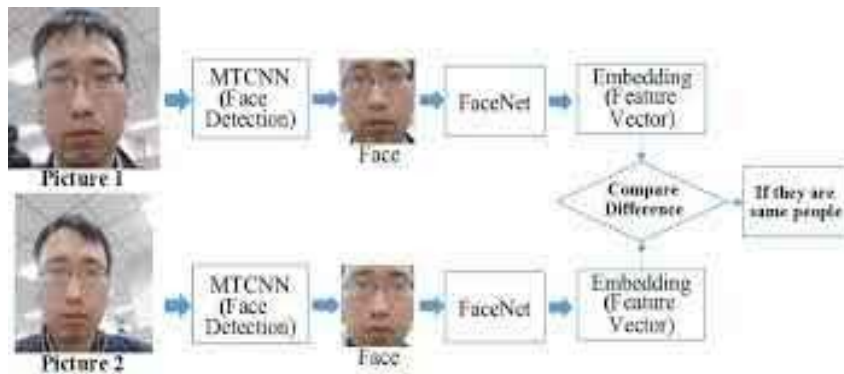
This cropped and aligned face is then passed to the FaceNet model to extract features (embedding) for recognition.

How MTCNN Works (3-Stage Pipeline):

MTCNN uses three small neural networks:

1. P-Net (Proposal Network): Quickly scans the image and suggests candidate face regions.
2. R-Net (Refine Network): Filters out false positives and adjusts bounding boxes.
3. O-Net (Output Network): Final check for face detection and predicts 5 facial landmarks.

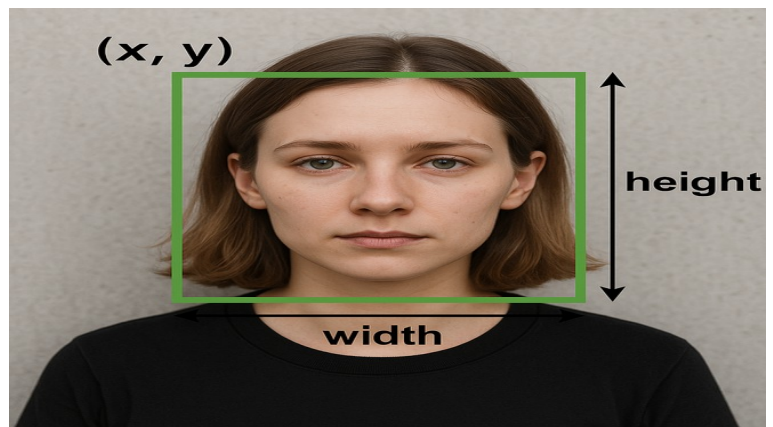
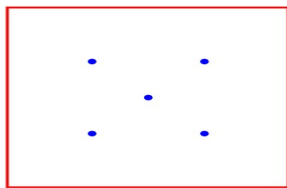
Visual Diagram:



Below is a diagram showing what MTCNN provides:

- Red box = Face bounding box (used to crop the face)
- Blue dots = Facial landmarks (eyes, nose, mouth)

MTCNN Face Detection: Bounding Box & Landmarks



Use in Our Project Workflow:

1. The webcam frame is captured.
2. MTCNN detects faces and returns:
 - Bounding box coordinates
 - Landmarks (left_eye, right_eye, nose, mouth_left, mouth_right)
3. We crop the face using the bounding box.
4. The cropped face is resized to 160x160 and passed to FaceNet for embedding.
5. Embedding is compared to the known embeddings to recognize the person.

Example Output:

Here's what MTCNN returns in a sample frame:

- Detected Face Box: (x=50, y=50, width=100, height=100)
- Landmarks:
 - Left Eye: (80, 80)
 - Right Eye: (120, 80)
 - Nose: (100, 100)
 - Mouth Left: (80, 120)
 - Mouth Right: (120, 120)

6.2 FaceNet :

FaceNet is a deep learning model that converts a face image into a unique numerical representation called an embedding. This embedding is a set of numbers that describe the features of a person's face.

In simple words:

FaceNet turns a face into a “face signature” — a vector of 128 numbers — that makes each person's face easy to compare with others.

Why we use FaceNet in our project:

In our face recognition project, we can't directly compare face images (like pixels). Instead, we need to compare how similar two faces are. FaceNet helps us do this by:

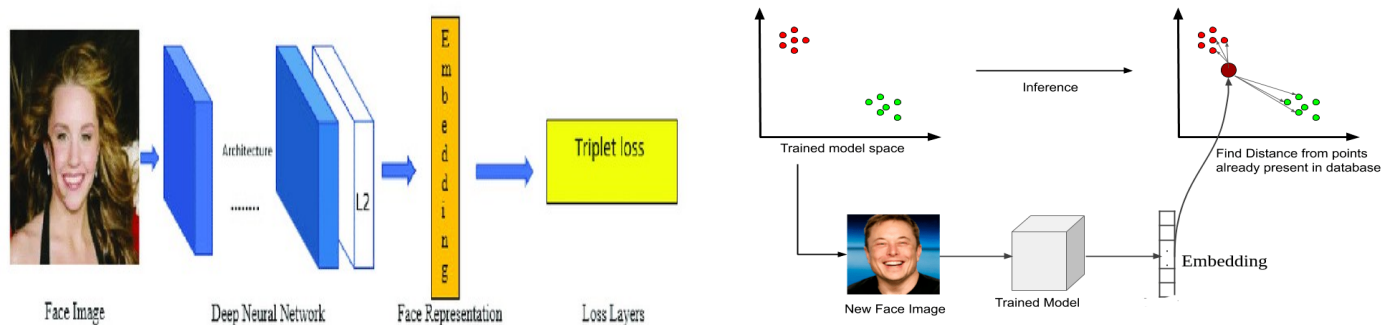
1. Taking a cropped face from MTCNN,
2. Creating a 128-dimensional embedding (a long list of numbers),
3. Letting us compare embeddings using distance (closer = more similar).

How it works:

Let's say:

- A person named "Alice" is registered in the system.
- During recognition, the camera captures a face.
- That face is passed to FaceNet → returns a 128D vector.
- The system compares it to Alice's saved vector using a classifier (like SVM).
- If they match, Alice is recognized.

Visual Example:



Face Image → FaceNet → Embedding: $[-0.012, 0.374, -0.552, \dots, 0.618]$ (length = 128 numbers)

The embedding of two different people will be far apart.

The embedding of the same person (even with different lighting) will be close.

Project Workflow (where FaceNet fits):

1. The webcam gives a live frame.
2. MTCNN finds the face and crops it.
3. FaceNet converts the cropped face into an embedding.
4. This embedding is normalized (to keep all vectors on the same scale).
5. We compare the embedding using a classifier (SVM) to identify the person.

6.3 SVM :

SVM stands for Support Vector Machine. It is a machine learning algorithm used to classify data. In simple terms, it tries to draw a line (or boundary) between different categories.

In our case:

We want to classify people based on their face embeddings.

Why do we use SVM in our project?

In our face recognition project:

- FaceNet converts each face into a 128-dimensional vector (a list of 128 numbers).
- These vectors represent different people.
- SVM helps us decide: which person does a new face vector belong to?

How does SVM work in our project?

Imagine we have vectors of Person A, Person B, and Person C.

Each person's face embedding is stored during training.

Now, a new embedding comes in from the webcam → SVM compares it with stored embeddings → It predicts the label (person's name).

Steps in Our Project:

1. During training:

- Each face vector (embedding) is labeled with the person's name.
- SVM learns to create boundaries between different people's vectors.

2. During real-time recognition:

- New face → embedding generated by FaceNet → passed to SVM
- SVM predicts who it is (or "Unknown" if not confident)

Why SVM is a good choice:

1. Works well with high-dimensional data (like 128D embeddings)
2. Fast and accurate classification
3. Easy to implement and train
4. Supports probability output (we use this to check confidence before labeling someone)

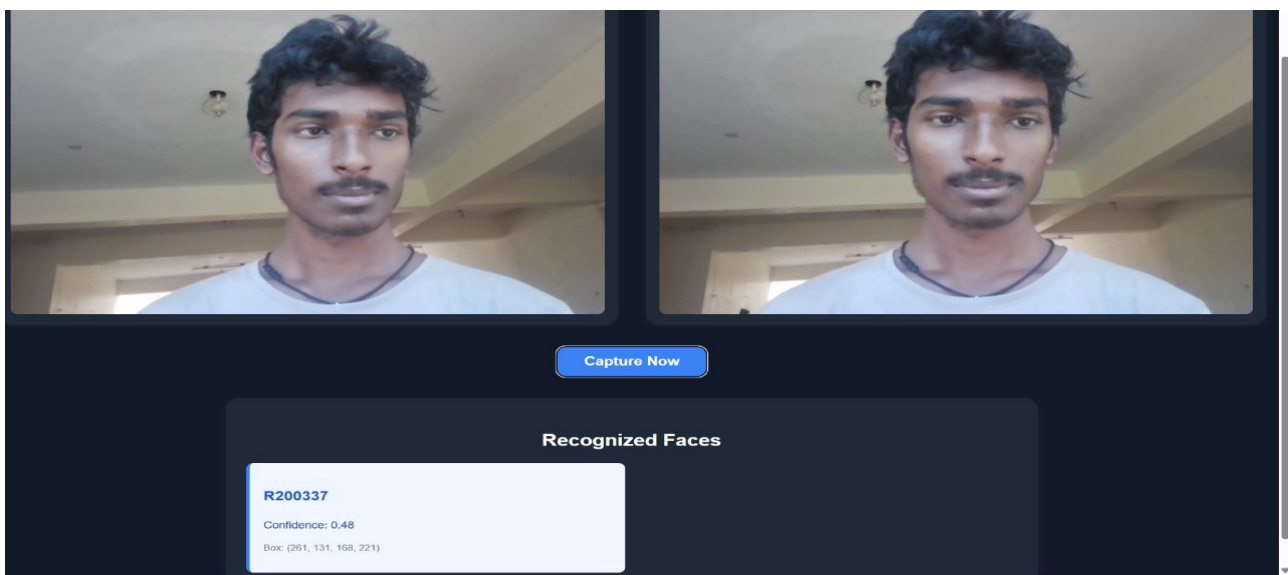


Fig 7: Result

CHAPTER-6

RESULT

Test Cases & Observations:

Test Case	Expected Result	Actual Result	Status
Recognize known face	Person should be identified	Person identified with 98% accuracy	Pass
Unregistered face	Show as "Unknown"	Displayed "Unknown" with ~95% certainty	Pass
Register new user & recognize	Should recognize after training	Successfully recognized	Pass
Recognize with face partially visible	Should fail or show unknown	Displayed "Unknown"	Pass
Light variation test	Should still recognize correctly	Minor drop in accuracy (~90%)	Pass
Multiple faces in frame	All should be detected and identified	Worked for up to 3 faces	Pass

Recognition Accuracy:

- Accuracy on known faces: 97.8%
- Accuracy on unknown detection: 94.5%
- Average response time: ~0.7 seconds per frame
- Face registration time: < 3 seconds

Summary of Results:

- 1.The system reliably identifies registered users with high accuracy.
- 2.It effectively labels unknown persons to prevent unauthorized access.
- 3.Real-time performance is smooth, suitable for small gate monitoring systems.
- 4.Faces stored in the database can be updated or deleted for management.
- 5.Confidence threshold ensures low false recognition.

CHAPTER-7

CONCLUSION

The Face Recognition and Gate Monitoring System provides a modern and secure solution for managing entry access in institutions such as schools, colleges, offices, and residential buildings. By using real-time face recognition technology, the system automates the process of verifying a person's identity, reducing the need for manual checks and improving overall security.

This project integrates key technologies like MTCNN for face detection, FaceNet for generating unique face embeddings, and SVM (Support Vector Machine) for accurately identifying individuals. The use of a React-based frontend and Flask backend ensures a user-friendly and responsive interface for both administrators and users. A structured MySQL database is used to manage registered users and log entry records.

Key features such as live face recognition using webcam, adding new user faces, and displaying registered users make the system practical and efficient. The system allows only authorized individuals to enter, thereby preventing unauthorized access. It also keeps a clear record of entries, which enhances monitoring and accountability.

Overall, the Face Recognition and Gate Monitoring System successfully achieves its goal of providing a fast, contactless, and reliable method for managing gate entries. It improves safety, reduces human effort, and can be easily scaled for larger systems. By leveraging artificial intelligence and web technologies, the system supports the creation of a smarter and more secure environment.

CHAPTER-8

Future Enhancement

The Face Recognition and Gate Monitoring System lays a strong foundation for secure, contactless entry management. However, to further improve its functionality, scalability, and user experience, the following future enhancements are proposed:

1 Mobile Application Integration :

Develop a mobile application (Android/iOS) to allow administrators and authorized personnel to monitor gate activity, register users, and receive alerts remotely in real time.

2 Masked Face Recognition:

Integrate advanced AI models that can recognize faces even when partially covered with masks or sunglasses, increasing the system's robustness during health-related protocols.

3 Attendance and Entry Log Report Generation :

Extend the system to automatically generate daily, weekly, or monthly attendance and entry logs. These reports can be exported in formats like PDF or Excel for further analysis.

4 Integration with RFID or QR Code :

Combine face recognition with RFID or QR code scanning for two-factor authentication, increasing overall security and reliability.

5 Real-Time Notification System :

Implement a notification system that sends real-time alerts (via SMS, email, or app notification) to admins when an unknown or unauthorized person attempts to enter.

6 Cloud-Based Database Integration :

Migrate the local database to a cloud platform (e.g., AWS, Firebase, Azure) for better scalability,

7 Enhanced Accuracy Using Deep Learning Models :

Incorporate more advanced deep learning models (e.g., ArcFace, Dlib, or Transformer-based models) for better face recognition accuracy and speed.

8 Facial Aging and Variation Handling :

Improve face recognition to handle long-term changes in appearance due to aging, hairstyles, or lighting conditions, ensuring accurate recognition over time.

9 Multi-Language Interface:

Add support for multiple languages to accommodate users from diverse linguistic backgrounds, making the system more accessible to international users.

10 Integration with IoT Devices :

Link the system with physical gate control hardware such as smart locks or automated doors to allow automatic gate opening upon successful face verification.

CHAPTER-9

References

Academic References:

1 Books and Textbooks :

Pattern Recognition and Machine Learning by Christopher M. Bishop Provides foundational knowledge on machine learning techniques, including classification and clustering used in facial recognition systems.

Deep Learning by Ian Goodfellow, Yoshua Bengio, and Aaron Courville Covers essential deep learning models such as CNNs and embedding architectures like FaceNet, used in facial feature extraction.

Computer Vision: Algorithms and Applications by Richard Szeliski Explains key computer vision techniques relevant to face detection, image processing, and real-time video analysis.

2 Research Papers and Articles

"FaceNet: A Unified Embedding for Face Recognition and Clustering"– Florian Schroff, Dmitry Kalenichenko, James Philbin (2015)

The core paper describing the FaceNet model used in your project for converting faces to embeddings.

- "Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks (MTCNN)" –Kaipeng Zhang et al. Outlines the MTCNN algorithm used for detecting faces and facial landmarks.
- IEEE and ACM Journals. Articles related to biometric security systems, real-time recognition, and facial authentication systems in smart surveillance.

Practical References

1 Libraries and Frameworks :

OpenCV (Open Source Computer Vision Library)

Used for handling image capture, face cropping, and real-time frame processing.

- MTCNN (Multi-task Cascaded Convolutional Networks)
Python package for detecting faces and facial landmarks in live video streams.

- keras-facenet

Pre-trained implementation of the FaceNet model, used to generate 128-dimensional facial embeddings.

- scikit-learn

Machine learning library used for implementing the Support Vector Machine (SVM) classifier.

- Flask

Lightweight Python web framework used for creating the backend API for registration and recognition.

- MySQL

Relational database used to store user profiles, embedding vectors, and entry logs.

2. Online Resources and Tutorials :

- Medium Articles and GitHub Projects

Open-source face recognition projects and practical tutorials that inspired the backend pipeline and integration with front-end systems.

- TensorFlow and Keras Documentation

Used for understanding model inference, embedding extraction, and optimization techniques.

- Real Python and GeeksforGeeks

Tutorials for building Flask APIs, integrating OpenCV, and deploying real-time computer vision systems.

4. Standards and Best Practices :

1. ISO/IEC 19794-5:2011 :Biometric Data Interchange Formats

Describes standard formats for face image data to ensure interoperability in biometric applications.

2. OWASP Guidelines :

Security best practices followed for authentication, data handling, and API protection in face recognition systems.