



Rajiv Gandhi University of Knowledge Technologies-Andhra Pradesh

RK Valley Institute

(Constituted under the A.P Govt. Act 18 of 2008 and recognized as per Section 2(f), 12(B) of UGC Act, 1956)

Accredited by 'NAAC' with 'B+' Grade

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AI-Powered Face Recognition And Entry Monitoring System

Under the Guidance of

E.SUSMITHA

M.Tech,(Ph.D)

Assistant Professor

Y CHENNA KESAVA REDDY

[R200337]

D VENKATESWARA NAIK

[R200491]

S GANGA MAHESWARA

[R201046]

Abstract

- The project implements an AI-based face recognition system for monitoring and logging entry of individuals (students/visitors) in real-time.
- Enhances security and automation by replacing manual entry logs with biometric identification.

Abstract

- Uses Machine learning Algorithms for face detection and recognition.
- Integrates database management for logging entries and generating reports.
- Applicable in institutions and organizations for attendance, visitor logs, and security.

Introduction

- Traditional entry systems rely on ID cards or manual logs, prone to errors and manipulation.
- Facial recognition provides a contactless, accurate, and secure alternative.
- Real-time face detection and recognition using webcam.
- Logs user data and timestamp into a backend database.

- Alerts for unauthorized entries and generates reports.

Technology Stack:

- **Frontend:** React.js
- **Backend:** Flask
- **Face Recognition:** MTCNN + FaceNet + SVM
- **Database:** MySQL

Motivation

- Increase in security breaches due to identity fraud.
- Manual entry systems are inefficient and prone to human error.
- Aim to build a smart, automated, and secure entry management system leveraging AI

Contribution

- Developed a modular, scalable system for automated entry logging.
- Integrated real-time face recognition with a robust backend.
- Implemented diverse image augmentation techniques (blur, noise, flip, lighting adjustments) to improve model robustness and accuracy.

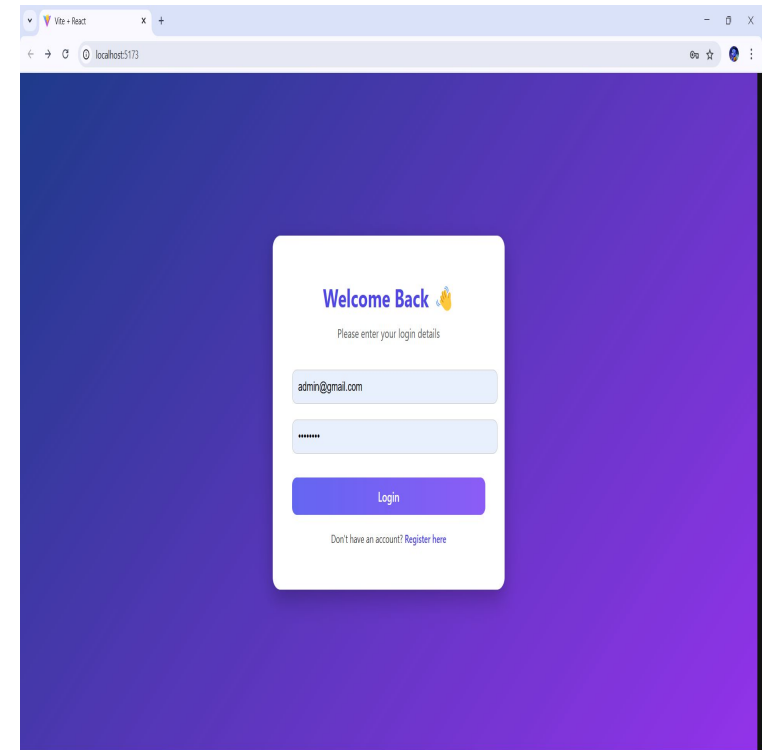
Module Overview

Login Page

Purpose: Secure authentication for admin/security users.

UI Elements:

- Email/Username input
- Password input
- Login button

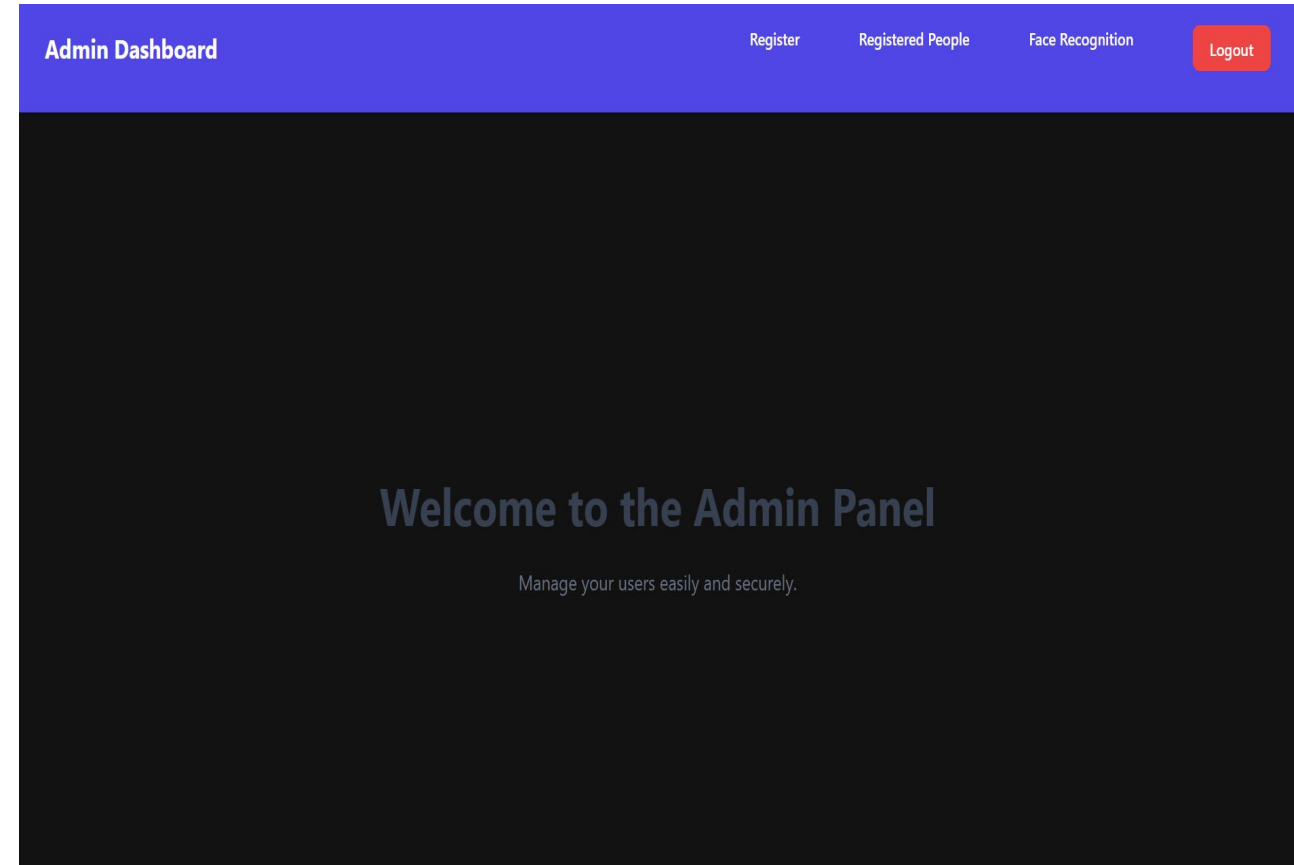


Login Page

```
<form onSubmit={handleLogin}>
  <input
    type="email"
    placeholder="Email"
    value={email}
    onChange={(e) => setEmail(e.target.value)}
    className="input-field"
    required
  />
  <input
    type="password"
    placeholder="Password"
    value={password}
    onChange={(e) => setPassword(e.target.value)}
    className="input-field"
    required
  />
  <button type="submit" className="btn-login" disabled={loading}>
    {loading ? 'Logging in...' : 'Login'}
  </button>
</form>
```

DashBoardPage

- **Centralized Control Panel** for managing users, face recognition, and registered profiles.
- **Easy Navigation** via top menu bar with quick access to Register, Registered People, and Recognition features.
- **Secure Access & Logout** to ensure only authorized personnel manage the system.



DashBoard


```
<div className= dashboard-container >
  <nav className="navbar">
    <h2 className="logo">Admin Dashboard</h2>
    <div className="nav-links">
      <Link to="/register" className="nav-link">Register</Link>
      <Link to="/registered-users" className="nav-link">Registered People</Link>
      <Link to="/face-recognition" className="nav-link">Face Recognition</Link>
      <button onClick={logout} className="logout-btn">Logout</button>
    </div>
  </nav>

  <div className="welcome-section">
    <h1>Welcome to the Admin Panel</h1>
    <p>Manage your users easily and securely.</p>
  </div>
</div>
```

FaceRegister

- **Live Camera Capture** for real-time face input, ensuring accurate data collection.
- **User Details Form** includes ID, name, email, role, and visit purpose for comprehensive profiling.
- **One-Click Actions** like *Preview*, *Register*, and *View Registered Faces* for smooth registration flow.

Register New Face



Enter ID Number

Enter Name

Enter Email

Select Role ▼

Purpose of Visit

Preview **Register**

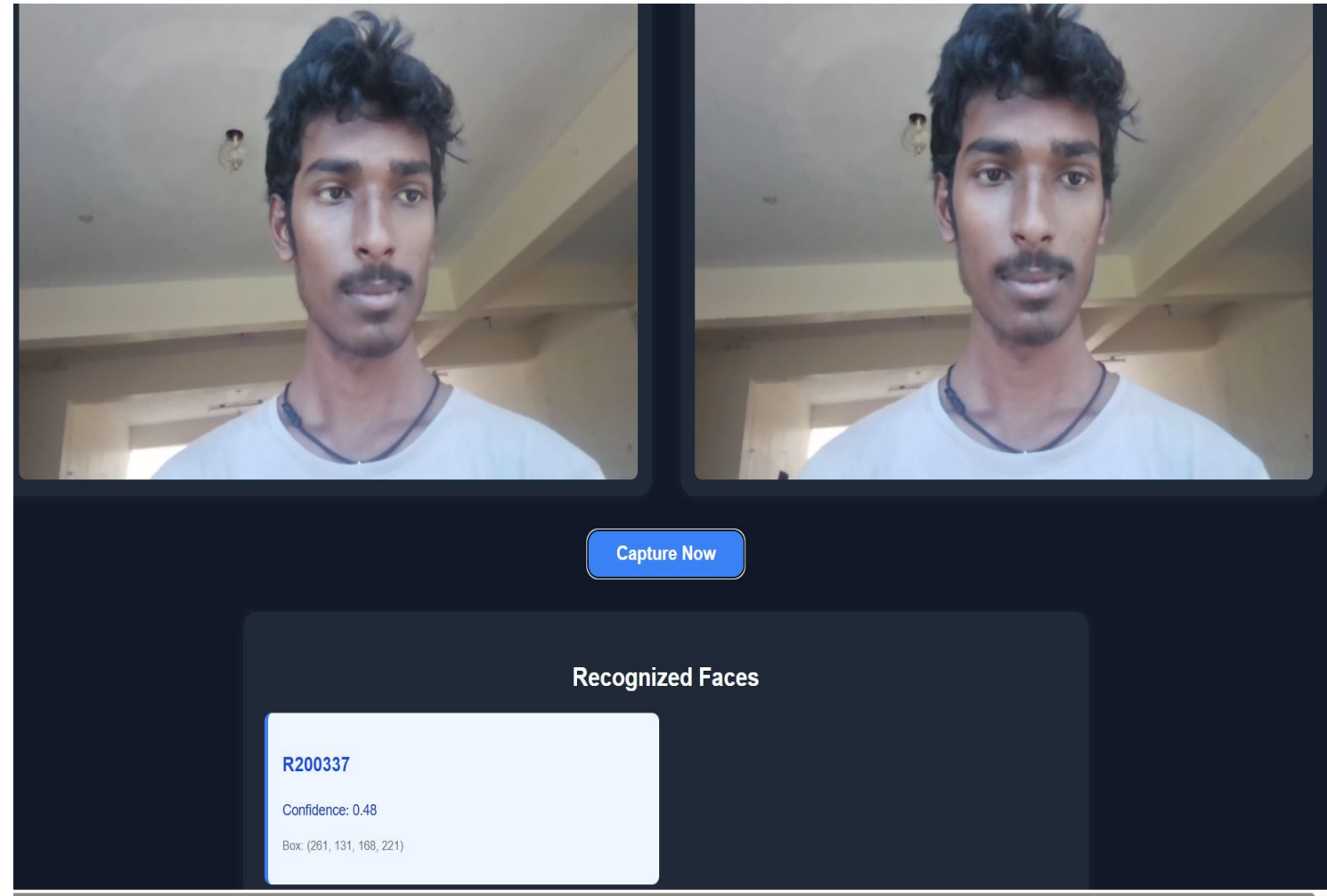
View Registered Faces

Face Registration

```
<input
  type="text"
  placeholder="Enter Name"
  value={name}
  onChange={(e) => setName(e.target.value)}
  className="input-field"
/>
<input
  type="email"
  placeholder="Enter Email"
  value={email}
  onChange={(e) => setEmail(e.target.value)}
  className="input-field"
/>
<select
  value={role}
  onChange={(e) => setRole(e.target.value)}
  className="input-field"
>
  <option value="">Select Role</option>
  <option value="Student">Student</option>
  {/* <option value="Staff">Staff</option> */}
  <option value="Visitor">Visitor</option>
</select>
<input
  type="text"
  placeholder="Purpose of Visit"
  value={purpose}
  onChange={(e) => setPurpose(e.target.value)}
  className="input-field"
/>
<div className="button-group">
  {!previewImage && (
    <button onClick={handlePreview} className="btn btn-warning">
      Preview
    </button>
  )}
  {previewImage && (
    <button onClick={() => setPreviewImage(null)} className="btn btn-secondary">
      Retake Photo
    </button>
  )}
  <button
    onClick={handleRegister}
    disabled={loading}
    className="btn btn-success">
```

Face Recognition Page

- **Live Face Comparison:** Captures and compares real-time face input with registered faces side-by-side.
- **Recognition Feedback:** Displays identified ID, confidence score, and bounding box location.
- **One-Click Capture:** Simple "Capture Now" button initiates recognition process instantly



Face Recognition Page

```
useEffect(() => {
  navigator.mediaDevices.getUserMedia({ video: true }).then((stream) => {
    videoRef.current.srcObject = stream;
    videoRef.current.play();
  });
}, []);


useEffect(() => {
  fetch('http://localhost:5000/registered_users')
    .then((res) => res.json())
    .then((data) => setUsers(data.users || []));
}, []);

const handleCapture = async () => {
  if (!videoRef.current) return;
  setLoading(true);
  const canvas = canvasRef.current;
  const ctx = canvas.getContext('2d');
  canvas.width = videoRef.current.videoWidth;
  canvas.height = videoRef.current.videoHeight;
  ctx.drawImage(videoRef.current, 0, 0);
  const imageData = canvas.toDataURL('image/jpeg');
  try {
    const res = await fetch('http://localhost:5000/recognize', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ image: imageData }),
    });
    const data = await res.json();
    setDetections(data.results || []);
  } catch (err) {
    console.error('Recognition error:', err);
  }
  setLoading(false);
};
```

Registered Users

- **User Overview Cards:** Displays detailed cards for each registered user including name, email, ID, role, and purpose.
- **Manage Profiles Easily:** Includes intuitive “Edit” and “Delete” buttons for each user, enabling quick updates or removals.
- **Navigation Controls:** Pagination and “Back to Registration” button improve usability and allow smooth transitions.

Registered Users



Srinivas


Email: srinu@gmail.com

ID: r20010003

Role: Student

Purpose: upsc

Edit Delete



Yenugu Chenna Kesava Reddy


Email: Chenna6305092639@gmail.com

ID: R200337

Role: Student

Purpose: To Study

Edit Delete



Deshavath Venkateswara Naik


Email: Venkatesh@gmail.com

ID: R200491

Role: Student

Purpose: Studying

Edit Delete



S GANGA MAHESHWARA REDDY

Email: rr201046@rguktrkv.ac.in

ID: R201046

Role: Student

Purpose: studies

Edit Delete

Previous Page 1 of 1 Next

Back to Registration

Registered Users Page

```
const fetchUsers = async () => {
  setLoading(true);
  try {
    const res = await fetch('http://localhost:5000/registered_users');
    const data = await res.json();
    setUsers(data.users || []);
  } catch (err) {
    console.error(err);
    toast.error('Failed to fetch users.');
```

```
  } finally {
    setLoading(false);
  }
};

const deleteUser = async (user_id) => {
  const confirmed = window.confirm(`Are you sure you want to delete ${user_id}?`);
  if (!confirmed) return;

  try {
    const response = await fetch(`http://localhost:5000/delete_user/${user_id}`, {
      method: 'DELETE',
    });

    const result = await response.json();

    if (response.ok) {
      toast.success(result.message || `${user_id} deleted successfully!`);
      fetchUsers();
    } else {
      toast.error(result.error || 'Failed to delete user.');
```

```
    }
  } catch (error) {
    console.error(error);
    toast.error('Something went wrong. Please try again later.');
```

```
  }
};

useEffect(() => {
  fetchUsers();
}, []);
```

Extract Face

- Detects the face in an image using **MTCNN**.
- Crops and aligns the face to ensure consistent input for the model.
- Converts the aligned face to a 160x160 image suitable for embedding.

Get Embeddings

- Passes the aligned face through the **FaceNet model**.
- Generates a **128-dimensional vector** that represents unique facial features.
- The embedding captures identity information for comparison or storage

Face Recognition

- Compares the new embedding with stored embeddings using **cosine or Euclidean distance**.
- If the distance is below a threshold (e.g., 0.6), it's considered a **match**.
- Returns the **closest matching name** or marks it as unknown.

Training

- New faces are embedded and stored in a PKL (as part of **one-shot learning**).
- No need for traditional retraining – just add new vectors.
- A simple **SVM classifier** can be trained on the embeddings for better scalability.

Augmentation

1.Improves Model Robustness

Applies transformations (e.g., brightness, contrast, blur) to simulate real-world variations like lighting and noise.

2.Generates Multiple Variants per Image

Each original image is augmented 20 times, increasing data size and diversity for better training.

3.Maintains Label Integrity

Augmented images are saved under the correct name/label folders, ensuring accurate training data.

Conclusion

- The project successfully demonstrates a secure, automated system for face-based entry monitoring using deep learning and real-time processing.
- It enhances traditional entry systems by eliminating manual processes, reducing fraud, and improving efficiency.
- The combination of MTCNN, FaceNet, and a scalable backend ensures accuracy and reliability in varied environments.

Future Enhancements

- Mobile app integration for live monitoring and approvals.
- Integration of additional biometric methods like voice recognition.
- Continuous model improvement using larger and diverse datasets.

References

React Official Documentation: [React Docs](#)

Flask Official Documentation: [Flash Docs](#)

OpenCV Python Tutorial (PylmageSearch): [PylmageSearch OpenCV Tutorial](#)

MySQL Official Documentation: [MySQL Docs](#)