**Dưới đây là cấu trúc thư mục của dự án và mã của tất cả các file:**

**Cấu trúc thư mục**

**face\_recognition\_project**

**│**

**├── app**

**│ ├── \_\_init\_\_.py**

**│ ├── routes.py**

**│ ├── static**

**│ │ └── styles.css**

**│ └── templates**

**│ ├── index.html**

**│ └── add\_face.html**

**│**

**├── models**

**│ └── face\_recognition\_model.h5**

**│**

**├── data**

**│ ├── raw\_images**

**│ └── processed\_images**

**│**

**├── students.json # File lưu trữ thông tin học sinh**

**├── app.py**

**└── requirements.txt**

**Nội dung các file**

**app/\_\_init\_\_.py**

**from flask import Flask**

**app = Flask(\_\_name\_\_)**

**from app import routes**

**app/routes.py**

**from flask import request, jsonify, render\_template, Response**

**from app import app**

**import os**

**import json**

**import cv2**

**model\_path = 'models/face\_recognition\_model.h5'**

**students\_file = 'students.json'**

**def get\_next\_id():**

**current\_id = 0**

**if os.path.exists('current\_id.json'):**

**with open('current\_id.json', 'r') as file:**

**current\_id = json.load(file).get('current\_id', 0)**

**next\_id = current\_id + 1**

**with open('current\_id.json', 'w') as file:**

**json.dump({'current\_id': next\_id}, file)**

**return next\_id**

**@app.route('/')**

**def home():**

**return render\_template('index.html')**

**@app.route('/add\_face', methods=['GET', 'POST'])**

**def add\_face():**

**if request.method == 'POST':**

**user\_id = get\_next\_id()**

**user\_name = request.form['user\_name']**

**success = collect\_face\_data(user\_id, user\_name)**

**if success:**

**return jsonify({'message': 'Đã thêm dữ liệu khuôn mặt thành công!'})**

**else:**

**return jsonify({'message': 'Có lỗi xảy ra khi thêm dữ liệu khuôn mặt.'}), 500**

**return render\_template('add\_face.html')**

**@app.route('/delete\_data', methods=['POST'])**

**def delete\_data():**

**num\_students = 0**

**if os.path.exists(students\_file):**

**with open(students\_file, 'r') as file:**

**try:**

**students = json.load(file)**

**num\_students = len(students)**

**except json.JSONDecodeError:**

**print("Lỗi: Không thể đọc tệp JSON")**

**return jsonify({'message': 'Lỗi: Không thể đọc tệp JSON'}), 500**

**for folder in ['data/raw\_images', 'data/processed\_images']:**

**if os.path.exists(folder):**

**for filename in os.listdir(folder):**

**file\_path = os.path.join(folder, filename)**

**if os.path.isfile(file\_path):**

**os.remove(file\_path)**

**if os.path.exists(model\_path):**

**os.remove(model\_path)**

**if os.path.exists(students\_file):**

**os.remove(students\_file)**

**if os.path.exists('current\_id.json'):**

**os.remove('current\_id.json')**

**return jsonify({'message': f'Đã xóa dữ liệu của {num\_students} học sinh và đặt lại ID thành công!'})**

**@app.route('/video\_feed')**

**def video\_feed():**

**return Response(gen\_frames(), mimetype='multipart/x-mixed-replace; boundary=frame')**

**def gen\_frames():**

**camera = cv2.VideoCapture(0)**

**while True:**

**success, frame = camera.read()**

**if not success:**

**break**

**gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)**

**face\_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade\_frontalface\_default.xml')**

**faces = face\_cascade.detectMultiScale(gray, 1.1, 4)**

**for (x, y, w, h) in faces:**

**cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 0), 2)**

**ret, buffer = cv2.imencode('.jpg', frame)**

**frame = buffer.tobytes()**

**yield (b'--frame\r\n'**

**b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')**

**camera.release()**

**@app.route('/get\_next\_id', methods=['GET'])**

**def get\_next\_id\_route():**

**next\_id = get\_next\_id()**

**return jsonify({'next\_id': next\_id})**

**def collect\_face\_data(user\_id, user\_name):**

**try:**

**# Tạo thư mục lưu trữ dữ liệu khuôn mặt nếu chưa tồn tại**

**raw\_images\_dir = 'data/raw\_images'**

**user\_dir = os.path.join(raw\_images\_dir, f'user\_{user\_id}\_{user\_name}')**

**if not os.path.exists(user\_dir):**

**os.makedirs(user\_dir)**

**# Mở camera**

**camera = cv2.VideoCapture(0)**

**if not camera.isOpened():**

**raise Exception("Không thể mở camera")**

**print("Bắt đầu thu thập dữ liệu khuôn mặt. Nhấn 'q' để dừng.")**

**count = 0**

**while True:**

**success, frame = camera.read()**

**if not success:**

**raise Exception("Không thể đọc khung hình từ camera")**

**# Phát hiện khuôn mặt**

**face\_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade\_frontalface\_default.xml')**

**faces = face\_cascade.detectMultiScale(frame, 1.1, 4)**

**for (x, y, w, h) in faces:**

**# Vẽ hình chữ nhật quanh khuôn mặt**

**cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 0), 2)**

**# Lưu khung hình chứa khuôn mặt vào thư mục**

**face\_img = frame[y:y+h, x:x+w]**

**img\_path = os.path.join(user\_dir, f'face\_{count}.jpg')**

**cv2.imwrite(img\_path, face\_img)**

**count += 1**

**# Hiển thị khung hình**

**cv2.imshow('Thu thập dữ liệu khuôn mặt', frame)**

**# Nhấn 'q' để dừng**

**if cv2.waitKey(1) & 0xFF == ord('q'):**

**break**

**# Giải phóng camera và đóng tất cả các cửa sổ**

**camera.release()**

**cv2.destroyAllWindows()**

**print(f"Đã thu thập {count} hình ảnh khuôn mặt cho người dùng {user\_name} (ID: {user\_id})")**

**return True**

**except Exception as e:**

**print(f"Lỗi: {e}")**

**return False**

**app/static/styles.css**

**body {**

**font-family: 'Roboto', sans-serif;**

**margin: 0;**

**padding: 0;**

**background-color: #f4f4f9;**

**color: #333;**

**overflow-x: hidden;**

**}**

**header {**

**display: flex;**

**justify-content: space-between;**

**align-items: center;**

**padding: 20px;**

**background-color: #4CAF50;**

**color: white;**

**}**

**header h1 {**

**margin: 0;**

**font-size: 24px;**

**}**

**.delete-button, .home-button, .camera-button, .submit-button {**

**background-color: #be1e1e;**

**color: white;**

**border: none;**

**padding: 10px 20px;**

**cursor: pointer;**

**font-size: 16px;**

**border-radius: 5px;**

**display: flex;**

**align-items: center;**

**transition: background-color 0.3s ease;**

**}**

**.delete-button i, .home-button i, .camera-button i, .submit-button i {**

**margin-right: 8px;**

**}**

**.delete-button:hover, .home-button:hover, .camera-button:hover, .submit-button:hover {**

**background-color: #cabd06;**

**}**

**main {**

**padding: 20px;**

**}**

**.hero {**

**text-align: center;**

**padding: 50px 20px;**

**background-color: white;**

**border-radius: 10px;**

**box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);**

**transition: transform 0.3s ease;**

**}**

**.hero:hover {**

**transform: translateY(-10px);**

**}**

**.hero h2 {**

**font-size: 32px;**

**margin-bottom: 20px;**

**}**

**.hero p {**

**font-size: 18px;**

**color: #666;**

**}**

**.form-container {**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**}**

**.form-container label {**

**font-size: 18px;**

**margin: 10px 0 5px;**

**}**

**.form-container input {**

**padding: 10px;**

**font-size: 16px;**

**margin-bottom: 20px;**

**width: 100%;**

**max-width: 400px;**

**border: 1px solid #ccc;**

**border-radius: 5px;**

**}**

**.camera-container {**

**margin: 20px 0;**

**width: 100%;**

**max-width: 400px;**

**height: 300px;**

**background-color: #e0e0e0;**

**border: 2px dashed #ccc;**

**display: flex;**

**justify-content: center;**

**align-items: center;**

**overflow: hidden**

**Nội dung các file (tiếp)**

**app/templates/index.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Trang web điểm danh học sinh tự động bằng công nghệ AI</title>

<link rel="stylesheet" href="{{ url\_for('static', filename='styles.css') }}">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/4.1.1/animate.min.css">

<script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>

</head>

<body>

<header class="animate\_\_animated animate\_\_fadeInDown">

<h1>Trang web điểm danh học sinh tự động bằng công nghệ AI</h1>

<button id="deleteDataButton" class="delete-button"><i class="fas fa-trash-alt"></i> Xóa dữ liệu đã train</button>

<button id="addFaceButton" class="delete-button"><i class="fas fa-user-plus"></i> Thêm dữ liệu khuôn mặt</button>

</header>

<main>

<section class="hero animate\_\_animated animate\_\_fadeInUp">

<h2>Chào mừng bạn đến với hệ thống điểm danh tự động</h2>

<p>Sử dụng công nghệ AI để nhận diện khuôn mặt và điểm danh học sinh một cách nhanh chóng và chính xác.</p>

</section>

</main>

<script>

document.getElementById('deleteDataButton').addEventListener('click', function() {

fetch('/delete\_data', { method: 'POST' })

.then(response => response.json())

.then(data => {

Swal.fire({

title: 'Thành công!',

text: data.message,

icon: 'success',

confirmButtonText: 'OK'

});

})

.catch(error => {

Swal.fire({

title: 'Lỗi!',

text: 'Có lỗi xảy ra khi xóa dữ liệu.',

icon: 'error',

confirmButtonText: 'OK'

});

console.error('Error deleting data:', error);

});

});

document.getElementById('addFaceButton').addEventListener('click', function() {

window.location.href = '/add\_face';

});

</script>

</body>

</html>

**app/templates/add\_face.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Thêm dữ liệu khuôn mặt</title>

<link rel="stylesheet" href="{{ url\_for('static', filename='styles.css') }}">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/4.1.1/animate.min.css">

<script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>

</head>

<body>

<header class="animate\_\_animated animate\_\_fadeInDown">

<h1>Thêm dữ liệu khuôn mặt</h1>

<button id="homeButton" class="home-button"><i class="fas fa-home"></i> Trở về trang chủ</button>

</header>

<main>

<section class="hero animate\_\_animated animate\_\_fadeInUp">

<form id="addFaceForm" class="form-container">

<label for="user\_id">ID:</label>

<input type="text" id="user\_id" name="user\_id" readonly>

<label for="user\_name">Tên:</label>

<input type="text" id="user\_name" name="user\_name" required>

<button type="button" id="startCameraButton" class="camera-button"><i class="fas fa-camera"></i> Bắt đầu camera</button>

<div id="camera" class="camera-container">

<img id="videoFeed" src="" alt="Video Feed">

</div>

<button type="submit" class="submit-button"><i class="fas fa-user-plus"></i> Thêm khuôn mặt</button>

</form>

</section>

</main>

<script>

document.getElementById('homeButton').addEventListener('click', function() {

window.location.href = '/';

});

document.getElementById('startCameraButton').addEventListener('click', function() {

startCamera();

});

document.getElementById('addFaceForm').addEventListener('submit', function(event) {

event.preventDefault();

const formData = new FormData(this);

fetch('/add\_face', {

method: 'POST',

body: formData

})

.then(response => response.json())

.then(data => {

Swal.fire({

title: 'Thành công!',

text: data.message,

icon: 'success',

confirmButtonText: 'OK'

});

})

.catch(error => {

Swal.fire({

title: 'Lỗi!',

text: 'Có lỗi xảy ra khi thêm dữ liệu khuôn mặt.',

icon: 'error',

confirmButtonText: 'OK'

});

console.error('Error adding face data:', error);

});

});

// Lấy ID tiếp theo từ server và hiển thị trong input

document.addEventListener('DOMContentLoaded', function() {

fetch('/get\_next\_id')

.then(response => response.json())

.then(data => {

const userIdInput = document.getElementById('user\_id');

userIdInput.value = data.next\_id;

})

.catch(error => {

console.error('Error fetching next ID:', error);

});

});

// Hàm để khởi động camera và hiển thị video

function startCamera() {

const videoFeed = document.getElementById('videoFeed');

videoFeed.src = "{{ url\_for('video\_feed') }}";

}

</script>

</body>

</html>

**app.py**

from app import app

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=True)

**requirements.txt**

Flask

opencv-python-headless

tensorflow

numpy

Pillow