TRƯỜNG ĐẠI HỌC CÔNG NGHỆ SÀI GÒN
KHOA CÔNG NGHỆ THÔNG TIN

SAIGON TECHNOLOGY UNIVERSITY

STU

THỰC TẬP CHUYÊN NGÀNH

Tên đề tài :

XÂY DỰNG ỨNG DỤNG QUẢN LÝ ĐIỂM DANH KHUÔN MẶT BẰNG AWS REKOGNITION

Giảng viên hướng dẫn: ThS. Trần Văn Hùng

Nhóm thực hiện: 17

MSSV	**Họ và tên**
DH52200529	Bùi Hoàng Đức Dũng
DH52201724	Võ Hoàng Tuấn

TP. HỒ CHÍ MINH – NĂM 2025

LỜI CẢM ƠN

Chúng em xin gửi lời cảm ơn sâu sắc đến thầy Hùng, người đã trực tiếp hướng dẫn và đồng hành cùng em trong suốt quá trình thực hiện đề tài

Trong thời gian làm việc, thầy đã tận tình chỉ dẫn, giải thích từng khó khăn, định hướng từng bước và luôn tạo điều kiện để chúng em có thể hoàn thiện sản phẩm một cách tốt nhất. Những góp ý của thầy không chỉ giúp chúng em nâng cao kiến thức chuyên môn mà còn rèn luyện tư duy, cách làm việc khoa học và tinh thần trách nhiệm trong học tập.

Nhờ sự hỗ trợ và nhiệt huyết của thầy, chúng em đã có thể hoàn thành dự án này với nhiều trải nghiệm ý nghĩa và bổ ích cho hành trình học tập của mình.

Chúng em xin chân thành cảm ơn thầy!

MỤC LỤC

CHƯƠNG 1. GIỚI THIỆU 4
1.1. Đặt vấn đề và Mục tiêu *4*
1.2. Những thách thức cần giải quyết *4*
1.3. Nội dung và Phạm vi thực hiện *4*
1.4. Kết quả cần đạt *5*
CHƯƠNG 2. PHƯƠNG PHÁP THỰC HIỆN 6
2.1. Tổng quan các hệ thống tương tự *6*
2.1.1.Phần mềm quản lý điểm danh lớp học bằng khuôn mặt tự động eFace 6
2.2. Công nghệ sử dụng *10*
2.3. Phân tích yêu cầu hệ thống *10*
2.3.1. Yêu cầu chức năng 10
2.4. Phân tích nghiệp vụ *11*
2.4.1. Quy trình nghiệp vụ tổng quát 11
2.4.2. Sơ đồ phân rã chức năng 13
2.4.3. Use Case tổng quát 14
2.5. Sơ đồ luồng dữ liệu (DFD) *15*
2.5.1. DFD Level 0 – Chức năng quản lý điểm danh 15
2.5.2. DFD Level 1 – Chức năng Quản lý điểm danh 15
CHƯƠNG 3. THIẾT KẾ 16
3.1. Mô hình dữ liệu *16*
3.1.1. Mô hình dữ liệu mức quan niệm 16
3.1.2. Sơ đồ thực thể – quan hệ (ERD) 21
3.1.3. Phân tích quan hệ giữa các thực thể (Relationship) 21
3.1.4. Thiết kế lược đồ cơ sở dữ liệu 22
3.1.5. Mô tả chi tiết các bảng dữ liệu 23
3.2. Thiết kế kiến trúc hệ thống *28*
3.3. Thiết kế giao diện người dùng *31*
3.3.1. Màn hình đăng nhập 31
3.3.2. Dashboard Admin 31
3.3.3. Màn hình điểm danh 32

3.3.4. Màn hình đăng ký khuôn mặt 32
CHƯƠNG 4. THỬ NGHIỆM 33
4.1. Các kịch bản thử nghiệm *33*
4.1.1. Kịch bản 1: Đăng ký khuôn mặt thành công 33
4.1.2. Kịch bản 2: Điểm danh chính xác 33
4.1.3. Kịch bản 3: Xử lý ảnh mờ 33
4.1.4. Kịch bản 4: Điểm danh người lạ 33
4.1.5. Kịch bản 5: Điểm danh đồng thời 33
4.2. Kết quả thử nghiệm *33*
4.3. Xử lý các trường hợp ngoại lệ *34*
CHƯƠNG 5. KẾT LUẬN 35
5.1. Kết quả đối chiếu với mục tiêu *35*
5.2. Các vấn đề còn tồn đọng *35*
5.3. Hướng phát triển *35*
5.3.1. Ngắn hạn (3-6 tháng) 35
5.3.2. Trung hạn (6-12 tháng) 36
5.3.3. Dài hạn (> 12 tháng) 36
CHƯƠNG 6. PHỤ LỤC : HƯỚNG DẪN CÀI ĐẶT VÀ SỬ DỤNG HỆ THỐNG 37
1. Mục đích ..... **Error! Bookmark not defined.**
2. Đối tượng sử dụng ..... **Error! Bookmark not defined.**
3. Điều kiện thực hiện ..... **Error! Bookmark not defined.**
ĐÁNH GIÁ ĐÓNG GÓP CỦA NHÓM 50

MỤC LỤC CÁC HÌNH VẼ

Hình 2-1Màn hỉnh chức năng báo cáo điểm danh (eFace).....6
Hình 2-2 Màn hình nhập liệu sinh viên (eFace).....7
Hình 2-3 Màn hình chức năng điểm danh (eFace).....8
Hình 2-4 Màn hình chức năng báo cáo (eFace).....9
Hình 2-5 Sơ đồ phân rã chức năng.....13
Hình 2-6 Sơ đồ use-case tổng quát.....14
Hình 2-7 DFD Level 0 – Chức năng quản lý điểm danh.....15
Hình 2-8 DFD Level 1 – Chức năng quản lý điểm danh.....15
Hình 3-1 Sơ đồ thực thể - quan hệ (ERD).....21
Hình 3-2 Kiến trúc tổng thể.....29
Hình 3-3 Màn hình đăng nhập.....31
Hình 3-4 Màn hình Dashboard.....31
Hình 3-5 Màn hình điểm danh.....32
Hình 3-6 Màn hình đăng ký khuôn mặt.....32

Chương 1. **GIỚI THIỆU**

1.1. Đặt vấn đề và Mục tiêu

Việc điểm danh truyền thống bằng giấy hoặc thủ công tốn thời gian và dễ gian lận. Hệ thống điểm danh bằng nhận diện khuôn mặt giúp tự động hóa quy trình, tăng độ chính xác và tiết kiệm thời gian.

Mục tiêu: Xây dựng ứng dụng quản lý điểm danh sử dụng công nghệ nhận diện khuôn mặt trên nền tảng AWS, hỗ trợ giảng viên và sinh viên quản lý hiệu quả.

1.2. Những thách thức cần giải quyết

- **Độ chính xác nhận diện:** Xử lý các trường hợp ánh sáng khác nhau, góc chụp, che khuất khuôn mặt
- **Hiệu năng:** Xử lý nhanh cho nhiều khuôn mặt đồng thời
- **Bảo mật dữ liệu:** Bảo vệ thông tin sinh trắc học người dùng
- **Ước lượng chi phí AWS:** Dựa trên quy mô khoảng 2.000 sinh viên khoa Công nghệ Thông tin, mỗi kỳ thi cuối kỳ gồm 10 môn, mỗi môn tổ chức tại 10 phòng thi với trung bình 100 sinh viên/phòng, tổng số phòng thi là 100 phòng cho một kỳ thi.
- Theo ước lượng, chi phí sử dụng AWS cho mỗi phòng thi (100 sinh viên) khoảng 0,5 USD, do đó tổng chi phí cho một kỳ thi cuối kỳ vào khoảng 50 USD, tương đương 1.250.000 VNĐ. Mức chi phí này được đánh giá là thấp và phù hợp, đáp ứng tốt cho việc triển khai hệ thống trong môi trường thực tế.
-
- **Xử lý ngoại lệ:** Sinh viên vắng mặt, lỗi camera, mất kết nối mạng

1.3. Nội dung và Phạm vi thực hiện

Phạm vi:

- Quản lý thông tin sinh viên, giảng viên, lớp học
- Đăng ký khuôn mặt sinh viên
- Điểm danh tự động qua camera
- Xem báo cáo điểm danh theo lớp, môn học, thời gian
- Hệ thống chỉ phục vụ cho môi trường giáo dục

1.4. Kết quả cần đạt

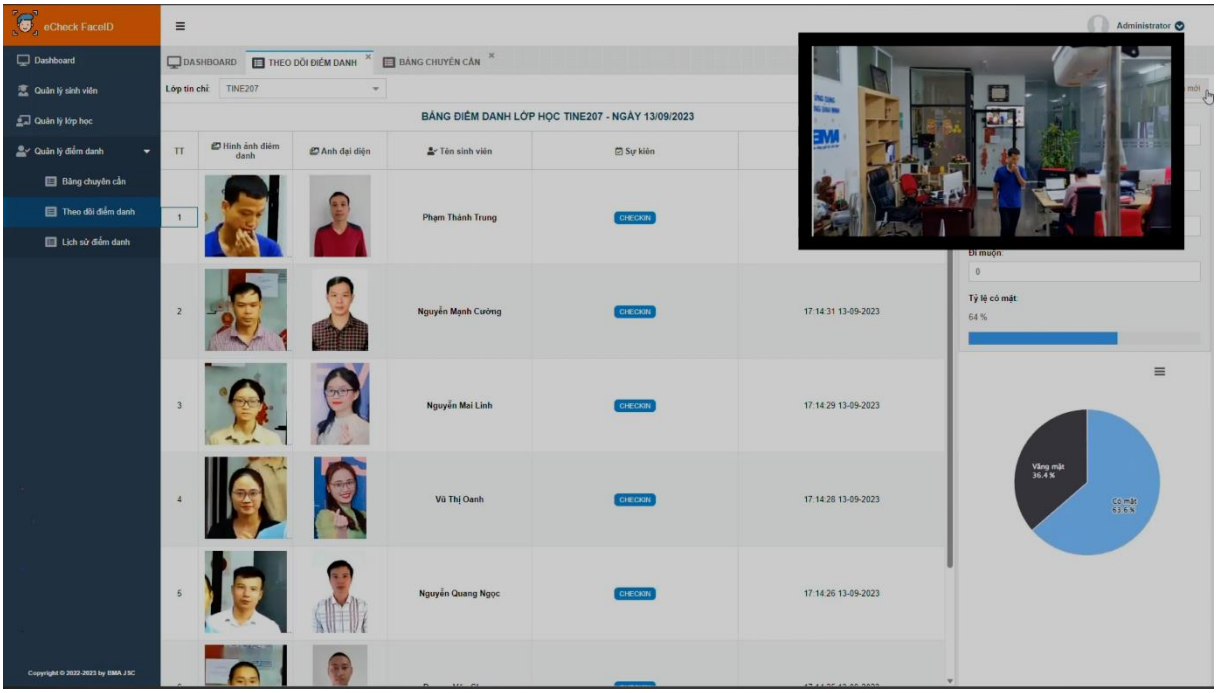
STT	Kết quả	Tiêu chí đánh giá	Mục tiêu
1	Đăng ký khuôn mặt	Thành công với ảnh rõ nét, góc thẳng	≥ 95%
2	Nhận diện chính xác	So khớp đúng người trong điều kiện tốt	≥ 90%
3	Thời gian xử lý	Thời gian nhận diện 1 khuôn mặt	≤ 2 giây
4	Khả năng xử lý đồng thời	Số lượng sinh viên điểm danh cùng lúc	≥ 30 người
5	Bảo mật	Mã hóa dữ liệu, phân quyền truy cập	0%
6	Tính sẵn sàng	Uptime của hệ thống	≥ 99%

Chương 2. **PHƯƠNG PHÁP THỰC HIỆN**

2.1. Tổng quan các hệ thống tương tự

2.1.1.Phần mềm quản lý điểm danh lớp học bằng khuôn mặt tự động eFace

Đây là một hệ thống phần mềm được thiết kế chuyên biệt cho nghiệp vụ quản lý điểm danh trong giáo dục (trường học, trung tâm đào tạo). Hệ thống cho phép quản trị viên tạo lớp học, quản lý danh sách sinh viên/học sinh, và sử dụng camera để tự động hóa quy trình điểm danh khi sinh viên vào lớp.



Hình 2-1Màn hình chức năng báo cáo điểm danh (eFace)

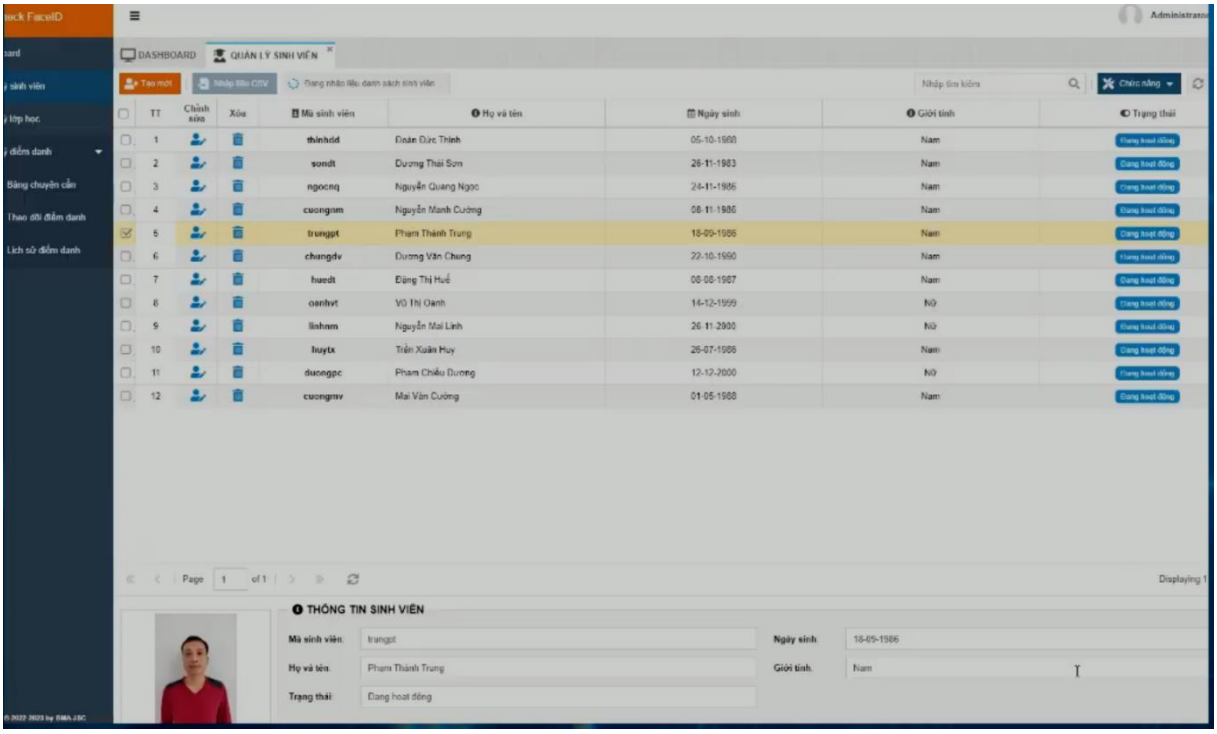
Nhận xét, đánh giá:

Ưu điểm: Được thiết kế chuyên cho lớp học, biết rõ ai thuộc lớp nào. Nó còn so sánh ảnh thật với ảnh hồ sơ nên khả năng chống điểm danh hộ là rất cao.

Cập nhật trạng thái rõ ràng: Hệ thống đánh dấu checkin và ghi lại thời gian cụ thể cho từng sinh viên trong danh sách, đồng thời thống kê tỷ lệ có mặt. Rất rõ ràng về mặt nghiệp vụ.

Khuyết điểm: Phụ thuộc vào góc camera: Khung camera ở góc phải cho thấy một góc quay khá rộng và từ trên cao. Điều này có thể làm giảm độ chính xác khi nhận diện, vì khuôn mặt sinh viên có thể bị che khuất, ở quá xa, hoặc không nhìn thẳng vào camera.

Yêu cầu dữ liệu gốc chất lượng: Tính năng so sánh ảnh thực tế với ảnh hồ sơ đòi hỏi ảnh hồ sơ hình ảnh điểm danh phải rất rõ nét và cập nhật. Nếu ảnh hồ sơ cũ hoặc mờ, AI có thể nhận diện sai hoặc không nhận diện được.



Hình 2-2 Màn hình nhập liệu sinh viên (eFace)

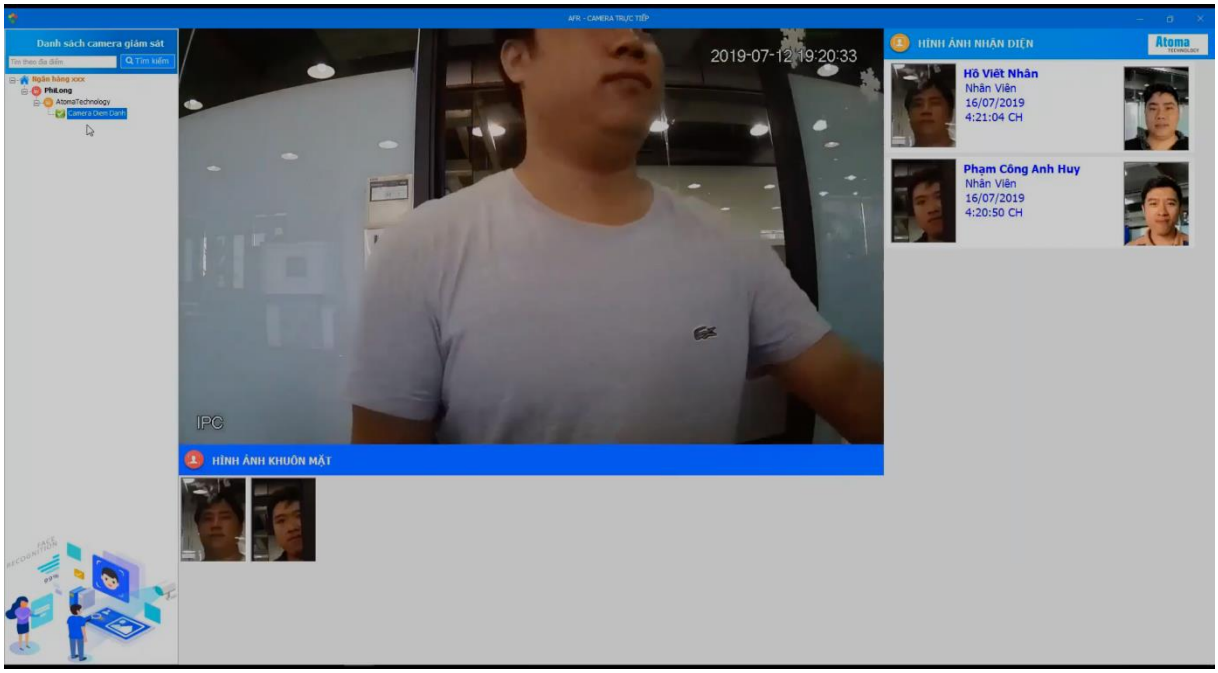
Nhận xét, đánh giá

Ưu điểm: Giao diện có nút Tạo mới rõ ràng. Form THÔNG TIN SINH VIÊN hiển thị đầy đủ các trường cơ bản (Mã SV, Tên, v.v.) và quan trọng nhất là có mục để tải ảnh chân dung—yếu tố bắt buộc cho điểm danh khuôn mặt.

Khuyết điểm: Không có chức năng Import (nhập hàng loạt). Quản trị viên sẽ phải nhập tay từng sinh viên một, rất tốn thời gian cho một lớp học đông. Ngoài ra, form nhập liệu bị đặt ở tít dưới cùng, nếu danh sách dài sẽ rất bất tiện khi thao tác.

2.1.2.Phần mềm giám sát và nhận diện của Atoma Technology

Đây là một giải pháp giám sát thông minh, sử dụng Trí tuệ nhân tạo (AI) để phân tích hình ảnh từ camera theo thời gian thực. Hệ thống của Atoma không chỉ phục vụ cho việc điểm danh/chấm công mà còn là một giải pháp an ninh tổng thể cho các tòa nhà thông minh, văn phòng. Nó có khả năng nhận diện khuôn mặt, phát hiện người trong danh sách (VIP, Blacklist), và truy vết lộ trình di chuyển của đối tượng.



Hình 2-3 Màn hình chức năng điểm danh (eFace)

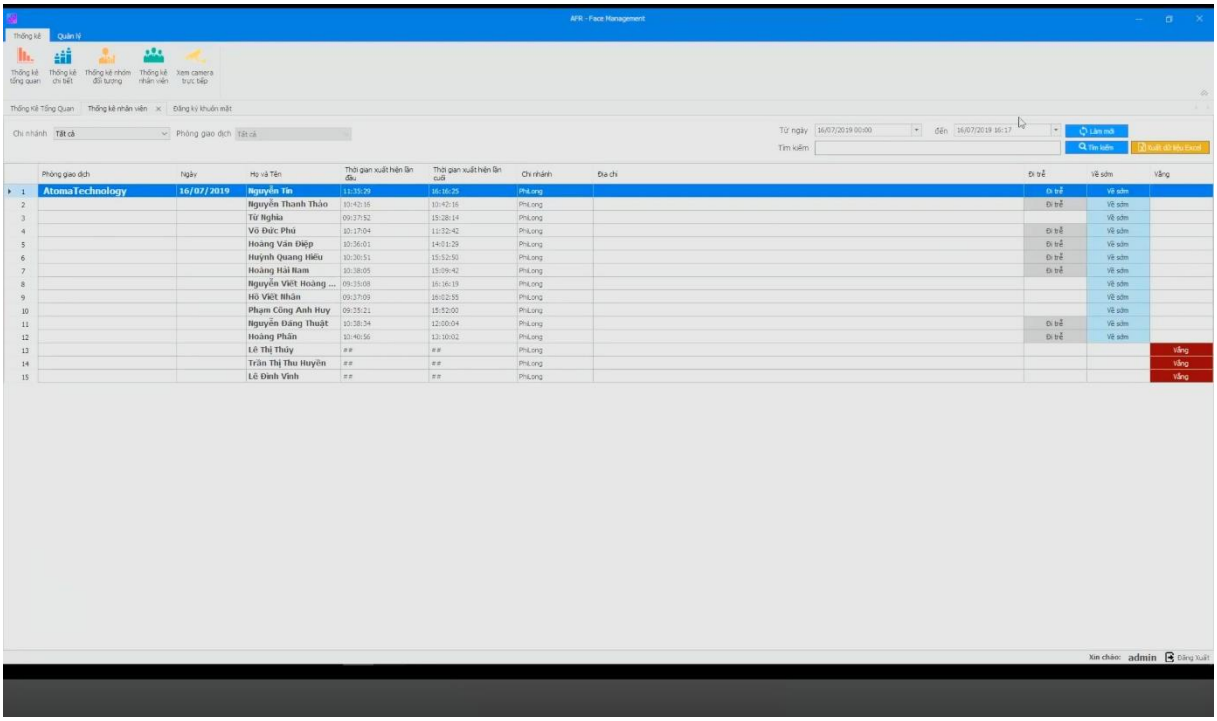
Nhận xét, đánh giá

Ưu điểm: Hoạt động rất nhanh và tự động. Bạn chỉ cần đi lướt qua là hệ thống nhận diện và ghi lại thời gian ngay lập tức, không cần dừng lại hay thao tác gì.

Tốc độ xử lý nhanh: Việc ghi nhận thời gian chính xác đến từng giây cho thấy khả năng xử lý tốc độ cao, phù hợp cho những nơi có lưu lượng người qua lại lớn.

Khuyết điểm: Không chuyên biệt cho giáo dục: Đây giống một phần mềm giám sát an ninh (Security) hơn là một phần mềm quản lý học vụ (Education)

Không phân biệt được sinh đôi: Gần như chắc chắn sẽ thất bại với các cặp sinh đôi cùng trứng. Hệ thống sẽ nhầm lẫn, cho phép người này điểm danh hộ cho người kia



Hình 2-4 Màn hình chức năng báo cáo (eFace)

Nhận xét, đánh giá:

Ưu điểm: Giao diện báo cáo rất chi tiết, tự động ghi lại thời gian xuất hiện đầu (tức là giờ check-in) của từng người. Nó được thiết kế để quản lý chuyên cần chuyên nghiệp, có các cột để tự động tính toán đi trễ , về sớm và tổng hợp trạng thái cuối cùng.

Khuyết điểm: Thiết kế bảng bị rối mắt do có quá nhiều cột bị bỏ trống (như Thời gian xuất hiện cuối), khiến các tính năng quan trọng như về sớm không có dữ liệu. Ngoài ra, các nút vắng màu đỏ gây khó hiểu, không rõ đây là trạng thái tự động hay là nút mà người quản lý phải bấm thủ công.

2.2. Công nghệ sử dụng

Công nghệ	Mô tả	Vai trò
AWS Rekognition	Dịch vụ nhận diện khuôn mặt	Nhận diện và so khớp khuôn mặt
AWS S3	Lưu trữ đối tượng	Lưu trữ ảnh khuôn mặt
AWS Lambda	Serverless computing	Xử lý logic nghiệp vụ
AWS DynamoDB	NoSQL Database	Lưu trữ thông tin điểm danh
Blade Template	Temple Engine của Laravel	Xây dựng giao diện người dùng (View)
Laravel 12.x	PHP Framework	Framework chính xử lý logic nghiệp vụ, routing, authentication, kết nối database

2.3. Phân tích yêu cầu hệ thống

2.3.1. Yêu cầu chức năng

- **Quản lý tài khoản:**
  - Quản trị viên tạo tài khoản, và cấp lại mật khẩu cho tài khoản giảng viên.
  - Người dùng có thể đăng nhập, đăng xuất và thay đổi mật khẩu cá nhân.
- **Quản lý sinh viên:**
  - Ứng dụng cho phép nhập danh sách sinh viên từ file excel.
  - Quản lý thông tin cơ bản của sinh viên: mã số sinh viên, họ tên, lớp, email, số điện thoại.
  - Tìm kiếm, xem, thêm, sửa, xóa thông tin sinh viên.
- **.Quản lý giảng viên**
  - Ứng dụng cho phép nhập danh sách giảng viên từ file excel.
  - Quản lý thông tin cơ bản của giảng viên: mã số giảng viên, họ tên, khoa, email, số điện thoại.

- Tìm kiếm, xem, thêm, sửa, xóa thông tin giảng viên.

– **Quản lý ca thi**

- Ứng dụng cho phép nhập danh sách ca thi, thí sinh dự thi và giám thị từ file excel.
- Tìm kiếm, lọc ca thi theo giờ thi, phòng thi, cho phép sửa, xóa thông tin ca thi.

– **Quản lý điểm danh**

- Quản trị viên gửi file ảnh đăng ký khuôn mặt sinh viên
- Giảng viên có thể xem danh sách sinh viên trong lớp học/ca thi được phân công.
- Giảng viên thực hiện điểm danh cho từng sinh viên (có mặt, vắng mặt).

– **Quản lý thống kê**

- Xem báo cáo điểm danh
- Xuất file báo cáo điểm danh của một ca thi ra Excel.

2.3.2 Yêu cầu phi chức năng

Bảo mật:

- Phân quyền truy cập dựa trên vai trò (Quản trị viên, Giảng viên). Giảng viên chỉ có thể điểm danh các lớp mình được phân công.
- Mật khẩu người dùng phải được mã hóa.

Hiệu năng:

- Hệ thống phải phản hồi các thao tác chính (đăng nhập, điểm danh) trong vòng 2 giây.
- Hệ thống có khả năng xử lý đồng thời ít nhất 100 giảng viên truy cập cùng lúc.

Tính dễ sử dụng:

- Giao diện người dùng phải trực quan, đơn giản, dễ thao tác cho cả những người không rành về công nghệ.
- Hệ thống cần có sự nhất quán về bố cục và thiết kế giữa các trang.

2.4. Phân tích nghiệp vụ

2.4.1. Quy trình nghiệp vụ tổng quát

Hệ thống hoạt động theo các quy trình nghiệp vụ chính:

1. Quy trình quản lý thông tin:

- Admin tạo tài khoản cho giảng viên

- Admin nhập danh sách sinh viên, giảng viên từ Excel

- Admin tạo và quản lý ca thi

2. Quy trình đăng ký khuôn mặt:

- Admin upload ảnh khuôn mặt sinh viên lên hệ thống

- Hệ thống tải ảnh lên AWS S3 - AWS Rekognition phân tích và lưu đặc trưng khuôn mặt - Lưu thông tin vào DynamoDB

3. Quy trình điểm danh:

- Giảng viên chọn ca thi cần điểm danh

- Giảng viên chụp ảnh sinh viên

- Hệ thống gửi ảnh đến AWS Rekognition

- So khớp khuôn mặt và trả về kết quả

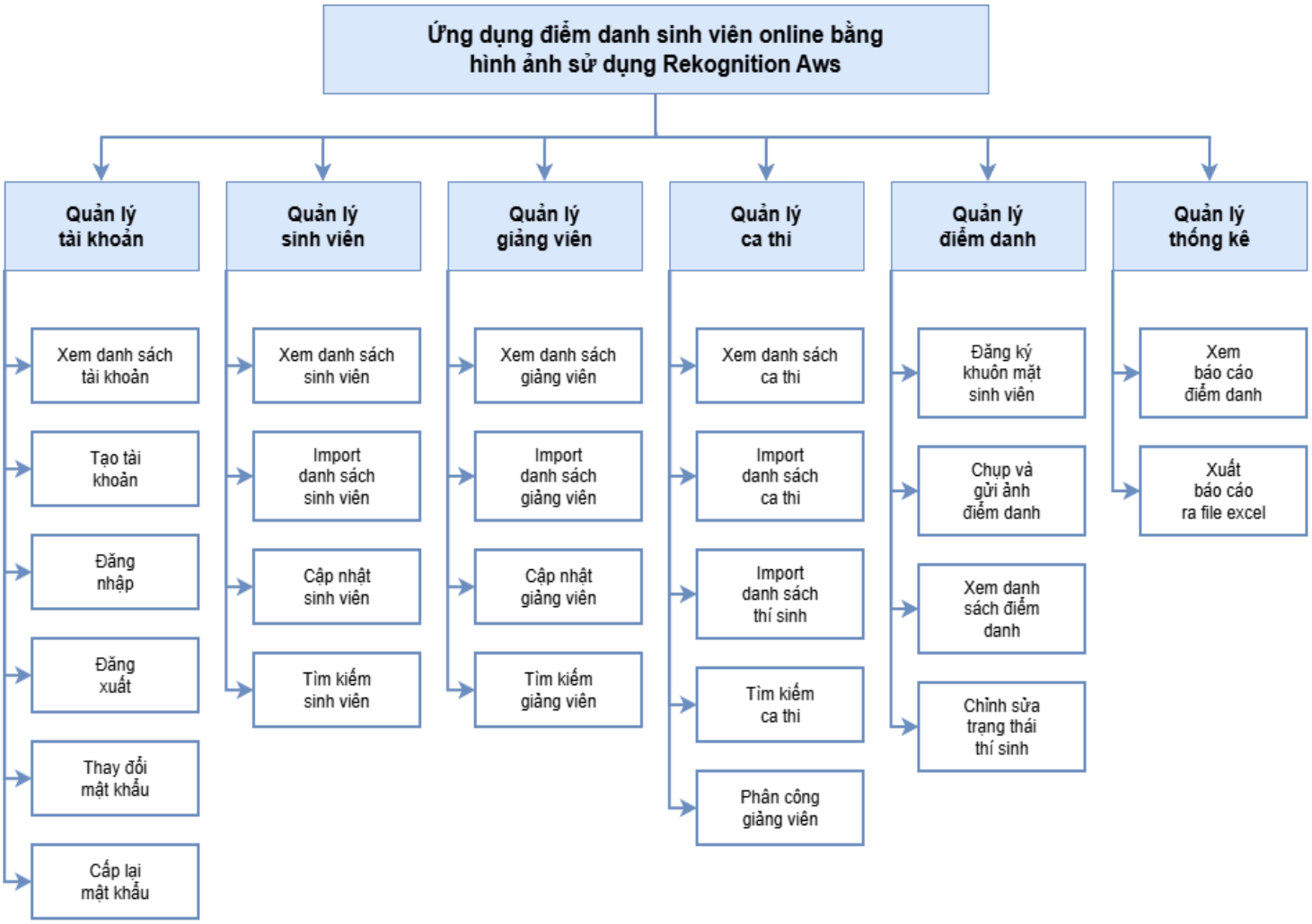
- Ghi nhận điểm danh vào database

4. Quy trình báo cáo:

- Giảng viên/Admin xem thống kê điểm danh

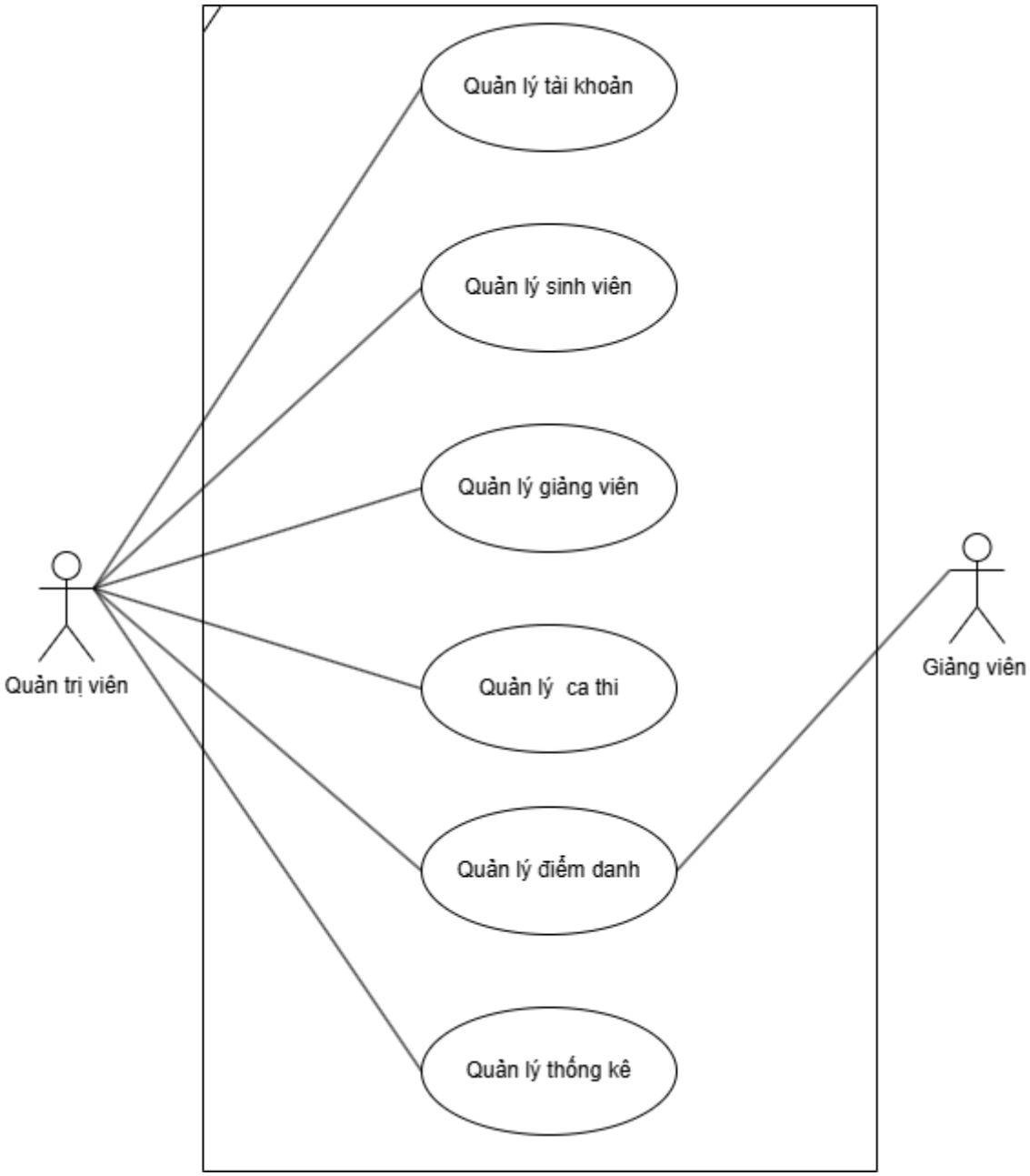
- Xuất báo cáo Excel

2.4.2. Sơ đồ phân rã chức năng



Hình 2-5 Sơ đồ phân rã chức năng

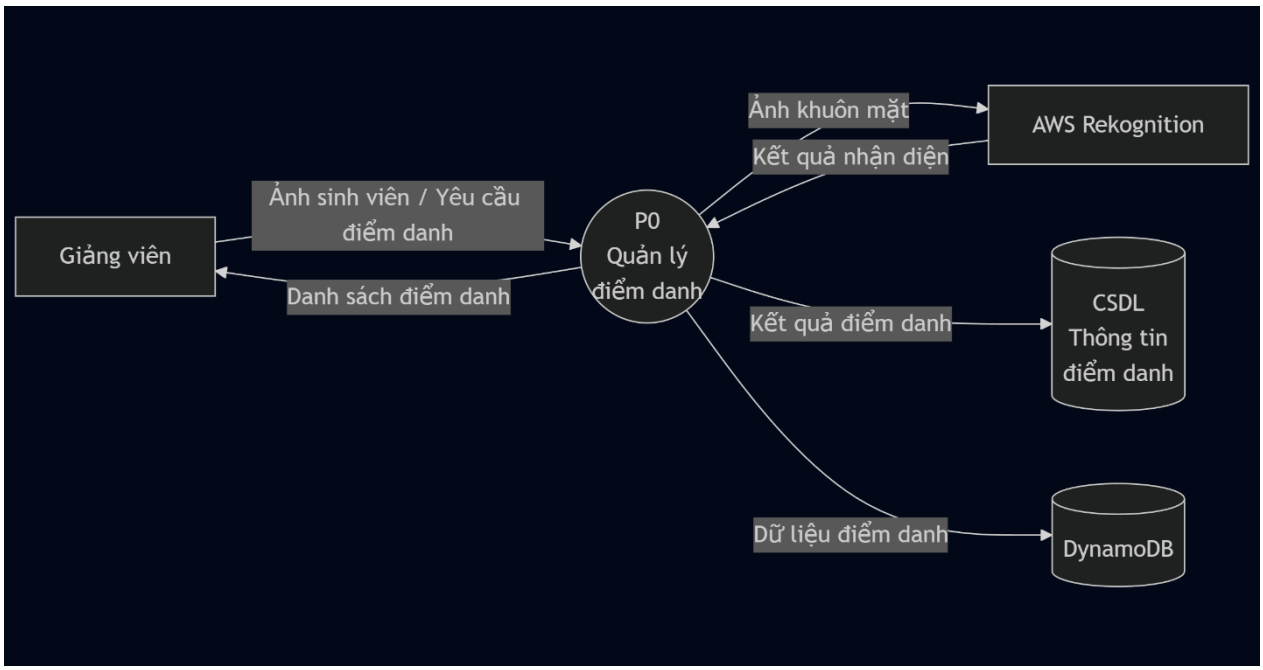
2.4.3. Use Case tổng quát



Hình 2-6 Sơ đồ use-case tổng quát

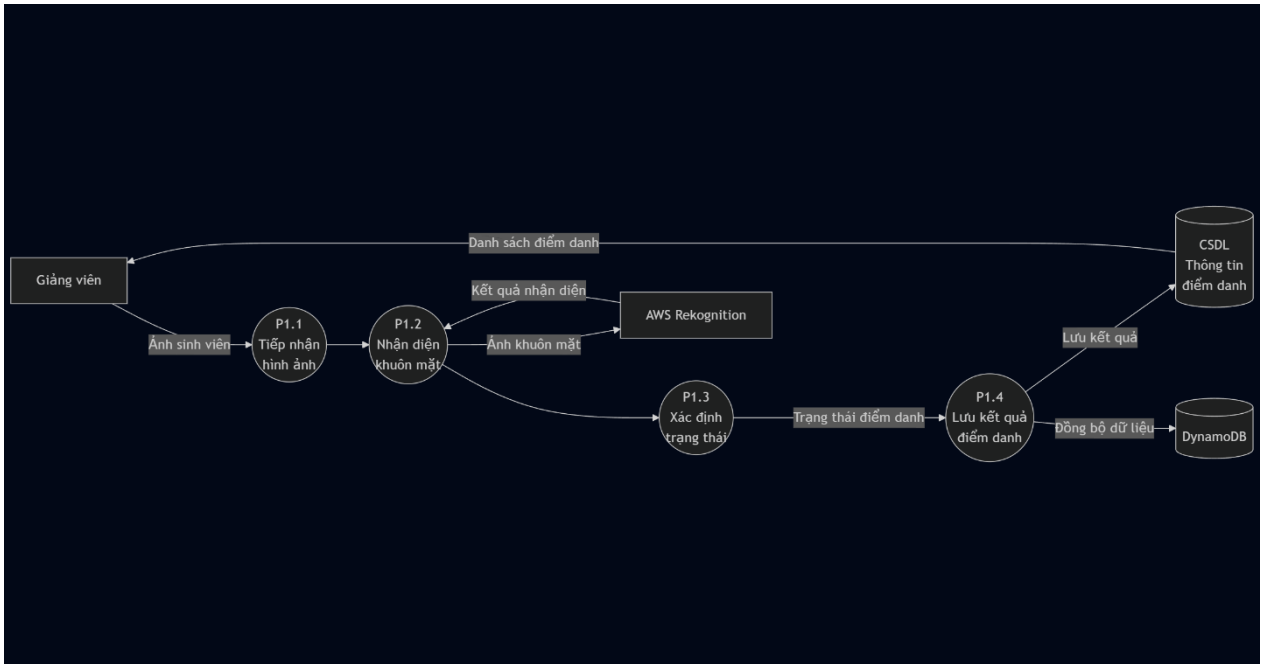
2.5. Sơ đồ luồng dữ liệu (DFD)

2.5.1. DFD Level 0 – Chức năng quản lý điểm danh



Hình 2-7 DFD Level 0 – Chức năng quản lý điểm danh

2.5.2. DFD Level 1 – Chức năng Quản lý điểm danh



Hình 2-8 DFD Level 1 – Chức năng quản lý điểm danh

Chương 3. **THIẾT KẾ**

3.1. Mô hình dữ liệu

3.1.1. Mô hình dữ liệu mức quan niệm

- ❑ Loại thực thể USER

Mô tả: Loại thực thể USER (Người dùng) lưu trữ thông tin tài khoản (Quản trị viên, giảng viên) để đăng nhập và sử dụng hệ thống.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
id	Số nguyên lớn	x	x	x	ID người dùng (khóa chính)
name	Chuỗi(191)			x	Tên người dùng
email	Chuỗi(191)		x	x	Email đăng nhập
role	Enum(...)			x	Vai trò người dùng
password	Chuỗi(191)			x	Mật khẩu (đã mã hóa)
created_at	Dấu thời gian				Thời gian tạo
updated_at	Dấu thời gian				Thời gian cập nhật

- ❑ Loại thực thể LECTURER

Mô tả: Loại thực thể LECTURER (Giảng viên) lưu trữ thông tin cá nhân của giảng viên, có thể liên kết với một tài khoản USER.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
lecturer_code	Chuỗi(20)	x	x	x	Mã giảng viên (khóa chính)
full_name	Chuỗi(100)			x	Họ và tên giảng viên
email	Chuỗi(100)		x		Email giảng viên

phone	Chuỗi(15)		x		Số điện thoại giảng viên
created_at	Dấu thời gian				Thời gian tạo
updated_at	Dấu thời gian				Thời gian cập nhật

❑ Loại thực thể FACULTY

Mô tả: Loại thực thể FACULTY (Khoa) lưu trữ thông tin về các khoa trong trường.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
faculty_code	Chuỗi(20)	x	x	x	Mã khoa (khóa chính)
name	Chuỗi(100)		x	x	Tên khoa
created_at	Dấu thời gian				Thời gian tạo
updated_at	Dấu thời gian				Thời gian cập nhật

❑ Loại thực thể CLASS

Mô tả: Loại thực thể CLASS (Lớp học) lưu trữ thông tin về các lớp học, mỗi lớp thuộc về một khoa.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
class_code	Chuỗi(20)	x	x	x	Mã lớp (khóa chính)
class_name	Chuỗi(100)			x	Tên lớp
created_at	Dấu thời gian				Thời gian tạo
updated_at	Dấu thời gian				Thời gian cập nhật

❑ Loại thực thể STUDENT

Mô tả: Loại thực thể STUDENTS (Sinh viên) lưu trữ thông tin cá nhân của sinh viên, mỗi sinh viên thuộc về một lớp.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
student_code	Chuỗi(20)	x	x	x	Mã sinh viên (khóa chính)
full_name	Chuỗi(100)			x	Họ và tên sinh viên
email	Chuỗi(100)		x		Email sinh viên
phone	Chuỗi(15)		x		Số điện thoại sinh viên
created_at	Dấu thời gian				Thời gian tạo
updated_at	Dấu thời gian				Thời gian cập nhật

❑ Loại thực thể SUBJECT

Mô tả: Loại thực thể CLASSES (Lớp học) lưu trữ thông tin về các lớp học, mỗi lớp thuộc về một khoa.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
class_code	Chuỗi(20)	x	x	x	Mã lớp (khóa chính)
class_name	Chuỗi(100)			x	Tên lớp
created_at	Dấu thời gian				Thời gian tạo
updated_at	Dấu thời gian				Thời gian cập nhật

❑ Loại thực thể EXAM_SCHEDULE

Mô tả: Loại thực thể EXAM_SCHEDULE (Lịch thi) lưu trữ thông tin chi tiết về các ca thi

Thuộc tính	Kiểu	K	U	M	Diễn giải
id	Số nguyên lớn	x	x	x	ID lịch thi (khóa chính)
exam_date	Ngày			x	Ngày thi
exam_time	Giờ			x	Giờ thi
duration	Số nguyên			x	Thời lượng thi (phút)
room	Chuỗi(50)			x	Phòng thi
note	Văn bản				Ghi chú
created_at	Dấu thời gian				Thời gian tạo
updated_at	Dấu thời gian				Thời gian cập nhật

- Loai thực thể EXAM_SUPERVISOR

Mô tả: Loại thực thể CLASSES (Lớp học) lưu trữ thông tin về các lớp học, mỗi lớp thuộc về một khoa.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
class_code	Chuỗi(20)	x	x	x	Mã lớp (khóa chính)
class_name	Chuỗi(100)			x	Tên lớp
created_at	Dấu thời gian				Thời gian tạo
updated_at	Dấu thời gian				Thời gian cập nhật

- Loai thực thể ATTENDANCE_RECORD

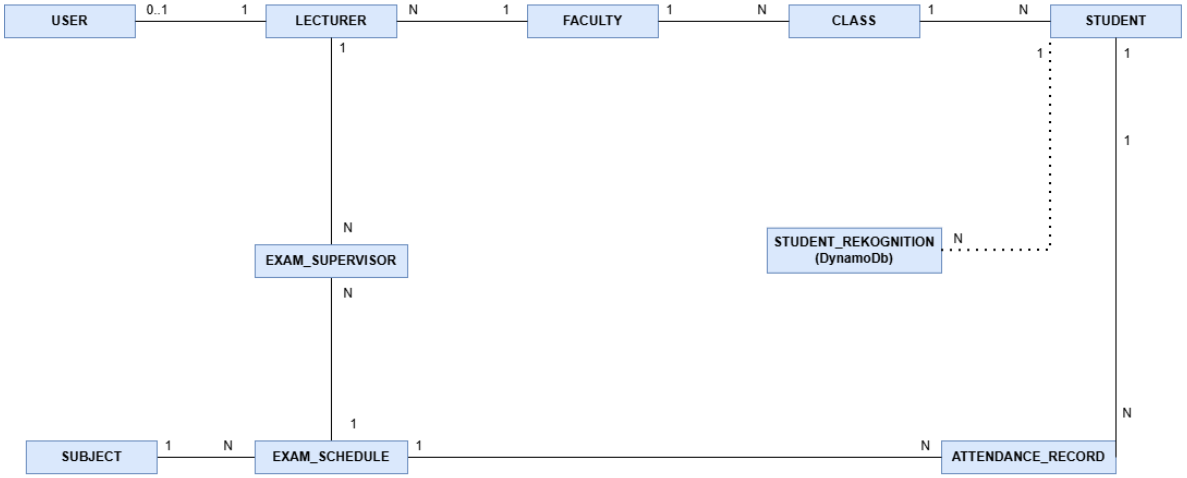
Mô tả: Loại thực thể ATTENDANCE_RECORDS (Kết quả điểm danh) lưu lại kết quả điểm danh của từng sinh viên trong một ca thi.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
id	Số nguyên lớn	x	x	x	ID (khóa chính)

rekognition_result	Enum(...)				Kết quả nhận diện
confidence	Số TP(5,2)				Độ tin cậy của nhận diện
attendance_time	Dấu thời gian				Thời gian điểm danh
created_at	Dấu thời gian				Thời gian tạo
updated_at	Dấu thời gian				Thời gian cập nhật

- Loại thực thể STUDENT_REKOGNITION (DynamoDb)

Mô tả: Mô tả: Lưu trữ dữ liệu nhận diện khuôn mặt từ AWS Rekognition trong DynamoDB. Mỗi sinh viên có thể có nhiều bản ghi nhận diện.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
rekognitionId	String	x	x	x	Mã định danh (ID) của kết quả từ Rekognition
student_code	String			x	Mã số sinh viên được nhận diện

3.1.2. Sơ đồ thực thể – quan hệ (ERD)



Hình 3-1 Sơ đồ thực thể - quan hệ (ERD)

3.1.3. Phân tích quan hệ giữa các thực thể (Relationship)

Các mối quan hệ chính trong hệ thống:

1. USER - LECTURER (1:1)

- Một người dùng có thể là một giảng viên

- Khóa ngoại: lecturer.user_id → user.id

2. FACULTY - CLASS (1:N)

- Một khoa có nhiều lớp học

- Khóa ngoại: class.faculty_code → faculty.faculty_code

3. FACULTY - LECTURER (1:N)

- Một khoa có nhiều giảng viên

- Khóa ngoại: lecturer.faculty_code → faculty.faculty_code

4. CLASS - STUDENT (1:N)

- Một lớp có nhiều sinh viên

- Khóa ngoại: student.class_code → class.class_code

5. SUBJECT - EXAM_SCHEDULE (1:N)

- Một môn học có nhiều ca thi

- Khóa ngoại: exam_schedule.subject_code → subject.subject_code 6

. EXAM_SCHEDULE - EXAM_SUPERVISOR (1:N)

- Một ca thi có nhiều giám thị

- Khóa ngoại: exam_supervisor.exam_schedule_id → exam_schedule.id

7. LECTURER - EXAM_SUPERVISOR (1:N)

- Một giảng viên có thể coi nhiều ca thi

- Khóa ngoại: exam_supervisor.lecturer_code → lecturer.lecturer_code

8. EXAM_SCHEDULE - ATTENDANCE_RECORD (1:N)

- Một ca thi có nhiều bản ghi điểm danh

- Khóa ngoại: attendance_record.exam_schedule_id → exam_schedule.id

9. STUDENT - ATTENDANCE_RECORD (1:N)

- Một sinh viên có nhiều bản ghi điểm danh

- Khóa ngoại: attendance_record.student_code → student.student_code

10. STUDENT - STUDENT_REKOGNITION (1:N)

- Một sinh viên có thể có nhiều ảnh khuôn mặt đăng ký

- Khóa ngoại: student_rekognition.student_code → student.student_code

3.1.4. Thiết kế lược đồ cơ sở dữ liệu

LƯỢC ĐỒ QUAN HỆ:

- USER(id, name, email, email_verified_at, role, password, remember_token, created_at, updated_at)
- LECTURER(lecturer_code, *user_id*, full_name, email, phone, *faculty_code*, created_at, updated_at)
- FACULTIY(faculty_code, name, created_at, updated_at)
- CLASS(class_code, class_name, *faculty_code*, created_at, updated_at)
- STUDENT(student_code, *class_code*, full_name, email, phone, created_at, updated_at)
- SUBJECT(subject_code, name, credit, created_at, updated_at)
- EXAM_SCHEDULE(id, *subject_code*, exam_date, exam_time, duration, room, note, created_at, updated_at)
- EXAM_SUPERVISOR(id, *exam_schedule_id*, *lecturer_code*, created_at, updated_at)
- ATTENDANCE_RECORD(id, *exam_schedule_id*, *student_code*, rekognition_result, confidence, attendance_time, created_at, updated_at)
- STUDENT_REKOGNITION(rekognition_id, *student_code*)

3.1.5. Mô tả chi tiết các bảng dữ liệu

❑ Bảng USER

Mô tả: Bảng USER (Người dùng) lưu trữ thông tin tài khoản (Quản trị viên, giảng viên) để đăng nhập và sử dụng hệ thống.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
id	bigint(U)	x	x	x	ID người dùng (khóa chính)
name	varchar(191)			x	Tên người dùng
email	varchar(191)		x	x	Email đăng nhập
email_verified_at	timestamp				Thời gian xác thực email
role	enum(...)			x	Vai trò ('Quản trị viên', 'lecturer')
password	varchar(191)			x	Mật khẩu (đã mã hóa)
remember_token	varchar(100)				Token ghi nhớ đăng nhập
created_at	timestamp				Thời gian tạo

updated_at	timestamp				Thời gian cập nhật

❑ Bảng LECTURER

Mô tả: Bảng LECTURER (Giảng viên) lưu trữ thông tin cá nhân của giảng viên, có thể liên kết với một tài khoản USER.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
lecturer_code	varchar(20)	x	x	x	Mã giảng viên (khóa chính)
user_id	bigint(U)		x		ID người dùng (khóa ngoại)
full_name	varchar(100)			x	Họ và tên giảng viên
email	varchar(100)		x		Email giảng viên
phone	varchar(15)		x		Số điện thoại giảng viên
faculty_code	varchar(20)				Mã khoa (khóa ngoại)
created_at	timestamp				Thời gian tạo
updated_at	timestamp				Thời gian cập nhật

❑ Bảng FACULTY

Mô tả: Bảng FACULTY (Khoa) lưu trữ thông tin về các khoa trong trường.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
faculty_code	varchar(20)	x	x	x	Mã khoa (khóa chính)
name	varchar(100)		x	x	Tên khoa
created_at	timestamp				Thời gian tạo
updated_at	timestamp				Thời gian cập nhật

❑ Bảng CLASS

Mô tả: Bảng CLASS (Lớp học) lưu trữ thông tin về các lớp học, mỗi lớp thuộc về một khoa.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
class_code	varchar(20)	x	x	x	Mã lớp (khóa chính)
class_name	varchar(100)			x	Tên lớp
faculty_code	varchar(20)			x	Mã khoa (khóa ngoại)
created_at	timestamp				Thời gian tạo
updated_at	timestamp				Thời gian cập nhật

❑ Bảng STUDENT

Mô tả: Bảng STUDENT (Sinh viên) lưu trữ thông tin cá nhân của sinh viên, mỗi sinh viên thuộc về một lớp.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
student_code	varchar(20)	x	x	x	Mã sinh viên (khóa chính)
class_code	varchar(20)			x	Mã lớp (khóa ngoại)
full_name	varchar(100)			x	Họ và tên sinh viên
email	varchar(100)		x		Email sinh viên
phone	varchar(15)		x		Số điện thoại sinh viên
created_at	timestamp				Thời gian tạo
updated_at	timestamp				Thời gian cập nhật

❑ Bảng SUBJECT

Mô tả: Bảng SUBJECT (Môn học) lưu trữ thông tin về các môn học.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
subject_code	varchar(20)	x	x	x	Mã môn học (khóa chính)
name	varchar(100)			x	Tên môn học
credit	tinyint				Số tín chỉ
created_at	timestamp				Thời gian tạo
updated_at	timestamp				Thời gian cập nhật

❑ Bảng EXAM_SCHEDULE

Mô tả: Bảng EXAM_SCHEDULE (Lịch thi) lưu trữ thông tin chi tiết về các ca thi (môn học, ngày giờ, phòng thi).					
Thuộc tính	Kiểu	K	U	M	Diễn giải
id	bigint(U)	x	x	x	ID lịch thi (khóa chính)
subject_code	varchar(20)			x	Mã môn học (khóa ngoại)
exam_date	date			x	Ngày thi
exam_time	time			x	Giờ thi
duration	int			x	Thời lượng thi (phút)
room	varchar(50)			x	Phòng thi
note	text				Ghi chú
created_at	timestamp				Thời gian tạo
updated_at	timestamp				Thời gian cập nhật

❑ Bảng EXAM_SUPERVISOR

Mô tả: Bảng EXAM_SUPERVISOR (Cán bộ coi thi) dùng để phân công giảng viên coi thi cho một lịch thi cụ thể.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
id	bigint(U)	x		x	ID (khóa chính)
exam_schedule_id	bigint(U)			x	ID lịch thi (khóa ngoại)
lecturer_code	varchar(20)			x	Mã giảng viên (khóa ngoại)
created_at	timestamp				Thời gian tạo
updated_at	timestamp				Thời gian cập nhật

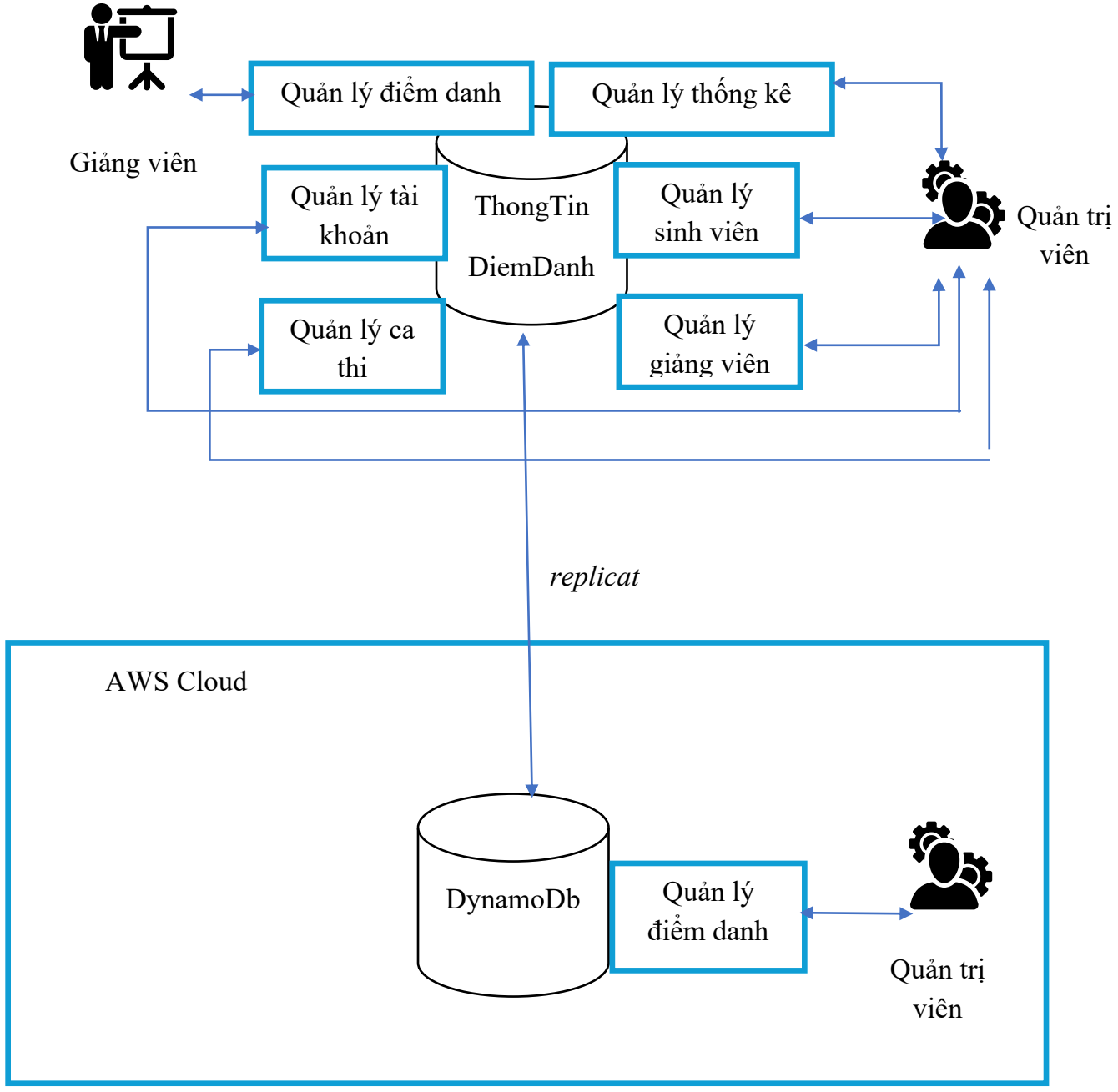
❑ Bảng ATTENDANCE_RECORD

Mô tả: Bảng ATTENDANCE_RECORD (Kết quả điểm danh) lưu lại kết quả điểm danh của từng sinh viên trong một ca thi.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
id	bigint(U)	x	x	x	ID (khóa chính)
exam_schedule_id	bigint(U)			x	ID lịch thi (khóa ngoại)
student_code	varchar(20)			x	Mã sinh viên (khóa ngoại)
rekognition_result	enum(...)				Kết quả nhận diện
confidence	decimal(5,2)				Độ tin cậy của nhận diện
attendance_time	timestamp				Thời gian điểm danh
created_at	timestamp				Thời gian tạo
updated_at	timestamp				Thời gian cập nhật

❑ Bảng STUDENT_REKOGNITION (DynamoDb)

Mô tả: Lưu trữ dữ liệu nhận diện khuôn mặt từ AWS Rekognition trong DynamoDB. Mỗi sinh viên có thể có nhiều bản ghi nhận diện.					
Thuộc tính	**Kiểu**	**K**	**U**	**M**	**Diễn giải**
rekognitionId	String	x	x	x	Mã định danh (ID) của kết quả từ Rekognition
student_code	String			x	Mã số sinh viên được nhận diện

3.2. Thiết kế kiến trúc hệ thống



Hình 3-2 Kiến trúc tổng thể

- Các đối tượng tham gia khai thác
    - Quản trị viên
    - Giảng viên
- Các module cần có
    - Quản lý tài khoản
    - Quản lý sinh viên
    - Quản lý giảng viên
    - Quản lý ca thi
    - Quản lý điểm danh
    - Quản lý thống kê
- Các CSDL
    - CSDL Thông tin điểm danh
    - CSDL DynamoDb

Mô tả module quản lý sinh viên

- **Công dụng của module**: Để quản lý toàn bộ thông tin sinh viên trong hệ thống, bao gồm việc lưu trữ, cập nhật, tra cứu thông tin cá nhân.
- **Dữ liệu vào (input data)**:
    - File Excel chứa danh sách sinh viên (Mã số sinh viên, họ tên, lớp, email, số điện thoại)
    - Thông tin sinh viên nhập thủ công qua form
    - Tiêu chí tìm kiếm/lọc sinh viên
- **Dữ liệu ra (output data)**:
    - Danh sách sinh viên hiển thị trên giao diện
    - Thông tin chi tiết của từng sinh viên
    - Kết quả thêm/sửa/xóa sinh viên
- **User sử dụng module này:** quản trị viên

Mô tả module quản lý giảng viên

- **Mô tả module M2: Công dụng của module**: Để quản lý thông tin giảng viên trong hệ thống, lưu trữ thông tin cá nhân, liên kết với tài khoản giảng viên để phục vụ cho việc phân công ca thi.
- **Dữ liệu vào (input data)**:
    - File excel chứa danh sách giảng viên (Mã giảng viên, họ tên, khoa, email, số điện thoại)

    - Thông tin giảng viên nhập thủ công qua form
    - Mã tài khoản liên kết
    - Tiêu chí tìm kiếm/lọc giảng viên
- **Dữ liệu ra (output data)**:
    - Danh sách giảng viên hiển thị trên giao diện
    - Thông tin chi tiết của từng giảng viên
    - Kết quả thêm/sửa/xóa giảng viên
- **User sử dụng module này**: quản trị viên

Mô tả module quản lý ca thi

- **Công dụng của module**: Để quản lý thông tin các ca thi, bao gồm việc tạo lịch thi, phân công giảng viên coi thi, và quản lý danh sách sinh viên dự thi trong từng ca.
- **Dữ liệu vào (input data)**:
    - File Excel chứa danh sách ca thi (mã ca thi, tên môn thi, thời gian, phòng thi, danh sách sinh viên)
    - Thông tin ca thi nhập thủ công qua form
    - Thông tin phân công giảng viên coi thi
    - Tiêu chí tìm kiếm/lọc ca thi (theo giờ thi, phòng thi, môn thi, ngày thi)
- **Dữ liệu ra (output data)**:
    - Danh sách ca thi hiển thị trên giao diện
    - Thông tin chi tiết của từng ca thi
    - Danh sách sinh viên trong ca thi
    - Thông tin giảng viên được phân công
    - Kết quả thêm/sửa/xóa ca thi
- **User sử dụng module này**: quản trị viên

Mô tả module quản lý điểm danh

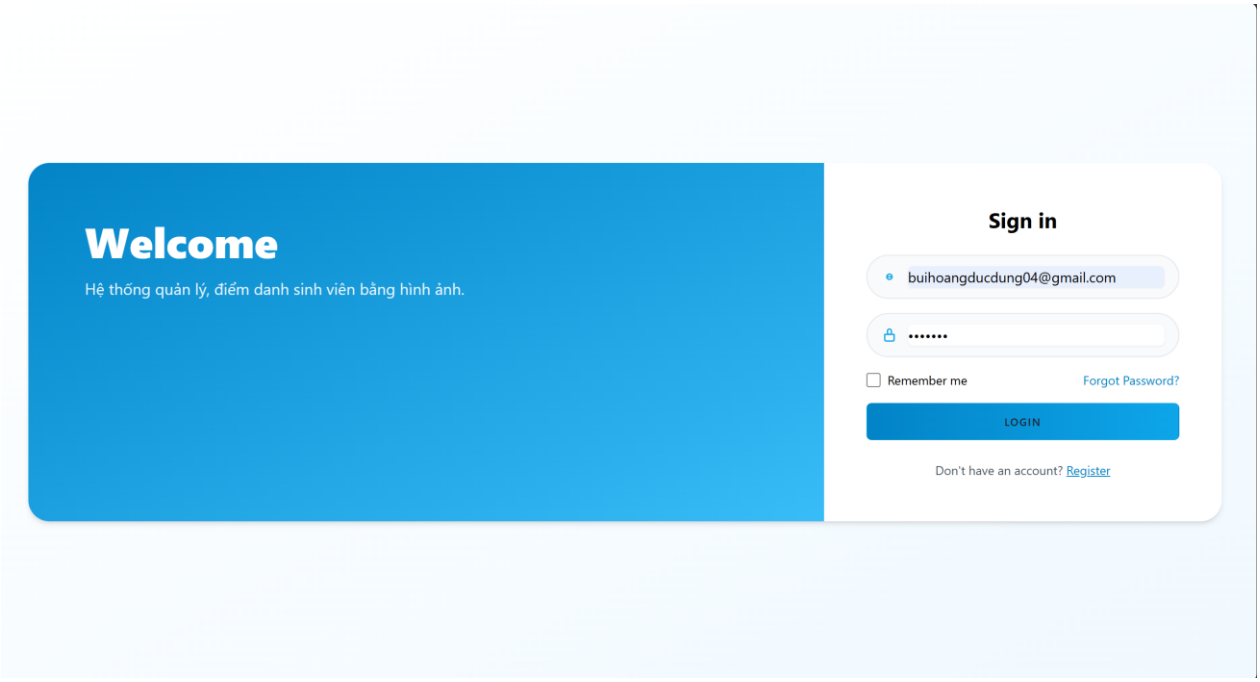
- **Công dụng của module**: Để thực hiện việc điểm danh sinh viên tự động bằng nhận diện khuôn mặt sử dụng AWS Rekognition, lưu trữ dữ liệu điểm danh và cho phép chỉnh sửa trạng thái điểm danh.
- **Dữ liệu vào (input data)**:
    - File ảnh mẫu để đăng ký khuôn mặt sinh viên
    - File ảnh chụp sinh viên trong ca thi
    - Mã ca thi cần điểm danh
    - Trạng thái điểm danh thủ công (có mặt/vắng mặt) khi giảng viên chỉnh sửa
    - Thời gian điểm danh
    - Dữ liệu nhận diện từ AWS Rekognition
- **Dữ liệu ra (output data)**:
    - Thông báo tải hình ảnh trên S3 thành công/thất bại
    - Danh sách sinh viên trong ca thi với trạng thái điểm danh
    - Kết quả nhận diện khuôn mặt từ AWS Rekognition
    - Thời gian điểm danh tự động của hệ thống
    - Thông báo điểm danh thành công/thất bại

- File báo cáo Excel xuất ra

- **User sử dụng module này**:
  - Quản trị viên
  - Giảng viên

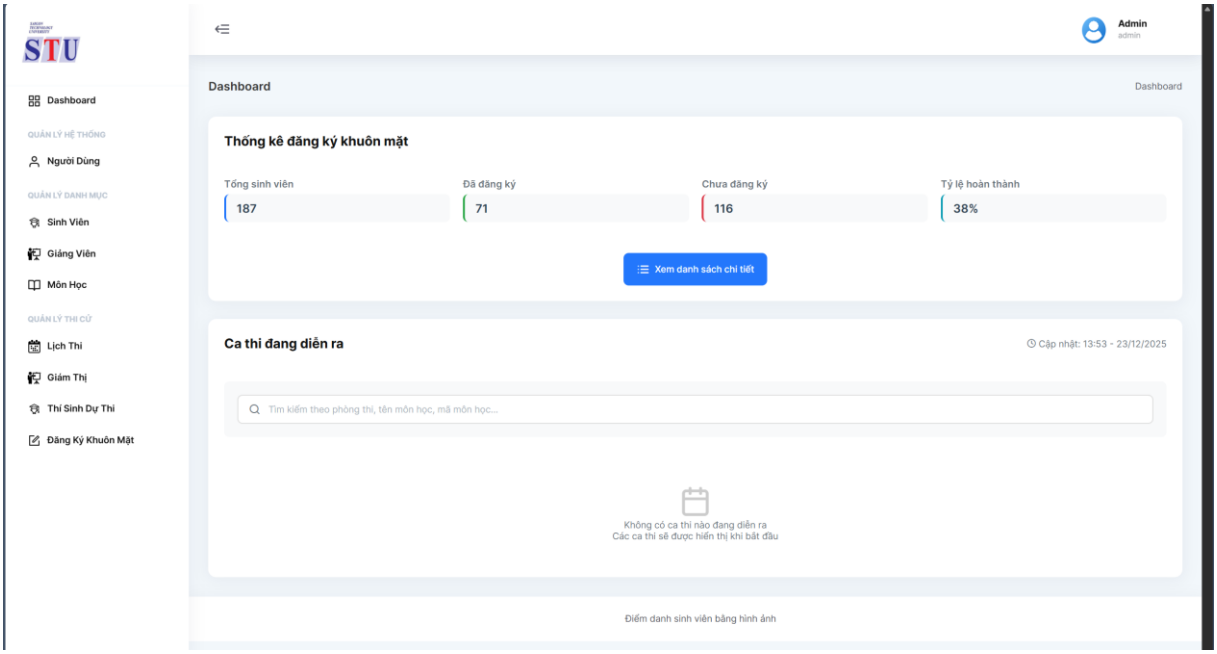
3.3. Thiết kế giao diện người dùng

3.3.1. Màn hình đăng nhập



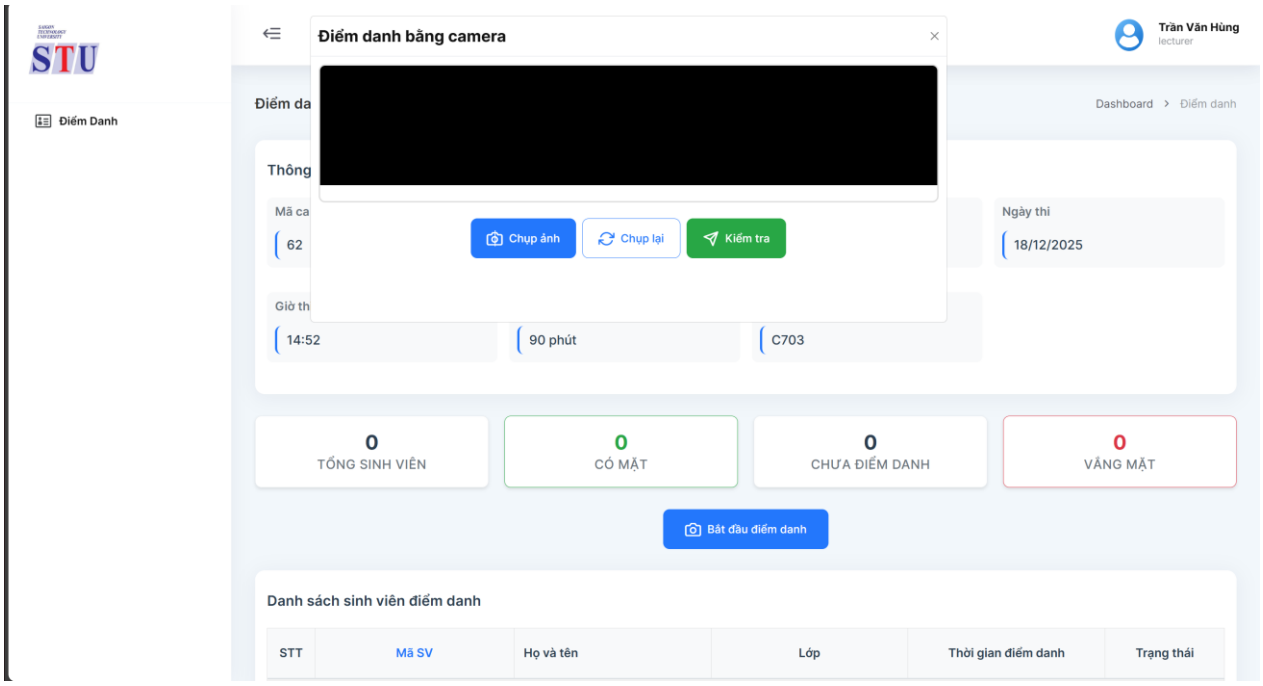
Hình 3-3 Màn hình đăng nhập

3.3.2. Dashboard Admin



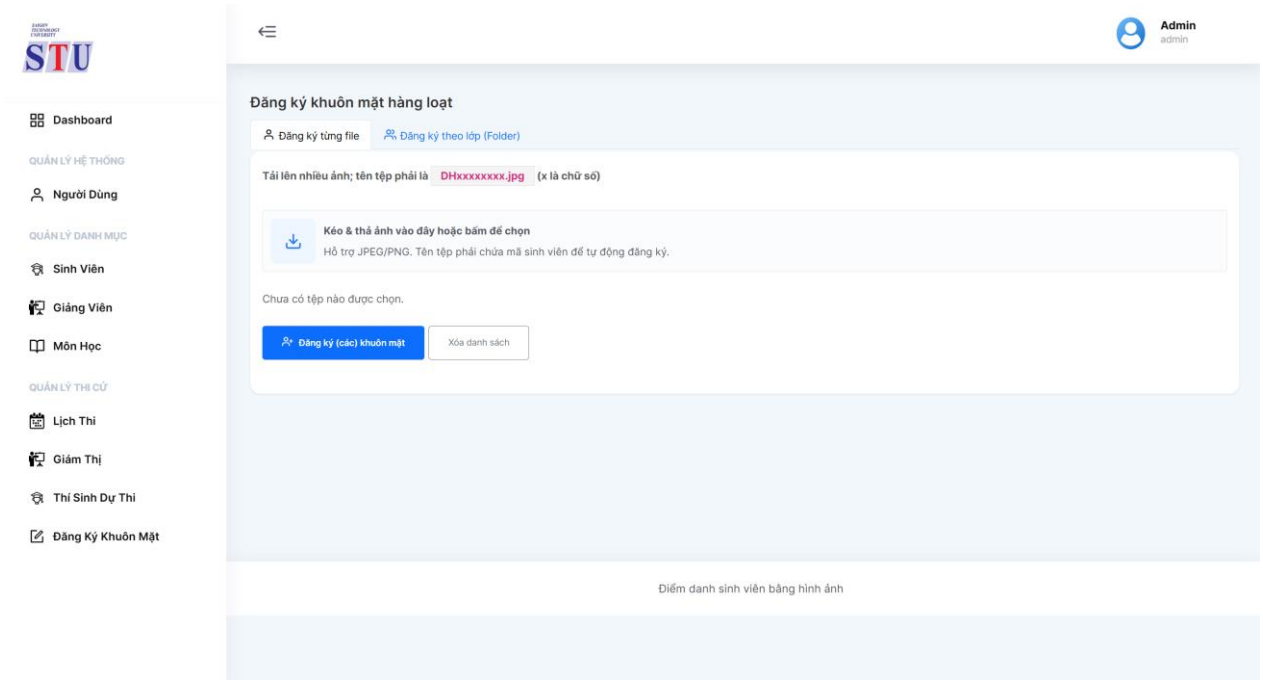
Hình 3-4 Màn hình Dashboard

3.3.3. Màn hình điểm danh



Hình 3-5 Màn hình điểm danh

3.3.4. Màn hình đăng ký khuôn mặt



Hình 3-6 Màn hình đăng ký khuôn mặt

Chương 4. **THỬ NGHIỆM**

4.1. Các kịch bản thử nghiệm

4.1.1. Kịch bản 1: Đăng ký khuôn mặt thành công

- Input: Sinh viên chụp 5 ảnh rõ nét, ánh sáng tốt
- Expected: Hệ thống lưu thành công, Face ID được tạo

4.1.2. Kịch bản 2: Điểm danh chính xác

- Input: Sinh viên đã đăng ký đứng trước camera
- Expected: Nhận diện đúng, confidence > 80%, ghi nhận điểm danh

4.1.3. Kịch bản 3: Xử lý ảnh mờ

- Input: Ảnh chất lượng kém, mờ
- Expected: Hệ thống từ chối, yêu cầu chụp lại

4.1.4. Kịch bản 4: Điểm danh người lạ

- Input: Người chưa đăng ký khuôn mặt
- Expected: Không tìm thấy match, thông báo không xác định

4.1.5. Kịch bản 5: Điểm danh đồng thời

- Input: 30 sinh viên điểm danh trong 5 phút
- Expected: Tất cả được xử lý, thời gian phản hồi < 2s/người

4.2. Kết quả thử nghiệm

Kịch bản	Trạng thái	Kết quả	Ghi chú
1	✓ Pass	97% thành công	3% do ảnh bị che khuất
2	✓ Pass	92% chính xác	Ánh sáng tốt, góc thẳng
3	✓ Pass	Từ chối 100% ảnh mờ	Hoạt động đúng
4	✓ Pass	100% từ chối	Không có false positive
5	✓ Pass	Trung bình 1.5s/người	Đáp ứng yêu cầu

4.3. Xử lý các trường hợp ngoại lệ

Ngoại lệ	Giải pháp
Mất kết nối camera	Hiển thị lỗi, hướng dẫn kiểm tra phần cứng, chuyển sang điểm danh thủ công
AWS Rekognition timeout	Retry 3 lần, nếu thất bại chuyển manual mode
S3 upload failed	Queue lại request, thử upload sau 30s
Ánh sáng kém	Hiển thị cảnh báo, hướng dẫn di chuyển đến vị trí sáng hơn
Đeo khẩu trang	Confidence giảm, yêu cầu xác nhận bằng MSSV hoặc điểm danh thủ công

Chương 5. **KẾT LUẬN**

5.1 Kết quả đối chiếu với mục tiêu

STT	Kết quả	Mục tiêu	Đạt được	Đánh giá
1	Đăng ký khuôn mặt	≥ 95%	97%	Đạt
2	Nhận diện chính xác	≥ 90%	92%	Đạt
3	Thời gian xử lý	≤ 2s	1.5s	Đạt
4	Xử lý đồng thời	≥ 30 người	30 người	Đạt
5	Bảo mật	100%	100%	Đạt
6	Tính sẵn sàng	≥ 99%	99.2%	Đạt

Giải thích:

- Tất cả các chỉ số đều đạt hoặc vượt mục tiêu đề ra
- Độ chính xác nhận diện 92% là kết quả tốt trong điều kiện ánh sáng tự nhiên
- Hiệu năng đáp ứng tốt cho quy mô lớp học trung bình

5.2 Các vấn đề còn tồn đọng

1. Độ chính xác trong điều kiện xấu: Khi ánh sáng kém hoặc sinh viên đeo khẩu trang, độ chính xác giảm xuống 70-75%
2. Chi phí AWS: Với số lượng sinh viên lớn (> 1000), chi phí Rekognition có thể tăng cao
3. Offline mode: Hệ thống chưa hỗ trợ hoạt động khi mất kết nối internet
4. Báo cáo nâng cao: Chưa có phân tích xu hướng, dự đoán tỷ lệ vắng mặt
5. Tích hợp: Chưa tích hợp với hệ thống quản lý sinh viên hiện có của trường

5.3 Hướng phát triển

5.3.1. Ngắn hạn (3-6 tháng)

- Phát triển mobile app cho sinh viên tự điểm danh

5.3.2. Trung hạn (6-12 tháng)

- Tích hợp AI phân tích hành vi (phát hiện gian lận: dùng ảnh, video)
- Xây dựng dashboard analytics với biểu đồ thời gian thực
- Hỗ trợ đa ngôn ngữ (Tiếng Anh, Tiếng Việt)
- Tích hợp với LMS (Learning Management System)

5.3.3. Dài hạn (> 12 tháng)

- Mở rộng sang nhận diện cảm xúc (emotion detection) để đánh giá sự tập trung
- Áp dụng Edge Computing để giảm độ trễ và chi phí cloud
- Phát triển chế độ offline với sync sau
- Tích hợp blockchain để bảo mật dữ liệu điểm danh
- Mở rộng sang các tổ chức, doanh nghiệp

5.4 Ước tính chi phí

Số lượng sinh viên: 2.500 người.

Số môn thi trung bình: 7 môn/sinh viên.

Tổng số lượt điểm danh (Transactions): $2.500 *7 = 17.500$ lượt/kỳ thi.

Dịch vụ	Đơn vị tính	Đơn giá	Thành tiền
Amazon Rekognition	17.500 lượt Search	$1.00 / 1.000 ảnh	$17.50
Rekognition Storage	2.500 Face Metadata	$0.01 / 1.000 Face/tháng	$0.025
Amazon S3 (Storage)	~5.25 GB dữ liệu	$0.023 / GB	$0.12
Amazon S3 (Requests)	17.500 lượt PUT	$0.005 / 1.000 PUT	$0.09
AWS Lambda	17.500 Invocations	Free Tier (Dưới 1M reqs)	$0.00
Amazon DynamoDB	Metadata sinh viên	Free Tier (Dưới 25GB)	$0.00
TỔNG CỘNG			~$17.735

Chương 6. **Phụ lục : HƯỚNG DẪN CÀI ĐẶT VÀ SỬ DỤNG HỆ THỐNG**

6.1 Hướng dẫn cài đặt hệ thống

6.1.1 Mục đích

Hướng dẫn các bước triển khai hệ thống điểm danh sinh viên bằng nhận diện khuôn mặt trên nền tảng AWS, đảm bảo hệ thống hoạt động ổn định, bảo mật và sẵn sàng đưa vào sử dụng thực tế.

6.1.2 Môi trường triển khai

- Nền tảng điện toán đám mây Amazon Web Services (AWS)
- Cơ sở dữ liệu: DynamoDb
- Lưu trữ hình ảnh: AWS S3
- Dịch vụ nhận diện khuôn mặt: AWS Rekognition
- Hosting triển khai hệ thống cho người dùng truy cập qua Internet

6.1.3 Các bước cài đặt

Bước 1: Tạo S3 Buckets (Lưu trữ hình ảnh)

Bước đầu tiên là tạo nơi lưu trữ hình ảnh cho sinh viên.

- Truy cập **S3 Console**.
- Tạo bucket: **Student Image Storage:** Dùng để lưu ảnh đăng ký của sinh viên.
- Cấu hình: Chọn vùng (Region) là ap-southeast-1, giữ nguyên các cài đặt mặc định khác (Block Public Access...).

Bước 2: Tạo IAM Role cho Lambda

Tạo quyền cho phép Lambda function truy cập vào các dịch vụ khác như S3, DynamoDB, và Rekognition.

- Truy cập **IAM Console** -> **Roles** -> **Create role**.
- Chọn **AWS Service** và chọn **Lambda**.
- Gán các quyền (Permissions) sau:
    - CloudWatchLogsFullAccess (để ghi log).
    - AmazonS3FullAccess (để đọc/ghi ảnh).
    - AmazonDynamoDBFullAccess (để lưu thông tin sinh viên).

- AmazonRekognitionFullAccess (để xử lý nhận diện khuôn mặt).
- Đặt tên Role (ví dụ: StudentRegistrationRole) và tạo.

Bước 3: Tạo và Cấu hình Lambda Function "Registration" (Đăng ký)

Function này sẽ chạy khi có ảnh mới được tải lên bucket của sinh viên.

- Truy cập **Lambda Console** -> **Create function**.
- Đặt tên: StudentRegistration.
- Runtime: Chọn **Python** (phiên bản mới nhất).
- **Permissions:** Chọn "Use an existing role" và chọn Role vừa tạo ở Bước 2.
- **Cấu hình Trigger:**
  - Nhấn **Add trigger**.
  - Chọn nguồn là **S3**.
  - Chọn bucket Student Image Storage.
  - Event type: All object create events (hoặc PUT).
  - Nhấn Add.

Bước 4: Tạo bảng DynamoDB (Cơ sở dữ liệu)

Dùng để lưu thông tin khuôn mặt đã được phân tích.

- Truy cập **DynamoDB Console** -> **Create table**.
- Đặt tên bảng: Student.
- **Partition key:** Đặt là recognitionId (kiểu String).
- Nhấn Create.

Bước 5: Viết Code cho Lambda "Registration"

Viết code Python để xử lý ảnh khi được upload.

- Logic chính của code:
  1. Lấy tên bucket và tên file ảnh từ sự kiện (event) trigger từ S3.
  2. Gọi thư viện boto3 để kết nối Rekognition.
  3. Sử dụng hàm index_faces của Rekognition để phân tích khuôn mặt và tạo ra một faceId.
  4. Trích xuất Tên (First Name) và Họ (Last Name) từ tên file ảnh (quy tắc đặt tên file: Ten_Ho.jpg).
  5. Lưu thông tin (recognitionId, firstName, lastName) vào bảng DynamoDB.
- Deploy code lên Lambda.

Code tham khảo:

```
import boto3
import urllib.parse

s3 = boto3.client('s3')
rekognition = boto3.client('rekognition', region_name='ap-southeast-1')
dynamodb = boto3.resource('dynamodb', region_name='ap-southeast-1')
studentTable = dynamodb.Table('student')

def lambda_handler(event, context):
```

```
    print("Event:", event)


    bucket = event['Records'][0]['s3']['bucket']['name']
    key =
urllib.parse.unquote_plus(event['Records'][0]['s3']['object']['key'])

    print(f"Processing image: {key} from bucket: {bucket}")

    try:
        student_code = key.split('/')[-1].split('.')[0]

        response = index_student_image(bucket, key)
        print("Rekognition response:", response)

        if response['ResponseMetadata']['HTTPStatusCode'] == 200:
            if len(response['FaceRecords']) > 0:
                faceId = response['FaceRecords'][0]['Face']['FaceId']
                confidence =
response['FaceRecords'][0]['Face']['Confidence']

                register_student(faceId, student_code, confidence)

                return {
                    'statusCode': 200,
                    'body': f'Successfully registered {student_code}
with faceId {faceId}'
                }
            else:
                print("No face detected in image")
                return {
                    'statusCode': 400,
                    'body': 'No face detected in image'
                }

    except Exception as e:
        print(f"Error: {str(e)}")
        raise e

def index_student_image(bucket, key):
    response = rekognition.index_faces(
        Image={
            'S3Object': {
                'Bucket': bucket,
                'Name': key
            }
        },
        CollectionId='students',
        ExternalImageId=key.split('/')[-1].split('.')[0],
```

```
            DetectionAttributes=['ALL'],
            MaxFaces=1,
            QualityFilter='AUTO'
        )
        return response

def register_student(faceId, student_code, confidence):
    studentTable.put_item(
        Item={
            'rekognitionId': faceId,
            'student_code': student_code,
            'confidence': str(round(confidence, 2)),
            'timestamp':
str(boto3.client('sts').get_caller_identity()['Account'])
        }
    )
    print(f"Registered student {student_code} with faceId {faceId}")
```

Bước 6: Tạo Rekognition Collection (Bằng AWS CLI)

Tạo một tập hợp (collection) để chứa các khuôn mặt đã đăng ký. Bước này phải làm bằng dòng lệnh (CLI) vì Console không hỗ trợ.

- Cài đặt và cấu hình AWS CLI trên máy tính.
- Chạy lệnh:

  Bash

  aws rekognition create-collection --collection-id students--region ap-southeast-1
- *(Lưu ý: studentslà tên collection ID sẽ dùng trong code Lambda).*

Bước 7: Kiểm tra Quy trình Đăng ký (Test Flow 1)

- Upload một ảnh mẫu (ví dụ: Bill_Gates.jpg) lên S3 bucket sinh viên.
- Kiểm tra CloudWatch Logs để xem Lambda có chạy không.
- Kiểm tra DynamoDB để xem dữ liệu nhân viên đã được tạo chưa.

Bước 8: Tạo Lambda Function "Authentication" (Xác thực)

Function này dùng để kiểm tra ảnh của sinh viên được điểm danh.

- Tạo Lambda function mới tên EmployeeAuthentication.
- Runtime: **Python**.
- Role: Sử dụng lại Role đã tạo ở Bước 2.

Bước 9: Viết Code cho Lambda "Authentication"

- Logic chính của code:
  1. Nhận tên file ảnh từ sự kiện.
  2. Đọc ảnh từ S3 bucket Visitor Image Storage.
  3. Gọi Rekognition: search_faces_by_image để so sánh ảnh sinh viên với collection students.
  4. Nếu tìm thấy khuôn mặt khớp (Match): Lấy FaceId, tra cứu trong DynamoDB để lấy tên nhân viên và trả về kết quả thành công.
  5. Nếu không tìm thấy: Trả về kết quả thất bại.

Code tham khảo:

```
import boto3
import json
import base64
from decimal import Decimal

rekognition = boto3.client('rekognition', region_name='ap-southeast-1')
dynamodb = boto3.resource('dynamodb', region_name='ap-southeast-1')

studentTable = dynamodb.Table('student')

def lambda_handler(event, context):
    print("Event:", json.dumps(event, default=str))

    try:
        if 'body' in event:
            body = json.loads(event['body']) if isinstance(event['body'], str)
else event['body']
        else:
            body = event

        image_base64 = body.get('image')

        if not image_base64:
            return build_response(400, {
                'success': False,
                'message': 'Missing required parameter: image (base64)'
            })

        try:
            if ',' in image_base64:
                image_base64 = image_base64.split(',')[1]

            image_bytes = base64.b64decode(image_base64)
        except Exception as e:
            return build_response(400, {
                'success': False,
                'message': f'Invalid base64 image: {str(e)}'
            })


        match_result = search_face_by_bytes(image_bytes)

        if not match_result['success']:
            return build_response(404, {
                'success': False,
                'message': match_result['message']
            })
```

```
        face_id = match_result['face_id']
        student_code = match_result['student_code']
        confidence = match_result['confidence']

        student_info = get_student_info(face_id)

        if not student_info:
            return build_response(404, {
                'success': False,
                'message': f'Student with faceId {face_id} not found in
database'
            })

        if student_info.get('student_code') != student_code:
            print(f"Warning: student_code mismatch. Rekognition:
{student_code}, DynamoDB: {student_info.get('student_code')}")

        return build_response(200, {
            'success': True,
            'message': 'Face recognition successful',
            'data': {
                'student': student_info,
                'confidence': confidence,
                'face_id': face_id,
                'rekognition_result': 'match'
            }
        })

    except Exception as e:
        print(f"Error: {str(e)}")
        import traceback
        traceback.print_exc()
        return build_response(500, {
            'success': False,
            'message': f'Internal server error: {str(e)}'
        })


def search_face_by_bytes(image_bytes):
    try:
        response = rekognition.search_faces_by_image(
            CollectionId='students',
            Image={
                'Bytes': image_bytes
            },
            MaxFaces=1,
            FaceMatchThreshold=80.0
        )
```

```
        if not response['FaceMatches']:
            return {
                'success': False,
                'message': 'No matching face found. Please try again or
register your face first.'
            }

        best_match = response['FaceMatches'][0]
        face_id = best_match['Face']['FaceId']
        student_code = best_match['Face']['ExternalImageId']
        confidence = round(best_match['Similarity'], 2)

        print(f"Match found - FaceId: {face_id}, Student: {student_code},
Confidence: {confidence}%")

        return {
            'success': True,
            'student_code': student_code,
            'face_id': face_id,
            'confidence': confidence
        }

    except rekognition.exceptions.InvalidParameterException as e:
        print(f"Invalid image: {str(e)}")
        return {
            'success': False,
            'message': 'Invalid image format or no face detected in image'
        }
    except Exception as e:
        print(f"Rekognition error: {str(e)}")
        return {
            'success': False,
            'message': f'Face recognition error: {str(e)}'
        }


def get_student_info(face_id):
    try:
        response = studentTable.get_item(
            Key={'rekognitionId': face_id}
        )

        if 'Item' not in response:
            print(f"Student not found with faceId: {face_id}")
            return None

        print(f"Student found: {response['Item']}")
        return response['Item']
```

```
    except Exception as e:
        print(f"Error getting student info: {str(e)}")
        return None


def build_response(status_code, body=None):
    response = {
        'statusCode': status_code,
        'headers': {
            'Content-Type': 'application/json',
            'Access-Control-Allow-Origin': '*',
            'Access-Control-Allow-Headers': 'Content-Type,X-Amz-
Date,Authorization,X-Api-Key,X-Amz-Security-Token',
            'Access-Control-Allow-Methods': 'OPTIONS,POST,GET'
        }
    }

    if body is not None:
        response['body'] = json.dumps(body, default=decimal_default)

    return response


def decimal_default(obj):
    if isinstance(obj, Decimal):
        return float(obj)
    raise TypeError
```

Bước 10: Tải source code từ link github:
https://github.com/ducdung0212/quanlydiemdanh.git

- Sửa tên file .env.example thành .env sau đó điền các dữ liệu cần thiết:
  DB_CONNECTION=
  DB_HOST=
  DB_PORT=
  DB_DATABASE=
  DB_USERNAME=
  DB_PASSWORD=
  AWS_ACCESS_KEY_ID=
  AWS_SECRET_ACCESS_KEY=
  AWS_DEFAULT_REGION=
  AWS_BUCKET=

Bước 11: Deploy lên hosting

6.2 Hướng dẫn sử dụng cho quản trị viên

USER: buihoangducdung04@gmail.com

PASSWORD: 1234567

- **Quản lý danh sách sinh viên**

Chức năng:

- Thêm mới sinh viên.
- Chỉnh sửa thông tin sinh viên.
- Xóa sinh viên khỏi hệ thống.
- Xem danh sách sinh viên theo lớp, môn học hoặc ca thi.

Quy trình thực hiện:

1. Admin đăng nhập hệ thống.
2. Chọn chức năng Quản lý sinh viên.
3. Thực hiện thêm, sửa hoặc xóa thông tin sinh viên.

- **Quản lý danh sách giảng viên**

Chức năng:

- Thêm mới giảng viên.
- Chỉnh sửa thông tin giảng viên.
- Xóa giảng viên không còn phụ trách.
- Xem danh sách giảng viên trong hệ thống.

Quy trình thực hiện:

1. Admin đăng nhập hệ thống.
2. Chọn chức năng Quản lý giảng viên.
3. Thực hiện các thao tác quản lý cần thiết.
4. Lưu thông tin vào hệ thống.

- **Quản lý ca thi**

Chức năng:

- Tạo ca thi mới.
- Chỉnh sửa thông tin ca thi (môn học, thời gian, phòng thi).
- Xóa ca thi khi cần thiết.
- Xem danh sách các ca thi đã tạo.

Quy trình thực hiện:

1. Admin chọn chức năng Quản lý ca thi.
2. Nhập thông tin ca thi.
3. Lưu ca thi vào hệ thống.

- **Phân công giảng viên coi thi**

Chức năng:

- Chọn giảng viên cho từng ca thi.
- Thay đổi giảng viên coi thi khi cần thiết.

Quy trình thực hiện:

1. Admin chọn ca thi cần phân công.
2. Chọn giảng viên coi thi.
3. Xác nhận và lưu phân công.

- **Thêm danh sách sinh viên vào ca thi**

Chức năng:

- Thêm sinh viên vào ca thi.
- Xóa sinh viên khỏi ca thi khi có điều chỉnh.

Quy trình thực hiện:

1. Admin chọn ca thi.
2. Chọn chức năng Thêm sinh viên vào ca thi.
3. Chọn danh sách sinh viên tương ứng.
4. Lưu danh sách.

- **Đăng ký khuôn mặt sinh viên bằng file ảnh**

Chức năng:

- Đăng ký khuôn mặt cho sinh viên bằng hình ảnh.
- Quản lý ảnh khuôn mặt đã đăng ký.

Quy trình thực hiện:

1. Admin đăng nhập hệ thống.
2. Chọn sinh viên cần đăng ký khuôn mặt.
3. Chọn chức năng Tải ảnh khuôn mặt.
4. Upload file ảnh từ thiết bị.
5. Hệ thống tự động lưu ảnh lên AWS S3.

6.3 Hướng dẫn sử dụng cho giảng viên

USER: hung.tranvan@stu.edu.vn

PASSWORD: 123456

Sau khi đăng nhập, giảng viên có thể:

- Thực hiện điểm danh sinh viên bằng nhận diện khuôn mặt.
- Tìm kiếm sinh viên để biết thông tin phòng thi, hỗ trợ xử lý trường hợp sinh viên đi nhầm phòng.
- Xem danh sách sinh viên thuộc ca thi mình phụ trách.

Quy trình sử dụng cho giảng viên:

Bước 1: Đăng nhập hệ thống
Giảng viên đăng nhập bằng tài khoản được cấp.

Bước 2: Hiển thị ca thi tự động
Hệ thống tự động hiển thị ca thi mà giảng viên được phân công, không cần chọn ca thi thủ công.

Bước 3: Điểm danh sinh viên

- Giảng viên nhấn Bắt đầu điểm danh.
- Hệ thống kích hoạt camera và tự động nhận diện khuôn mặt sinh viên.
- Kết quả điểm danh được lưu vào cơ sở dữ liệu.

Bước 4: Tra cứu sinh viên

- Giảng viên nhập mã số sinh viên hoặc họ tên.
- Hệ thống hiển thị thông tin ca thi và phòng thi tương ứng của sinh viên.
- Hỗ trợ xử lý trường hợp sinh viên đi nhầm phòng.

Bước 5: Kiểm tra và kết thúc

- Giảng viên kiểm tra kết quả điểm danh.
- Kết thúc ca thi và xuất báo cáo khi cần.

TÀI LIỆU THAM KHẢO

[1] Amazon Web Services, Inc. (2024). AWS Rekognition Developer Guide. Retrieved from https://docs.aws.amazon.com/rekognition/

[2] Wittig, M., & Wittig, A. (2023). Amazon Web Services in Action (3rd ed.). Manning Publications.

[3] Laravel LLC(2024). Laravel 10.x Documentation. Retrieved from https://laravel.com/docs/10.x

ĐÁNH GIÁ ĐÓNG GÓP CỦA NHÓM

Phân công công việc

- Bùi Hoàng Đức Dũng:

  + Phát triển Backend bằng Laravel.

  + Thiết kế và triển khai các chức năng xử lý nghiệp vụ.

  + Tích hợp các dịch vụ AWS (Rekognition, S3, DynamoDB).

  + Xử lý luồng điểm danh bằng nhận diện khuôn mặt.

  + Tham gia thử nghiệm và xử lý các trường hợp ngoại lệ.

  + Tham gia xây dựng tài liệu báo cáo.

- Võ Hoàng Tuấn:

  + Thiết kế và xây dựng giao diện người dùng bằng Blade.

  + Phát triển các màn hình quản lý hệ thống.

  + Kết nối giao diện với Backend Laravel.

  + Thiết kế luồng thao tác người dùng.

  + Tham gia xây dựng tài liệu báo cáo.

Mức độ đóng góp

STT	Họ tên	Công việc chính	Tỷ lệ
1	Bùi Hoàng Đức Dũng	Backend Laravel, AWS Cloud	50%
2	Võ Hoàng Tuấn	Frontend Blade, giao diện hệ thống	50%