**ĐỀ TÀI: XÂY DỰNG HỆ THỐNG ĐIỂM DANH BẰNG NHẬN DIỆN KHUÔN MẶT**

**LỜI CẢM ƠN**

Để hoàn thành bài báo cáo đồ án tốt nghiệp này, em xin gửi lời cảm ơn chân thành và sâu sắc nhất đến thầy/cô (Tên giảng viên hướng dẫn), người đã tận tình chỉ bảo, hướng dẫn và cung cấp những kiến thức quý báu trong suốt quá trình em thực hiện đề tài.

Em cũng xin gửi lời cảm ơn đến quý thầy cô trong khoa (Tên khoa) của trường (Tên trường) đã tạo điều kiện và môi trường học tập tốt nhất để em có thể tiếp thu kiến thức và hoàn thành chương trình học.

Cuối cùng, em xin cảm ơn gia đình và bạn bè đã luôn ở bên cạnh động viên, khích lệ và giúp đỡ em trong những lúc khó khăn nhất.

Mặc dù đã rất cố gắng, nhưng do kiến thức và kinh nghiệm còn hạn chế nên bài báo cáo không thể tránh khỏi những thiếu sót. Em rất mong nhận được sự góp ý của quý thầy cô và các bạn để đề tài được hoàn thiện hơn.

Em xin chân thành cảm ơn!

**MỤC LỤC**

**CHƯƠNG 1: GIỚI THIỆU**

1.1. Lý do chọn đề tài

1.2. Mục tiêu của đề tài

1.3. Mô tả đề tài

1.4. Phạm vi và giới hạn

1.5. Phương pháp thực hiện

**CHƯƠNG 2: CƠ SỞ LÝ THUYẾT**

2.1. Tổng quan về vấn đề nghiên cứu

2.2. Các khái niệm, công nghệ liên quan

2.2.1. Trí tuệ nhân tạo (AI) và Nhận diện khuôn mặt (Face Recognition)

2.2.2. Các Framework và Công nghệ sử dụng

2.3. Các giải pháp tương tự

**CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG**

3.1. Khảo sát yêu cầu

3.2. Yêu cầu chức năng và phi chức năng

3.3. Use Case Diagram và mô tả các Use Case

3.4. Biểu đồ hoạt động (Activity Diagram)

3.5. Kiến trúc tổng thể hệ thống

3.6. Thiết kế giao diện người dùng (UI)

3.7. Thiết kế cơ sở dữ liệu

3.8. Thiết kế chi tiết các chức năng chính

**CHƯƠNG 4: CÀI ĐẶT VÀ TRIỂN KHAI**

4.1. Môi trường phát triển

4.2. Cấu trúc thư mục mã nguồn

4.3. Mô tả các chức năng đã cài đặt

4.4. Vấn đề gặp phải và cách khắc phục

**KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

1. Những gì đã đạt được
2. Hạn chế còn tồn tại
3. Hướng phát triển trong tương lai

**TÀI LIỆU THAM KHẢO**

**PHỤ LỤC**

Phụ lục A:

Mã nguồn Phụ lục B:

Cấu hình hệ thống

**(Nội dung chi tiết của từng chương sẽ được trình bày ở các trang tiếp theo)**

**CHƯƠNG 1: GIỚI THIỆU**

**1.1. Lý do chọn đề tài**

Trong bối cảnh cuộc cách mạng công nghiệp 4.0 đang diễn ra mạnh mẽ, việc ứng dụng các công nghệ tiên tiến như Trí tuệ nhân tạo (AI) vào mọi lĩnh vực của đời sống là một xu hướng tất yếu. Giáo dục cũng không nằm ngoài xu thế đó. Một trong những bài toán thực tế và cấp thiết tại các cơ sở giáo dục là công tác quản lý sinh viên và đặc biệt là quy trình điểm danh.

Các phương pháp điểm danh truyền thống như gọi tên, ký giấy không chỉ tốn thời gian, công sức của giảng viên mà còn dễ xảy ra sai sót, gian lận. Việc điểm danh thủ công còn gây gián đoạn bài giảng và không tạo ra được một bộ dữ liệu đồng bộ, khó khăn cho việc thống kê và báo cáo sau này.

Nhận thấy những bất cập đó, đề tài **“Hệ thống điểm danh bằng nhận diện khuôn mặt”** được đề xuất với mục tiêu xây dựng một giải pháp tự động hóa, hiện đại hóa quy trình điểm danh. Hệ thống ứng dụng công nghệ nhận diện khuôn mặt – một nhánh quan trọng của AI – để xác thực danh tính sinh viên một cách nhanh chóng, chính xác và minh bạch. Giải pháp này không chỉ giúp tiết kiệm thời gian, giảm thiểu sai sót mà còn nâng cao trải nghiệm cho cả giảng viên và sinh viên, đồng thời cung cấp một hệ thống quản lý dữ liệu điểm danh thông minh và hiệu quả.

**1.2. Mục tiêu của đề tài**

Đề tài tập trung vào việc xây dựng một hệ thống hoàn chỉnh với các mục tiêu cụ thể sau:

* **Xây dựng hệ thống nhận diện khuôn mặt:** Phát triển một module AI có khả năng phát hiện, trích xuất đặc trưng và nhận diện khuôn mặt của sinh viên từ hình ảnh hoặc video với độ chính xác cao.
* **Tự động hóa quy trình điểm danh:** Cho phép giảng viên tạo và quản lý các phiên điểm danh. Sinh viên có thể thực hiện điểm danh nhanh chóng thông qua ứng dụng di động bằng khuôn mặt của mình.
* **Phát triển hệ thống quản lý đa nền tảng:**
  + **Web Application (React):** Xây dựng một giao diện web cho quản trị viên và giảng viên để quản lý toàn bộ hệ thống, bao gồm quản lý người dùng, lớp học, môn học, lịch học và xem các báo cáo, thống kê điểm danh.
  + **Mobile Application (Flutter):** Xây dựng một ứng dụng di động cho sinh viên và giảng viên. Sinh viên dùng để điểm danh, xem lịch học, lịch sử điểm danh. Giảng viên dùng để tạo và quản lý phiên điểm danh trực tiếp trên lớp.
* **Đảm bảo tính bảo mật và hiệu năng:** Xây dựng hệ thống với kiến trúc vững chắc, bảo mật thông tin người dùng và dữ liệu, đồng thời tối ưu hóa hiệu năng để đáp ứng số lượng lớn người dùng truy cập đồng thời.

**1.3. Mô tả đề tài**

Hệ thống được thiết kế là một giải pháp quản lý điểm danh thông minh, bao gồm ba thành phần chính:

1. **Frontend Web (React):** Một trang quản trị dành cho Admin và Giảng viên.
   * **Admin:** Có toàn quyền quản lý hệ thống, bao gồm quản lý người dùng (sinh viên, giảng viên), phân quyền, quản lý lớp học, môn học, và theo dõi các thống kê tổng quan.
   * **Giảng viên:** Có thể tạo và quản lý các phiên điểm danh cho lớp học mình phụ trách, xem lịch dạy và theo dõi tình hình điểm danh của sinh viên trong thời gian thực.
2. **Mobile App (Flutter):** Một ứng dụng di động tiện lợi dành cho Sinh viên và Giảng viên.
   * **Sinh viên:** Đăng nhập, xem thời khóa biểu, tìm các phiên điểm danh đang hoạt động và thực hiện điểm danh bằng cách chụp ảnh khuôn mặt.
   * **Giảng viên:** Có thể sử dụng các chức năng quản lý phiên điểm danh ngay trên điện thoại.
3. **Backend (Node.js):** Máy chủ xử lý toàn bộ logic nghiệp vụ, quản lý cơ sở dữ liệu và đặc biệt là tích hợp module AI để xử lý nhận diện khuôn mặt. Backend cung cấp các API cho Frontend Web và Mobile App.

**Quy trình điểm danh:**

* **Bước 1:** Giảng viên đăng nhập vào hệ thống (Web hoặc Mobile App) và tạo một phiên điểm danh mới cho lớp học và môn học cụ thể.
* **Bước 2:** Sinh viên đăng nhập vào ứng dụng di động, chọn phiên điểm danh đang hoạt động.
* **Bước 3:** Ứng dụng sẽ mở camera. Sinh viên đưa khuôn mặt vào khung hình. Hệ thống sẽ tự động phát hiện và chụp lại hình ảnh.
* **Bước 4:** Hình ảnh được gửi lên server. Backend sẽ sử dụng mô hình AI đã được huấn luyện để nhận diện và xác thực danh tính của sinh viên.
* **Bước 5:** Kết quả điểm danh (thành công/thất bại) được trả về ngay lập tức cho sinh viên và cập nhật vào hệ thống trong thời gian thực, cho phép giảng viên theo dõi trực tiếp.

**1.4. Phạm vi và giới hạn**

* **Phạm vi:**
  + Hệ thống tập trung vào các chức năng cốt lõi: Quản lý người dùng, lớp học, môn học, lịch học và quy trình điểm danh bằng nhận diện khuôn mặt.
  + Hỗ trợ ba vai trò chính: Quản trị viên, Giảng viên và Sinh viên với các quyền hạn được phân chia rõ ràng.
  + Hệ thống được xây dựng trên hai nền tảng: Web cho quản lý và Mobile (Android) cho người dùng cuối.
  + Module AI sử dụng thuật toán LBPH để huấn luyện và nhận diện.
* **Giới hạn:**
  + Mô hình nhận diện khuôn mặt được huấn luyện dựa trên bộ dữ liệu được tạo ra từ video do người dùng cung cấp, độ chính xác có thể bị ảnh hưởng bởi các yếu tố môi trường như ánh sáng, góc mặt, biểu cảm.
  + Hệ thống chưa tích hợp các biện pháp chống giả mạo nâng cao như liveness detection (phát hiện đối tượng sống).
  + Tính năng thông báo real-time mới ở mức có thể triển khai, chưa hoàn thiện.
  + Hệ thống hiện tại hỗ trợ tốt cho nền tảng Android, việc hỗ trợ iOS vẫn đang trong kế hoạch.

**1.5. Phương pháp thực hiện**

Để thực hiện đề tài, các phương pháp sau đã được áp dụng:

* **Nghiên cứu tài liệu:** Tìm hiểu các kiến thức về Trí tuệ nhân tạo, xử lý ảnh, các thuật toán nhận diện khuôn mặt (Haar Cascade, LBPH), cũng như các công nghệ phát triển Web (React, Node.js) và Mobile (Flutter).
* **Phân tích và thiết kế hệ thống:** Sử dụng phương pháp hướng đối tượng, thiết kế hệ thống dựa trên các biểu đồ UML như Use Case Diagram, Activity Diagram và thiết kế cơ sở dữ liệu quan hệ (ERD).
* **Phát triển theo mô hình Agile:** Chia nhỏ dự án thành các phần (Frontend, Backend, Mobile, AI) và phát triển song song, liên tục tích hợp và kiểm thử để đảm bảo chất lượng.
* **Kiểm thử và đánh giá:** Thực hiện kiểm thử chức năng trên từng module và kiểm thử tích hợp toàn hệ thống. Đánh giá độ chính xác của mô hình AI và hiệu năng của hệ thống.

**CHƯƠNG 2: CƠ SỞ LÝ THUYẾT**

**2.1. Tổng quan về vấn đề nghiên cứu**

Điểm danh là một tác vụ cơ bản nhưng quan trọng trong quản lý giáo dục. Vấn đề đặt ra là làm thế nào để tự động hóa quy trình này một cách hiệu quả, chính xác và bảo mật. Nhận diện khuôn mặt, một lĩnh vực của thị giác máy tính và trí tuệ nhân tạo, nổi lên như một giải pháp sinh trắc học ưu việt. Không giống như thẻ từ hay mật khẩu, khuôn mặt là một đặc điểm độc nhất và khó có thể giả mạo hoặc làm mất.

Nghiên cứu này tập trung vào việc áp dụng kỹ thuật nhận diện khuôn mặt để xây dựng một hệ thống điểm danh tự động. Hệ thống cần giải quyết các bài toán cốt lõi:

1. **Phát hiện khuôn mặt (Face Detection):** Xác định vị trí của một hoặc nhiều khuôn mặt trong một ảnh hoặc video.
2. **Trích xuất đặc trưng (Feature Extraction):** Chuyển đổi dữ liệu khuôn mặt đã được phát hiện thành một tập hợp các đặc trưng số học.
3. **Nhận diện khuôn mặt (Face Recognition):** So sánh các đặc trưng vừa trích xuất với các đặc trưng đã được lưu trữ trong cơ sở dữ liệu để tìm ra danh tính tương ứng.

**2.2. Các khái niệm, công nghệ liên quan**

**2.2.1. Trí tuệ nhân tạo (AI) và Nhận diện khuôn mặt (Face Recognition)**

* **Trí tuệ nhân tạo (AI):** Là lĩnh vực khoa học máy tính nhằm tạo ra các hệ thống thông minh có khả năng thực hiện các nhiệm vụ mà thông thường đòi hỏi trí tuệ của con người.
* **Nhận diện khuôn mặt:** Là một ứng dụng của AI, cho phép máy tính xác định hoặc xác minh danh tính của một người từ hình ảnh kỹ thuật số hoặc một khung hình video. Quy trình này thường bao gồm các bước sau:
  + **Phát hiện khuôn mặt:** Trong dự án này, bộ phân loại **Haar Cascade** được sử dụng để phát hiện khuôn mặt trong ảnh. Đây là một phương pháp dựa trên máy học hiệu quả, sử dụng các đặc trưng giống Haar để xác định các vùng có khả năng chứa khuôn mặt.
  + **Chuẩn hóa và Tiền xử lý:** Sau khi phát hiện, hình ảnh khuôn mặt được cắt ra, chuyển đổi sang ảnh xám và chuẩn hóa kích thước để đảm bảo tính nhất quán.
  + **Huấn luyện mô hình:** Dự án sử dụng thuật toán **LBPH (Local Binary Patterns Histograms)** để huấn luyện mô hình nhận diện. LBPH là một thuật toán nhận diện khuôn mặt hiệu quả, hoạt động bằng cách phân tích các mẫu texture cục bộ của ảnh. Nó có ưu điểm là kháng tốt với sự thay đổi về điều kiện ánh sáng.
  + **Nhận diện:** Khi có một ảnh đầu vào mới, hệ thống sẽ trích xuất đặc trưng LBPH của nó và so sánh với các đặc trưng đã được huấn luyện trong mô hình để tìm ra người có khuôn mặt khớp nhất.

**2.2.2. Các Framework và Công nghệ sử dụng**

Hệ thống được xây dựng dựa trên một ngăn xếp công nghệ hiện đại và mạnh mẽ:

* **Backend (Node.js & Express.js):**
  + **Node.js:** Là một môi trường chạy JavaScript phía máy chủ, cho phép xây dựng các ứng dụng mạng có khả năng mở rộng.
  + **Express.js:** Là một framework ứng dụng web tối giản và linh hoạt cho Node.js, được sử dụng để xây dựng các API RESTful.
  + **Cơ sở dữ liệu:** Sử dụng **MySQL**, một hệ quản trị cơ sở dữ liệu quan hệ phổ biến, kết hợp với connection pool để tối ưu hóa hiệu năng.
  + **Xác thực:** Sử dụng **JSON Web Tokens (JWT)** và **bcrypt** để mã hóa mật khẩu, đảm bảo xác thực an toàn và quản lý phiên đăng nhập.
  + **Tích hợp Python:** Các tác vụ xử lý AI (tạo dataset, huấn luyện, nhận diện) được viết bằng Python với thư viện **OpenCV** và được gọi từ Node.js thông qua child\_process.
* **Frontend Web (React):**
  + **React:** Là một thư viện JavaScript để xây dựng giao diện người dùng, được phát triển bởi Facebook. Dự án sử dụng React phiên bản 18.2.0.
  + **React Router DOM:** Được sử dụng để quản lý việc định tuyến giữa các trang trong ứng dụng web.
  + **State Management:** Sử dụng các React Hooks cơ bản như useState, useEffect để quản lý trạng thái của component.
  + **Styling:** Sử dụng phương pháp CSS-in-JS để tạo kiểu cho các component một cách linh hoạt và có thể tái sử dụng.
* **Mobile App (Flutter):**
  + **Flutter:** Là một UI toolkit của Google để xây dựng các ứng dụng đẹp, được biên dịch tự nhiên cho di động, web và máy tính để bàn từ một cơ sở mã duy nhất. Dự án sử dụng Flutter với ngôn ngữ Dart.
  + **Kiến trúc:** Sử dụng kiến trúc hướng dịch vụ (Service-oriented) với StatefulWidget để quản lý trạng thái.
  + **Tích hợp Camera:** Sử dụng package camera kết hợp với OpenCV để xử lý các tác vụ liên quan đến camera và phát hiện khuôn mặt thời gian thực trên thiết bị.

**2.3. Các giải pháp tương tự**

Trên thị trường hiện nay đã có một số giải pháp điểm danh bằng nhận diện khuôn mặt, tuy nhiên, chúng thường có những nhược điểm như:

* **Chi phí cao:** Các giải pháp thương mại thường yêu cầu chi phí bản quyền và phần cứng đắt đỏ.
* **Thiếu linh hoạt:** Hệ thống thường đóng gói, khó tùy chỉnh và tích hợp với các hệ thống quản lý sinh viên hiện có của trường học.
* **Yêu cầu thiết bị chuyên dụng:** Một số hệ thống đòi hỏi camera chuyên dụng, gây khó khăn trong việc triển khai rộng rãi.

Đề tài này hướng tới việc xây dựng một giải pháp mã nguồn mở, linh hoạt, dễ dàng triển khai và sử dụng các thiết bị phổ thông (webcam, camera điện thoại), giúp giảm thiểu chi phí và tăng khả năng tiếp cận cho các cơ sở giáo dục.

**CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG**

**3.1. Khảo sát yêu cầu**

Qua quá trình khảo sát thực tế tại các trường học và tìm hiểu nhu cầu của các bên liên quan (Quản trị viên, Giảng viên, Sinh viên), các yêu cầu chính đối với hệ thống được xác định như sau:

* **Đối với Quản trị viên:** Cần một công cụ quản lý tập trung, có thể quản lý toàn bộ dữ liệu người dùng, lớp học, môn học và theo dõi hoạt động của hệ thống.
* **Đối với Giảng viên:** Cần một phương pháp điểm danh nhanh chóng, tự động, giảm thiểu công việc thủ công. Họ cần khả năng tạo phiên điểm danh, theo dõi sinh viên tham gia và xuất báo cáo một cách dễ dàng.
* **Đối với Sinh viên:** Cần một cách thức điểm danh đơn giản, tiện lợi, không cần mang theo thẻ sinh viên. Họ muốn xem được lịch học và lịch sử điểm danh của mình.

**3.2. Yêu cầu chức năng và phi chức năng**

**Yêu cầu chức năng (Functional Requirements):**

* **Quản lý xác thực:**
  + Hệ thống phải cho phép người dùng đăng ký, đăng nhập với vai trò cụ thể (Admin, Teacher, Student).
  + Hệ thống phải phân quyền truy cập chức năng dựa trên vai trò của người dùng.
* **Quản lý người dùng:**
  + Admin có thể thực hiện các thao tác CRUD (Tạo, Đọc, Cập nhật, Xóa) với tài khoản người dùng.
  + Admin có thể kích hoạt/vô hiệu hóa tài khoản người dùng.
* **Quản lý lớp học và môn học:**
  + Admin có thể quản lý thông tin lớp học và môn học.
  + Admin có thể thêm/xóa sinh viên khỏi một lớp học.
* **Quản lý lịch học:**
  + Admin có thể tạo và quản lý lịch học, phân công giảng viên cho các lớp và môn học.
* **Đăng ký khuôn mặt:**
  + Hệ thống cho phép người dùng (sinh viên) tải lên video để tạo bộ dữ liệu khuôn mặt.
  + Hệ thống phải tự động huấn luyện lại mô hình sau khi có dữ liệu mới.
* **Quản lý điểm danh:**
  + Giảng viên có thể tạo, quản lý và đóng các phiên điểm danh.
  + Sinh viên có thể xem các phiên điểm danh đang hoạt động và thực hiện điểm danh bằng khuôn mặt.
  + Hệ thống phải cung cấp phản hồi điểm danh (thành công/thất bại) tức thì.
* **Báo cáo và thống kê:**
  + Hệ thống phải cho phép xem lịch sử điểm danh cá nhân và của cả lớp.
  + Admin có thể xem các thống kê tổng quan về hệ thống.

**Yêu cầu phi chức năng (Non-functional Requirements):**

* **Hiệu năng:** Thời gian nhận diện khuôn mặt và trả về kết quả phải dưới 3 giây. Hệ thống phải đáp ứng được ít nhất 100 request/giây.
* **Bảo mật:**
  + Mật khẩu người dùng phải được mã hóa.
  + Tất cả các API yêu cầu quyền truy cập phải được bảo vệ bằng JWT.
  + Hệ thống phải có các biện pháp chống lại các cuộc tấn công phổ biến như SQL Injection, XSS.
  + Áp dụng giới hạn tỷ lệ truy cập (Rate limiting) để chống tấn công DDoS.
* **Tính khả dụng:** Hệ thống phải hoạt động 24/7. Giao diện người dùng phải thân thiện, dễ sử dụng trên cả web và mobile.
* **Khả năng mở rộng:** Kiến trúc hệ thống phải được thiết kế để dễ dàng mở rộng, thêm các tính năng mới trong tương lai.

**3.3. Use Case Diagram và mô tả các Use Case**

**(Sơ đồ Use Case được vẽ ở đây)**

**Mô tả các Use Case chính:**

* **Use Case: Đăng nhập**
  + **Tác nhân:** Admin, Giảng viên, Sinh viên.
  + **Mô tả:** Người dùng nhập tên đăng nhập và mật khẩu. Hệ thống xác thực thông tin và nếu hợp lệ, chuyển hướng người dùng đến trang dashboard tương ứng với vai trò của họ.
* **Use Case: Quản lý người dùng**
  + **Tác nhân:** Admin.
  + **Mô tả:** Admin có thể xem danh sách người dùng, tạo người dùng mới, cập nhật thông tin và thay đổi trạng thái (active/inactive) của họ.
* **Use Case: Đăng ký khuôn mặt**
  + **Tác nhân:** Sinh viên.
  + **Mô tả:** Sinh viên tải lên một video ngắn quay khuôn mặt của mình. Hệ thống xử lý video này để tạo bộ dữ liệu và huấn luyện mô hình nhận diện.
* **Use Case: Tạo phiên điểm danh**
  + **Tác nhân:** Giảng viên.
  + **Mô tả:** Giảng viên chọn lớp, môn học và thiết lập thời gian bắt đầu/kết thúc để tạo một phiên điểm danh mới.
* **Use Case: Thực hiện điểm danh**
  + **Tác nhân:** Sinh viên.
  + **Mô tả:** Sinh viên chọn một phiên điểm danh đang mở, ứng dụng sẽ kích hoạt camera. Sinh viên chụp ảnh khuôn mặt mình và gửi lên hệ thống để xác thực.
* **Use Case: Xem báo cáo điểm danh**
  + **Tác nhân:** Giảng viên, Admin.
  + **Mô tả:** Giảng viên có thể xem báo cáo điểm danh chi tiết của một phiên. Admin có thể xem các thống kê tổng hợp trên toàn hệ thống.

**3.4. Biểu đồ hoạt động (Activity Diagram)**

**(Biểu đồ hoạt động cho quy trình điểm danh được vẽ ở đây)**

* **Bắt đầu:** Giảng viên tạo phiên điểm danh.
* **Hành động:** Hệ thống tạo và lưu phiên với trạng thái "Active".
* **Rẽ nhánh:** Sinh viên mở ứng dụng.
* **Hành động:** Sinh viên chọn phiên đang hoạt động.
* **Hành động:** Ứng dụng mở camera.
* **Hành động:** Sinh viên chụp ảnh khuôn mặt.
* **Hành động:** Ảnh được gửi lên server.
* **Hành động:** Server thực hiện nhận diện.
* **Điều kiện:** Khuôn mặt có khớp không?
  + **[Đúng]:** Ghi nhận "Có mặt", cập nhật trạng thái và trả về kết quả "Thành công".
  + **[Sai]:** Trả về kết quả "Thất bại".
* **Hành động:** Hiển thị kết quả cho sinh viên.
* **Hành động:** Cập nhật real-time trên dashboard của giảng viên.
* **Kết thúc:** Quy trình kết thúc khi sinh viên nhận được kết quả.
* **Song song:** Giảng viên có thể đóng phiên điểm danh bất cứ lúc nào, khi đó sinh viên sẽ không thể tham gia nữa.

**3.5. Kiến trúc tổng thể hệ thống**

Hệ thống được thiết kế theo kiến trúc 3 lớp (3-Tier Architecture) kết hợp với kiến trúc Microservices cho phần xử lý AI.

1. **Presentation Layer (Tầng trình diễn):**
   * **Frontend Web (React):** Giao diện web cho người dùng quản trị.
   * **Mobile App (Flutter):** Giao diện di động cho người dùng cuối.
   * Lớp này chịu trách nhiệm hiển thị dữ liệu và tương tác với người dùng.
2. **Application Layer / Business Logic Layer (Tầng ứng dụng):**
   * **Backend (Node.js/Express.js):** Chứa toàn bộ logic nghiệp vụ của hệ thống. Nó xử lý các yêu cầu từ client, tương tác với cơ sở dữ liệu và gọi đến các dịch vụ xử lý AI.
   * Cung cấp các API RESTful cho client.
3. **Data Layer (Tầng dữ liệu):**
   * **MySQL Database:** Lưu trữ toàn bộ dữ liệu của hệ thống như thông tin người dùng, lớp học, lịch học, bản ghi điểm danh, v.v..
   * **File System:** Lưu trữ các file video và ảnh (dataset khuôn mặt) do người dùng tải lên.

Data Flow (Luồng dữ liệu):

Frontend (React/Flutter) gửi HTTP request đến Backend (Node.js) -> Backend xử lý request, truy vấn hoặc cập nhật dữ liệu trong MySQL -> Đối với các tác vụ AI, Backend gọi các script Python -> Script Python xử lý và trả kết quả về cho Backend -> Backend trả response (JSON) về cho Frontend.

**3.6. Thiết kế giao diện người dùng (UI)**

* **Web (React):**
  + **Layout:** Sử dụng layout 2 cột chính gồm Sidebar (menu điều hướng) và Content (nội dung chính).
  + **Thiết kế:** Gọn gàng, chuyên nghiệp, tập trung vào việc hiển thị dữ liệu dạng bảng, biểu đồ và form.
  + **Responsive:** Giao diện có khả năng co giãn, tương thích tốt trên cả desktop và các thiết bị di động.
  + **Thành phần UI:** Sử dụng các component tái sử dụng như Header, Notification, LoadingOverlay, và các component form tùy chỉnh.
* **Mobile (Flutter):**
  + **Thiết kế:** Theo phong cách Material Design của Google, tối ưu cho trải nghiệm cảm ứng.
  + **Điều hướng:** Sử dụng Tab-based navigation cho các dashboard chính (Sinh viên, Giảng viên) để dễ dàng chuyển đổi giữa các chức năng.
  + **Thành phần UI:** Xây dựng các widget tùy chỉnh như AttendanceCard, CustomButton, LoadingDialog để đảm bảo giao diện đồng nhất và dễ sử dụng.

**3.7. Thiết kế cơ sở dữ liệu**

Thiết kế cơ sở dữ liệu được thể hiện qua biểu đồ quan hệ thực thể (ERD) được cung cấp.

**(Chèn ảnh ERD vào đây)**

**Mô tả các bảng chính:**

* **users**: Lưu trữ thông tin chung của tất cả người dùng (admin, teacher, student), bao gồm thông tin đăng nhập, thông tin cá nhân và vai trò.
* **classes**: Lưu trữ thông tin về các lớp học.
* **subjects**: Lưu trữ thông tin về các môn học.
* **class\_students**: Bảng trung gian, thể hiện mối quan hệ nhiều-nhiều giữa sinh viên và lớp học.
* **schedules**: Lưu trữ thông tin lịch học, kết nối lớp học, môn học và giảng viên.
* **attendance\_sessions**: Lưu trữ thông tin về mỗi phiên điểm danh được tạo ra bởi giảng viên.
* **attendances**: Lưu trữ bản ghi điểm danh chi tiết của từng sinh viên trong một phiên, bao gồm thời gian điểm danh và trạng thái.
* **face\_images**: Lưu trữ đường dẫn đến các hình ảnh trong bộ dữ liệu khuôn mặt của mỗi người dùng.

Các mối quan hệ khóa chính - khóa ngoại được thiết lập rõ ràng để đảm bảo tính toàn vẹn dữ liệu. Ví dụ, attendances.session\_id tham chiếu đến attendance\_sessions.id và attendances.student\_id tham chiếu đến users.id.

**3.8. Thiết kế chi tiết các chức năng chính**

**1. Chức năng Đăng ký khuôn mặt:**

* **Input:** Video file (định dạng mp4, avi,...), userId.
* **API Endpoint:** POST /face/upload-video.
* **Xử lý:**
  1. Backend nhận file video qua multer.
  2. Gọi script Python

01\_face\_dataset.py.

* 1. Script sử dụng OpenCV để đọc video, frame by frame.
  2. Với mỗi frame, sử dụng Haar Cascade để phát hiện khuôn mặt.
  3. Nếu phát hiện, cắt vùng khuôn mặt, chuyển sang ảnh xám, resize và lưu thành file ảnh .jpg trong thư mục dataset của userId.
  4. Sau khi tạo xong dataset, tự động gọi script

02\_face\_training.py.

* 1. Script này sẽ đọc tất cả các ảnh trong thư mục dataset, trích xuất đặc trưng LBPH và huấn luyện một mô hình mới, lưu lại dưới dạng file .yml.
* **Output:** Thông báo thành công và kết quả của quá trình tạo dataset và huấn luyện.

**2. Chức năng Điểm danh:**

* **Input:** Image file, session\_id, user\_id.
* **API Endpoint:** POST /attendance/check.
* **Xử lý:**
  1. Backend nhận file ảnh.
  2. Kiểm tra xem session\_id có hợp lệ và đang hoạt động không.
  3. Gọi script Python

03\_face\_recognition.py.

* 1. Script tải mô hình .yml đã được huấn luyện.
  2. Đọc ảnh đầu vào, phát hiện khuôn mặt.
  3. Thực hiện nhận diện bằng model.predict().
  4. Hàm này sẽ trả về id của người được nhận diện và một giá trị confidence (độ tin cậy).
  5. Nếu confidence đủ thấp (cho thấy độ khớp cao) và id trả về khớp với user\_id gửi lên, thì việc nhận diện được xem là thành công.
  6. Backend ghi nhận một bản ghi mới vào bảng attendances với trạng thái "present".
* **Output:** JSON chứa kết quả điểm danh (thành công/thất bại).

**CHƯƠNG 4: CÀI ĐẶT VÀ TRIỂN KHAI**

**4.1. Môi trường phát triển**

* **Backend:**
  + Hệ điều hành: Windows/Linux/macOS
  + Runtime: Node.js v16.x trở lên
  + Framework: Express.js v4.18.2
  + Database: MySQL 8.0
  + Ngôn ngữ: JavaScript (ES6+), Python 3.8+
  + Thư viện Python chính: OpenCV-python
* **Frontend Web:**
  + Hệ điều hành: Windows/Linux/macOS
  + Runtime: Node.js v16.x trở lên (cho npm)
  + Framework: React 18.2.0
  + Công cụ build: Create React App
* **Mobile App:**
  + Hệ điều hành: Windows/Linux/macOS
  + SDK: Flutter SDK v3.x, Dart SDK v2.18+
  + IDE: Android Studio, Visual Studio Code
  + Nền tảng mục tiêu: Android API 21+
* **Công cụ khác:**
  + Quản lý mã nguồn: Git
  + Kiểm thử API: Postman
  + IDE: Visual Studio Code

**4.2. Cấu trúc thư mục mã nguồn**

* **backendjs/**
  + controllers/: Chứa logic xử lý cho mỗi route.
  + routes/: Định nghĩa các API endpoints (e.g., auth.js, face.js).
  + services/: Chứa các logic nghiệp vụ phức tạp (e.g., faceRecognitionService.js, databaseService.js).
  + middlewares/: Chứa các middleware (e.g., xác thực JWT, xử lý lỗi).
  + models/: Định nghĩa cấu trúc dữ liệu.
  + python\_scripts/: Chứa các file Python cho xử lý AI (01\_face\_dataset.py, 02\_face\_training.py, 03\_face\_recognition.py).
  + config/: Chứa các file cấu hình (database, JWT secret).
* **my-app/ (React)**
  + src/
    - components/: Chứa các component UI tái sử dụng (Header, Sidebar).
    - pages/: Chứa các component tương ứng với mỗi trang (Login, AdminDashboard).
    - services/: Chứa các module xử lý logic (api-service, auth-service).
    - routes/: Định nghĩa cấu trúc định tuyến của ứng dụng.
    - styles/: Chứa các file style chung.
* **lib/ (Flutter)**
  + screens/: Chứa các file Dart cho mỗi màn hình của ứng dụng (login\_screen.dart, student\_dashboard.dart).
  + widgets/: Chứa các widget tùy chỉnh, có thể tái sử dụng (camera\_preview\_widget.dart).
  + services/: Chứa các lớp xử lý logic nghiệp vụ (api\_service.dart, auth\_service.dart).
  + models/: Định nghĩa các lớp model dữ liệu (user.dart, attendance\_session.dart).

**4.3. Mô tả các chức năng đã cài đặt (kèm ảnh minh họa giao diện)**

**(Phần này sẽ bao gồm các ảnh chụp màn hình của ứng dụng web và mobile cho từng chức năng và mô tả chi tiết cách chúng hoạt động)**

* **Giao diện Đăng nhập (Web & Mobile):**
  + **(Ảnh màn hình trang đăng nhập)**
  + Mô tả: Form đăng nhập cho phép người dùng nhập username và password. Có validation cho các trường và hiển thị thông báo lỗi. Sau khi đăng nhập thành công, người dùng được chuyển hướng đến trang phù hợp với vai trò của họ.
* **Dashboard của Admin (Web):**
  + **(Ảnh màn hình dashboard Admin)**
  + Mô tả: Hiển thị các thẻ thống kê tổng quan về số lượng người dùng, lớp học, phiên điểm danh.... Các thẻ này cũng là lối tắt để điều hướng đến các trang quản lý tương ứng.
* **Trang Quản lý Lớp học (Web):**
  + **(Ảnh màn hình trang quản lý lớp học)**
  + Mô tả: Giao diện hiển thị danh sách các lớp học dưới dạng bảng, có chức năng tìm kiếm, phân trang. Admin có thể thực hiện các thao tác Thêm, Sửa, Xóa lớp học và quản lý danh sách sinh viên trong từng lớp.
* **Giao diện Điểm danh (Mobile):**
  + **(Ảnh màn hình camera khi điểm danh)**
  + Mô tả: Sau khi chọn phiên điểm danh, camera trước của điện thoại được kích hoạt. Khuôn mặt người dùng được phát hiện và đóng khung trong thời gian thực. Người dùng nhấn nút chụp để gửi ảnh đi xác thực.
* **Lịch sử điểm danh (Mobile):**
  + **(Ảnh màn hình lịch sử điểm danh của sinh viên)**
  + Mô tả: Sinh viên có thể xem lại lịch sử các buổi học đã tham gia, với trạng thái rõ ràng (Có mặt, Vắng, Trễ).

**4.4. Vấn đề gặp phải và cách khắc phục**

* **Vấn đề 1: Tích hợp Node.js và Python không hiệu quả.**
  + **Mô tả:** Ban đầu, việc gọi script Python từ Node.js bằng child\_process.spawn gặp khó khăn trong việc truyền dữ liệu lớn (ảnh) và xử lý lỗi.
  + **Khắc phục:** Thay vì truyền trực tiếp dữ liệu, giải pháp được chọn là lưu file ảnh/video vào một thư mục tạm, sau đó truyền đường dẫn file làm tham số cho script Python. Kết quả từ Python được trả về qua stdout dưới dạng chuỗi JSON, giúp Node.js dễ dàng phân tích.
* **Vấn đề 2: Độ chính xác của mô hình nhận diện bị ảnh hưởng bởi ánh sáng.**
  + **Mô tả:** Trong điều kiện ánh sáng yếu hoặc ngược sáng, thuật toán Haar Cascade khó phát hiện khuôn mặt và thuật toán LBPH nhận diện sai.
  + **Khắc phục:**
    - **Tiền xử lý ảnh:** Áp dụng kỹ thuật cân bằng biểu đồ độ sáng (Histogram Equalization) trên ảnh xám trước khi đưa vào nhận diện để cải thiện độ tương phản.
    - **Hướng dẫn người dùng:** Hiển thị hướng dẫn trên màn hình điểm danh, yêu cầu người dùng đảm bảo đủ sáng và khuôn mặt nhìn thẳng vào camera.
* **Vấn đề 3: Quản lý trạng thái phức tạp trong Flutter.**
  + **Mô tả:** Việc cập nhật trạng thái UI real-time (ví dụ: danh sách phiên điểm danh) trên nhiều màn hình trở nên phức tạp khi chỉ dùng StatefulWidget.
  + **Khắc phục:** Áp dụng kiến trúc hướng dịch vụ, tạo ra các AuthService, ApiService hoạt động như những Singleton. Các service này quản lý trạng thái và dữ liệu, các widget chỉ cần "lắng nghe" sự thay đổi từ service để cập nhật lại UI. Điều này giúp tách biệt logic và UI, làm cho code dễ quản lý và bảo trì hơn.

**KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

**1. Những gì đã đạt được**

Đề tài đã xây dựng thành công một hệ thống điểm danh bằng nhận diện khuôn mặt hoàn chỉnh, đáp ứng đầy đủ các mục tiêu đã đề ra.

* **Hệ thống đa nền tảng:** Đã xây dựng thành công một ứng dụng web quản trị (React) và một ứng dụng di động (Flutter), cung cấp trải nghiệm nhất quán và đồng bộ.
* **Tự động hóa hoàn toàn:** Quy trình từ tạo phiên điểm danh đến ghi nhận kết quả được tự động hóa, giúp tiết kiệm thời gian và công sức.
* **Tích hợp AI hiệu quả:** Đã tích hợp thành công module nhận diện khuôn mặt sử dụng OpenCV và thuật toán LBPH, cho kết quả nhận diện nhanh và có độ chính xác chấp nhận được trong môi trường được kiểm soát.
* **Kiến trúc vững chắc:** Hệ thống được xây dựng trên kiến trúc 3 lớp, sử dụng các công nghệ phổ biến và mạnh mẽ, đảm bảo tính bảo mật, hiệu năng và khả năng mở rộng.
* **Quản lý toàn diện:** Hệ thống cung cấp đầy đủ các chức năng quản lý cho cả ba vai trò Admin, Giảng viên và Sinh viên.

**2. Hạn chế còn tồn tại**

* **Chống giả mạo:** Hệ thống hiện tại có thể bị qua mặt bằng cách sử dụng ảnh hoặc video của người khác. Chưa tích hợp công nghệ Liveness Detection để phân biệt khuôn mặt thật và giả.
* **Độ chính xác của mô hình:** Độ chính xác của thuật toán LBPH vẫn bị ảnh hưởng bởi các yếu-tố-bên-ngoài-như-góc-mặt,-biểu-cảm-và-ánh-sáng-phức-tạp.
* **Trải nghiệm người dùng:** Quá trình đăng ký khuôn mặt bằng video có thể chưa thực sự tối ưu và cần có hướng dẫn chi tiết hơn cho người dùng.
* **Triển khai:** Hệ thống mới được triển khai ở môi trường local (localhost, 10.0.2.2), chưa được thử nghiệm trên môi trường production với tải trọng lớn.

**3. Hướng phát triển trong tương lai**

Để cải thiện và hoàn thiện hệ thống, các hướng phát triển sau đây có thể được xem xét:

* **Nâng cấp mô hình AI:**
  + **Tích hợp Liveness Detection:** Sử dụng các kỹ thuật như phân tích chớp mắt, chuyển động đầu, hoặc các mô hình CNN chuyên dụng để chống giả mạo.
  + **Sử dụng Deep Learning:** Thay thế thuật toán LBPH bằng các mô hình Deep Learning hiện đại như FaceNet, ArcFace, hoặc Dlib, vốn cho độ chính xác cao hơn và khả năng kháng nhiễu tốt hơn.
* **Mở rộng tính năng:**
  + **Điểm danh tự động:** Thay vì sinh viên phải chủ động mở app, có thể nghiên cứu lắp đặt camera tại cửa lớp để tự động quét và điểm danh sinh viên khi họ bước vào.
  + **Tích hợp hệ thống thông báo:** Phát triển hệ thống thông báo đẩy (push notification) để nhắc nhở sinh viên khi có phiên điểm danh sắp bắt đầu hoặc thông báo kết quả điểm danh.
  + **Phân tích và cảnh báo:** Xây dựng module phân tích dữ liệu điểm danh, tự động phát hiện các sinh viên thường xuyên vắng mặt hoặc đi trễ và gửi cảnh báo đến cố vấn học tập.
* **Cải thiện hiệu năng và triển khai:**
  + **Tối ưu hóa API:** Sử dụng các kỹ thuật caching, tối ưu hóa truy vấn cơ sở dữ liệu để tăng tốc độ phản hồi.
  + **Triển khai lên Cloud:** Đóng gói ứng dụng (Backend, Frontend) bằng Docker và triển khai lên các nền tảng đám mây như AWS, Google Cloud để đảm bảo tính sẵn sàng và khả năng mở rộng.

**TÀI LIỆU THAM KHẢO**

1. Tài liệu chính thức của React. (<https://reactjs.org/>)
2. Tài liệu chính thức của Flutter. (<https://flutter.dev/>)
3. Tài liệu chính thức của Node.js và Express.js. (<https://nodejs.org/>, <https://expressjs.com/>)
4. Thư viện OpenCV và tài liệu hướng dẫn. (<https://opencv.org/>)
5. Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), pages 886-893.
6. Ahonen, T., Hadid, A., & Pietikäinen, M. (2006). Face Description with Local Binary Patterns: Application to Face Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(12), 2037–2041.