

HỆ THỐNG TỬ KHÓA THÔNG MINH SỬ DỤNG NHẬN DIỆN KHUÔN MẶT

Lê Vũ Ngọc Anh : 22001230
Hoàng Trung Kiên : 22001266
Trần Văn Lâm : 22001268
Lường Duy Thái : 22001284



Ngày 30 tháng 11 năm 2025

- ① Giới Thiệu
- ② Phương pháp và triển khai
- ③ Kết quả
- ④ Kết luận

① Giới Thiệu

② Phương pháp và triển khai

③ Kết quả

④ Kết luận

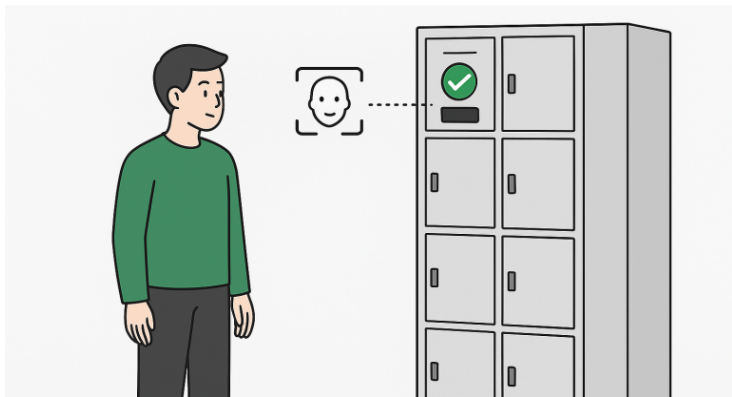
Bối cảnh & Mục tiêu

Bối cảnh

- AI và Computer Vision phát triển mạnh, đặc biệt trong nhận diện khuôn mặt.
- Ứng dụng rộng rãi: kiểm soát ra vào, thanh toán, xác thực thiết bị, camera giám sát.
- Nhu cầu tủ thông minh (Smart Locker) tăng trong trường học, KTX, siêu thị, thư viện, coworking,...

Bối cảnh & Mục tiêu

Mục tiêu

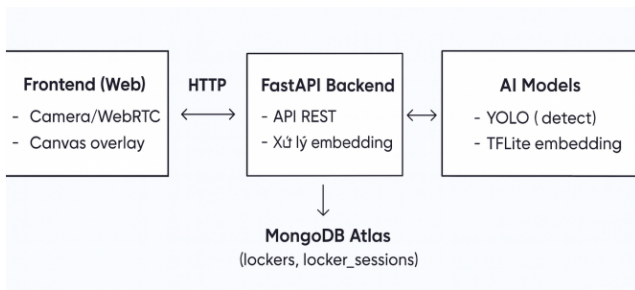


- Xây dựng hệ thống nhận diện khuôn mặt chính xác – ổn định.
- Thiết kế giao diện đăng ký và cơ sở dữ liệu người dùng.

Kiến trúc hệ thống tổng thể

Hệ thống được xây dựng theo kiến trúc phân tán gồm 4 thành phần chính:

- **Frontend (Web):** Giao diện người dùng, sử dụng camera và WebRTC
- **Backend (FastAPI):** Xử lý logic, cung cấp API REST
- **Server AI:** Chạy các mô hình YOLO và Face Embedding
- **Database:** MongoDB Atlas lưu trữ thông tin tử, phiên gửi đồ và vector khuôn mặt
- **Phần Cứng:** ESP32 điều khiển khóa điện tử



Kiến trúc hệ thống tổng thể

① Giới Thiệu

② Phương pháp và triển khai

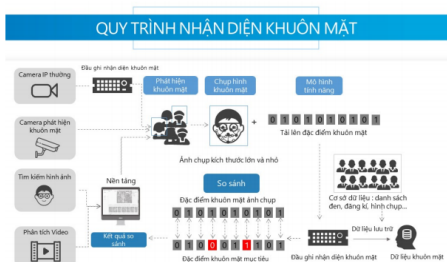
③ Kết quả

④ Kết luận

Quy trình AI & Công nghệ cốt lõi

Luồng xử lý AI 4 bước:

- 1 **Phát hiện người:** YOLOv8 trên tập COCO-person
- 2 **Phát hiện khuôn mặt:** YOLOv8 fine-tune trên WIDER FACE
- 3 **Trích xuất đặc trưng:** ArcFace/InsightFace tạo embedding
- 4 **So khớp:** Cosine Similarity tính toán trực tiếp trong MongoDB



Quy trình nhận diện khuôn mặt tổng quan

Thành phần	Công nghệ
AI Models	YOLOv8, InsightFace (ArcFace), OpenCV
Backend	FastAPI, Uvicorn, Python-dotenv
Database	MongoDB Atlas, Aggregation Framework
Frontend	HTML/CSS/JS, WebRTC, Fetch API
Phần cứng	ESP32, Relay, Solenoid Lock, Nguồn 12V

Điểm nổi bật về kỹ thuật:

- Cosine Similarity được tính trực tiếp trong MongoDB
- Kiến trúc cascade: Person Detection → Face Detection
- Vector embedding 256 chiều được chuẩn hóa L2

Luồng gửi đồ (Store)

Quy trình thực hiện:

- 1 Người dùng nhấn nút "Lưu đồ" trên giao diện
- 2 Trình duyệt bật camera, thu thập 15–20 khung hình
- 3 Hệ thống trích xuất embedding khuôn mặt
- 4 Backend tìm tử trùng, tạo phiên gửi đồ mới
- 5 Gửi lệnh mở tủ đến ESP32 qua API
- 6 Người dùng đặt đồ vào tủ, tủ tự động đóng lại

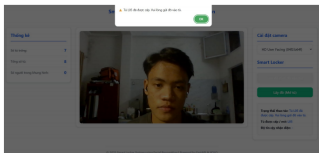


Giao diện gửi đồ thực tế

Luồng lấy đồ (Retrieve)

Quy trình thực hiện:

- 1 Người dùng nhấn nút "Lấy đồ" trên giao diện
- 2 Camera thu thập khung hình khuôn mặt
- 3 Trích xuất embedding và so khớp với database
- 4 Kiểm tra cosine similarity (ngưỡng > 0.6)
- 5 Nếu khớp: mở đúng tủ chứa đồ của người dùng
- 6 Cập nhật trạng thái phiên thành "closed"



Giao diện lấy đồ thực tế

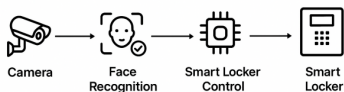
Tích hợp phần cứng

Thành phần phần cứng:

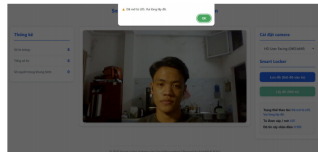
- ESP32 DevKit V1
- Solenoid Lock 12V
- Relay Module
- Nguồn 12V–2A

Giao thức giao tiếp:

- HTTP REST API
- JSON format
- /open/{locker_id}
- /status/{locker_id}



Luồng camera → AI → phần cứng



Kết quả nhận diện thành công

API Backend System

Các endpoint chính:

- POST /process_frame
- POST /store
- POST /retrieve
- GET /lockers/summary
- POST /init_lockers
- GET /health

Ví dụ request mở tủ:

```
{  
  "locker_id": "L01",  
  "action": "open",  
  "timestamp": "2025-11-30T10:22:53Z"  
}
```

① Giới Thiệu

② Phương pháp và triển khai

③ Kết quả

④ Kết luận

Kết quả mô hình phát hiện người

Mô hình YOLOv8 fine-tune trên tập COCO-person:

- **Dữ liệu:** 7,000 ảnh training, 1,000 ảnh validation
- **Tham số:** 50 epochs, batch size 16, optimizer AdamW
- **Kết quả đạt được:**

Bảng 3.1: Quá trình hội tụ của mô hình phát hiện người trên tập validation COCO-person

Epoch	Precision P	Recall R	mAP@0.5	mAP@0.5:0.95
1	0.620	0.436	0.488	0.270
10	0.685	0.522	0.593	0.360
20	0.742	0.569	0.654	0.414
30	0.740	0.600	0.681	0.437
50	0.758	0.624	0.701	0.464

- Precision: **0.758**, Recall: **0.624**
- mAP@0.5: **0.701**, mAP@0.5:0.95: **0.464**
- Mô hình ổn định, phù hợp cho bước tiền xử lý

Kết quả mô hình nhận diện khuôn mặt

Mô hình YOLO8n fine-tune trên WIDER FACE:

- **Kiến trúc:** 100 tầng, 2,582,347 tham số
- **Đánh giá:** 3,226 ảnh validation với 39,707 khuôn mặt
- **Điều kiện:** CPU Intel Xeon 2.20 GHz

Bảng 3.2: Kết quả đánh giá mô hình YOLO8n trên tập validation WIDER FACE

Chỉ số	Precision P	Recall R	mAP@0.5	mAP@0.5:0.95
Giá trị	0.836	0.567	0.648	0.354

Kết quả đánh giá mô hình YOLO8n trên WIDER FACE

- Precision cao: **0.836** → ít false positives
- Recall trung bình: **0.567** → bỏ sót khuôn mặt nhỏ/che khuất
- mAP@0.5: **0.648** - Khá tốt cho bài toán thực tế

Kết quả mô hình nhúng khuôn mặt

Thử nghiệm so sánh vector embedding:

- Sử dụng 3 ảnh: 2 ảnh cùng người (A1, A2) và 1 ảnh khác người (B1)
- Vector embedding 256 chiều được trích xuất
- So sánh bằng Euclidean Distance và Cosine Similarity

Cặp ảnh	Euclidean Distance	Cosine Similarity	Nhận xét
Cùng người (A1 vs A2)	Giá trị nhỏ	Giá trị gần 1	Mô hình hoạt động đúng
Khác người (A1 vs B1)	Giá trị lớn	Giá trị nhỏ	Phân biệt tốt giữa các người

Kết luận:

- Mô hình embedding hoạt động hiệu quả
- Vector cùng người có khoảng cách gần, similarity cao
- Vector khác người có khoảng cách xa, similarity thấp

Kết quả thực nghiệm hệ thống - Gửi đồ

Thử nghiệm chức năng Gửi đồ:

- **Số lần thử:** 30 lần
- **Tỷ lệ thành công:** 100%
- **Thời gian phản hồi:** 1.2 - 1.8 giây
- **Độ tương đồng trung bình:**
 0.85 ± 0.05

Quy trình thành công:

- 1 Nhận diện khuôn mặt chính xác
- 2 Tìm tử trống và tạo phiên
- 3 Gửi lệnh mở tử đến ESP32
- 4 Người dùng gửi đồ thành công

Bảng 3.1: Quá trình hội tụ của mô hình phát hiện người trên tập validation COCO-person

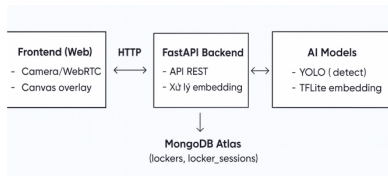
Epoch	Precision P	Recall R	mAP@0.5	mAP@0.5:0.95
1	0.620	0.436	0.488	0.270
10	0.685	0.522	0.593	0.360
20	0.742	0.569	0.654	0.414
30	0.740	0.600	0.681	0.437
50	0.758	0.624	0.701	0.464

Giao diện gửi đồ thực tế

Kết quả thực nghiệm hệ thống - Lấy đồ

Thử nghiệm chức năng Lấy đồ:

- **Số lần thử:** 30 lần
- **Tỷ lệ thành công:** 93.3% (28/30)
- **Thời gian phản hồi:** 1.5 giây
- **Lỗi chủ yếu:**
 - Ánh sáng yếu
 - Góc quay mặt không tốt
 - Che khuất một phần khuôn mặt



Giao diện lấy đồ thực tế

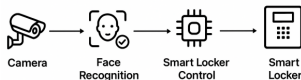
Đánh giá hiệu năng hệ thống

Kết quả tổng quan:

- **Tốc độ xử lý:** 30 FPS trên CPU i5
- **Độ trễ toàn hệ thống:** < 2 giây
- **Độ ổn định phần cứng:** Hoạt động tốt qua 50 lần thử
- **Tỷ lệ nhận diện tổng:** > 95% trong điều kiện chuẩn

Ưu điểm nổi bật:

- Xác thực nhanh, không cần vật mang theo
- Giao diện thân thiện, dễ sử dụng
- Hệ thống ổn định, ít lỗi phần cứng
- Bảo mật cao với vector embedding



Kết quả nhận diện thành công trong thực tế

- ① Giới Thiệu
- ② Phương pháp và triển khai
- ③ Kết quả
- ④ Kết luận**

Kết luận chung

- Hoàn thiện hệ thống tử thông minh sử dụng nhận diện khuôn mặt: AI model + backend + giao diện + phần cứng.
- Sử dụng YOLOv8n (phát hiện người) và YOLOv8n-Face (phát hiện khuôn mặt), tốc độ 30 FPS, phù hợp realtime.
- Nhận diện khuôn mặt bằng InsightFace (ArcFace), độ chính xác >95%, dữ liệu embedding lưu bằng MongoDB.
- Tích hợp ESP32 điều khiển khóa solenoid, mở tủ trong <2 giây, giao diện web trực quan.

Hướng phát triển

- Tích hợp liveness detection (phát hiện thật – giả) để chống ảnh/video spoofing.
- Triển khai hệ thống lên cloud (AWS/GCP), dùng MongoDB Atlas để xử lý dữ liệu lớn, phản hồi $<1s$.
- Nâng cấp phần cứng: Raspberry Pi 5 để chạy mô hình trực tiếp (edge computing), giảm độ trễ còn $<50ms$.
- Tích hợp cảm biến 3D (Intel RealSense) để cải thiện độ chính xác và góc nhìn.
- Phát triển app mobile (React Native) và mở rộng phân tích cảm xúc, cá nhân hóa trải nghiệm.

Cảm ơn mọi người đã lắng
nghe!