PLAN CẢI THIỆN ĐỒ ÁN TÌM NGƯỜI MẤT TÍCH KẾT HỢP VỚI AI

Tuyệt vời! Lựa chọn **Project 2 (Missing Person Tracing)** để nâng cấp là một quyết định chiến lược rất thông minh.

Lý do: Xu hướng tuyển dụng GenAI hiện nay đang chuyển dịch mạnh sang **Multimodal RAG** (Kết hợp tìm kiếm văn bản - hình ảnh) và **Vector Search**. Project này là môi trường hoàn hảo để demo kỹ năng đó.

Dưới đây là **Plan hành động 4 tuần (Sprint 1 tháng)** để biến đồ án này thành một sản phẩm GenAI ấn tượng.

Mục tiêu sản phẩm sau nâng cấp

Hệ thống không chỉ tìm người bằng khuôn mặt (FaceNet cũ) mà còn có thể:

1. **Tìm kiếm bằng mô tả tự nhiên (Semantic Search):** "Tìm cho tôi bé gái khoảng 6 tuổi, mặc áo màu đỏ, đang khóc".
2. **Trợ lý ảo phân tích hồ sơ (AI Agent):** Tự động tóm tắt thông tin vụ việc và soạn thảo thông báo tìm người dựa trên dữ liệu tìm thấy.

KẾ HOẠCH CHI TIẾT 4 TUẦN

Tuần 1: Xây dựng "Bộ não" tìm kiếm mới (Vector Database & CLIP)

Mục tiêu: Hiểu và triển khai được Multimodal Embedding.

- **Ngày 1-2: Setup Vector Database.**
  - Thay vì chỉ lưu metadata trong MongoDB, hãy cài đặt **Qdrant** hoặc **ChromaDB** (dùng Docker). Đây là tiêu chuẩn của ngành GenAI.
  - Nhiệm vụ: Chạy được Qdrant trên local/Docker.
- **Ngày 3-5: Tích hợp mô hình CLIP (OpenAI CLIP hoặc SigLIP).**
  - Tìm hiểu thư viện `sentence-transformers` hoặc `transformers` của HuggingFace.
  - Viết script Python: Load toàn bộ ảnh hiện có trong S3/MongoDB -> Chạy qua model CLIP để ra vector (embedding) -> Lưu vector đó vào Qdrant.
- **Ngày 6-7: Test chức năng tìm kiếm cơ bản.**
  - Viết code input text: "người đàn ông đeo kính đen" -> Model CLIP chuyển thành vector -> Query Qdrant -> Trả về ảnh.

- *Kết quả tuần 1:* Bạn có thể tìm ảnh bằng chữ (điều mà đồ án cũ không làm được).

Tuần 2: Nâng cao độ chính xác (Hybrid Search)

Mục tiêu: Kết hợp cái cũ (FaceNet) và cái mới (CLIP) để tạo ra hệ thống tìm kiếm siêu việt.

- **Ngày 8-10: Logic Hybrid Search.**
  - Vấn đề: CLIP giỏi tìm đặc điểm (áo đỏ, tóc dài) nhưng dở trong việc xác định chính xác danh tính (identity). FaceNet giỏi danh tính nhưng không hiểu ngữ nghĩa.
  - Giải pháp: Xây dựng pipeline tìm kiếm 2 lớp.
    - *Bước 1 (Lọc sơ bộ):* Dùng CLIP lọc ra 50 người có đặc điểm "áo đỏ, tóc ngắn".
    - *Bước 2 (Xác thực):* (Nếu có ảnh input) Dùng FaceNet so khớp khuôn mặt trong tập 50 người đó để tìm người chính xác nhất.
- **Ngày 11-14: Viết API (Backend Refactoring).**
  - Viết thêm API endpoint `/search-smart` bằng FastAPI/Flask.
  - Input: Text mô tả + (Optional) Ảnh upload.
  - Output: JSON danh sách người trùng khớp kèm độ tin cậy (Confidence Score).

Tuần 3: Tích hợp LLM & Agent (Generative Features)

Mục tiêu: Thêm phần "Generative" để đúng chất GenAI Intern.

- **Ngày 15-17: Tính năng "Tự động tạo hồ sơ tìm kiếm" (Auto-Generative Alert).**
  - Khi tìm thấy một người khớp (hoặc khi thêm người mới), sử dụng LLM (GPT-4o-mini hoặc Gemini Flash qua API).
  - **Prompt:** *"Dựa trên dữ liệu: Tên A, 10 tuổi, mất tích tại B, đặc điểm hình ảnh (lấy từ CLIP tags)... hãy viết một bài đăng Facebook kêu gọi tìm kiếm khẩn cấp, văn phong xúc động và cung cấp hotline."*
- **Ngày 18-21: Xây dựng RAG Chatbot (Optional nhưng điểm cộng lớn).**
  - Cho phép admin chat với database: *"Tháng vừa rồi có bao nhiêu vụ mất tích ở Hà Nội? Xu hướng độ tuổi là bao nhiêu?"*.
  - Sử dụng **LangChain** để text-to-query (tự động query MongoDB dựa trên câu hỏi tự nhiên).

Tuần 4: Giao diện (Frontend) & Đóng gói Portfolio

Mục tiêu: Làm cho nó đẹp để quay video demo gửi nhà tuyển dụng.

- **Ngày 22-25: Dựng UI demo bằng Streamlit.**
  - Đừng dùng lại UI cũ nếu nó phức tạp. Dựng một trang Streamlit đơn giản gồm 2 tab:
    - Tab 1: **Smart Search** (Ô nhập text, nút upload ảnh, hiển thị kết quả dạng lưới).

    - Tab 2: **AI Assistant** (Chatbot hỏi đáp số liệu và nút "Generate Poster").
- **Ngày 26-28: Dockerize & Viết Readme.**
    - Tạo `docker-compose.yml` chạy cả Backend + Qdrant + Streamlit.
    - Viết lại Readme trên Github. Nhấn mạnh vào kiến trúc **Multimodal RAG**.
- **Ngày 29-30: Quay video & Nộp đơn.**
    - Quay video màn hình 2 phút demo tính năng tìm kiếm bằng ngôn ngữ tự nhiên. Đây là vũ khí mạnh nhất của bạn.

Tech Stack mới (Bạn cần thêm vào CV sau 1 tháng)

Vector Database:

- Qdrant (recommend) hoặc Pinecone
- Lý do: Better metadata filtering cho age/location
  Embedding Models:
- CLIP: openai/clip-vit-large-patch14
- Text-only fallback: sentence-transformers/paraphrase-multilingual-mpnet-base-v2
  (Hỗ trợ tiếng Việt tốt hơn)
  LLM Orchestration:
- LangChain hoặc LlamaIndex (prefer LlamaIndex cho RAG use case)
  LLM Provider:
- Development: Ollama + Llama 3.2 (free, local)
- Production: GPT-4o-mini hoặc Claude Haiku (cost-effective)

Lưu ý quan trọng cho bạn

1. **Về dữ liệu:** Bạn đã có 18,962 mẫu từ Scrapy. Đây là tài sản lớn. Việc chuyển đổi (embedding) 19k ảnh này tốn thời gian tính toán. Hãy dùng GPU (Google Colab) để chạy script embedding rồi tải vector về nạp vào database local.
2. **Đừng quá sa đà vào Fine-tune:** Trong 1 tháng, đừng cố train lại CLIP hay FaceNet. Hãy dùng pre-trained models. Nhà tuyển dụng cần thấy khả năng **ứng dụng (Engineering)** chứ không phải khả năng training model (Research) ở vị trí Intern.

Bạn có đồng ý với lộ trình này không? Nếu OK, hãy bắt đầu **Tuần 1** và tôi có thể cung cấp code mẫu để bạn kết nối **CLIP với Qdrant**.

KẾ HOẠCH DO CLAUDE ĐỀ XUẤT

🚀 Roadmap thực hiện chi tiết (4-6 tuần)

Phase 1: CLIP Integration & Multimodal Search (2 tuần)

Week 1: Vector Database Setup & Embedding Generation

Công việc cần làm:

- Research và setup Qdrant vector database (local trước, sau đó deploy lên cloud)
- Tiền xử lý 18,962 ảnh hiện có: resize, normalize theo chuẩn CLIP input
- Generate CLIP embeddings cho toàn bộ database ảnh
- Thiết kế metadata schema (tuổi, giới tính, địa điểm, timestamp) để filter khi search
- Index embeddings vào Qdrant kèm metadata

Đầu ra đạt được:

- ✅ Qdrant instance hoạt động với ~19K vectors đã được index
- ✅ Script tự động để add ảnh mới vào vector DB
- ✅ Baseline search: query bằng ảnh → trả về top-K similar faces
- ✅ Document architecture: Tại sao chọn CLIP model nào (base/large), embedding dimension

---

Week 2: Text-to-Image Search API & Evaluation

Công việc cần làm:

- Build REST API endpoint nhận text input tiếng Việt/Anh
- Implement text embedding pipeline: preprocess query → CLIP text encoder → vector search
- Tạo hybrid search logic: kết hợp CLIP search + FaceNet reranking
- Thiết kế 50 test cases với ground truth (ví dụ: "bé trai 7 tuổi áo xanh" → expected match IDs)
- Đo metrics: Recall@5, Recall@10, Mean Reciprocal Rank (MRR), Average search latency

Đầu ra đạt được:

- ✅ API endpoint `/search/semantic` hoạt động với text queries
- ✅ Evaluation report: Recall@10 ≥ 80%, latency < 3 giây

- ✅ Demo interface đơn giản (có thể dùng Gradio/Streamlit) để test search
- ✅ So sánh performance: CLIP-only vs Hybrid (CLIP + FaceNet reranking)

Checkpoint: Có thể demo tính năng "Tìm người bằng mô tả văn bản" end-to-end

---

Phase 2: LLM Agent & Automated Reporting (2 tuần)

Week 3: Report Generation với LLM

Công việc cần làm:

- Setup LLM environment: Local (Ollama + Llama 3.2) cho dev, cloud API cho production
- Thiết kế prompt template cho report generation (có cấu trúc cụ thể, không cho LLM tự do sáng tác)
- Implement structured output parsing: LLM phải trả về JSON với fields cố định
- Build trigger logic: Khi confidence score > 85% → tự động tạo draft report
- Tạo database table `pending_reports` để lưu draft chờ admin duyệt
- Test với 20 real match cases, đánh giá chất lượng report (grammar, completeness, accuracy)

Đầu ra đạt được:

- ✅ Module tự động generate report từ match data
- ✅ Template chuẩn cho 3 loại report: High-confidence match, Medium-confidence, Negative report
- ✅ Validation layer: Kiểm tra output không có hallucination (so sánh với database)
- ✅ Metrics: 95% reports không có factual error, 100% follow template structure

---

Week 4: Multi-Agent System & Human-in-the-Loop

Công việc cần làm:

- Thiết kế 3 agents với vai trò riêng biệt:
  - **Analyst Agent**: Phân tích match quality, đánh giá risk level (urgent/normal/low)
  - **Writer Agent**: Viết report dựa trên phân tích của Analyst
  - **Validator Agent**: Cross-check thông tin trong report với database
- Implement workflow orchestration bằng LangChain hoặc LlamaIndex
- Build admin dashboard đơn giản: hiển thị pending reports, nút approve/reject/edit

- Tạo notification system: Email/Slack alert khi có high-confidence match
- Test end-to-end workflow với 10 simulated cases

Đầu ra đạt được:

- ✅ Multi-agent system hoạt động: 3 agents collaborate để tạo report
- ✅ Admin dashboard functional: Có thể xem, duyệt, sửa reports
- ✅ Audit trail: Log đầy đủ quyết định của mỗi agent (tại sao chọn risk level này, tại sao phrase như vậy)
- ✅ Performance metrics: Thời gian tạo report < 30 giây, cost < $0.05/report

Checkpoint: Có thể demo quy trình từ khi phát hiện match → agent tạo report → admin duyệt → gửi alert

---

Phase 3: Integration, Deployment & Documentation (1-2 tuần)

Week 5: System Integration & Testing

Công việc cần làm:

- Tích hợp CLIP search vào UI hiện có: Thêm search bar cho text queries
- Connect LLM agent module với existing notification system
- End-to-end testing: Upload ảnh mới → CLIP index → Search test → Trigger report generation
- Load testing: System handle được 100 concurrent searches, 50 reports/hour
- Cost analysis: Tính toán chi phí vận hành cho 1000 queries/ngày
- Security audit: Đảm bảo không leak thông tin nhạy cảm qua LLM logs

Đầu ra đạt được:

- ✅ Unified system: CV features + GenAI features hoạt động seamlessly
- ✅ Performance report: Latency, throughput, cost breakdown
- ✅ Security checklist: Data encryption, API key management, PII protection
- ✅ Bug fixes: Xử lý edge cases (query không rõ ràng, không có match, LLM timeout)

Week 6: Deployment & Portfolio Preparation

Công việc cần làm:

- Deploy full system lên AWS/Railway/Render (recommendation: Railway cho đơn giản)
- Setup monitoring: Tracking API calls, error rates, LLM token usage
- Tạo public demo với sample data (không dùng real missing person data vì privacy)
- Viết technical blog post (Medium/Dev.to): Architecture breakdown, lessons learned
- Record demo video 3-5 phút: Show use cases, explain technical decisions
- Chuẩn bị presentation deck: 5-7 slides giải thích project cho interview

Đầu ra đạt được:

- ✅ Live demo URL functional (uptime > 95%)
- ✅ GitHub repo organized: Clean code, README with architecture diagram, setup instructions
- ✅ Blog post published: 1500-2000 words về technical challenges
- ✅ Demo video: Upload lên YouTube/LinkedIn
- ✅ Interview prep document: Anticipate 20 câu hỏi recruiter sẽ hỏi + câu trả lời mẫu

---

📊 Success Metrics cho toàn bộ dự án

Technical Metrics

- CLIP Semantic Search: Recall@10 ≥ 80%, latency < 3s
- LLM Report Quality: 95% factual accuracy, 0% hallucination rate
- System Uptime: 95%+ sau khi deploy
- Cost Efficiency: < $20/tháng cho 1000 queries

Portfolio Impact Metrics

- GitHub stars/forks (target: 20+ stars sau 1 tháng)
- Blog post views (target: 500+ views)
- LinkedIn engagement (target: 50+ reactions trên demo post)
- Interview conversion: Project được nhắc đến ở ≥80% interviews

---

🎯 Deliverables cuối cùng (Checklist)

Code & Infrastructure:

- [ ] GitHub repo với CI/CD pipeline
- [ ] Deployed demo với public URL

- [ ] Docker containers cho easy setup
- [ ] Postman collection cho API testing

Documentation:

- [ ] System architecture diagram (draw.io/Excalidraw)
- [ ] API documentation (Swagger/Postman)
- [ ] Technical blog post
- [ ] README with clear setup instructions

Demo Materials:

- [ ] 3-5 phút demo video
- [ ] 5-7 slides presentation deck
- [ ] Sample queries showcase document
- [ ] Performance benchmark report

Interview Prep:

- [ ] List 20 anticipated questions + answers
- [ ] Cost breakdown spreadsheet
- [ ] Failure cases analysis document

⏰ Timeline Flexibility

Nếu chỉ có 4 tuần:

- Skip Week 4 multi-agent complexity → Chỉ làm single LLM agent
- Skip load testing → Focus vào functional demo

Nếu có 8 tuần:

- Add A/B testing cho prompt variations
- Implement multilingual support (Việt + Anh + Trung)
- Build analytics dashboard cho admin

Lưu ý: Mỗi tuần nên có 1 mini-demo checkpoint để đảm bảo progress đúng hướng. Đừng code liên tục 2 tuần mới test lần đầu! 🎯