**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**

**NGUYỄN THANH NỞ**

**<TÊN ĐỒ ÁN NGÀNH>**

**ĐỒ ÁN NGÀNH**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**TP. HỒ CHÍ MINH, 2025BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**

**NGUYỄN THANH NỞ**

**< TÊN ĐỒ ÁN NGÀNH >**

**Mã số sinh viên: 2251050056**

**ĐỒ ÁN NGÀNH**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**Giảng viên hướng dẫn: TS. Trương Hoàng Vinh**

**TP. HỒ CHÍ MINH, 2025**

LỜI CẢM ƠN

Trong thời gian làm đồ án ngành, em đã nhận được những sự giúp đỡ, đóng góp ý kiến và chỉ bảo nhiệt tình của thầy cô, gia đình, bạn bè.

Em xin phép được gửi sự tri ân sâu sắc và lời cảm ơn chân thành nhất đối với các thầy cô của khoa Công nghệ thông tin đã truyền đạt những tri thức quý báu cho em trong suốt quá trình học tập tại trường Đại học Mở Thành phố Hồ Chí Minh. Đặc biệt, em xin trân trọng cảm ơn thầy Trương Hoàng Vinh đã nhiệt tình hướng dẫn để em có thể hoàn thành tốt bài báo cáo đồ án ngành này.

Em xin kính chúc sức khỏe đến tất cả quý thầy cô đang công tác tại trường Đại học Mở Thành phố Hồ Chí Minh và sự nghiệp giảng dạy sẽ ngày càng thành công hơn, gặt hái được nhiều thành quả tốt. Chúc trường sẽ luôn là nền tảng vững chắc cho nhiều thế hệ sinh viên trên con đường học tập.

Có thể đồ án ngành hoặc bài báo cáo của em sẽ có những chỗ thiếu sót, chưa làm hài lòng được thầy cô nhưng em mong sẽ nhận được sự cảm thông của thầy cô. Em sẽ cố gắng rút kinh nghiệm sau khi nhận được những lời nhận xét của thầy cô và cố gắng hơn nữa trên con đường học tập của mình cũng như con đường để em bước gần hơn với sự nghiệp tương lai phía trước.

Em xin chân thành cảm ơn!

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

TÓM TẮT ĐỒ ÁN NGÀNH

Đồ án xây dựng trọn vẹn một hệ thống “kiểm thử giao diện quảng cáo” từ mô hình đến ứng dụng web. Về mô hình, hệ thống thiết kế vector đặc trưng 5 chiều bao quát cả thay đổi toàn cục và cục bộ: SSIM (cấu trúc), OCR Jaccard (nội dung chữ), Histogram màu (tông màu), ORB (keypoints cục bộ), và Layout IoU (bố cục). Dữ liệu được tổ chức theo cặp (original, variant) và đánh giá bằng GroupKFold theo nhóm original\_x để tránh rò rỉ. Các thuật toán SVM, kNN, Random Forest, XGBoost được so sánh; XGBoost cho hiệu năng ổn định nhất (xấp xỉ ~90%) và được đóng gói kèm metadata.json (ánh xạ nhãn, thứ tự đặc trưng) phục vụ suy luận nhất quán. Ba mức phân loại similar/almost/different phù hợp yêu cầu nghiệp vụ: tách được sai lệch nhẹ cần kiểm chứng khỏi vi phạm rõ rệt.

Về triển khai, ứng dụng Flask cung cấp hai luồng: so sánh trực tiếp 2 ảnh và kiểm thử tự động nhiều chi nhánh với 1 ảnh gốc dùng chung. Crawler headless (Selenium) duyệt từng URL + CSS selector, xử lý lazy loading/ẩn bởi JS, ép hiển thị phần tử, thêm timeout và tải ảnh thật (src/data-src/srcset/background-image) rồi so sánh với ảnh gốc. API /crawl-all xử lý batch, dọn file tạm, trả ảnh base64; /predict phục vụ so sánh nhanh; /health cho giám sát. Giao diện HTML/CSS tối giản, 1 nút “Chạy tất cả”, hiển thị kết quả dạng grid kèm summary (Safe/Warning/Danger/Failed) và modal chi tiết từng chi nhánh, giúp quét nhanh số lượng lớn website mà vẫn dễ tra cứu.

Giá trị cốt lõi là tự động hóa kiểm tra tuân thủ quảng cáo đa chi nhánh, giảm chi phí kiểm thử thủ công, chuẩn hóa tiêu chí đánh giá và rút ngắn thời gian phản hồi. Các giới hạn đã được nhận diện: phụ thuộc chất lượng OCR, chi phí/độ ổn định của Selenium trước anti-bot, xử lý tuần tự chưa tối ưu, và dữ liệu huấn luyện còn khiêm tốn. Từ đây có thể mở rộng theo hướng Playwright/đi song song qua hàng đợi, lưu lịch sử và báo cáo, tích hợp DB/API doanh nghiệp, tinh chỉnh ngưỡng theo chiến dịch, và nghiên cứu học sâu (Siamese/Triplet) để tăng độ bền với các biến thiên phức tạp hơn.

ABSTRACT

**MỤC LỤC**

[LỜI CẢM ƠN 1](#_Toc208845950)

[NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN 2](#_Toc208845951)

[TÓM TẮT ĐỒ ÁN NGÀNH 3](#_Toc208845952)

[ABSTRACT 4](#_Toc208845953)

[DANH MỤC TỪ VIẾT TẮT 8](#_Toc208845954)

[DANH MỤC HÌNH VẼ 9](#_Toc208845955)

[DANH MỤC BẢNG 10](#_Toc208845956)

[No table of figures entries found. 10](#_Toc208845957)

[MỞ ĐẦU 11](#_Toc208845958)

[Chương 1. TỔNG QUAN ĐỀ TÀI 12](#_Toc208845959)

[1.1. Giới thiệu đề tài 12](#_Toc208845960)

[1.2. Lý do chọn đề tài 12](#_Toc208845961)

[1.3. Mục tiêu đề tài 13](#_Toc208845962)

[1.4. Phương pháp thực hiện 13](#_Toc208845963)

[1.5. Bố cục đề tài 14](#_Toc208845964)

[Chương 2. CƠ SỞ LÝ THUYẾT 15](#_Toc208845965)

[2.1. Tổng quan bài toán kiểm thử giao diện tự động 15](#_Toc208845966)

[2.2. Kiểm thử giao diện: khái niệm và phân loại 16](#_Toc208845967)

[2.2.1. Khái niệm chung 16](#_Toc208845968)

[2.2.2. Phân loại 16](#_Toc208845969)

[2.3. Các đặc trưng xử lý ảnh 17](#_Toc208845970)

[2.3.1. SSIM (Structural Similarity Index Measurement) 17](#_Toc208845971)

[2.3.2. OCR (Optical Character Recognition) 18](#_Toc208845972)

[2.3.3. Layout 19](#_Toc208845973)

[2.3.4. Histogram 20](#_Toc208845974)

[2.3.5. ORB (Oriented FAST and Rotated BRIEF) 21](#_Toc208845975)

[2.4. Các thuật toán học máy áp dụng trong phân loại giao diện 22](#_Toc208845976)

[2.4.1. Thuật toán SVM (Support Vector Machine) 22](#_Toc208845977)

[2.4.2. Thuật toán RF (Random Forest) 23](#_Toc208845978)

[2.4.3. Thuật toán kNN (k-Nearest Neighbors) 24](#_Toc208845979)

[2.4.4. Thuật toán XGBoost (Extreme Gradient Boosting) 26](#_Toc208845980)

[Chương 3. HỆ THỐNG KIỂM THỬ GIAO DIỆN TỰ ĐỘNG 27](#_Toc208845981)

[3.1. Giới thiệu hệ thống 27](#_Toc208845982)

[3.2. Phân tích hệ thống 28](#_Toc208845983)

[3.2.1. Bài toán và Phạm vi 28](#_Toc208845984)

[3.2.2. Phân tích Dữ liệu 29](#_Toc208845985)

[3.2.3. Đặc trưng và lý do lựa chọn 29](#_Toc208845986)

[3.2.4. Thuật toán và lý do lựa chọn 30](#_Toc208845987)

[3.3. Thiết kế hệ thống 31](#_Toc208845988)

[3.3.1. Kiến trúc hệ thống 31](#_Toc208845989)

[3.3.2. Các thành phần của hệ thống 32](#_Toc208845990)

[3.3.3. Luồng hoạt động của hệ thống 34](#_Toc208845991)

[3.4. Các giai đoạn phát triển hệ thống 35](#_Toc208845992)

[3.4.1. Giai đoạn 1 35](#_Toc208845993)

[3.4.2. Giai đoạn 2 37](#_Toc208845994)

[3.4.3. Giai đoạn 3 39](#_Toc208845995)

[3.5. Kết quả đạt được 40](#_Toc208845996)

[3.5.1. Giai đoạn 1 (Mô hình sơ khai) 40](#_Toc208845997)

[3.5.2. Giai đoạn 2 (Phát triển mô hình) 42](#_Toc208845998)

[3.5.3. Giai đoạn 3 (So sánh mô hình) 44](#_Toc208845999)

[3.5.4. Giai đoạn 4 51](#_Toc208846000)

[Chương 4. KẾT LUẬN VÀ HƯƠNG PHÁT TRIỂN 52](#_Toc208846001)

[4.1. Kết luận 52](#_Toc208846002)

[4.2. Hướng phát triển 53](#_Toc208846003)

[TÀI LIỆU THAM KHẢO 55](#_Toc208846004)

[PHỤ LỤC TEST CASE 57](#_Toc208846005)

DANH MỤC TỪ VIẾT TẮT

DANH MỤC HÌNH VẼ

[Hình 2‑1. Pipeline tính toán SSIM từ ảnh đầu vào 16](#_Toc208840492)

[Hình 2‑2. Pipeline nhận diện và so sánh văn bản bằng OCR từ ảnh đầu vào 17](#_Toc208840493)

[Hình 2‑3. Pipeline so sánh bố cục bằng IoU 18](#_Toc208840494)

[Hình 2‑4. Pipeline tính toán Histogram từ ảnh màu 19](#_Toc208840495)

[Hình 2‑5. Pipeline tính toán ORB từ ảnh đầu vào 20](#_Toc208840496)

[Hình 2‑6. Pipeline so sánh ảnh đa tính năng với SVM 21](#_Toc208840497)

[Hình 2‑7. Pipeline so sánh ảnh đa tính năng với RF 22](#_Toc208840498)

[Hình 2‑8. Pipeline so sánh ảnh đa tính năng với kNN 24](#_Toc208840499)

[Hình 2‑9. Pipeline so sánh ảnh đa tính năng với XGBoost 25](#_Toc208840500)

[Hình 3‑1. Kiến trúc hệ thống 30](#_Toc208840501)

[Hình 3‑2. Sơ đồ luồng hoạt động của hệ thống 33](#_Toc208840502)

[Hình 3‑3. Pipeline tiền xử lý và đặc trưng cơ bản 35](#_Toc208840503)

[Hình 3‑4. Pipeline mở rộng với OCR và ORB 37](#_Toc208840504)

[Hình 3‑5. Biểu đồ phân bố nhãn giai đoạn 1 39](#_Toc208840505)

[Hình 3‑6. Ma trận nhầm lẫn (Version 1 - SVM) 40](#_Toc208840506)

[Hình 3‑7. Biểu đồ phân bố nhãn giai đoạn 2 41](#_Toc208840507)

[Hình 3‑8. Độ chính xác Cross-validation theo từng Fold (Version 2 - SVM) 42](#_Toc208840508)

[Hình 3‑9. Ma trận nhầm lẫn (Version 2 - SVM) 42](#_Toc208840509)

[Hình 3‑10. Độ chính xác Cross-validation theo từng Fold (kNN) 44](#_Toc208840510)

[Hình 3‑11. Ma trận nhầm lẫn kNN 44](#_Toc208840511)

[Hình 3‑12. Độ chính xác Cross-validation theo từng Fold (SVM) 45](#_Toc208840512)

[Hình 3‑13. Ma trận nhầm lẫn SVM 46](#_Toc208840513)

[Hình 3‑14. Độ chính xác Cross-validation theo từng Fold (RF) 46](#_Toc208840514)

[Hình 3‑15. Ma trận nhầm lẫn RF 47](#_Toc208840515)

[Hình 3‑16. Độ chính xác Cross-validation theo từng Fold (XGBoost) 48](#_Toc208840516)

[Hình 3‑17. Ma trận nhầm lẫn (Version 5 - XGBoost) 48](#_Toc208840517)

DANH MỤC BẢNG

[Bảng 1. Bảng kết quả so sánh các mô hình 50](#_Toc208846006)

MỞ ĐẦU

Trong bối cảnh doanh nghiệp triển khai cùng một chiến dịch quảng cáo trên nhiều website chi nhánh, việc bảo đảm nội dung hiển thị thống nhất với mẫu gốc là yêu cầu quan trọng nhằm duy trì uy tín thương hiệu. Tuy nhiên, kiểm tra thủ công trên từng trang vừa tốn nhiều thời gian, chi phí, vừa dễ bỏ sót các thay đổi nhỏ nhưng nhạy cảm như giá bán, khuyến mãi hoặc logo.

Xuất phát từ thực tế đó, đồ án hướng tới việc xây dựng một hệ thống kiểm thử giao diện quảng cáo tự động trên web. Người dùng chỉ cần cung cấp ảnh quảng cáo gốc và danh sách chi nhánh; hệ thống sẽ tự động thu thập ảnh thực tế từ từng trang, so sánh với mẫu gốc và trả về kết quả theo ba mức similar/almost/different, qua đó hỗ trợ ra quyết định nhanh, giảm chi phí thủ công và chuẩn hóa quy trình kiểm thử.

Về mặt kỹ thuật, hệ thống kết hợp xử lý ảnh truyền thống và mô hình học máy nhằm phát hiện sự khác biệt cả về bố cục, nội dung chữ và chi tiết đồ họa. Kết quả so sánh được trực quan hóa bằng báo cáo và hình ảnh đánh dấu điểm khác biệt, giúp người dùng dễ dàng xác định và đánh giá lỗi hiển thị.

Mục tiêu cuối cùng của đồ án là đề xuất một quy trình kiểm thử tự động có thể tích hợp vào pipeline CI/CD, từ đó mở rộng thành dịch vụ kiểm thử giao diện cho doanh nghiệp đa chi nhánh. Điều này không chỉ góp phần giảm thiểu sai sót trong vận hành mà còn nâng cao chất lượng quản lý thương hiệu trong môi trường kinh doanh trực tuyến.

# TỔNG QUAN ĐỀ TÀI

## Giới thiệu đề tài

Trong bối cảnh hiện nay, các hệ thống quảng cáo được triển khai đồng bộ trên website của nhiều chi nhánh hoặc cửa hàng. Quảng cáo này có thể ở nhiều dạng như tĩnh (hình ảnh), động (video), âm thanh (audio)… nhưng việc có tuân thủ quảng cáo gốc hay không lại là một vấn đề rất lớn. Thực tế, những sự sai lệch nhỏ như thêm bớt nội dung, chỉnh sửa màu, thay đổi chữ, di chuyển biểu tượng, cắt bỏ hoặc thay đổi lớn có thể làm sai đi thông điệp vốn có của quảng cáo.

Cách để giải quyết vấn đề này là phải đi từng website của các chi nhánh để kiểm tra, quá trình kiểm tra thủ công này rất tốn thời gian, dễ bỏ sót những thay đổi nhỏ nhưng quan trọng (ví dụ thông tin giá, khuyến mãi…). Lâu dài có thể gây ảnh hưởng nghiêm trọng đến uy tín thương hiệu cũng như độ tin cậy của khách hàng.

Sau khi nhận thấy nguy cơ tiềm ẩn của vấn đề, em quyết định tiến hành nghiên cứu và triển khai hệ thống kiểm thử giao diện tự động cho quảng cáo trực tuyến. Hệ thống này có thể tự động thu thập những hình ảnh quảng cáo được chèn trực tiếp trên website sau đó so sánh với quảng cáo gốc dựa trên vị trí HTML đã được thống nhất. Qua đó, giúp phát hiện nhanh chóng các vi phạm trong việc triển khai quảng cáo tại từng chi nhánh, đảm bảo tính đồng bộ, chính xác và uy tín cho doanh nghiệp.

## Lý do chọn đề tài

Trong hoạt động quảng cáo, việc triển khai đồng bộ và chính xác nội dung quảng cáo từ nhà cung cấp đến các chi nhánh là yếu tố then chốt để đảm bảo tính thống nhất của thương hiệu. Nhưng thực tế, yếu tố trên không dễ dàng thực hiện, nếu chỉ có vài chi nhánh sử dụng hình ảnh quảng cáo thì có thể đảm bảo nhưng nếu có đến vài trăm thậm chí là vài nghìn chi nhánh sử dụng quảng cáo và đâu đó trong những chi nhánh này không tuân thủ quảng cáo ban đầu thì việc kiểm tra thủ công của từng chi nhánh rất tốn chi phí, thời gian, nhân lực…

Để giải quyết vấn đề này trong phạm vi đồ án, em tập trung nghiên cứu và xây dựng hệ thống kiểm thử giao diện quảng cáo tự động với các chức năng như tự động thu thập hình ảnh quảng cáo được chèn trực tiếp trên website chi nhánh. Người dùng chỉ cần cung cấp hình ảnh quảng cáo gốc, đường dẫn website và vị trí thẻ HTML (đã được thống nhất giữa nhà cung cấp và chi nhánh). Hệ thống sẽ tiến hành so sánh hình ảnh quảng cáo hiển thị thực tế với ảnh gốc, từ đó phát hiện các trường hợp sai lệch hoặc vi phạm nguyên tắc ban đầu.

## Mục tiêu đề tài

Mục tiêu đề tài là nghiêm cứu và xây dựng một hệ thống kiểm thử giao diện quảng cáo tự động, cho phép phát hiện và phân loại mức độ sai lệch giữa quảng cáo gốc và quảng cáo được triển khai trên các website chi nhánh. Thay vì phải phụ thuộc vào phương pháp kiểm tra thủ công vốn tốn nhiều thời gian, chi phí và dễ bỏ sót chi tiết, hệ thống hướng tới việc tự động hóa toàn bộ quy trình kiểm tra, từ khâu thu thập dữ liệu đến phân tích, đối chiều và báo cáo kết quả.

## Phương pháp thực hiện

Trải qua quá trình phát triển mô hình, em quyết định đưa mô hình đến 3 nhãn để phân loại sai lệch: giống (similar), sai lệch nhẹ chấp nhận được (almost), sai lệch cần cảnh báo (different).

Ngôn ngữ và thư viện: Python, OpenCV (xử lý ảnh, Canny, ORB, histogram), cikit-learn (SVM, RandomForest, kNN, GroupKFold), XGBoost, Tesseract OCR (trích xuất văn bản), Numpy / Pandas / Seaborn (phân tích)

Các giai đoạn tiến hóa mô hình:

* Giai đoạn 1: SSIM, Histogram và Layout với nhãn 2 lớp (similar/different).
* Giai đoạn 2: Thêm OCR (tỷ lệ tương đồng từ khóa) đồng thời tích hợp ORB để xử lý các thay đổi bố cục nhỏ như icon, nút hay chi tiết đồ họa cục bộ, mở rộng thành nhãn 3 lớp (similar/almost/different) để tách sai lệch nhẹ. tạo vector đặc trưng đầy đủ (SSIM, OCR, Histogram, ORB, Layout).
* Giai đoạn 3: Tiến hành so sánh đa thuật toán (SVM, RF, kNN, XGBoost) với mục đích chọn phương án cân bằng hiệu năng, ổn định.
* Giai đoạn 4: Triển khai mô hình đã chọn vào hệ thống.

## Bố cục đề tài

Báo cáo có 4 chương

Chương 1. Tổng quan đề tài: -> mục tiêu chương

Chương 2. Cơ sở lý thuyết: -> mục tiêu chương

...

...

# CƠ SỞ LÝ THUYẾT

## Tổng quan bài toán kiểm thử giao diện tự động

Bài toán đặt ra là trong lĩnh vực quảng cáo trực tuyến, tính đồng bộ và nhất quán của nội dung giữa mẫu gốc và các phiên bản triển khai tại chi nhánh đóng vai trò rất quan trọng trong việc duy trì uy tín thương hiệu. Tuy nhiên, trong thực tế, quá trình triển khai thường phát sinh nhiều sai lệch. Những sai lệch này có thể biểu hiện ở nhiều mức độ khác nhau như thay đổi nội dung chữ, chỉnh sửa hoặc dịch chuyển biểu tượng, biến đổi bố cục trình bày, điều chỉnh tông màu hiển thị hoặc thậm chí thay thế hoàn toàn hình ảnh và thông tin khuyến mãi. Các sai lệch nhỏ đôi khi rất khó nhận biết bằng mắt thường, nhưng lại có khả năng làm sai lệch thông điệp ban đầu mà doanh nghiệp muốn truyền tải đến khách hàng.

Việc phát hiện và phân loại sai lệch theo phương pháp thủ công là một nhiệm vụ phức tạp và tốn kém. Khi số lượng chi nhánh và quảng cáo cần kiểm tra ngày càng nhiều, việc con người phải đi từng website để so sánh với mẫu gốc sẽ dẫn đến lãng phí thời gian, tăng chi phí nhân sự và tiềm ẩn nguy cơ bỏ sót những thay đổi quan trọng. Chính vì vậy, nhu cầu đặt ra là cần một giải pháp tự động, có khả năng thay thế hoặc hỗ trợ quá trình kiểm tra giao diện, từ đó giúp doanh nghiệp kiểm soát được tính tuân thủ quảng cáo ở quy mô lớn.

Bài toán kiểm thử giao diện tự động trong bối cảnh này có thể được khái quát như sau: hệ thống nhận vào hình ảnh mẫu gốc do nhà cung cấp phát hành và hình ảnh quảng cáo được hiển thị thực tế tại một chi nhánh, sau đó tiến hành so sánh để đánh giá mức độ khác biệt. Kết quả đầu ra là nhãn phân loại thể hiện mức độ sai lệch, có thể chia thành ba mức: similar (giống hoặc đúng chuẩn), almost (có sai lệch nhẹ nhưng chấp nhận được) và different (khác biệt rõ rệt, vi phạm nguyên tắc triển khai).

Khó khăn lớn nhất của bài toán này nằm ở chỗ không chỉ dừng lại ở so sánh pixel thuần túy mà còn cần đạt được mức hiểu ngữ nghĩa. Ví dụ, nếu văn bản trên quảng cáo bị thay đổi một từ, việc so sánh màu sắc hoặc cấu trúc hình ảnh thông thường sẽ không đủ, mà cần đến kỹ thuật nhận dạng ký tự quang học (OCR). Tương tự, khi một biểu tượng nhỏ bị dịch chuyển hoặc thay đổi vị trí, chỉ số so sánh toàn cục như SSIM hay histogram chưa chắc đã phát hiện được, mà cần đến phương pháp so khớp đặc trưng cục bộ như ORB. Bên cạnh đó, những thay đổi về tông màu hoặc cách bố trí khung quảng cáo cũng cần được phân tích bằng các đặc trưng riêng như histogram hoặc layout.

Với những thách thức đó, một hướng tiếp cận hiệu quả là kết hợp nhiều đặc trưng khác nhau, bao gồm SSIM, histogram, OCR, ORB và layout, để nắm bắt được cả thay đổi toàn cục và cục bộ trong hình ảnh. Các đặc trưng này sau đó được đưa vào mô hình học máy giám sát để huấn luyện và phân loại mức độ sai lệch. Đây là cách tiếp cận thuộc nhóm Visual Compliance Testing, tức là kiểm tra tính tuân thủ trực quan của giao diện, và thường áp dụng các mô hình học máy cổ điển như SVM, Random Forest, k-NN hoặc XGBoost nhằm đảm bảo cân bằng giữa độ chính xác, tính ổn định và khả năng giải thích kết quả.

## Kiểm thử giao diện: khái niệm và phân loại

### Khái niệm chung

Kiểm thử giao diện người dùng là quá trình kiểm tra tính đúng đắn. nhất quán và khả năng sử dụng của giao diện phần mềm hoặc ứng dụng. Kiểm thử giao diện không chỉ dừng lại ở việc đảm bảo các nút bấm, form nhập liệu hoạt động mà còn bao gồm các khía cạnh trực quan, bố cục, màu sắc và nội dung.

### Phân loại

Kiểm thử giao diện thủ công tức là con người trực tiếp quan sát và so sánh giao diện với bản thiết kế hoặc yêu cầu ban đầu. Với ưu điểm linh hoạt, phát hiện được lỗi trải nghiệm nhưng nhược điểm tốn nhiều thời gian, chi phí và dễ bỏ sót.

Kiểm thử giao diện tự động tức là sử dụng công cụ hoặc thuật toán để so sánh giao diện với mẫu gốc. Có thể tiếp cận theo hướng kiểm thử chức năng giao diện và kiểm thử giao diện trực quan. Việc kiểm thử chức năng giao diện tập trung vào việc xác minh chức năng như click button, nhập dữ liệu, chuyển trang… còn kiểm thử giao diện trực quan tập trung vào so sánh giao diện hiển thị (pixel, màu sắc, bố cục văn bản…) nhằm phát hiện sai lệch về hình ảnh, nội dung, bố cục.

## Các đặc trưng xử lý ảnh

### SSIM (Structural Similarity Index Measurement)

#### Giới thiệu SSIM

Đặc trưng SSIM phản ánh mức tương đồng cấu trúc sáng và tương phản giữa hai ảnh, thay thế so sánh pixel thô vốn nhạy với dịch chuyển nhỏ hoặc nhiễu. Nó giúp ghi nhận chênh lệch lớn về bố cục tổng thể, cắt xén vùng lớn hay thay đổi nền rõ rệt mà vẫn giữ chi phí tính toán thấp. Lý do sử dụng vì nhanh, ổn định và phù hợp vai trò trục “nền” để các đặc trưng khác bổ sung phần thông tin bị bỏ sót như màu, chữ hay điểm cục bộ. SSIM được sử dụng vì tốc độ thực thi nhanh, cảm nhận thị giác gần giống với mắt thường khi chúng ta nhìn và so sánh hình ảnh.

#### Triển khai SSIM

A diagram of a photo processing process

AI-generated content may be incorrect.

Hình ‑. Pipeline tính toán SSIM từ ảnh đầu vào

Trong quá trình triển khai, hai ảnh đầu vào trước tiên được chuyển sang dạng grayscale nhằm giảm bớt kênh màu và tập trung vào cấu trúc thị giác chính. Sau đó, ảnh được kiểm tra kích thước: nếu tổng số điểm ảnh vượt quá 700.000 thì hệ thống sẽ tự động thu nhỏ theo tỉ lệ nhằm đảm bảo cân bằng giữa việc giữ chi tiết quan trọng và giảm chi phí tính toán. Tiếp theo, bộ lọc median blur được áp dụng để triệt tiêu nhiễu muối tiêu nhưng vẫn giữ được biên và cấu trúc lớn, tránh làm mờ các yếu tố cần so sánh. Sau bước tiền xử lý, chỉ số SSIM (Structural Similarity Index) được tính toán giữa hai ảnh grayscale, với giá trị dao động trong khoảng từ gần 0 đến 1, trong đó giá trị càng cao biểu thị mức độ tương đồng càng lớn. Giá trị SSIM này được đưa trực tiếp vào vector đặc trưng mà không cần chuẩn hóa bổ sung do bản chất phân bố đã hẹp. Đặc biệt, trong trường hợp xảy ra ngoại lệ như ảnh lỗi hoặc không đọc được dữ liệu, hệ thống sẽ gán giá trị trung tính là 0.5. Việc chọn ngưỡng này nhằm tránh phá vỡ cân bằng dữ liệu huấn luyện, đồng thời giữ kết quả ở mức “không thiên lệch” giữa hai trạng thái giống và khác.

### OCR (Optical Character Recognition)

#### Giới thiệu OCR

Thành phần OCR nắm thông tin nội dung chữ và là lớp rủi ro nghiệp vụ (giá, khuyến mãi) mà các phép đo thuần thị giác thường bỏ sót vì hình dạng tổng thể không đổi. Trong nhiều trường hợp banner bị sửa con số hoặc thay cụm chữ ngắn nhưng nền và icon giữ nguyên, SSIM, Histogram và Layout đều cao, bổ sung điểm tương đồng văn bản làm giảm false negative với vi phạm nội dung.

#### Triển khai OCR

A diagram of a process

AI-generated content may be incorrect.

Hình ‑. Pipeline nhận diện và so sánh văn bản bằng OCR từ ảnh đầu vào

Trong quá trình triển khai, sử dụng thư viện Tesseract để nhận diện chữ sau khi ảnh được xử lý bằng adaptive threshold nhằm tăng độ tương phản. Văn bản được chuẩn hóa (loại bỏ ký tự đặc biệt, chuyển thành chữ thường) rồi tách thành tập hợp từ khóa. Từ đó, mức độ tương đồng giữa hai giao diện được tính bằng Jaccard Index, thông qua tỷ lệ giữa số lượng từ chung và tổng số từ của cả hai ảnh. Cách tiếp cận này có ưu điểm là tập trung vào nội dung hiển thị (textual content) thay vì toàn bộ pixel, do đó đặc biệt hiệu quả trong các trường hợp lỗi liên quan đến thay đổi chữ, font hoặc nội dung hiển thị. Tuy nhiên, kỹ thuật này cũng có hạn chế là chất lượng ảnh chụp màn hình, font chữ phức tạp hoặc ngôn ngữ đặc thù có thể làm giảm độ chính xác của OCR.

### Layout

#### Giới thiệu Layout

Đặc trưng Layout tập trung vào hình dạng tổng thể và vị trí khối nội dung. Khi một banner bị cắt bớt, chèn thêm panel, dịch chuyển cụm chữ lớn hoặc đảo bố cục, sự trùng khớp biên giảm mạnh dù màu hoặc văn bản còn giống. Nó giúp tách những thay đổi cấu trúc khỏi thay đổi nhỏ về màu hay ký tự, giảm phụ thuộc đơn thuần vào SSIM vốn có thể giữ cao khi dịch chuyển nhẹ kèm nền đồng đều.

#### Triển khai Layout

A diagram of a process

AI-generated content may be incorrect.

Hình ‑. Pipeline so sánh bố cục bằng IoU

Trong kỹ thuật so sánh bố cục (layout), hai ảnh sau khi được tiền xử lý (chuyển sang grayscale và đảm bảo cùng kích thước) sẽ được trích xuất biên bằng thuật toán Canny với ngưỡng 50–150. Kết quả thu được là các mặt nạ nhị phân thể hiện đường viền trong từng ảnh. Tiếp đó, hệ thống tính toán giao (intersection) và hợp (union) giữa hai mặt nạ để xác định chỉ số IoU (Intersection over Union) phản ánh mức độ trùng khớp của cấu trúc. Nếu hợp bằng 0 (trường hợp hiếm gặp khi ảnh hầu như trắng hoàn toàn), hệ thống trả về 1.0 để tránh lỗi chia cho 0. Giá trị IoU càng thấp cho thấy sự thay đổi bố cục càng lớn. Đặc trưng này không cần chuẩn hóa bổ sung và được đưa trực tiếp vào vector đặc trưng. Ngoài ra, do văn bản hoặc chi tiết nhỏ có thể tạo ra biên giả, việc áp dụng median blur trước Canny giúp giảm nhiễu và làm kết quả ổn định hơn.

### Histogram

#### Giới thiệu Histogram

Histogram màu biểu diễn phân bố cường độ của từng kênh RGB, giúp phát hiện thay đổi bảng màu nền, pha tông hoặc can thiệp chỉnh màu tổng thể khi cấu trúc hình học vẫn gần như giữ nguyên (khi đó SSIM còn cao). Việc thêm đặc trưng này giảm rủi ro bỏ sót trường hợp chỉnh tông thương hiệu, đảo màu hoặc thay background phẳng. Nó đóng vai trò mở rộng chiều “ngữ cảnh thị giác toàn cục” mà edge hay keypoint không thể hiện.

#### Triển khai Histogram

A diagram of a calculator and magnifying glass

AI-generated content may be incorrect.

Hình ‑. Pipeline tính toán Histogram từ ảnh màu

Ảnh được đọc ở dạng màu gốc (không chuyển về grayscale). Với mỗi kênh R, G, B, hệ thống tính histogram 32 bins và chuẩn hóa theo chuẩn L1 để loại bỏ ảnh hưởng tuyệt đối của độ sáng. Sau đó, mức độ tương đồng giữa hai histogram được đo bằng hệ số tương quan (CORREL) và được ánh xạ từ [−1,1][-1, 1][−1,1] về [0,1][0, 1][0,1] thông qua công thức (corr+1)/2(corr+1)/2(corr+1)/2. Giá trị cuối cùng là trung bình cộng của ba kênh màu. Trường hợp không đọc được ảnh, hệ thống trả về giá trị mặc định 0.5. Phương pháp này không sử dụng chia ô không gian (spatial bins) để giữ tính đơn giản, bởi đặc trưng về cấu trúc đã được đảm nhiệm bởi SSIM và Layout.

### ORB (Oriented FAST and Rotated BRIEF)

#### Giới thiệu ORB

Đặc trưng ORB bắt thay đổi cục bộ tinh vi như thêm hoặc xóa icon nhỏ, dịch chuyển logo, chỉnh nút CTA, thay mảng chi tiết nhỏ mà histogram và SSIM không nhạy. Khi phần lớn ảnh giữ nguyên nhưng một đối tượng thương hiệu bị thay, số và chất lượng cặp điểm đặc trưng trùng khớp sẽ giảm rõ, giúp mô hình đẩy mẫu vào almost hoặc different đúng mức.

#### Triển khai ORB

A diagram of a puzzle

AI-generated content may be incorrect.

Hình ‑. Pipeline tính toán ORB từ ảnh đầu vào

Trong kỹ thuật so sánh dựa trên đặc trưng ORB, hai ảnh sau khi được tiền xử lý sang dạng grayscale sẽ được đưa vào bộ phát hiện ORB (với tham số nfeatures = 150) để xác định các điểm đặc trưng và trích xuất descriptor nhị phân. Tiếp đó, hệ thống sử dụng BFMatcher với khoảng cách Hamming để so khớp descriptor, đồng thời áp dụng phương pháp KNN (k = 2) kết hợp tiêu chuẩn Lowe’s ratio (0.75) nhằm loại bỏ những cặp khớp kém tin cậy. Từ các cặp khớp tốt, hai chỉ số được tính toán: tỷ lệ phủ (ratio) – thể hiện mức độ tương ứng giữa số match tốt và số lượng keypoint tối thiểu của hai ảnh, và chất lượng khớp (quality) – được tính dựa trên khoảng cách trung bình giữa các match, quy đổi về thang điểm [0,1]. Điểm ORB cuối cùng được xác định bằng trung bình cộng có trọng số đều giữa ratio và quality, đồng thời được kẹp trong khoảng [0,1]. Trong trường hợp không có descriptor hoặc không tìm thấy match hợp lệ, hệ thống sẽ trả về giá trị trung lập 0.5. Việc lựa chọn số lượng đặc trưng vừa phải (150) giúp cân bằng giữa tốc độ xử lý và độ nhạy, đồng thời tránh việc trích xuất dư thừa ở các vùng phẳng vốn ít thông tin.

## Các thuật toán học máy áp dụng trong phân loại giao diện

### Thuật toán SVM (Support **Vector** Machine)

#### Giới thiệu SVM

SVM được chọn làm trụ cột giai đoạn đầu vì phù hợp không gian đặc trưng chiều thấp (5 đặc trưng) và khả năng tạo ranh giới quyết định rõ ràng giữa các mức sai lệch. Kernel linear là lựa chọn nhất quán tốt nhất trong toàn bộ quá trình phát triển và lý do em chọn thuật toán này ngay từ ban đầu bởi vì lý do ít cần nhiều siêu tham số, dễ phát hiện overfitting qua biến thiên giữa các fold, ổn định trên dữ liệu không quá lớn.

#### Triển khai SVM

A diagram of a diagram of a checklist and a brain

AI-generated content may be incorrect.

Hình ‑. Pipeline so sánh ảnh đa tính năng với SVM

Trong triển khai SVM, mô hình được huấn luyện trên vector đặc trưng 5 chiều gồm SSIM, OCR, Histogram, Layout và ORB. Do các đặc trưng này khác nhau về thang đo và mức độ phân tán, bước chuẩn hóa bằng StandardScaler là cần thiết để tránh việc một đặc trưng chi phối toàn bộ quá trình học.

Ở giai đoạn baseline, kernel tuyến tính được sử dụng nhằm kiểm tra khả năng phân tách đơn giản của dữ liệu, nhưng khi xuất hiện các đặc trưng phi tuyến như OCR hoặc ORB thì ranh giới tuyến tính không đủ mạnh, vì vậy kernel RBF được lựa chọn để mô hình hóa ranh giới phức tạp hơn. Trong đó, tham số C điều chỉnh mức phạt với các điểm phân loại sai, C lớn giúp tăng độ chính xác trên tập huấn luyện nhưng dễ dẫn đến overfitting, còn C nhỏ tạo biên mềm hơn, khái quát tốt hơn. Tham số gamma xác định phạm vi ảnh hưởng của một điểm dữ liệu, gamma cao làm ranh giới tập trung chặt quanh các điểm nhưng dễ quá khớp, trong khi gamma thấp tạo ranh giới mượt mà hơn nhưng có thể bỏ qua chi tiết. Quá trình tuning được thực hiện với GridSearchCV để tìm tổ hợp tối ưu giữa C, gamma và kernel, đồng thời GroupKFold được dùng để đảm bảo dữ liệu cùng gốc (original\_x) không bị chia cắt giữa tập huấn luyện và kiểm tra, hạn chế rò rỉ dữ liệu. Kết quả được đánh giá bằng ma trận nhầm lẫn gộp, trong đó lớp almost thường có xu hướng bị nhầm sang similar do ranh giới chưa rõ ràng; trong trường hợp này có thể điều chỉnh tham số C hoặc mở rộng thêm đặc trưng để cải thiện khả năng phân tách.

### Thuật toán RF (Random Forest)

#### Giới thiệu RF

Random Forest được đưa vào như một đối trọng dạng “tập hợp cây” để kiểm chứng mức ổn định và để đọc độ quan trọng đặc trưng. Việc nhiều cây trên các bootstrap khác nhau giảm nguy cơ lệ thuộc vào nhiễu cục bộ của một đặc trưng (ví dụ OCR dao động khi chữ mờ). Nó bổ sung góc nhìn khác nhau và cho thấy nếu một đặc trưng kỳ vọng quan trọng nhưng weight thấp, đó là tín hiệu cần xem lại cách tính.

#### Triển khai RF

A diagram of a brain

AI-generated content may be incorrect.

Hình ‑. Pipeline so sánh ảnh đa tính năng với RF

Mô hình Random Forest được xây dựng dựa trên 5 đặc trưng đầu vào: SSIM (so sánh cấu trúc), OCR Jaccard (từ khóa văn bản), Histogram (tương quan màu đã chuẩn hóa L1), ORB (tỷ lệ và chất lượng match đặc trưng cục bộ) và Layout (IoU đường biên Canny).

Trong cấu hình, tham số n\_estimators quy định số lượng cây trong rừng, càng nhiều cây thì mô hình càng ổn định nhưng chi phí tính toán cũng tăng. max\_depth kiểm soát độ sâu tối đa của mỗi cây, giúp tránh việc cây học quá chi tiết (overfitting). max\_features xác định số đặc trưng được xem xét tại mỗi lần chia, với lựa chọn 'sqrt' thường giúp tăng tính ngẫu nhiên và giảm tương quan giữa các cây. min\_samples\_split và min\_samples\_leaf lần lượt đặt ngưỡng tối thiểu số mẫu để một node có thể được chia tiếp hoặc để tồn tại như một lá, các tham số này đảm bảo cây không bị chia quá vụn, từ đó tăng tính tổng quát. Tham số random\_state đảm bảo tính tái lập kết quả.

Do bản chất RF tách dữ liệu bằng ngưỡng tại từng node, mô hình này không yêu cầu chuẩn hóa dữ liệu đầu vào. Quá trình đánh giá sử dụng GroupKFold theo nhóm original\_x để tránh rò rỉ thông tin, kết hợp với ma trận nhầm lẫn gộp để quan sát xu hướng phân loại nhầm giữa các lớp. Ngoài ra, các tham số được tinh chỉnh qua GridSearchCV, và sau khi chọn được cấu hình tốt nhất, mô hình được huấn luyện lại trên toàn bộ dữ liệu.

### Thuật toán kNN (k-Nearest Neighbors)

#### Giới thiệu kNN

kNN đóng vai trò baseline phi tham số nhằm kiểm tra “tính phân tách tự nhiên” của không gian đặc trưng. Nếu không gian đủ rõ ràng, kNN với k nhỏ sẽ đạt kết quả gần các mô hình tinh vi hơn, qua đó xác nhận chiến lược thiết kế đặc trưng đúng hướng. Ưu điểm của kNN là không giả định hình dạng ranh giới, nhược điểm là suy luận chậm khi dữ liệu lớn và rất nhạy với tỷ lệ thang đo giữa các chiều.

#### Triển khai kNN

A diagram of a checklist and brain

AI-generated content may be incorrect.

Hình ‑. Pipeline so sánh ảnh đa tính năng với kNN

Triển khai kNN trong hệ thống được thiết kế nhằm kiểm tra khả năng phân loại dựa trên khoảng cách trực tiếp giữa các mẫu. Mô hình sử dụng đủ 5 đặc trưng đầu vào: SSIM, OCR (Jaccard trên tập từ khóa), Histogram (tương quan màu đã chuẩn hóa L1), ORB (tỷ lệ và chất lượng match đặc trưng cục bộ) và Layout (IoU từ biên Canny). Do các đặc trưng này khác nhau về thang đo và độ phân tán, bước chuẩn hóa bằng StandardScaler là bắt buộc để đảm bảo không có đặc trưng nào (như SSIM) lấn át các đặc trưng khác.

Trong cấu hình, n\_neighbors xác định số lượng láng giềng được xét đến, đóng vai trò cân bằng giữa độ mượt và độ chi tiết của biên phân loại. Tham số weights='distance' giúp giảm tác động của các điểm xa, ưu tiên láng giềng gần hơn nhằm tăng độ chính xác. metric='manhattan' (L1) được chọn để phù hợp hơn với dữ liệu đa dạng thang đo, trong khi algorithm='brute' đảm bảo tính đúng đắn tuyệt đối khi tính toán khoảng cách (dù chi phí suy luận cao hơn).

Đánh giá mô hình được thực hiện bằng GroupKFold để tránh rò rỉ thông tin từ cùng một nhóm original\_x, kết quả được theo dõi qua accuracy từng fold, classification\_report và ma trận nhầm lẫn gộp. Quá trình tuning với GridSearchCV thử nghiệm trên các tham số quan trọng (n\_neighbors, weights, metric, algorithm) để tìm ra cấu hình tối ưu.

Tuy hiệu năng kNN thường thấp hơn các mô hình cây như RF hay XGBoost, nó đóng vai trò quan trọng trong việc kiểm chứng pipeline đặc trưng: nếu kNN thể hiện xu hướng phân loại tương tự với các mô hình mạnh hơn, ta có thể tin tưởng rằng đặc trưng và tiền xử lý đang hoạt động ổn định. Ngược lại, sự tụt giảm lớn ở kNN có thể gợi ý rằng cần xem xét lại chất lượng đặc trưng hoặc phương pháp chuẩn hóa.

### Thuật toán XGBoost (Extreme Gradient Boosting)

#### Giới thiệu XGBoost

XGBoost được bổ sung ở giai đoạn sau nhằm tối ưu sự cân bằng giữa độ chính xác trung bình và độ ổn định giữa các fold. Cơ chế boosting tuần tự cho phép tập trung sửa các lỗi còn sót lại sau khi đã có đủ đặc trưng (đặc biệt các trường hợp biên almost với different). Ngoài ra, XGBoost cung cấp nhiều dạng độ quan trọng đặc trưng, giúp xác thực lại đóng góp tương đối của OCR và ORB so với các thành phần toàn cục.

#### Triển khai XGBoost

A diagram of a brain

AI-generated content may be incorrect.

Hình ‑. Pipeline so sánh ảnh đa tính năng với XGBoost

Trong triển khai XGBoost, hệ thống khai thác đồng thời cả năm đặc trưng đầu vào gồm SSIM, OCR (tính bằng Jaccard trên tập từ khóa), Histogram (tương quan giữa các kênh sau khi chuẩn hóa L1), ORB (kết hợp tỷ lệ và chất lượng match) và Layout (IoU từ biên Canny). Vì các đặc trưng này đều có ý nghĩa định lượng riêng, không nhất thiết phải chuẩn hóa thang đo trước khi đưa vào mô hình. Mô hình XGBClassifier được thiết lập với mục tiêu phân loại đa lớp theo xác suất, trong đó tham số số lượng cây và độ sâu cây giúp kiểm soát độ phức tạp: cây nông hạn chế hiện tượng ghi nhớ quá mức, còn tốc độ học (learning\_rate) điều chỉnh mức cập nhật để cân bằng giữa tốc độ hội tụ và khả năng khái quát. Các tham số subsample và colsample\_bytree lần lượt kiểm soát tỷ lệ dữ liệu và số lượng đặc trưng được sử dụng ở mỗi cây, đóng vai trò như một dạng regularization ngẫu nhiên để giảm overfit. Ngoài ra, reg\_alpha (L1) và reg\_lambda (L2) tạo thêm ràng buộc, lần lượt giúp loại bỏ đặc trưng ít quan trọng và giữ ổn định hệ số.

Mô hình được đánh giá với GroupKFold nhằm đảm bảo rằng các biến thể cùng nhóm gốc không bị phân tán vào cả train và test, tránh rò rỉ thông tin. Mỗi fold sẽ huấn luyện, dự đoán và in ra các chỉ số như accuracy, classification report và confusion matrix. Sau đó, kết quả được tổng hợp cả về trung bình, độ lệch chuẩn và một báo cáo gộp toàn bộ folds. Để tối ưu, lưới tham số được thiết kế xoay quanh các yếu tố chính: số lượng cây, độ sâu cây, tốc độ học, tỷ lệ lấy mẫu, mức regularization L1/L2. GridSearchCV với GroupKFold cho phép tìm tổ hợp cân bằng nhất giữa bias–variance. Cuối cùng, mô hình tốt nhất sẽ được huấn luyện lại trên toàn bộ dữ liệu để sử dụng.

XGBoost còn cho phép trích xuất feature importances, nhờ đó có thể phân tích mức đóng góp tương đối của từng đặc trưng (SSIM, OCR, ORB, Histogram, Layout) và so sánh trực tiếp với kết quả của Random Forest để củng cố luận điểm trong báo cáo. Về khả năng mở rộng, có thể giảm learning\_rate và tăng số lượng cây trong khi vẫn giữ độ sâu nông và mức regularization ổn định, nhờ vậy mô hình sẽ học tinh chỉnh hơn trên dữ liệu lớn mà không cần thay đổi kiến trúc tổng thể.

# HỆ THỐNG KIỂM THỬ GIAO DIỆN TỰ ĐỘNG

## Giới thiệu hệ thống

Hệ thống kiểm thử giao diện tự động được thiết kế nhằm giải quyết bài toán so khớp và đánh giá mức độ tương đồng giữa ảnh quảng cáo gốc và các biến thể được triển khai tại nhiều chi nhánh. Trong thực tế, cùng một mẫu quảng cáo có thể xuất hiện dưới nhiều phiên bản khác nhau do sự khác biệt về thiết bị, môi trường triển khai, thao tác thủ công hoặc thậm chí lỗi trong quá trình phân phối nội dung. Những sai lệch này có thể biểu hiện dưới nhiều hình thức, chẳng hạn như thay đổi về màu sắc, dịch chuyển hoặc mất cân đối bố cục, thay đổi nội dung văn bản, thay thế hoặc biến dạng các biểu tượng, cũng như các sai khác cục bộ ở mức chi tiết nhỏ.

Để đáp ứng yêu cầu phát hiện và phân loại sai lệch một cách chính xác, hệ thống được xây dựng theo hướng tự động hóa toàn bộ quy trình từ khâu tiền xử lý ảnh, trích xuất đặc trưng cho đến giai đoạn huấn luyện mô hình phân loại. Thay vì chỉ dựa trên một đặc trưng duy nhất, hệ thống tận dụng sự kết hợp của nhiều đặc trưng bổ sung lẫn nhau như SSIM để đo cấu trúc toàn cục, Histogram để phân tích màu sắc, Layout để đánh giá bố cục, OCR để nhận diện và so sánh nội dung văn bản, và ORB để bắt chi tiết cục bộ. Các đặc trưng này được hợp nhất thành vector đặc trưng tổng hợp và đưa vào mô hình học máy giám sát để phân loại mức độ tương đồng (similar / almost similar / different).

Việc triển khai hệ thống không chỉ giúp tăng độ chính xác và tính ổn định của quá trình kiểm thử, mà còn giảm thiểu đáng kể thời gian và công sức so sánh thủ công vốn dễ xảy ra sai sót. Hơn nữa, hệ thống được thiết kế theo hướng linh hoạt, có thể mở rộng để tích hợp vào pipeline kiểm thử tự động hoặc kết hợp với các nền tảng quản lý chất lượng hiện có, từ đó nâng cao hiệu quả giám sát chất lượng quảng cáo trong môi trường triển khai đa chi nhánh.

## Phân tích hệ thống

### Bài toán và Phạm vi

Hệ thống được xây dựng nhằm giải quyết bài toán kiểm thử giao diện người dùng (UI) và quảng cáo đa chi nhánh theo cách tự động. Trong thực tế, các trang web hoặc ứng dụng thường hiển thị banner quảng cáo, hình ảnh sản phẩm, hay nội dung UI khác nhau ở từng chi nhánh. Việc đảm bảo tính nhất quán và tuân thủ về mặt hình ảnh là rất quan trọng, đặc biệt đối với doanh nghiệp có nhiều điểm bán.

Mục tiêu của hệ thống là phát hiện và phân loại mức độ khác biệt giữa ảnh gốc (chuẩn) và ảnh quan sát được từ chi nhánh/web. Hệ thống xuất ra 3 mức phân loại:

* Similar (Không vi phạm): hai ảnh hầu như giống nhau, sai khác nhỏ hoặc không đáng kể.
* Almost (Cần kiểm chứng): có sự thay đổi về chi tiết như font chữ, màu chữ, bố cục nhỏ… cần được kiểm tra thêm.
* Different (Vi phạm): khác biệt rõ rệt, thay đổi toàn diện hoặc sai lệch nghiêm trọng.

Phạm vi ứng dụng là so sánh 2 ảnh trực tiếp do người dùng cung cấp (manual testing) hoặc so sánh 1 ảnh gốc với nhiều ảnh crawl từ web chi nhánh (automated web testing).

Use case chính: Hệ thống hỗ trợ doanh nghiệp kiểm thử tính tuân thủ quảng cáo trên nhiều chi nhánh/website, nhằm đảm bảo nội dung và giao diện đồng bộ, tránh sai lệch hoặc vi phạm chính sách truyền thông.

### Phân tích Dữ liệu

* Cấu trúc dữ liệu:
* original\_image/: chứa ảnh gốc
* variant\_image/original\_X/: chứa các biến thể của ảnh gốc original\_X
* Mapping nhãn:
* identical, font\_color → Similar (không vi phạm).
* text\_changes, layout\_changes → Almost (cần kiểm chứng).
* changes, completely\_different → Different (vi phạm).
* Chiến lược đánh giá: sử dụng GroupKFold theo nhóm original\_X để đảm bảo các biến thể của cùng một ảnh gốc không bị phân tán đồng thời vào train và test, giúp tránh rò rỉ dữ liệu (data leakage).

### Đặc trưng và lý do lựa chọn

Bộ đặc trưng được xây dựng gồm 5 yếu tố bổ sung lẫn nhau, vừa khai thác pixel-level, vừa xem xét cấu trúcvà ngữ nghĩa:

* SSIM (Structural Similarity Index): đo mức độ tương đồng cấu trúc ở mức pixel, nhạy với biến đổi nhỏ về hình dạng và độ sáng.
* OCR Jaccard: trích xuất text bằng Tesseract, sau đó so sánh tập từ khóa bằng Jaccard similarity; giúp phát hiện thay đổi văn bản hoặc thông tin quảng cáo.
* Histogram Correlation: mô tả phân bố màu sắc qua histogram 32 bins trên ba kênh RGB, chuẩn hóa L1 để loại bỏ ảnh hưởng tuyệt đối của độ sáng; so sánh bằng hệ số tương quan.
* ORB Matching: phát hiện keypoint và so khớp descriptor nhị phân bằng Hamming distance, phù hợp để nhận diện logo, biểu tượng hoặc chi tiết hình ảnh cục bộ.
* Layout IoU: trích biên bằng Canny Edge, sau đó tính giao/union giữa hai mask nhị phân; giúp phát hiện thay đổi trong bố cục.

### Thuật toán và lý do lựa chọn

Các mô hình học máy được thử nghiệm trên tập dữ liệu với đặc trưng đầu vào gồm 5 chiều:

* SVM (Support Vector Machine): SVM được lựa chọn vì khả năng phân tách dữ liệu hiệu quả trong không gian đặc trưng thấp chiều. Với kernel RBF, mô hình có thể xử lý quan hệ phi tuyến phức tạp giữa các đặc trưng như OCR hay ORB, trong khi kernel tuyến tính đóng vai trò baseline. Tham số C điều chỉnh độ cứng của biên, còn gamma quyết định phạm vi ảnh hưởng của điểm dữ liệu. Mô hình yêu cầu chuẩn hóa dữ liệu bằng StandardScaler để các đặc trưng không chi phối lẫn nhau. Nhược điểm là chi phí tính toán cao khi dữ liệu lớn và khó giải thích trực quan.
* k-Nearest Neighbors (kNN): kNN được chọn như một thuật toán đơn giản, trực quan để kiểm tra mức độ tách lớp tự nhiên của vector đặc trưng. Mỗi dự đoán dựa trên nhãn của k láng giềng gần nhất, trong đó weights='distance' giảm ảnh hưởng của điểm xa. Do dựa trên khoảng cách, mô hình nhạy với thang đo nên cần chuẩn hóa trước khi huấn luyện. Ưu điểm là dễ hiểu và không cần quá trình huấn luyện phức tạp, nhưng kém hiệu quả khi dữ liệu lớn vì phải so sánh toàn bộ mẫu trong dự đoán.
* Random Forest (RF): RF được chọn vì tính ổn định cao và khả năng xử lý tốt dữ liệu không chuẩn hóa. Mô hình kết hợp nhiều cây quyết định (bagging) để giảm overfitting và tăng độ chính xác. Tham số quan trọng gồm số lượng cây (n\_estimators), độ sâu (max\_depth) và số đặc trưng chia tại mỗi node (max\_features). Ngoài độ chính xác, RF còn cung cấp feature importance để giải thích mức độ đóng góp của SSIM, OCR hay ORB. Tuy nhiên, ranh giới phân loại của RF thường ít mượt mà hơn so với boosting.
* XGBoost (Extreme Gradient Boosting): XGBoost được lựa chọn vì khả năng đạt hiệu năng cao nhất trong nhiều bài toán phân loại nhờ cơ chế boosting. Mỗi cây mới học từ sai số của cây trước, cùng với regularization L1/L2 để tránh overfitting. Các tham số như learning\_rate, n\_estimators, max\_depth, subsample và colsample\_bytree giúp điều chỉnh linh hoạt giữa bias và variance. XGBoost hỗ trợ đa luồng, tối ưu tốc độ, đồng thời cung cấp importance và SHAP để giải thích mô hình. Nhược điểm là cần tuning nhiều tham số và huấn luyện lâu hơn RF.

## Thiết kế hệ thống

### Kiến trúc hệ thống

A diagram of a software development

AI-generated content may be incorrect.

Hình ‑. Kiến trúc hệ thống

Hệ thống bắt đầu từ các cặp Input, được chuyển vào Loader & Group để liệt kê các trường hợp và nhóm theo group\_ids. Sau đó dữ liệu được đưa qua bước Preprocessing gồm chuyển sang ảnh xám, thay đổi kích thước và giảm nhiễu. Tiếp theo, 5 đặc trưng được tính toán song song, sau đó hợp nhất lại thành 1 vector 5 chiều. Quá trình huẩn luyện và đánh giá sử dụng GroupKFold, chia dữ liệu thành nhiều fold dựa trên nhóm. Với mỗi fold, mô hình được huẩn luyện trên tập train, sau đó dự đoán trên tập test và ghi lại output cùng các chỉ số đánh giá. Từ kết quả từng fold, hệ thống gom thống kê và tiến hành đánh giá tổng hợp, bao gồm trung bình và độ lệch chuẩn, ma trận nhầm lẫn và tầm quan trọng của từng đặc trưng. Cuối cùng dựa trên kết quả tổng hợp này, em sẽ đưa ra quyết định mô hình và thực hiện các điều chỉnh cần thiết.

### Các thành phần của hệ thống

- Input: Cặp (original.png, variant.png). Mỗi original\_x có nhiều biến thể với mục tiêu là so sánh một – một theo cặp.

- Loader & Group: Đọc thư mục original\_image và variant\_image/original\_x. Sau đó tạo danh sách theo đường dẫn dựa vào tên thư mục (path\_original, path\_variant, label). Cuối cùng là gán group\_id = số x để dùng GroupKFold.

- Preprocessing: Đọc grayscale. Downscale nếu tổng pixel vượt ngưỡng 700k, medianBlur(3) giảm nhiễu nhẹ.

- Feature Extraction: Mỗi đặc trưng chuẩn hóa về thang [0,1] để dễ học:

+ SSIM: Đánh giá cấu trúc và độ tương đồng cường độ sau đó đưa đầu ra gần bằng [0,1]

+ Histogram: So sánh phân bố màu (3 kênh, 32 bins) và dùng hệ số tương quan CORREL rồi map (corr+1)/2.

+ Layout: Canny edges để có thể IoU giữa hai mặt nạ cạnh và phản ánh thay đổi bố cục lớn

+ OCR: Ảnh được chuyển grayscale, áp dụng adaptive threshold rồi đưa vào Tesseract để trích xuất văn bản. Kết quả được làm sạch và tách thành token, sau đó so sánh bằng Jaccard giữa hai tập từ

+ ORB Keypoints: Phát hiện và mô tả điểm đặc trưng, lọc Lowe ratio, tính score = trung bình giữa mật độ match tương đối và chất lượng (1 - mean(distance)/256)

- 5D Vector: Ghép các đặc trưng SSIM, OCR, Hist, ORB, Layout để tiến hành quá trình huẩn luyện và kiểm tra

- Models:

+ SVM: Đầu vào được chuẩn hóa bằng StandardScaler để đưa các đặc trưng về cùng thang đo, giúp khoảng cách trong không gian đặc trưng không bị lệch. Mô hình SVC sử dụng kernel RBF (Radial Basis Function), cho phép ánh xạ phi tuyến các đặc trưng đầu vào sang không gian chiều cao hơn để phân tách tốt hơn các lớp khó phân biệt tuyến tính. Tham số C=10.0 điều chỉnh mức phạt cho các điểm phân loại sai, cân bằng giữa độ khái quát và độ chính xác. Tham số gamma='scale' tự động tính toán mức ảnh hưởng của một điểm huấn luyện trong không gian kernel, đảm bảo sự ổn định. Random state được cố định để kết quả tái lập

+ kNN: Do đặc trưng đầu vào có thang đo khác nhau, dữ liệu được chuẩn hóa bằng StandardScaler để tránh việc một đặc trưng (như SSIM) lấn át các đặc trưng khác. Mô hình KNeighborsClassifier sử dụng 11 láng giềng gần nhất (n\_neighbors=11) và gán trọng số theo khoảng cách (weights='distance'), nghĩa là các điểm gần sẽ ảnh hưởng mạnh hơn đến dự đoán. Khoảng cách được đo bằng Manhattan (L1) (metric='manhattan'), phù hợp khi dữ liệu đa chiều và có sự phân tán khác nhau giữa các đặc trưng. Việc tính toán khoảng cách được thực hiện bằng phương pháp vét cạn (algorithm='brute'), đảm bảo chính xác tuyệt đối dù chi phí tính toán cao hơn.

+ Random Forest: Mô hình gồm 140 cây (n\_estimators=140), mỗi cây có độ sâu tối đa giới hạn ở 10 (max\_depth=10) để tránh overfitting. Số đặc trưng được xem xét tại mỗi lần chia là căn bậc hai của tổng số đặc trưng (max\_features='sqrt'), giúp tăng tính ngẫu nhiên và giảm tương quan giữa các cây. Các tham số min\_samples\_split=6 và min\_samples\_leaf=2 đảm bảo node không bị chia quá nhỏ, từ đó tăng tính tổng quát. Mô hình không yêu cầu chuẩn hóa dữ liệu và có thể cung cấp feature importance để phân tích mức đóng góp tương đối của từng đặc trưng.

+ XGBoost: Mô hình sử dụng XGBClassifier để khai thác các quan hệ phi tuyến giữa đặc trưng. Số lượng cây được đặt là 70 (n\_estimators=70) với độ sâu tối đa 3 (max\_depth=3) giúp hạn chế overfitting. Tham số learning\_rate=0.05 điều chỉnh mức đóng góp của từng cây mới, cân bằng giữa tốc độ hội tụ và khả năng khái quát. Các tham số subsample=0.8 và colsample\_bytree=1.0 lần lượt kiểm soát tỷ lệ mẫu và tỷ lệ đặc trưng được sử dụng cho mỗi cây, đóng vai trò như regularization ngẫu nhiên. Bên cạnh đó, reg\_alpha=0.0 (L1) và reg\_lambda=1.0 (L2) tăng cường ràng buộc, giúp giảm overfitting và ổn định mô hình.

- Output: nhãn dự đoán similar / almost / different

- Evaluation:

+ GroupKFold (k=5) với mỗi original\_x chỉ nằm trong 1 fold test để loại bỏ leakage

+ Thu thập các giá trị mean accuracy ± std và confusion matrix, tầm quan trọng của từng feature với mục đích so sánh độ ổn định std. Sau đó có thể đưa ra quyết định có cần cải tiến mô hình nữa hay không.

### Luồng hoạt động của hệ thống

A diagram of a comparison between different types of objects

AI-generated content may be incorrect.

Hình ‑. Sơ đồ luồng hoạt động của hệ thống

Đối với so sánh ảnh trực tiếp, luồng này giao diện cho phép người dùng tải lên hai ảnh – một ảnh gốc và một ảnh cần kiểm thử. Người dùng có thể thêm hoặc xóa ảnh và nhấn nút So sánh để bắt đầu quá trình. Hệ thống tiến hành trích xuất năm loại đặc trưng gồm SSIM (đo mức độ tương đồng cấu trúc), OCR (so sánh từ khóa văn bản), Histogram (tương quan màu sắc đã chuẩn hóa), ORB (đặc trưng cục bộ) và Layout (IoU biên cạnh). Các đặc trưng được đưa vào mô hình học máy để dự đoán mức độ tương đồng giữa hai ảnh. Kết quả cuối cùng được hiển thị dưới ba trạng thái: Không vi phạm (Similar), Cần kiểm chứng (Almost) hoặc Vi phạm (Different).

Đối với kiểm thử web tự động, luồng này người dùng tải lên một ảnh gốc và nhập danh sách các chi nhánh, mỗi chi nhánh bao gồm tên, URL và CSS Selector của phần tử quảng cáo cần kiểm tra. Giao diện cho phép thêm hoặc xóa chi nhánh trước khi nhấn Chạy tất cả test để bắt đầu. Hệ thống sử dụng Selenium để truy cập từng trang web, tải ảnh theo selector và so sánh với ảnh gốc bằng pipeline đặc trưng và mô hình học máy. Kết quả được hiển thị dưới dạng lưới, trong đó mỗi ô tương ứng với một chi nhánh và trạng thái kiểm thử: Không vi phạm, Cần kiểm chứng hoặc Vi phạm. Người dùng có thể nhấp vào từng chi nhánh để mở modal chi tiết, xem trực quan ảnh gốc và ảnh crawl để đối chiếu.

## Các giai đoạn phát triển hệ thống

### Giai đoạn 1 (Mô hình sơ khai)

#### Mục tiêu và bối cảnh

Giai đoạn khởi đầu nhằm xây dựng baseline ổn định đó là khai thác đồng thời ba đặc trưng toàn cục SSIM (cấu trúc), Histogram (màu) và Layout (cạnh) để phát hiện sự khác biệt lớn giữa các ảnh, vẫn giữ ánh xạ nhãn ở mức 2 lớp similar/different. Mục tiêu chính là kiểm chứng pipeline trích xuất đặc trưng đa chiều, đánh giá hiệu quả phân loại bước đầu và xác lập nền so sánh cho các cải tiến tiếp theo.

#### Dữ liệu và đánh giá

Bản chất tổ chức dữ liệu (cặp original\_x và nhiều biến thể) và cơ chế gán group mỗi cặp (original, variant, nhãn) được gán group\_id theo ảnh gốc, tránh rò rỉ dữ liệu khi chia tập nên GroupKFold đóng vai trò then chốt trong đánh giá. Các thước đo gồm độ chính xác tổng, cấu trúc nhầm lẫn hai chiều và quan sát định tính các mẫu biên. Các giá trị số được dời sang phần kết quả, ở đây nhấn mạnh phương pháp luận nhất quán nhằm bảo đảm tính so sánh.

#### Quy trình kỹ thuật

A diagram of a data processing process

AI-generated content may be incorrect.

Hình ‑. Pipeline tiền xử lý và đặc trưng cơ bản

Chuỗi tiền xử lý ảnh xám, tiếp tục thu nhỏ đẳng tỷ lệ theo ngưỡng pixel và median blur nhẹ. Đặc trưng SSIM được giữ làm trục nền, Histogram được tính trên ảnh màu gốc bằng phân phối kênh trong số bin cố định rồi chuẩn hóa để loại bỏ ảnh hưởng độ sáng tuyệt đối, sau đó chuyển hệ số tương quan về miền [0,1] và Layout được suy ra qua phát hiện biên Canny trên hai ảnh xám đã xử lý và tính IoU giữa mặt nạ biên nhằm đo mức chồng lấp cấu trúc hình dạng. Bộ phân loại tiếp tục là SVM tuyến tính với cân bằng lớp để đảm bảo thay đổi hiệu năng đến từ chiều đặc trưng mới chứ không phải thay đổi thuật toán; chưa áp dụng tinh chỉnh mở rộng vì mục tiêu vẫn là đánh giá đóng góp đặc trưng.

#### Hạn chế và ý tưởng cải tiến

Mặc dù bắt đầu với ba đặc trưng giúp giảm đáng kể các trường hợp đổi tông màu hoặc dịch chuyển bố cục lớn bị xem là similar, những lỗi liên quan đến sửa chữ nhỏ, thay con số hoặc chỉnh chi tiết biểu tượng vẫn tồn tại vì hệ thống chưa có khả năng đọc văn bản hay định vị thay đổi cục bộ tinh vi. Đồng thời việc chỉ có hai lớp khiến một mảng biến thể trung gian bị dồn vào different tạo ra nhiễu cho quy trình nghiệp vụ vốn cần ưu tiên những sai lệch mạnh hơn. Điều này chỉ ra hai khoảng trống là thiếu thông tin ngữ nghĩa chữ và thiếu cấp phân loại trung gian để giảm nhầm lẫn hai cực. Quyết định cho giai đoạn kế tiếp là bổ sung OCR nhằm trích nội dung văn bản và mở rộng không gian nhãn thêm lớp almost, với giả thuyết rằng lớp chuyển tiếp sẽ giảm trao đổi sai giữa similar và different, đồng thời kênh văn bản giúp bắt các thay đổi mà SSIM, Histogram và Layout cùng bỏ sót.

### Giai đoạn 2 (Phát triển mô hình)

#### Mục tiêu và bối cảnh

Sau khi đã kiểm soát tốt các thay đổi về màu sắc và bố cục tổng thể, nhu cầu tiếp theo là phát hiện chính xác các chỉnh sửa nội dung chữ như giá, khuyến mãi hay câu slogan và tránh việc đẩy mọi sai lệch vừa phải vào nhóm different, gây quá tải kiểm thử thủ công. Để làm được điều này, không gian nhãn được mở rộng từ hai lên ba lớp bằng cách thêm almost làm tầng đệm ngữ nghĩa cho mức sai lệch chấp nhận được, đồng thời tích hợp đặc trưng OCR để mô hình nắm thông tin trực tiếp về mức tương đồng văn bản giữa hai ảnh. Bên cạnh đó, với trục cấu trúc (SSIM, Layout) và màu (Histogram) đã được khai thác, khả năng phát hiện các chỉnh sửa cục bộ tinh vi như thêm/xóa icon, dịch chuyển logo nhỏ hay thay nút hành động được bổ sung nhờ đặc trưng ORB, hoàn thiện vector năm chiều. Sự kết hợp này giúp tăng độ nhạy với các thay đổi vi mô và cải thiện khả năng phân tách giữa almost và different tại những ranh giới trước đó còn mềm.

#### Dữ liệu và đánh giá

Cấu trúc thư mục vẫn được giữ nguyên, nhưng ánh xạ nhãn được cập nhật: các biến thể liên quan đến thay đổi cục bộ nhẹ, biến đổi văn bản hoặc layout mức nhỏ được gom vào lớp almost thay vì ép vào similar hay different. GroupKFold tiếp tục đảm bảo mỗi original\_x chỉ xuất hiện một lần trong tập kiểm tra ở mỗi vòng, tăng độ tin cậy khi thêm lớp mới vốn dễ làm tăng phương sai đánh giá nếu phân bổ folds không đồng đều. Việc đo lường vẫn dựa trên độ chính xác tổng và ma trận nhầm lẫn, nhưng diễn giải tập trung hơn vào hai cặp nhầm lẫn then chốt similar với almost và almost với different vì chúng đại diện cho các ranh giới cần tối ưu. Phân bố kỳ vọng giữa ba lớp được theo dõi qua biểu đồ để xác nhận không có lệch quá mạnh, đảm bảo kết quả cross-validation vẫn đáng tin cậy. Sự quan sát ma trận nhầm lẫn tập trung vào hai cải thiện mong đợi: tăng khả năng bắt đúng different khi thay đổi cục bộ nhỏ và hạn chế việc hoán đổi giữa similar và almost. Nhìn chung, phương pháp đánh giá được duy trì nhất quán, cho phép so sánh xuyên giai đoạn trong khi thêm chiều đặc trưng mới, còn các số liệu chi tiết được trình bày ở phần kết quả.

#### Quy trình kỹ thuật

A diagram of a funnel with a diagram and a diagram of a diagram

AI-generated content may be incorrect.

Hình ‑. Pipeline mở rộng với OCR và ORB

Pipeline kế thừa toàn bộ bước tiền xử lý ảnh xám, tính SSIM, Histogram và Layout từ giai đoạn trước, đồng thời mở rộng vector đặc trưng bằng hai thành phần mới. Trước hết, OCR được áp dụng: mỗi ảnh xám sau median blur được nhị phân hóa thích nghi để tăng tương phản ký tự, đưa qua OCR trích chuỗi, làm sạch và đánh giá mức tương đồng bằng cách kết hợp các độ đo chuỗi và tập từ, nhằm cân bằng độ nhạy theo thứ tự và phổ từ vựng; kết quả được nén về một điểm ổn định, tránh cực trị khi thiếu văn bản. Sau đó, ORB được tính trên ảnh xám đã chuẩn hóa bằng cách phát hiện tập điểm đặc trưng với số lượng tối ưu, trích descriptor nhị phân, so khớp KNN với ngưỡng ratio để loại bỏ match kém, và tổng hợp mật độ match cùng chất lượng trung bình thành một điểm chuẩn hóa, với fallback trung tính khi không đủ descriptor hợp lệ. Vector đặc trưng năm chiều (SSIM, Histogram, Layout, OCR, ORB) được chuẩn hóa đồng nhất (nếu mô hình yêu cầu) trước khi huấn luyện SVM, sử dụng kernel tuyến tính hoặc RBF nhẹ nếu có tương tác phi tuyến giữa các đặc trưng, đồng thời hạn chế tuning sâu để bảo toàn khả năng quy kết cải tiến cho các đặc trưng mới. Việc thêm lớp almost giúp mô hình học ranh giới mềm giữa tương đồng hoàn toàn và khác biệt rõ, tạo tiền đề cho đánh giá chi tiết hơn ở các giai đoạn tiếp theo.

#### Hạn chế và ý tưởng cải tiến

Mặc dù việc bổ sung OCR và lớp almost giảm đáng kể nhầm lẫn trực tiếp giữa similar và different, vẫn tồn tại các trường hợp thay đổi rất nhỏ về biểu tượng, logo hoặc chi tiết đồ họa cục bộ mà Histogram, Layout hay OCR khó phát hiện, khiến mô hình đôi khi giữ chúng ở similar hoặc almost thay vì cảnh báo. Nguyên nhân là thiếu đặc trưng nhạy với cấu trúc vi mô và quan hệ keypoint giữa hai ảnh, trong khi các đặc trưng hiện tại thiên về thống kê toàn cục. Do đó, ORB được tích hợp để khai thác mô tả keypoint và phép so khớp cục bộ, với kỳ vọng tăng recall của lớp different cho các thay đổi nhỏ nhưng quan trọng mà các chiều toàn cục và nội dung chưa nắm bắt. Sau khi tích hợp ORB, phần lớn lỗi bỏ sót thay đổi cục bộ giảm rõ, nhưng trọng tâm chuyển sang tối ưu lựa chọn mô hình theo tiêu chí ổn định giữa các folds, khả năng giải thích và chi phí suy luận. SVM cho kết quả nhất quán nhưng vẫn có dao động ở một số nhóm và không cung cấp trực tiếp độ quan trọng tương đối của từng đặc trưng như mô hình cây, hay mức nhạy khoảng cách như kNN. Vì vậy, vector năm chiều được giữ nguyên và bước tiếp theo là so sánh đa mô hình (SVM, kNN, Random Forest, XGBoost) để chọn ứng viên triển khai cuối cùng dựa trên sự cân bằng giữa mean performance, độ lệch chuẩn, khả năng giải thích và tốc độ dự đoán.

### Giai đoạn 3 (So sánh mô hình)

#### Mục tiêu và bối cảnh

Với không gian đặc trưng đã được củng cố đủ chiều toàn cục và cục bộ, nhu cầu cuối cùng là đánh giá khách quan các thuật toán phân loại khác nhau trên cùng một vector nhằm xác định mô hình triển khai tối ưu và kiểm chứng rằng hiệu năng thu được không phải sản phẩm ngẫu nhiên của một kiến trúc đơn lẻ. Giai đoạn này tiến hành huấn luyện và đánh giá song song các mô hình đại diện cho nhiều họ: SVM (margin-based), kNN (phi tham số đo khoảng cách), Random Forest (tập hợp cây bagging) và XGBoost (boosting tuần tự).

#### Dữ liệu và đánh giá

Tập dữ liệu và ánh xạ ba lớp giữ nguyên hoàn toàn so với giai đoạn ngay trước, đảm bảo mọi khác biệt hiệu năng phản ánh thuần túy lựa chọn thuật toán. GroupKFold tiếp tục là khung đánh giá duy nhất để loại trừ leakage và cho phép so sánh độ ổn định bằng cách đối chiếu độ dao động giữa các folds cho từng mô hình. Các thước đo sẽ gồm độ chính xác trung bình, ma trận nhầm lẫn gộp và (đối với mô hình cây / boosting) phân tích tầm quan trọng đặc trưng, kNN bổ sung góc nhìn về sự phân tách tự nhiên của không gian vì nó không học tham số toàn cục. Các trị số cụ thể được dành cho phần kết quả, ở đây chỉ mô tả thiết kế thử nghiệm đồng nhất giữa các thuật toán để đảm bảo tính công bằng.

#### Quy trình kỹ thuật và chọn mô hình

Tầng trích đặc trưng giữ nguyên năm chiều. SVM tiếp tục sử dụng chuẩn hóa với StandardScaler để đảm bảo các chiều có cùng thang đo. kNN cũng áp dụng scaler vì nhạy mạnh với scale. Random Forest và XGBoost sử dụng trực tiếp đặc trưng thô vì cây quyết định theo ngưỡng và boosting xử lý chuẩn hóa nội bộ không bắt buộc. Mỗi mô hình được huấn luyện độc lập trên từng fold, lưu lại dự đoán để xây dựng ma trận nhầm lẫn hợp nhất, quá trình tinh chỉnh được thực hiện ở mức lưới nhỏ có kiểm soát nhằm tránh tối ưu quá sâu trên tập giới hạn và vẫn ưu tiên so sánh tương đối giữa họ thuật toán. Feature importance từ RF và XGBoost cung cấp bằng chứng bổ sung về đóng góp cân bằng giữa OCR, ORB và các trục toàn cục (SSIM, Histogram, Layout), trong khi kNN đóng vai trò kiểm tra lại rằng cấu trúc khoảng cách sau chuẩn hóa không gây lệch chiều.

### Giai đoạn 4 (Triển khai)

#### Mục tiêu triển khai

Giai đoạn triển khai hướng đến việc đưa mô hình phân loại đã lựa chọn vào vận hành trong hệ thống kiểm thử giao diện quảng cáo. Trọng tâm là tích hợp mô hình vào backend để cung cấp các API so sánh ảnh và xử lý batch chi nhánh, đồng thời xây dựng giao diện web thân thiện cho phép người dùng tải ảnh gốc, nhập danh sách chi nhánh và chạy toàn bộ kiểm thử chỉ với một thao tác.

Bên cạnh đó, triển khai cũng đảm bảo tính ổn định thông qua đóng gói các artifact (model, metadata), tiến hành kiểm thử bằng test case minh họa để xác nhận yêu cầu, và tạo nền tảng mở rộng về sau như tăng quy mô dữ liệu, tối ưu crawler hay tích hợp vào pipeline CI/CD.

#### Quá trình triển khai

Quá trình triển khai được thực hiện theo hướng tiếp cận tuần tự, từ chuẩn bị môi trường đến kiểm thử kết quả. Trước tiên, em tiến hành cài đặt và cấu hình môi trường cần thiết, bao gồm các thư viện và công cụ hỗ trợ cho việc so sánh ảnh. Sau đó, dữ liệu đầu vào (gồm ảnh gốc và các biến thể) được thu thập, tiền xử lý và chuẩn hóa kích thước nhằm đảm bảo tính đồng nhất khi đưa vào mô hình so sánh.

Tiếp theo, các mô hình và thuật toán đã lựa chọn được áp dụng để tiến hành so sánh và trích xuất kết quả. Trong giai đoạn này, em cũng thiết kế các bộ test case nhằm đánh giá khả năng nhận diện và phân loại khác biệt của hệ thống. Cuối cùng, kết quả được ghi nhận, tổng hợp và minh họa bằng bảng số liệu cũng như hình ảnh trực quan để chứng minh hiệu quả triển khai.

## Kết quả đạt được từng giai đoạn

### Giai đoạn 1

A blue rectangular object with white text

AI-generated content may be incorrect.

Hình ‑. Biểu đồ phân bố nhãn giai đoạn 1

Biểu đồ phân bố nhãn cho thấy dữ liệu không cân bằng, với lớp similar khoảng 120 mẫu và lớp different khoảng 240 mẫu, xấp xỉ tỷ lệ 1:2. Sự mất cân bằng này làm cho các chỉ số accuracy tổng thể có thể thiên lệch về lớp different chiếm ưu thế, do đó chỉ số này chưa phản ánh đầy đủ khả năng phân loại chính xác từng lớp. Để đánh giá công bằng, GroupKFold với k=5 được sử dụng, đảm bảo mỗi original\_x chỉ xuất hiện ở một fold test, từ đó hạn chế rò rỉ thông tin theo nhóm và phản ánh đúng biến thiên thực tế giữa các nhóm ảnh. Cách đánh giá này giúp mô hình được kiểm tra khả năng tổng quát hóa trên các nhóm khác nhau, đồng thời cho phép so sánh nhất quán giữa các fold và giữa các giai đoạn thử nghiệm khác nhau.

A graph with a line and a dotted line

AI-generated content may be incorrect.

Hình ‑4. Độ chính xác Cross-validation theo từng Fold (Version 1 - SVM)

Độ chính xác của mô hình dao động trong khoảng 76%–82% trên 5 fold, với trung bình khoảng 79% và độ lệch chuẩn tương đối nhỏ (khoảng 2–3%). Sự biến thiên này chủ yếu xuất phát từ khác biệt trong phân bố các thay đổi giữa các nhóm original\_x; ví dụ, một số nhóm chứa nhiều thay đổi chữ hoặc bố cục rõ rệt hơn, trong khi các nhóm khác ít thay đổi hơn, khiến hiệu năng của mô hình trên từng fold có sự chênh lệch. Đánh giá theo GroupKFold giúp phản ánh đúng biến thiên thực tế này, đồng thời cung cấp cái nhìn ổn định về khả năng tổng quát hóa của mô hình trên các nhóm dữ liệu khác nhau.

A graph with blue squares

AI-generated content may be incorrect.

Hình ‑. Ma trận nhầm lẫn (Version 1 - SVM)

Tổng hợp kết quả phân loại cho thấy mô hình gặp khó khăn chủ yếu ở việc phân biệt các mẫu similar và different. Cụ thể, trong lớp similar, 80 mẫu được dự đoán chính xác, còn 40 mẫu bị nhầm thành different. Ngược lại, trong lớp different, 205 mẫu được phân loại đúng, 35 mẫu bị nhầm sang similar. Khi tính toán các chỉ số theo lớp, lớp different đạt precision khoảng 84%, recall khoảng 85% và F1-score khoảng 0.85, trong khi lớp similar có precision khoảng 70%, recall 67% và F1-score 0.68. Kết quả này cho thấy mô hình hoạt động tốt hơn với lớp chiếm ưu thế (different), trong khi lớp similar bị ảnh hưởng bởi sự mất cân bằng và các thay đổi nhỏ khó nhận diện.

### Giai đoạn 2

A bar chart with blue squares

AI-generated content may be incorrect.

Hình ‑. Biểu đồ phân bố nhãn giai đoạn 2

Bộ dữ liệu hiện tại được cân bằng theo lớp, với mỗi lớp xấp xỉ 120 mẫu, đảm bảo tỷ lệ gần như đồng đều giữa similar, almost và different. Nhờ sự cân bằng này, các chỉ số đánh giá như accuracy, macro-F1 hay weighted-F1 phản ánh khách quan hơn khả năng phân loại từng lớp, không bị chi phối bởi lớp chiếm đa số như trong phiên bản trước. Điều này giúp các lỗi phân loại được phân bổ đều hơn và cho phép so sánh hiệu năng giữa các lớp một cách công bằng, đồng thời hỗ trợ việc đánh giá tác động của việc bổ sung đặc trưng mới hoặc mở rộng nhãn mà không bị lệch bởi sự mất cân bằng dữ liệu.

A graph with a line

AI-generated content may be incorrect.

Hình ‑. Độ chính xác Cross-validation theo từng Fold (Version 2 - SVM)

Độ chính xác của mô hình trên 5 fold dao động khoảng 69%–76%, với trung bình xấp xỉ 74%. Fold 4 có kết quả thấp hơn một chút so với các fold còn lại, gợi ý rằng mô hình nhạy cảm với một số nhóm original\_x chứa thay đổi chữ hoặc bố cục đặc thù, khiến khả năng phân loại trên nhóm này giảm. Để đảm bảo các đặc trưng khác nhau (SSIM, Histogram, Layout, OCR, ORB) được so sánh công bằng, bước chuẩn hóa bằng StandardScaler vẫn được duy trì, giúp cân bằng thang đo và hạn chế một số đặc trưng chiếm ưu thế, đồng thời giữ ổn định quá trình huấn luyện trên các fold khác nhau.

A graph with blue squares

AI-generated content may be incorrect.

Hình ‑. Ma trận nhầm lẫn (Version 2 - SVM)

Kết quả tổng hợp cho thấy lớp almost thực sự đóng vai trò như một vùng đệm giữa các lớp similar và different. Cụ thể, trong lớp similar, 66 mẫu được dự đoán chính xác, 53 mẫu bị nhầm thành almost và chỉ 1 mẫu sang different, dẫn đến recall khoảng 55% và precision khoảng 86%. Lớp almost có 93 mẫu được phân loại đúng, 11 mẫu sang similar và 16 mẫu sang different, tương ứng recall 77.5% và precision 59%. Lớp different ghi nhận 108 mẫu đúng, 12 mẫu bị nhầm sang almost và không có mẫu nào sang similar, với recall 90% và precision 86%. Nhìn chung, nhầm lẫn chủ yếu xuất hiện giữa các lớp kề nhau (similar với almost và almost với different), trong khi nhầm trực tiếp giữa similar với different gần như không xảy ra. Kết quả này xác nhận hiệu quả của việc bổ sung lớp trung gian almost, giúp tách các thay đổi nhỏ hoặc cục bộ liên quan đến văn bản và các chi tiết mảnh, đồng thời làm giảm đáng kể các lỗi dự đoán trực tiếp giữa các lớp cực đoan.

### Giai đoạn 3

Trên vector 5 đặc trưng (SSIM, OCR, Histogram, ORB, Layout):

* kNN: Accuracy 79.2% với std 1.5%
* SVM: Accuracy 81.9% với std 2.5%
* Random Forest: Accuracy 87.8% với std 3.4%
* XGBoost: Accuracy 88.6% với std 2%

Trên vector năm đặc trưng (SSIM, OCR, Histogram, ORB, Layout), hiệu năng các mô hình cho thấy sự khác biệt rõ rệt. XGBoost đạt accuracy cao nhất 88.6% với độ lệch chuẩn 2%, tiếp theo là Random Forest 87.8% ±3.4%, SVM 81.9% ±2.5% và kNN thấp nhất 79.2% ±1.5%. Kết quả này cho thấy các mô hình cây (RF, XGBoost) tận dụng hiệu quả các đặc trưng đa chiều, ổn định hơn SVM và kNN, trong khi kNN kém hơn về cả độ chính xác và khả năng mở rộng theo số lượng mẫu.

#### kNN (k-Nearest Neighbors)

A graph with a line

AI-generated content may be incorrect.

Hình ‑. Độ chính xác Cross-validation theo từng Fold (kNN)

kNN đạt accuracy trung bình 79.2% ±1.5%, thấp nhất trong bốn mô hình. Các fold dao động hẹp (77%–82%), cho thấy độ ổn định cao nhưng mức chính xác thấp. Chuẩn hóa bằng StandardScaler giữ cho các đặc trưng ở cùng thang đo, tuy nhiên bản chất của kNN khiến mô hình nhạy với nhiễu và khó mở rộng khi số mẫu tăng. So với SVM, kNN kém hơn một chút về chính xác, và thua xa RF, XGBoost ở khả năng tận dụng đặc trưng phức tạp.

A blue squares with white text

AI-generated content may be incorrect.

Hình ‑. Ma trận nhầm lẫn kNN

kNN phân biệt khá tốt “similar” và “different” với nhầm trực tiếp rất ít (1 mẫu mỗi chiều). Tuy nhiên, lớp “almost” vẫn là điểm yếu: precision (~66%) thấp hơn so với SVM, RF và XGBoost, do nhầm lẫn cả từ similar sang almost và almost sang different. Dù recall ở các lớp vẫn ở mức chấp nhận (~79%–89%), kNN không tạo được sự cân bằng tốt giữa precision và recall, khẳng định rằng đây không phải mô hình phù hợp để triển khai chính.

#### SVM (Support Vector Machine)

A graph with a line and a dotted line

AI-generated content may be incorrect.

Hình ‑. Độ chính xác Cross-validation theo từng Fold (SVM)

SVM đạt accuracy trung bình 81.9% ±2.5%, dao động trong khoảng 77%–85%. Độ ổn định giữa các fold khá tốt nhưng mức chính xác thấp hơn so với Random Forest và XGBoost, chỉ nhỉnh hơn kNN. Fold 2 cho kết quả thấp nhất (~77%), gợi ý sự nhạy cảm với phân phối đặc thù ở một số nhóm dữ liệu. Việc sử dụng StandardScaler giúp cân bằng thang đo giữa các đặc trưng SSIM, OCR, Histogram, ORB và Layout, đảm bảo hiệu năng nhất quán, tuy nhiên khả năng tổng quát hóa của SVM vẫn kém hơn các mô hình cây.

A blue squares with white text

AI-generated content may be incorrect.

Hình ‑. Ma trận nhầm lẫn SVM

SVM phân biệt tốt hai cực “similar” và “different”, với nhầm lẫn trực tiếp gần như không đáng kể (1 mẫu mỗi chiều). Tuy vậy, lớp “almost” khó xử lý hơn: recall cao (~86%) nhưng precision thấp (~69%), cho thấy nhiều nhầm lẫn từ similar sang almost và almost sang different. So với RF và XGBoost, SVM kém cân bằng hơn giữa precision và recall, đặc biệt ở lớp trung gian, phản ánh hạn chế của siêu phẳng tuyến tính khi xử lý biên phức tạp.

#### RF (Random Forest)

A graph with a line and a point

AI-generated content may be incorrect.

Hình ‑. Độ chính xác Cross-validation theo từng Fold (RF)

Random Forest đạt accuracy trung bình 87.8% ±3.4%, cao hơn hẳn SVM và kNN, và chỉ thấp hơn XGBoost một chút. Độ dao động giữa các fold (83%–93%) cho thấy khả năng tổng quát hóa tốt trên các nhóm dữ liệu khác nhau. Là mô hình cây, RF không yêu cầu chuẩn hóa đặc trưng, và vẫn tận dụng hiệu quả các chiều SSIM, OCR, Histogram, ORB, Layout. So sánh chung, RF ổn định hơn SVM và kNN, và gần tương đương XGBoost về độ chính xác.

A blue squares with white text

AI-generated content may be incorrect.

Hình ‑. Ma trận nhầm lẫn RF

RF phân biệt rõ ràng cả ba lớp, đặc biệt với “different” (recall ~92.5%, precision ~89.5%). Lớp “similar” và “almost” cũng đạt kết quả cân bằng với precision/recall đều trên 80%. Nhầm lẫn chủ yếu vẫn nằm ở ranh giới gần nhau (similar và almost, almost và different), trong khi nhầm trực tiếp similar và different cực hiếm (2 mẫu). So với SVM và kNN, RF thể hiện sự vượt trội cả về độ chính xác lẫn độ cân bằng giữa các lớp.

#### XGBoost (Extreme Gradient Boosting)

A graph with a line and a dotted line

AI-generated content may be incorrect.

Hình ‑. Độ chính xác Cross-validation theo từng Fold (XGBoost)

XGBoost đạt accuracy trung bình 88.6% ±2%, cao nhất trong các mô hình. Các fold dao động trong khoảng 86%–93%, độ lệch chuẩn nhỏ phản ánh sự ổn định và khả năng tổng quát hóa mạnh. Là mô hình boosting trên cây, XGBoost tận dụng tốt mối tương quan phi tuyến giữa các đặc trưng và ít nhạy với thang đo, vượt trội hơn SVM và kNN, đồng thời nhỉnh hơn RF ở mức chính xác trung bình.

A blue squares with white text

AI-generated content may be incorrect.

Hình ‑. Ma trận nhầm lẫn (Version 5 - XGBoost)

XGBoost phân loại ba lớp với độ chính xác cao, đặc biệt “different” (recall ~94.2%, precision ~91.1%) và “similar” (recall/precision đều ~88.3%). Lớp “almost” cũng đạt cân bằng hơn RF và SVM với precision ~86.2%, recall ~83.3%. Nhầm lẫn trực tiếp giữa “similar” và “different” rất hiếm (2–3 mẫu), xác nhận vai trò “đệm” hiệu quả của lớp almost. Nhìn chung, XGBoost không chỉ đạt độ chính xác cao nhất mà còn duy trì cân bằng tốt nhất giữa ba lớp, thể hiện rõ ưu thế so với các mô hình còn lại.

#### Kết quả so sánh mô hình

Bảng . Bảng kết quả so sánh các mô hình

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mô hình | Accuracy (±std) | Precision lớp Almost | Ưu điểm chính | Hạn chế chính |
| kNN | 79.2% ±1.5% | ~66% | Đơn giản, dễ triển khai; phân biệt khá rõ Similar và Different | Chính xác thấp; rất nhạy với nhiễu; lớp Almost khó phân tách; kém mở rộng |
| SVM | 81.9% ±2.5% | ~69% | Phân biệt tốt Similar/Different; recall lớp Almost cao (~86%); ổn định hơn kNN | Precision thấp ở lớp Almost; cần tuning kernel/gamma; kém cân bằng giữa lớp |
| Random Forest | 87.8% ±3.4% | >80% | Hiệu năng cao; ổn định giữa các fold; không cần chuẩn hóa; feature importance | Std cao hơn XGBoost; đôi khi nhầm ở biên Similar/Almost và Almost/Different |
| XGBoost | 88.6% ±2.0% | ~86% | Accuracy cao nhất; std nhỏ cho thấy sự ổn định; cân bằng tốt giữa 3 lớp; mạnh với dữ liệu phi tuyến | Nhiều tham số cần tuning; tính toán nặng hơn kNN và SVM |

Dựa trên bảng so sánh, có thể thấy hiệu năng giữa các mô hình có sự khác biệt rõ rệt. kNN và SVM tuy đơn giản, ổn định nhưng bộc lộ hạn chế lớn ở lớp trung gian “almost”, với precision lần lượt chỉ khoảng 66% và 69%, dẫn đến mất cân bằng giữa precision và recall. Trong khi đó, Random Forest và XGBoost tận dụng tốt đặc trưng đa chiều, giúp đạt độ chính xác cao hơn hẳn, đặc biệt XGBoost không chỉ đạt accuracy trung bình cao nhất 88.6% ±2% mà còn duy trì độ cân bằng tốt giữa cả ba lớp, với precision lớp “almost” đạt tới 86%. Như vậy, có thể khẳng định các mô hình cây, đặc biệt là XGBoost, phù hợp hơn để triển khai trong giai đoạn tiếp theo.

### Giai đoạn 4

# KẾT LUẬN VÀ HƯƠNG PHÁT TRIỂN

## Kết luận

Hệ thống đã hoàn thiện một quy trình kiểm thử giao diện quảng cáo tự động, từ huấn luyện mô hình đến triển khai web. Về mô hình, pipeline đặc trưng 5 chiều (SSIM, OCR Jaccard, Histogram màu, ORB keypoints, Layout IoU) kết hợp XGBoost cho kết quả ổn định qua GroupKFold, đạt độ chính xác cao (xấp xỉ 90% tùy fold), tách được ba mức similar/almost/different theo đúng ý nghĩa nghiệp vụ.

Về ứng dụng, giao diện Flask đã hỗ trợ hai luồng dùng: so sánh ảnh trực tiếp 2 ảnh và kiểm thử tự động với 1 ảnh gốc dùng chung cho nhiều chi nhánh, crawl ảnh theo CSS selector bằng Selenium, xử lý lazy loading/ẩn bởi JS, và hiển thị kết quả dạng grid kèm summary và modal chi tiết. Các artifact (model.pkl/json, metadata.json) giúp đồng bộ hóa thứ tự đặc trưng và ánh xạ nhãn khi suy luận. Tổng thể, bài làm giải quyết được nhu cầu kiểm thử tuân thủ quảng cáo ở quy mô nhiều chi nhánh: tự động thu thập ảnh, so sánh thống nhất theo mô hình ML, giảm thời gian kiểm tra thủ công và chuẩn hóa tiêu chí đánh giá.

Ưu điểm:

1. Thiết kế đặc trưng mang tính bổ sung lẫn nhau giúp mô hình vững trước nhiều kiểu sai lệch (nội dung chữ, màu sắc, bố cục, chi tiết cục bộ).
2. Đánh giá bằng GroupKFold tránh rò rỉ theo nhóm original\_x, tăng độ tin cậy.
3. Ứng dụng web đơn giản, thân thiện: 1 ảnh gốc + danh sách URL/selector, 1 nút “Chạy tất cả”, kết quả theo grid có tổng quan (safe/warning/danger/failed) và modal chi tiết.
4. Crawler đã tính đến các trường hợp ảnh bị ẩn/lazy-load, thêm timeout và force-visible để tăng tỉ lệ thành công.

Khuyết điểm:

1. Phụ thuộc chất lượng OCR; chữ nhỏ, font lạ hoặc ảnh nén mạnh có thể làm giảm độ chính xác.
2. Selenium headless chậm và dễ bị anti-bot; một số selector động, slider hoặc background-image phức tạp vẫn có thể thất bại.
3. Xử lý tuần tự, chưa có hàng đợi/đồng thời; không lưu lịch sử, chưa có cơ chế theo dõi, retry hay báo cáo dài hạn.
4. Dữ liệu huấn luyện còn hạn chế quy mô và cân bằng lớp; mô hình chưa học từ dữ liệu động (video/animation) hay đa ngôn ngữ.
5. Một số tham số xử lý ảnh còn cố định (ngưỡng Canny, nfeatures ORB…), chưa có auto-tuning theo từng loại banner/chiến dịch.

## Hướng phát triển

* Về dữ liệu và mô hình: Mở rộng dữ liệu theo chiến dịch/nhãn hàng, cân bằng lớp và bổ sung case biên (chữ rất nhỏ, màu gần nhau, thay đổi cục bộ tinh vi). Khảo sát hướng học biểu diễn bằng deep learning (Siamese/Triplet loss) để học tương đồng trực tiếp từ ảnh, kết hợp học chữ (OCR) và bố cục (vision transformer/layout). Chuẩn hóa pipeline đánh giá với nested CV và theo dõi drift sau triển khai; bổ sung cơ chế hiệu chỉnh ngưỡng ra quyết định theo từng doanh nghiệp/đợt chiến dịch.
* Về giải pháp crawl và hiệu năng: Chuyển Selenium sang Playwright để ổn định và dễ bypass lazy-loading/visibility; xây dựng browser pool và timeout thích ứng. Tách crawler thành dịch vụ riêng, chạy song song bằng hàng đợi (Celery/RQ) và worker pool; có cơ chế retry theo mã lỗi (DNS, 403, content-type không phải ảnh, selector không khớp). Bổ sung logic xử lý ảnh nền (background-image), srcset, data-src và CDN có query ký tự đặc biệt; cache theo (url, selector) để giảm tải.
* Về kiến trúc hệ thống: Tách microservices theo 3 nhánh chính: crawling service, image-processing/inference service, reporting service. Đóng gói Docker, triển khai bằng Docker Compose/Kubernetes với scale theo hàng đợi. Bổ sung giám sát (Prometheus/Grafana), log tập trung (ELK), health-check và alert khi tỉ lệ crawl fail vượt ngưỡng. Lưu kết quả vào database (chiến dịch, chi nhánh, nhãn dự đoán, confidence, đường dẫn ảnh, thời gian) và cung cấp REST API để hệ thống khác tích hợp.
* Về trải nghiệm người dùng và quy trình: Thêm tiến trình hiển thị theo thời gian thực (progress per site), phân trang/lọc theo trạng thái, xuất báo cáo PDF/Excel, lưu và so sánh lịch sử theo chiến dịch. Cho phép cấu hình ngưỡng cảnh báo theo nhãn hàng, rule nghiệp vụ (từ khóa bắt buộc, vùng ROI quan trọng). Bổ sung cơ chế human-in-the-loop: hàng đợi duyệt thủ công cho lớp “almost”, học tăng cường từ phản hồi để cập nhật ngưỡng/mô hình.
* Về khía cạnh kinh doanh: Đóng gói thành dịch vụ SaaS đa tenant; định giá theo số lượng website/lượt kiểm thử/tháng. Cung cấp mẫu báo cáo compliance cho marketing/legal, dashboard xu hướng vi phạm theo khu vực/chi nhánh/chiến dịch, lịch kiểm thử định kỳ và cảnh báo sớm. Tích hợp SSO, phân quyền theo vai trò (admin, reviewer, viewer) và webhook/Email/Slack để thông báo tự động. Xây dựng lộ trình mở rộng sang so sánh video/ngữ nghĩa nội dung, và gắn với chuẩn brand guideline để tăng giá trị doanh nghiệp.

TÀI LIỆU THAM KHẢO

[1] A. A. Fadul and K. A. Ibrahim, “Challenges in GUI Test Automation for Dynamic Web Applications: A Systematic Review,” Aug. 2025. [Online]. Available: https://www.researchgate.net/publication/394449177\_Challenges\_in\_GUI\_Test\_Automation\_for\_Dynamic\_Web\_Applications\_A\_Systematic\_Review. [Accessed: Sep. 15, 2025].

[2] J. Yao, J. Shen, and C. Yao, “Image quality assessment based on the perceived structural similarity index of an image,” Mathematical Biosciences & Engineering, vol. 20, no. 5, pp. 9385–9409, Mar. 2023, doi: 10.3934/mbe.2023412. [Online]. Available: https://www.researchgate.net/publication/369365215\_Image\_quality\_assessment\_based\_on\_the\_perceived\_structural\_similarity\_index\_of\_an\_image. [Accessed: Sep. 15, 2025].

[3] H. Putra, A. M. Arymurthy, and D. Chahyati, “Enhancing Bounding Box Regression for Object Detection: Dimensional Angle Precision IoU-Loss,” IEEE Access, Jan. 2025, doi: 10.1109/ACCESS.2025.3567767. [Online]. Available: https://www.researchgate.net/publication/391528309\_Enhancing\_Bounding\_Box\_Regression\_for\_Object\_Detection\_Dimensional\_Angle\_Precision\_IoU-Loss. [Accessed: Sep. 15, 2025].

[4] F. Lu, G.-H. Liu, and X.-Z. Gao, “Image Retrieval Using Multilayer Feature Aggregation Histogram,” Cognitive Computation, vol. 16, no. 6, pp. 2902–2915, Aug. 2024, doi: 10.1007/s12559-024-10334-9. [Online]. Available: https://www.researchgate.net/publication/383459245\_Image\_Retrieval\_Using\_Multilayer\_Feature\_Aggregation\_Histogram. [Accessed: Sep. 15, 2025].

[5] R. Jothy, “A Comparative Analysis of SIFT, SURF and ORB on Sketch and Paint based images,” Feb. 2023. [Online]. Available: https://www.researchgate.net/publication/368189389\_A\_Comparative\_Analysis\_of\_SIFT\_SURF\_and\_ORB\_on\_Sketch\_and\_Paint\_based\_images. [Accessed: Sep. 15, 2025].

[6] P. Sharma, “Advancements in OCR: A Deep Learning Algorithm for Enhanced Text Recognition,” International Journal of Inventive Engineering and Sciences, vol. 10, no. 8, pp. 1–7, Aug. 2023, doi: 10.35940/ijies.F4263.0810823. [Online]. Available: https://www.researchgate.net/publication/373513855\_Advancements\_in\_OCR\_A\_Deep\_Learning\_Algorithm\_for\_Enhanced\_Text\_Recognition. [Accessed: Sep. 15, 2025].

[7] D. Zhang and T. Chen, “Scikit-ANFIS: A Scikit-Learn Compatible Python Implementation for Adaptive Neuro-Fuzzy Inference System,” Int. J. Fuzzy Syst., vol. 26, no. 6, Jun. 2024, doi: 10.1007/s40815-024-01697-0. [Online]. Available: https://www.researchgate.net/publication/381131594\_Scikit-ANFIS\_A\_Scikit-Learn\_Compatible\_Python\_Implementation\_for\_Adaptive\_Neuro-Fuzzy\_Inference\_System. [Accessed: Sep. 15, 2025].

[8] S. E. B. Simatupang, A. Nababan, R. A. E. Tarihoran, and J. Banjarnahor, “Comparative Performance Analysis of Decision Tree And SVM Algorithms in Detecting Multiple System Atrophy Based on Clinical Features,” SinkrOn, vol. 9, no. 3, pp. 1146–1152, Jul. 2025, doi: 10.33395/sinkron.v9i3.15073. [Online]. Available: https://www.researchgate.net/publication/394048129\_Comparative\_Performance\_Analysis\_of\_Decision\_Tree\_And\_SVM\_Algorithms\_in\_Detecting\_Multiple\_System\_Atrophy\_Based\_on\_Clinical\_Features. [Accessed: Sep. 15, 2025].

[9] M. Nalluri, M. Pentela, and N. Eluri, “A Scalable Tree Boosting System: XG Boost,” Jan. 2020, doi: 10.22259/2349-476X.0712005. [Online]. Available: https://www.researchgate.net/publication/372479561\_A\_Scalable\_Tree\_Boosting\_System\_XG\_Boost. [Accessed: Sep. 15, 2025].

PHỤ LỤC TEST CASE

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Ảnh gốc | Ảnh biến thể | Kết quả thực tế | Kết quả mong muốn | Kết quả |
| 1 |  | font\_color | Không vi phạm | Không vi phạm | Pass |
| 2 |  | layout | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 3 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 4 |  | changes | Vi phạm | Vi phạm | Pass |
| 5 |  | font\_color | Không vi phạm | Không vi phạm | Pass |
| 6 |  | layout | Vi phạm | Cần kiểm chứng | Fail |
| 7 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 8 |  | changes | Vi phạm | Vi phạm | Pass |
| 9 |  | font\_color | Cần kiểm chứng | Không vi phạm | Fail |
| 10 |  | layout | Không vi phạm | Cần kiểm chứng | Fail |
| 11 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 12 |  | changes | Vi phạm | Vi phạm | Pass |
| 13 |  | font\_color | Không vi phạm | Không vi phạm | Pass |
| 14 |  | layout | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 15 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 16 |  | changes | Không vi phạm | Vi phạm | Fail |
| 17 |  | font\_color | Không vi phạm | Không vi phạm | Pass |
| 18 |  | layout | Vi phạm | Cần kiểm chứng | Fail |
| 19 |  | text | Không vi phạm | Cần kiểm chứng | Fail |
| 20 |  | changes | Vi phạm | Vi phạm | Pass |
| 21 |  | font\_color | Không vi phạm | Không vi phạm | Pass |
| 22 |  | layout | Vi phạm | Cần kiểm chứng | Fail |
| 23 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 24 |  | changes | Vi phạm | Vi phạm | Pass |
| 25 |  | font\_color | Không vi phạm | Không vi phạm | Pass |
| 26 |  | layout | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 27 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 28 |  | changes | Vi phạm | Vi phạm | Pass |
| 29 |  | font\_color | Không vi phạm | Không vi phạm | Pass |
| 30 |  | layout | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 31 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 32 |  | changes | Vi phạm | Vi phạm | Pass |
| 33 |  | font\_color | Không vi phạm | Không vi phạm | Pass |
| 34 |  | layout | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 35 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 36 |  | changes | Vi phạm | Vi phạm | Pass |
| 37 |  | font\_color | Không vi phạm | Không vi phạm | Pass |
| 38 |  | layout | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 39 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 40 |  | changes | Vi phạm | Vi phạm | Pass |
| 41 |  | font\_color | Không vi phạm | Không vi phạm | Pass |
| 42 |  | layout | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 43 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 44 |  | changes | Vi phạm | Vi phạm | Pass |
| 45 |  | font\_color | Không vi phạm | Không vi phạm | Pass |
| 46 |  | layout | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 47 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 48 |  | changes | Vi phạm | Vi phạm | Pass |
| 49 |  | font\_color | Không vi phạm | Không vi phạm | Pass |
| 50 |  | layout | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 51 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 52 |  | changes | Vi phạm | Vi phạm | Pass |
| 53 |  | font\_color | Không vi phạm | Không vi phạm | Pass |
| 54 |  | layout | Vi phạm | Cần kiểm chứng | Fail |
| 55 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 56 |  | changes | Vi phạm | Vi phạm | Pass |
| 57 |  | font\_color | Không vi phạm | Không vi phạm | Pass |
| 58 |  | layout | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 59 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 60 |  | changes | Vi phạm | Vi phạm | Pass |
| 61 |  | font\_color | Cần kiểm chứng | Không vi phạm | Fail |
| 62 |  | layout | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 63 |  | text | Cần kiểm chứng | Cần kiểm chứng | Pass |
| 64 |  | changes | Vi phạm | Vi phạm | Pass |