HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG



ĐỖ HƯƠNG HÀ

PHÁT TRIỂN HỆ THỐNG HỖ TRỢ CỐ VẤN HỌC TẬP TRONG CÔNG TÁC QUẢN LÝ SINH VIÊN

ĐỀ ÁN TỐT NGHIỆP THẠC SĨ KỸ THUẬT

(*Theo định hướng ứng dụng*)

Hà Nội, 2026

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG



ĐỖ HƯƠNG HÀ

PHÁT TRIỂN HỆ THỐNG HỖ TRỢ CỐ VẤN HỌC TẬP TRONG CÔNG TÁC QUẢN LÝ SINH VIÊN

Chuyên ngành: Hệ thống thông tin

Mã số: 8.48.01.04

ĐỀ ÁN TỐT NGHIỆP THẠC SĨ KỸ THUẬT

(***Theo định hướng ứng dụng***)

NGƯỜI HƯỚNG DẪN KHOA HỌC: PGS.TS. NGÔ QUỐC DŨNG

Hà Nội, 2026

LỜI CAM ĐOAN

Tôi xin cam đoan đây là công trình nghiên cứu của riêng tôi.

Các số liệu, kết quả nêu trong đề án là trung thực và chưa từng được ai công bố trong bất kỳ công trình nào khác.

Tôi xin cam đoan rằng mọi sự giúp đỡ cho việc thực hiện đề án này đã được cảm ơn và các thông tin trích dẫn trong đề án đã được chỉ rõ nguồn gốc.

Hà Nội, ngày ... tháng ... năm ...

Đỗ Hương Hà

LỜI CẢM ƠN

Trong suốt quá trình học tập, nghiên cứu để thực hiện đề án, ngoài nỗ lực của bản thân, tôi cũng đã nhận được sự hướng dẫn nhiệt tình của quý thầy cô, cùng với sự động viên và ủng hộ của gia đình, bạn bè và đồng nghiệp. Với lòng kính trọng và biết ơn sâu sắc, tôi xin gửi lời cảm ơn chân thành tới: Ban Giám Đốc, Phòng đào tạo sau đại học và quý thầy cô đã tạo mọi điều kiện thuận lợi giúp tôi hoàn thành đề án.

Tôi xin chân thành cảm ơn Thầy PGS. TS. Ngô Quốc Dũng đã hết lòng giúp đỡ, hướng dẫn, động viên, tạo điều kiện cho tôi trong suốt quá trình thực hiện và hoàn thành đề án.

Tôi rất mong nhận được sự góp ý của quý thầy cô cùng bạn bè đồng nghiệp để kiến thức của tôi ngày một hoàn thiện hơn.

Xin chân thành cảm ơn!

Hà Nội, ngày … tháng … năm 20…

Học viên thực hiện đề án

Đỗ Hương Hà

MỤC LỤC

LỜI CAM ĐOAN i
LỜI CẢM ƠN ii
MỤC LỤC iii
DANH MỤC HÌNH VẼ ii
DANH MỤC BẢNG BIỂU iii
DANH MỤC TỪ VIẾT TẮT iv
MỞ ĐẦU vi
1. Bối cảnh đề tài vi
2. Thực trạng vi
3. Tính cấp thiết của đề tài vii
4. Phạm vi và đối tượng nghiên cứu viii
CHƯƠNG I: TỔNG QUAN VỀ HỆ THỐNG HỖ TRỢ CỐ VẤN HỌC TẬP 2
1.1. Tổng quan về công việc cố vấn học tập 2
1.2. Tổng quan về trực quan hóa dữ liệu 3
1.3. Tổng quan về công nghệ dữ liệu 4
1.3.1. Data Pipeline 4
1.3.2. Các kiến trúc triển khai data pipeline hiện đại 7
1.3.3. Data Lake 11
1.3.4. Data Warehouse 11
1.3.5. ETL (Extract, Transform, Load) 14

CHƯƠNG II: PHÂN TÍCH VÀ XÂY DỰNG HỆ THỐNG 19
2.1. Phân tích yêu cầu bài toán 19
2.1.1. Phân tích nghiệp vụ cho công tác CVHT 19
2.1.2. Phân tích các use case chính của hệ thống 24
2.2. Thiết kế và xây dựng hệ thống 28
2.2.1. Phân tích và thiết kế hệ thống..... 28
2.2.2. Công nghệ sử dụng 34
CHƯƠNG III: THỰC NGHIỆM VÀ ĐÁNH GIÁ HỆ THỐNG..... 39
3.1. Môi trường triển khai..... 39
3.2. Xây dựng các mô-đun chức năng 39
3.2.1. Mô-đun theo dõi học tập..... 39
3.2.2. Mô-đun theo dõi rèn luyện 41
3.2.3. Mô-đun cảnh báo và gửi email tự động..... 43
3.2.4. Mô-đun báo cáo thống kê 45
3.3. Đánh giá hệ thống..... 47
3.3.1. Kịch bản thử nghiệm hệ thống..... 47
3.3.2. Đánh giá hiệu quả hỗ trợ CVHT..... 49
KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN..... 51
TÀI LIỆU THAM KHẢO 54

DANH MỤC HÌNH VẼ

Hình 1. Mô hình dữ liệu 3 lớp 9
Hình 2. Tổng quan kiến trúc hệ thống 28
Hình 3. Thiết kế pipeline chi tiết 30
Hình 4. Dữ liệu điểm danh học tập sinh viên 40
Hình 5. Dữ liệu điểm thành phần..... 40
Hình 6. Lập lịch xử lý dữ liệu..... 41
Hình 8. Dữ liệu điểm rèn luyện thu thập 42
Hình 9. Kho dữ liệu lưu trữ dữ liệu đã được xử lý 43
Hình 10. Xử lý dữ liệu và gửi email 44
Hình 11. Email cảnh báo sinh viên 44
Hình 12. Hồ dữ liệu lưu trữ dữ liệu gốc 45
Hình 13. Dữ liệu lưu trữ trong hồ dữ liệu dưới dạng parquet 46
Hình 14. Dashboard điểm rèn luyện sinh viên 47
Hình 15.Dashboard điểm học tập sinh viên..... 47

DANH MỤC BẢNG BIỂU

Bảng 1. Cấu hình máy chủ thử nghiệm 39

DANH MỤC TỪ VIẾT TẮT

Từ viết tắt	Từ gốc	Nghĩa Tiếng Việt
ACID	Atomicity, Consistency, Isolation, Durability	Tính chất ACID trong giao dịch dữ liệu
ACM	Association for Computing Machinery	Hiệp hội Máy tính Hoa Kỳ
AI	Artificial Intelligence	Trí tuệ nhân tạo
BI	Business Intelligence	Hệ thống phân tích & khai thác dữ liệu doanh nghiệp
CAP	Consistency, Availability, Partition Tolerance	Định lý CAP trong hệ phân tán
CDC	Change Data Capture	Thu thập thay đổi dữ liệu
CLI	Command Line Interface	Giao diện dòng lệnh
COPY	COPY Command (SQL)	Lệnh tải dữ liệu hàng loạt (COPY)
CPU	Central Processing Unit	Bộ xử lý trung tâm
CSV	Comma-Separated Values	Tệp dữ liệu phân tách bằng dấu phẩy
DAG	Directed Acyclic Graph	Đồ thị có hướng không chu trình
ELT	Extract, Load, Transform	Trích xuất – Nạp – Chuyển đổi
ETL	Extract, Transform, Load	Trích xuất – Chuyển đổi – Nạp
GPA	Grade Point Average	Điểm trung bình học tập
HDFS	Hadoop Distributed File System	Hệ thống tệp phân tán Hadoop
HTTP	Hypertext Transfer Protocol	Giao thức truyền tải siêu văn bản

INSERT	INSERT Statement (SQL)	Lệnh chèn dữ liệu trong SQL
KPI	Key Performance Indicators	Chỉ số đánh giá hiệu suất
LMS	Learning Management System	Hệ thống quản lý học tập
MERGE	MERGE Statement (SQL)	Lệnh MERGE (kết hợp INSERT/UPDATE)
MinIO	MinIO Object Storage	Hệ thống lưu trữ đối tượng tương thích S3
OLAP	Online Analytical Processing	Xử lý phân tích trực tuyến
SCD	Slowly Changing Dimensions	Dimension thay đổi chậm
SQL	Structured Query Language	Ngôn ngữ truy vấn có cấu trúc
UPDATE	UPDATE Statement (SQL)	Lệnh cập nhật dữ liệu trong SQL
VLDB	Very Large DataBase	Cơ sở dữ liệu cực lớn

MỞ ĐẦU

1. Bối cảnh đề tài

Hiện nay, với tác động của cuộc "Cách mạng công nghiệp 4.0" đang đặt ra yêu cầu buộc giáo dục đại học phải thực hiện những cải tiến, cập nhật nhanh chóng và phù hợp. Theo đó, khái niệm "chuyển đổi số" trong giáo dục cần được hiểu là không chỉ bao gồm ở việc áp dụng E-learning hay số hóa giáo trình, mà là quá trình tích hợp tư duy hướng dữ liệu (data-driven) vào mọi khâu từ quản trị, đào tạo đến hỗ trợ sinh viên [1].

Mặc dù các cơ sở giáo dục đã đưa vào vận hành nhiều nền tảng "quản lý học tập (LMS)" và phần mềm hỗ trợ trực tuyến, nhưng thực tế cho thấy dòng chảy dữ liệu vẫn chưa được khơi thông. Thông tin hiện nay chủ yếu vẫn nằm phân tán tại các đơn vị phòng ban riêng biệt hoặc được lưu trữ thủ công trên các bảng tính Excel thay vì một kho dữ liệu tập trung [2]. Sự phân mảnh dữ liệu này tạo ra rào cản, đồng thời ảnh hưởng "tiêu cực" đến hoạt động cố vấn học tập (CVHT) - công tác vốn yêu cầu nguồn thông tin phải có tính thống nhất cao và được cập nhật liên tục.

Do đó, việc triển khai các giải pháp "phân tích dữ liệu"với quy trình xử lý tự động (data pipeline) trong công tác CVHT đang được xem là xu thế tất yếu. Cách tiếp cận này có"ý nghĩa quan trọng"trong việc cải thiện việc giám sát tiến độ, nâng cao hiệu quả hỗ trợ học vụ, và cố vấn [3].

2. Thực trạng

Hiện nay, thông tin chi tiết về sinh viên, bao gồm dữ liệu học tập, điểm rèn luyện đại học còn tồn tại các vấn đề chính:

Việc thiếu một kho dữ liệu tập trung để lưu trữ làm giảm khả năng khai thác để hỗ trợ sinh viên [2]. Dữ liệu phân tán, rời rạc, thiếu đồng bộ: Nhiều dữ liệu lưu trữ ở các dạng, các nơi không đồng nhất với nhau như Excel, các hệ

thống nội bộ, hoặc các báo cáo thủ công, khiến việc tổng hợp và phân tích gặp khó khăn.

Công tác CVHT còn thủ công và hạn chế: Phần lớn CVHT phải tự tổng hợp bằng tay từ nhiều nguồn, nhiều nơi, tăng khả năng sai sót do lỗi con người và gây mất thời gian. Điều này dẫn đến việc hỗ trợ sinh viên chưa kịp thời, khó phát hiện sớm sinh viên có nguy cơ học vụ hoặc rèn luyện kém [4].

Dù dữ liệu thô có thể có sẵn, song nhiều cơ sở giáo dục vẫn chưa trang bị các hệ thống phân tích học tập (learning analytics) đủ chiều sâu. Sự thiếu hụt công cụ này làm "giảm đáng kể khả năng phân tích" và sử dụng dữ liệu cho các bài toán dự báo và nhận diện "xu hướng hành vi" của sinh viên[5].

Mặc dù tầm quan trọng của "quy trình xử lý dữ liệu" (pipeline) và phân tích học tập trong việc tối ưu hóa quản lý đào tạo đã được một số các nghiên cứu có khẳng định [3], [6], nhưng mức độ triển khai thực tế tại Việt Nam vẫn còn khá hạn chế.

Những bất cập nêu trên phản ánh nhu cầu cấp thiết về một nền tảng dữ liệu có tính tích hợp cao, vận hành tự động và trực quan hóa để hỗ trợ đắc lực cho đội ngũ CVHT.

3. Tính cấp thiết của đề tài

Việc "xây dựng và phát triển" hệ thống Data Pipeline giúp khép kín quy trình từ xử lý, phân tích đến biểu diễn biểu đồ trực quan dữ liệu. Đây là "bước đi thiết thực" nhằm hiện thực hóa mô hình "đại học số" và tự động hóa quản trị mà các trường đại học đang hướng tới.

Thay thế các phương pháp thủ công bằng kiến trúc tích hợp Data Lake - ETL - Data Warehouse giúp quy tụ dữ liệu về một mối. Giải pháp này không chỉ chuẩn hóa thông tin để "đảm bảo sự chính xác", mà còn giúp cải thiện thời gian xử lý và triệt tiêu các sai sót do thao tác con người [2], [6].

Thông qua các Dashboard trực quan, giảng viên được trang bị cái "nhìn tổng quan" và kịp thời về tình trạng sinh viên (GPA, tiến độ tín chỉ, điểm rèn luyện...). Việc tiếp cận nhanh chóng với các cảnh báo học vụ này là cơ sở để cải thiện chất lượng cố vấn và đưa ra các can thiệp sớm [4].

Một cơ sở dữ liệu đầy đủ là tiền đề bắt buộc, là nền tảng cho các mô hình Học máy (Machine Learning) phục vụ dự báo rủi ro, phân nhóm đối tượng hoặc cá nhân hóa lộ trình đào tạo trong tương lai [3], [6].

Từ những phân tích trên, đề tài có tính cấp thiết cao và giá trị thực tiễn sâu sắc trong giáo dục đại học.

4. Phạm vi và đối tượng nghiên cứu

"Đối tượng nghiên cứu trọng tâm"của đề án hướng vào các luồng "dữ liệu liên quan đến kết quả về học tập" và thông tin về "điểm rèn luyện" của sinh viên. Nghiên cứu cũng khảo sát các quy trình kỹ thuật về "thu thập", "lưu trữ", "phân tích", "trực quan" và các kiến trúc đường ống dữ liệu (data pipeline) trong môi trường giáo dục đại học.

Về mặt kỹ thuật: Đề tài giới hạn trong việc thiết kế và cài đặt thử nghiệm một kiến trúc dữ liệu hoàn chỉnh bao gồm các phân hệ: Hồ dữ liệu, quy trình ETL, Kho dữ liệu và Bảng chỉ số (Dashboard).

Về mặt dữ liệu: "Quá trình đánh giá" hệ thống sẽ thử nghiệm với bộ dữ liệu giả lập (dummy data) hoặc dữ liệu đã được phi định danh nhằm "đảm bảo tuyệt đối" các yêu cầu về "bảo mật thông tin".

Giới hạn đề tài: Việc kết nối với hệ thống đào tạo hiện hành (Live Integration) nằm ngoài phạm vi và được đề xuất là hướng mở rộng sau này.

Đề tài này được tổ chức thành ba chương chính. Chương I trình bày tổng quan về hệ thống hỗ trợ cố vấn học tập. Chương II đi sâu vào các nội dung về phân tích và xây dựng hệ thống. Chương III trình bày quá trình thực nghiệm và kết quả đáng giá. Thông qua việc triển khai thực tế này, đề tài không chỉ cung cấp một giải pháp công nghệ cụ thể mà còn đóng góp vào quá trình chuyển đổi số trong giáo dục đại học, giúp tối ưu hóa công tác quản lý và hỗ trợ sinh viên.

CHƯƠNG I: TỔNG QUAN VỀ HỆ THỐNG HỖ TRỢ CỐ VẤN HỌC TẬP

1.1. Tổng quan về công việc cố vấn học tập

Cố vấn học tập (CVHT) là một thành tố quan trọng trong mô hình quản lý đào tạo tại các cơ sở giáo dục đại học, đặc biệt trong bối cảnh đào tạo theo học chế tín chỉ. CVHT giữ vai trò là người định hướng, hỗ trợ và đồng hành cùng sinh viên trong suốt quá trình học tập, rèn luyện và phát triển cá nhân. Thông qua hoạt động cố vấn học tập, nhà trường có thể kịp thời nắm bắt tình hình học tập của sinh viên, từ đó đưa ra các biện pháp hỗ trợ phù hợp nhằm nâng cao chất lượng đào tạo và giảm thiểu tỷ lệ sinh viên học tập sa sút, kéo dài thời gian đào tạo hoặc bỏ học.

Về mặt chức năng, công tác cố vấn học tập bao gồm nhiều nhiệm vụ khác nhau, trong đó trọng tâm là theo dõi và đánh giá kết quả học tập của sinh viên qua từng học kỳ. CVHT có trách nhiệm giám sát số lượng tín chỉ tích lũy, điểm trung bình học tập, tình trạng đăng ký học phần, cũng như việc tuân thủ các quy chế đào tạo của sinh viên. Bên cạnh đó, CVHT còn tham gia đánh giá kết quả rèn luyện, mức độ tham gia các hoạt động học thuật, phong trào và công tác xã hội của sinh viên, qua đó góp phần đánh giá toàn diện quá trình phát triển của người học.

Ngoài nhiệm vụ theo dõi, CVHT còn đóng vai trò tư vấn và hỗ trợ ra quyết định cho sinh viên. Điều này thể hiện qua việc hướng dẫn sinh viên xây dựng kế hoạch học tập cá nhân phù hợp với năng lực và mục tiêu nghề nghiệp; tư vấn lựa chọn học phần, chuyên ngành, định hướng nghiên cứu hoặc thực tập; cũng như hỗ trợ sinh viên giải quyết các khó khăn phát sinh trong quá trình học tập. Đối với các trường hợp sinh viên có dấu hiệu học tập yếu kém hoặc vi phạm quy chế, CVHT là người trực tiếp trao đổi, cảnh báo và phối hợp với các đơn vị chức năng để đưa ra giải pháp can thiệp kịp thời.

Trong thực tế triển khai, công tác cố vấn học tập phải xử lý một khối lượng lớn thông tin liên quan đến sinh viên, bao gồm dữ liệu học tập, rèn luyện, thông tin cá nhân và các dữ liệu hành chính khác. Các dữ liệu này thường được lưu trữ phân tán tại nhiều hệ thống khác nhau, dẫn đến khó khăn trong việc tổng hợp, theo dõi và phân tích một cách toàn diện. Khi số lượng sinh viên phụ trách ngày càng tăng, việc theo dõi thủ công hoặc dựa trên các báo cáo rời rạc không còn đáp ứng được yêu cầu về tính kịp thời và chính xác. Do đó, việc xây dựng các hệ thống hỗ trợ công tác cố vấn học tập dựa trên nền tảng dữ liệu tập trung là xu hướng tất yếu, góp phần nâng cao hiệu quả hoạt động của CVHT và chất lượng đào tạo của nhà trường.

1.2. Tổng quan về trực quan hóa dữ liệu

Trực quan hóa dữ liệu (Data Visualization) là một lĩnh vực quan trọng trong khoa học dữ liệu, tập trung vào việc biểu diễn dữ liệu dưới dạng hình ảnh nhằm hỗ trợ con người trong quá trình phân tích và ra quyết định. Thông qua các biểu đồ, bảng tổng hợp, đồ thị và dashboard, dữ liệu được chuyển đổi từ dạng số liệu thô sang dạng trực quan, giúp người dùng dễ dàng nhận biết xu hướng, mối quan hệ và các điểm bất thường tiềm ẩn trong tập dữ liệu.

Trong bối cảnh các hệ thống thông tin ngày càng tạo ra khối lượng dữ liệu lớn, trực quan hóa dữ liệu đóng vai trò cầu nối giữa dữ liệu và người sử dụng cuối. Một hệ thống trực quan hóa hiệu quả không chỉ trình bày dữ liệu một cách rõ ràng mà còn hỗ trợ tương tác, cho phép người dùng lọc, so sánh và phân tích dữ liệu theo nhiều góc nhìn khác nhau. Điều này đặc biệt quan trọng đối với các đối tượng không chuyên sâu về kỹ thuật dữ liệu, như cán bộ quản lý hoặc cố vấn học tập.

Đối với lĩnh vực giáo dục đại học, trực quan hóa dữ liệu được ứng dụng rộng rãi trong công tác quản lý đào tạo, đánh giá chất lượng học tập và hỗ trợ sinh viên. Các dashboard thống kê cho phép tổng hợp nhanh chóng thông tin

về kết quả học tập, tiến độ đào tạo, tỷ lệ cảnh báo học vụ, phân bố điểm số và nhiều chỉ tiêu khác. Nhờ đó, người sử dụng có thể nắm bắt tình hình chung cũng như chi tiết của từng nhóm sinh viên hoặc cá nhân sinh viên trong thời gian ngắn.

1.3. Tổng quan về công nghệ dữ liệu

1.3.1. Data Pipeline

Data Pipeline là "một kiến trúc hệ thống được thiết kế để tự động hóa quá trình di chuyển và chuyển đổi dữ liệu từ nhiều nguồn khác nhau" đến các hệ thống đích phục vụ mục đích phân tích, lưu trữ hoặc xử lý [7]. Trong bối cảnh bùng nổ dữ liệu hiện nay, Data Pipeline đóng vai trò như xương sống của các hệ thống phân tích dữ liệu, cho phép các tổ chức "tận dụng được giá trị từ dữ liệu có sẵn một cách hiệu quả" [8].

Về bản chất, Data Pipeline là "một chuỗi các bước xử lý dữ liệu được tự động hóa, trong đó đầu ra của bước này trở thành đầu vào của bước tiếp theo" [9]. Mỗi bước trong pipeline có một chức năng riêng như trích xuất từ nguồn, làm sạch, chuyển đổi định dạng, tính toán các chỉ số phái sinh, hoặc tải dữ liệu vào hệ thống lưu trữ. Một Data Pipeline hoàn chỉnh cần đảm bảo các đặc tính quan trọng bao gồm: tính nhất quán của dữ liệu, khả năng xử lý lỗi và phục hồi, mở rộng theo quy mô dữ liệu, và giám sát toàn diện các hoạt động xử lý [10].

Kiến trúc tổng thể của một Data Pipeline thông thường được cấu thành từ năm giai đoạn cơ bản [11]. Giai đoạn đầu tiên là thu thập dữ liệu (Data Ingestion), trong đó dữ liệu được tiếp nhận từ nhiều nguồn: cơ sở dữ liệu quan hệ, các tệp dữ liệu (CSV, Excel), dịch vụ API, cũng như các luồng dữ liệu phát sinh theo thời gian thực.

Tiếp theo là giai đoạn lưu trữ dữ liệu (Data Storage), nơi dữ liệu thô sau khi thu thập được tập trung và lưu giữ trong Data Lake nhằm đảm bảo tính đầy đủ, toàn vẹn và khả năng sẵn sàng cho các bước xử lý tiếp theo. Ở giai đoạn xử

lý dữ liệu (Data Processing), dữ liệu được làm sạch, chuyển đổi và bổ sung thông tin thông qua các quy trình ETL (Extract, Transform, Load) hoặc ELT (Extract, Load, Transform) [12], qua đó nâng cao chất lượng và giá trị sử dụng của dữ liệu.

Sau khi được xử lý, dữ liệu sẽ được tổ chức và lưu trữ trong kho dữ liệu (Data Warehousing). Tại đây, dữ liệu được xây dựng theo các mô hình phù hợp nhằm tối ưu cho hoạt động truy vấn và phân tích. Cuối cùng, giai đoạn trực quan hóa dữ liệu (Data Visualization) cho phép người dùng cuối khai thác và tương tác với dữ liệu thông qua các dashboard, báo cáo và các công cụ Business Intelligence, hỗ trợ hiệu quả cho công tác phân tích và ra quyết định [13].

Có hai phương pháp xử lý dữ liệu chính trong Data Pipeline là batch processing và stream processing [14]. Batch processing xử lý theo từng lô dữ liệu định kỳ (hàng giờ, hàng ngày), hướng tới với các tác vụ không yêu bắt buộc thời gian thực và có khối lượng dữ liệu lớn. Ngược lại, với stream processing, dữ liệu được xử lý ngay khi chúng được tạo ra và có tính liên tục, hướng tới các ứng dụng yêu cầu với độ trễ thấp như giám sát thời gian thực, phát hiện bất thường, hoặc hệ thống cảnh báo [15]. Trong bối cảnh của hệ thống phục vụ CVHT, phương pháp batch processing thường được ưa chuộng do dữ liệu học tập được cập nhật định kì, không đòi hỏi xử lý tức thời [16].

Data Pipeline đầu tiên giúp tự động hóa các quy trình, giảm công việc thủ công và giảm rủi ro do con người gây ra [17]. Thứ hai, với dữ liệu lớn, pipeline cho phép xử lý một cách hiệu quả bằng cách song song hóa các tác vụ và tối ưu hóa hệ thống [18]. Thứ ba, việc tiêu chuẩn hóa quy trình xử lý dữ liệu trong pipeline "đảm bảo tính nhất quán và chất lượng của dữ liệu đầu ra" [19]. Thứ tư, các công cụ điều phối workflow hiện đại như Apache Airflow hay Prefect cung cấp khả năng giám sát, logging chi tiết và xử lý lỗi tự động [20]. Cuối cùng, kiến trúc pipeline có tính mô-đun cao cho phép dễ dàng mở rộng,

bổ sung nguồn dữ liệu mới hoặc điều chỉnh logic xử lý một cách độc lập, tránh ảnh hưởng toàn bộ hệ thống [21].

Tuy nhiên, triển khai Data Pipeline cũng có nhiều thách thức như là đảm bảo chất lượng và chuẩn hóa dữ liệu khi tích hợp từ nhiều nguồn với các định dạng và tiêu chuẩn khác nhau [22]. Vấn đề về data lineage - khả năng truy vết nguồn gốc và quá trình biến đổi của dữ liệu - cũng là một yêu cầu quan trọng nhưng khó thực hiện [23]. Ngoài ra, việc xử lý lỗi và phục hồi khi có sự cố trong pipeline đòi hỏi cơ chế retry thông minh và chiến lược backup phù hợp. Cuối cùng, khi quy mô dữ liệu tăng lên, việc cải thiện hiệu năng và tối ưu chi phí vận hành đòi hỏi kiến thức chuyên sâu về kiến trúc hệ thống phân tán [24].

Trong lĩnh vực giáo dục, Data Pipeline được ứng dụng phổ biến để hỗ trợ các hoạt động quản lý và ra quyết định. Nghiên cứu của Daniel [25] cho thấy các trường đại học tại Mỹ đã triển khai Data Pipeline để tích hợp dữ liệu từ hệ thống quản lý học tập, hệ thống thông tin sinh viên, và các nền tảng học tập trực tuyến nhằm tạo ra thông tin toàn diện về hành trình học tập. Việc sử dụng pipeline phân tích dữ liệu giáo dục (Educational Data Mining) đã giúp các nhà giáo dục nắm bắt được sớm tình trạng các sinh viên học yếu, từ đó kịp thời có các biện pháp hỗ trợ để nâng cao tỷ lệ tốt nghiệp.

Trong bối cảnh của đề tài này, Data Pipeline được thiết kế để xử lý dữ liệu học tập và rèn luyện của sinh viên hỗ trợ các công tác của CVHT. Pipeline sẽ quy trình hóa tự động từ thu thập dữ liệu Excel từ các phòng ban, Data Lake tổng hợp dữ liệu thô, xử lý và chuyển đổi dữ liệu thông qua các bước ETL, dữ liệu đã xử lý được tải lên Data Warehouse, và cuối cùng trình bày thông tin qua dashboard trực quan. Kiến trúc này giúp giảm thiểu thời gian và công sức xử lý dữ liệu thủ công và cải thiện chất lượng thông tin hỗ trợ cho CVHT.

1.3.2. Các kiến trúc triển khai data pipeline hiện đại

Trong phần này, ba kiến trúc Data Pipeline phổ biến và hiện đại nhất sẽ được phân tích chi tiết bao gồm Lambda Architecture, Kappa Architecture và Medallion Architecture.

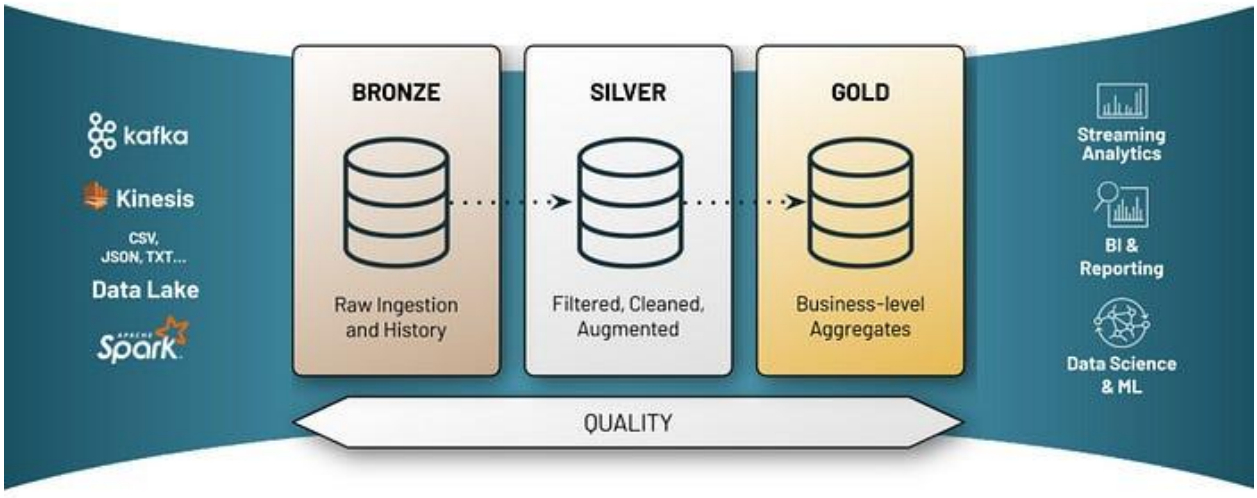
Lambda Architecture được Nathan Marz giới thiệu lần đầu năm 2011 [26] với ba tầng chính: "Batch Layer, Speed Layer và Serving Layer". Batch Layer lưu trữ toàn bộ dữ liệu thô và thực hiện các tính toán batch trên toàn bộ tập dữ liệu để tạo ra batch views - các kết quả được tính toán trước với độ chính xác cao. Dữ liệu trong tầng này được lưu trữ trong các hệ thống phân tán như HDFS (Hadoop Distributed File System) và xử lý bằng các framework như Apache Hadoop hoặc Apache Spark. Speed Layer xử lý dữ liệu thời gian thực để bù đắp độ trễ của Batch Layer, tạo ra real-time views từ dữ liệu mới nhất. Tầng này sử dụng các công nghệ xử lý thời gian thực như Apache Storm, Apache Flink hoặc Apache Kafka Streams để đạt độ trễ thấp. Cuối cùng, Serving Layer hợp nhất kết quả từ cả batch views và real-time views để phục vụ các truy vấn của người dùng cuối.

Ưu điểm lớn nhất của Lambda Architecture là khả năng cân bằng giữa độ chính xác và độ trễ thấp. Batch Layer đảm bảo tính toàn vẹn và chính xác của dữ liệu thông qua việc xử lý lại toàn bộ dataset, trong khi Speed Layer cung cấp kết quả gần thời gian thực. Kiến trúc này cũng có khả năng chịu lỗi tốt vì dữ liệu thô được lưu trữ bất biến trong Batch Layer, cho phép tính toán lại khi cần thiết. Tuy nhiên, Lambda Architecture cũng đối mặt với nhiều hạn chế đáng kể. Việc duy trì hai xử lý riêng biệt cho xử lý theo lô và xử lý thời gian thực làm tăng độ phức tạp trong phát triển và bảo trì hệ thống. Ngoài ra, việc hợp nhất kết quả từ hai tầng xử lý khác nhau đòi hỏi logic phức tạp và có thể dẫn đến sự không nhất quán tạm thời trong dữ liệu.

Kappa Architecture được Jay Kreps đề xuất năm 2014 như một giải pháp đơn giản hóa Lambda Architecture bằng cách loại bỏ hoàn toàn Batch Layer [27]. Kiến trúc này dựa trên nguyên tắc rằng tất cả dữ liệu đều có thể được coi là các event trong một stream vô hạn, và mọi xử lý đều được thực hiện thông qua stream processing. Kappa Architecture chỉ bao gồm hai tầng chính: Stream Processing Layer và Serving Layer. Stream Processing Layer xử lý tất cả dữ liệu đến dưới dạng stream, sử dụng các công nghệ như Apache Kafka làm nơi lưu trữ log sự kiện để lưu trữ dữ liệu thô và Apache Flink hoặc Apache Samza để xử lý stream. Khi cần tính toán lại dữ liệu lịch sử, hệ thống đơn giản là tua ngược lại quá trình và xử lý các sự kiện từ đầu thay vì phải xây dựng một pipeline riêng biệt.

Kappa Architecture mang lại nhiều lợi ích về mặt đơn giản hóa kiến trúc và giảm chi phí bảo trì. Việc chỉ có một code base duy nhất cho xủ lý thời gian thực giúp giảm đáng kể độ phức tạp trong phát triển và đảm bảo tính nhất quán của logic xử lý. Hơn nữa, kiến trúc này tận dụng tốt các công nghệ xử lý stream hiện đại có khả năng xử lý băng thông cao và độ trễ thấp. Tuy nhiên, không phải mọi loại tính toán đều phù hợp với mô hình xử lý stream, đặc biệt là các phép toán phức tạp yêu cầu truy cập ngẫu nhiên vào toàn bộ tập dữ liệu.

Medallion Architecture, còn được gọi là Multi-Hop Architecture hoặc Bronze-Silver-Gold Architecture, là một kiến trúc tương đối mới được phổ biến rộng rãi bởi Databricks và các nền tảng Data Lakehouse [28]. Kiến trúc này tổ chức dữ liệu thành các tầng với mức độ tinh chế và chất lượng tăng dần, tương tự như quá trình lọc và tinh chế kim loại quý. Medallion Architecture bao gồm ba tầng chính: Bronze Layer (tầng đồng), Silver Layer (tầng bạc) và Gold Layer (tầng vàng).



Hình 1. Mô hình dữ liệu 3 lớp

Bronze Layer là tầng đầu tiên trong kiến trúc, lưu trữ dữ liệu thô nguyên bản từ các nguồn khác nhau mà không thực hiện bất kỳ chuyển đổi nào. Dữ liệu trong tầng này được lưu trữ theo định dạng gần giống với nguồn gốc, bao gồm cả metadata như timestamp, source system, và các thông tin khác. Bronze Layer đóng vai trò như một vùng tiếp nhận đảm bảo tính đầy đủ của dữ liệu thu thập và có thể truy xuất lại khi cần thiết.

Silver Layer là tầng trung gian nơi dữ liệu được làm sạch, chuẩn hóa và làm giàu. Các hoạt động thực hiện ở tầng này bao gồm: loại bỏ dữ liệu trùng lặp, xử lý dữ liệu trống, chuẩn hóa định dạng (ví dụ: định dạng ngày tháng, số điện thoại), chuẩn hóa dữ liệu theo quy tắc nghiệp vụ, và kết hợp với các dimension tables để bổ sung thông tin. Dữ liệu trong Silver Layer thường được tổ chức theo các lĩnh vực hoặc đối tượng cụ thể, tạo thành một nguồn dữ liệu tin cậy cho từng loại. Silver Layer cung cấp dữ liệu có chất lượng cao đủ để phục vụ cho các phân tích không yêu cầu tính toán phức tạp.

Gold Layer là tầng cuối cùng, chứa dữ liệu đã được tối ưu hóa cho các yêu cầu nghiệp vụ cụ thể. Dữ liệu trong tầng này thường được tính toán, xử lý và tổ chức với các mô hình theo nhu cầu báo cáo và phân tích như star schema hoặc snowflake schema. Gold Layer có thể chứa các thông tin đã được tính

toán trước cho học máy, các KPI (Key Performance Indicators) cho dashboard, hoặc các fact tables được tối ưu cho việc truy vấn. Việc tách biệt Gold Layer cho phép tối ưu hóa riêng cho từng yêu cầu mà không ảnh hưởng đến các tầng trước đó.

Medallion Architecture có nhiều ưu điểm so với các kiến trúc truyền thống. Đầu tiên, kiến trúc này tách biệt một cách rõ ràng các giai đoạn xử lý dữ liệu, giúp dễ dàng gỡ lỗi và bảo trì. Thứ hai, việc lưu trữ dữ liệu ở nhiều mức độ tinh chế khác nhau cho phép phục vụ đa dạng nhu cầu từ nghiên cứu trong khoa học dữ liệu đến báo cáo nghiệp vụ. Thứ ba, kiến trúc này hỗ trợ tốt cho cả xử lý batch và stream, phù hợp với xu hướng Data Lakehouse hiện đại. Cuối cùng, Medallion Architecture dễ hiểu và triển khai, không đòi hỏi kiến thức chuyên sâu như Lambda hay Kappa Architecture.

Tuy nhiên, Medallion Architecture cũng có một số hạn chế cần lưu ý. Việc lưu trữ dữ liệu ở nhiều tầng có thể dẫn đến tăng chi phí lưu trữ và độ phức tạp trong quản lý. Ngoài ra, việc xác định ranh giới giữa các tầng và quyết định logic nào thuộc về tầng nào đôi khi không rõ ràng và phụ thuộc vào kinh nghiệm. Hơn nữa, với các tình huống yêu cầu độ trễ thấp, việc phải đi qua nhiều tầng xử lý có thể không phù hợp.

So sánh ba kiến trúc trên, Lambda Architecture phù hợp với các hệ thống yêu cầu cả xử lý batch và real-time với độ chính xác cao, nhưng đổi lại là độ phức tạp trong khi triển khai. Kappa Architecture đơn giản hơn và phù hợp khi phần lớn use case có thể được giải quyết bằng xử lý stream, tuy nhiên việc xử lý lại dữ liệu lịch sử có thể tốn kém. Medallion Architecture cân bằng giữa đơn giản và linh hoạt, đặc biệt phù hợp với các tổ chức đang xây dựng Data Lakehouse và cần phục vụ đa dạng từ phần tích đến học máy.

Trong nghiên cứu này, Medallion Architecture được lựa chọn làm kiến trúc chính vì những lý do sau. Thứ nhất, dữ liệu học tập và rèn luyện không yêu

cầu xử lý thời gian thực mà chủ yếu là xử lý batch theo kỳ học, do đó không cần đến độ phức tạp của Lambda hay Kappa Architecture. Thứ hai, việc tổ chức dữ liệu thành các tầng Bronze-Silver-Gold giúp dễ dàng quản lý quá trình chuyển đổi dữ liệu từ file Excel thô đến các báo cáo phân tích cuối cùng. Thứ ba, kiến trúc này linh hoạt trong việc thêm các nguồn dữ liệu mới hoặc thay đổi logic xử lý.

1.3.3. Data Lake

Data Lake được Dixon lần đầu tiên đề xuất năm 2010 như "một giải pháp lưu trữ dữ liệu thô ở định dạng nguyên bản", không cần "định nghĩa schema trước" [29]. Data Lake là một "kho lưu trữ tập trung cho phép lưu giữ tất cả dữ liệu có cấu trúc (structured), bán cấu trúc (semi-structured) và phi cấu trúc (unstructured)" ở bất kỳ quy mô nào. Khác với Data Warehouse yêu cầu định nghĩa schema trước khi nạp dữ liệu (schema-on-write), Data Lake áp dụng nguyên tắc schema-on-read, nghĩa là schema chỉ được áp dụng khi dữ liệu được đọc ra để phân tích. Các công nghệ phổ biến để triển khai "Data Lake" bao gồm "Hadoop Distributed File System (HDFS)", "Amazon S3", "Microsoft Azure Data Lake Storage", và "Google Cloud Storage". Một Data Lake hoàn chỉnh không chỉ là một kho lưu trữ đơn thuần mà còn bao gồm các thành phần như "công cụ thu thập dữ liệu để thu thập dữ liệu từ nhiều nguồn", quản lý metadata để theo dõi và tổ chức dữ liệu, data catalog để giúp người dùng "tìm kiếm và khám phá" dữ liệu, và bảo mật, phân quyền dữ liệu.

1.3.4. Data Warehouse

Data Warehouse được định nghĩa như một kho dữ liệu với 4 đặc tính: "hướng chủ đề (subject-oriented); tích hợp (integrated); bất biến theo thời gian (time-variant); không thay đổi (non-volatile)". Data Warehouse được sử dụng cho việc lưu trữ các dữ liệu đã được "làm sạch và chuẩn hóa" theo các schema có cấu trúc rõ ràng, để phục vụ tốt nhất cho các truy vấn phân tích phức tạp.

Quá trình xây dựng Data Warehouse bắt đầu với việc xác định các quy trình nghiệp vụ cần hỗ trợ, sau đó thiết kế các dimension và fact tables phù hợp. Dimension tables chứa "thông tin mô tả về các đối tượng", trong khi fact tables chứa các số liệu đo lường và các khóa ngoại liên kết đến dimension tables. Kiến trúc star schema, nơi fact table ở trung tâm và trực tiếp kết nối với các dimension tables. Snowflake schema, một biến thể của star schema với các dimension được biến đổi thành nhiều bảng con, giúp tiết kiệm bộ nhớ nhưng đổi lại là độ phức tạp cao hơn trong truy vấn. Các công nghệ Data Warehouse hiện đại bao gồm các hệ thống on-premise như "Oracle Exadata", "IBM Db2 Warehouse", "Microsoft SQL Server", và các giải pháp cloud như "Amazon Redshift", "Google BigQuery", "Snowflake", và "Azure Synapse Analytics". Các hệ thống này được thiết kế với kiến trúc columnar storage để đạt hiệu năng cao trong việc xử lý các truy vấn phân tích trên khối lượng dữ liệu lớn. "Columnar storage" đặc biệt "hiệu quả cho phân tích" vì nó cho phép đọc chỉ các cột cần thiết, áp dụng các điều kiện tốt hơn, và tận dụng hiệu quả CPU cache.

Về mặt dữ liệu, Data Lake lưu trữ dữ liệu ở định dạng gốc trong khi Data Warehouse chứa dữ liệu đã được xử lý với schema được định nghĩa rõ ràng. Về schema, Data Lake áp dụng schema-on-read cho phép linh hoạt trong việc diễn giải dữ liệu, trong khi Data Warehouse yêu cầu schema-on-write với cấu trúc cố định trước khi load dữ liệu. Về mục đích sử dụng, Data Lake phù hợp cho khai phá dữ liệu, khoa học dữ liệu và học máy, trong khi Data Warehouse tối ưu cho báo cáo nghiệp vụ, và phân tích chuẩn. Về hiệu năng, Data Warehouse thường nhanh hơn cho các truy vấn được định nghĩa trước, nhưng Data Lake linh hoạt hơn cho các phân tích ad-hoc. Về chi phí, Data Lake có chi phí lưu trữ thấp hơn nhưng có thể tốn kém về tính toán khi truy vấn, trong khi Data

Warehouse có chi phí thiết lập cao nhưng chi phí hiệu năng có thể tính toán trước.

Xu hướng Data Lakehouse đang nổi lên như một giải pháp kết hợp ưu điểm của cả Data Lake và Data Warehous. Khái niệm Lakehouse như một kiến trúc mới cho phép quản lý dữ liệu raw giống Data Lake, thêm vào đó cung cấp các giao dịch ACID và "hiệu năng truy vấn như Data Warehouse". Các nền tảng như "Delta Lake", "Apache Iceberg", "Apache Hudi" hỗ trợ xây dựng "Lakehouse" bằng cách thêm lớp metadata và giao dịch lên trên hồ dữ liệu. Lakehouse architecture hứa hẹn sẽ "đơn giản hóa data architecture" bằng cách "loại bỏ sự cần thiết phải duy trì" cả Data Lake và Data Warehouse riêng biệt.

Trong bối cảnh của hệ thống pipeline phục vụ CVHT, cả Data Lake và Data Warehouse đều "đóng vai trò quan trọng" và bổ trợ cho nhau. Data Lake được sử dụng để lưu trữ các file Excel gốc từ các phòng ban như Phòng Đào tạo và Phòng Công tác Sinh viên. Việc lưu trữ dữ liệu thô này đảm bảo tính audit trail, cho phép truy xuất lại dữ liệu gốc khi cần thiết, và cung cấp khả năng xử lý lại với logic mới trong tương lai. Trong đề tài này, MinIO được lựa chọn làm công nghệ Data Lake vì nó cung cấp S3-compatible object storage có thể triển khai on-premise với chi phí thấp và dễ quản lý.

Data Warehouse trong hệ thống được thiết kế để lưu trữ dữ liệu đã được chuẩn hóa theo star schema với các dimension tables và fact table. Kiến trúc này tối ưu hóa cho các truy vấn phân tích như tính GPA trung bình theo lớp, theo kỳ, xác định sinh viên cảnh báo học vụ, và phân tích xu hướng học tập. ClickHouse được chọn làm công nghệ Data Warehouse do khả năng xử lý truy vấn nhanh với columnar storage và khả năng mở rộng tốt. Việc kết hợp Data Lake để lưu trữ dữ liệu gốc và Data Warehouse để phục vụ phân tích tạo nên một kiến trúc cân bằng giữa tính linh hoạt và hiệu năng, đáp ứng tốt yêu cầu của hệ thống CVHT.

1.3.5. ETL (Extract, Transform, Load)

"ETL (Extract, Transform, Load)" là một quy trình quan trọng trong Data Pipeline, đóng vai trò cầu nối giữa các nguồn và hệ thống lưu trữ phân tích. Thuật ngữ ETL được sử dụng đầu tiên vào năm 1970 khi các tổ chức xây dựng Data Warehouse để tích hợp dữ liệu đa nguồn [30].

Giai đoạn "Extract (Trích xuất)" là bước đầu tiên trong quy trình ETL, có nhiệm vụ thu thập dữ liệu từ "một hoặc nhiều nguồn khác nhau". Các nguồn dữ liệu có thể bao gồm cơ sở dữ liệu quan hệ (RDBMS) như "MySQL", "PostgreSQL", "Oracle", các file như CSV, Excel, JSON, XML, các API (Application Programming Interface), nguồn dữ liệu stream, hoặc dữ liệu được thu thập từ các trang web . Quá trình trích xuất phải đối mặt với nhiều "thách thức" như xử lý các schema khác nhau, "đảm bảo tính toàn vẹn của dữ liệu" trong quá trình trích xuất, và tối ưu hóa hiệu năng để không làm ảnh hưởng đến hệ thống nguồn đang hoạt động.

Có hai phương pháp trích xuất chính là trích xuất toàn bộ và trích xuất tăng cường. Trích xuất toàn bộ thực hiện trích xuất toàn bộ dữ liệu từ nguồn mỗi lần chạy ETL, phương pháp này đơn giản nhưng"tốn nhiều thời gian và tài nguyên, đặc biệt với các tập dữ liệu lớn". Trích xuất tăng cường chỉ trích xuất dữ liệu mới hoặc đã thay đổi kể từ lần trích xuất trước, thường dựa vào timestamp, change data capture (CDC), hoặc database triggers. Trích xuất tăng cường là phương pháp được ưa chuộng trong môi trường thực tế vì giúp giảm tải cho hệ thống nguồn và rút ngắn thời gian xử lý.

Trong quá trình trích xuất, việc xử lý kết nối đến nguồn dữ liệu là một yếu tố quan trọng. Các tham số kết nối như host, port, username, password, database name cần được quản lý an toàn, thường thông qua các công cụ quản lý chuyên dụng như HashiCorp Vault, AWS Secrets Manager, hoặc environment variables. Xử lý lỗi cũng cần được thiết kế cẩn thận để xử lý các

tình huống như thời gian tải quá lâu hoặc nguồn dữ liệu không khả dụng. Giám sát trong giai đoạn trích xuất giúp theo dõi số lượng bản ghi được trích xuất, thời gian thực hiện, và phát hiện các bất thường.

Giai đoạn Transform (Chuyển đổi) là phần phức tạp và quan trọng nhất của quy trình ETL, nơi dữ liệu thô được làm sạch, chuẩn hóa, và biến đổi thành định dạng phù hợp với yêu cầu của Data Warehouse. Các thao tác biến đổi có thể được phân loại thành nhiều nhóm chính: “data cleaning”, “data integration”, “data transformation”, và “data reduction”.

Data cleaning thực hiện phát hiện và sửa chữa các lỗi, không nhất quán trong dữ liệu. Các hoạt động cleaning phổ biến bao gồm: xử lý missing values thông qua việc thay thế bằng giá trị mặc định, trung bình, hoặc loại bỏ bản ghi có quá nhiều giá trị rỗng; loại bỏ bản ghi trùng lặp dựa trên khóa chính hoặc quy tắc nghiệp vụ; “chuẩn hóa định dạng dữ liệu” như định dạng ngày tháng (DD/MM/YYYY hoặc MM/DD/YYYY), số điện thoại, địa chỉ email. Vấn đề về chất lượng dữ liệu có thể chiếm đến 20-30% tổng chi phí trong các dự án kho dữ liệu, do đó việc đầu tư vào data cleaning là cực kỳ quan trọng.

Data integration là quá trình “kết hợp dữ liệu từ nhiều nguồn khác nhau” thành một view thống nhất. Các thách thức chính trong data integration bao gồm: schema matching - xác định các fields tương ứng giữa các nguồn khác nhau (ví dụ "student_id" trong hệ thống này tương ứng với "sid" trong hệ thống khác); entity resolution - xác định các records khác nhau đại diện cho cùng một entity trong thế giới thực; data fusion - quyết định giá trị nào sẽ được giữ lại khi có conflict giữa các nguồn.

Data transformation bao gồm các phép biến đổi cấu trúc và giá trị của dữ liệu. Các thao tác biến đổi phổ biến bao gồm: chuyển đổi kiểu dữ liệu như chuyển đổi chuỗi thành số nguyên, thời gian, hoặc giá trị đúng sai; tính toán như tính toán GPA từ điểm các môn học, tính tuổi từ ngày sinh, hoặc phân loại

sinh viên thành các nhóm (xuất sắc, giỏi, khá, trung) dựa trên GPA; tính toán như tính tổng, trung bình, đếm, lấy nhỏ nhất, lớn nhất theo các tiêu chí; phân chia và gộp như tách họ và tên từ fullname, hoặc kết hợp các thành phần như thành phố, tỉnh, huyện, xã thành địa chỉ; "mã hóa và giải mã" như chuyển đổi danh mục thành số tượng trưng, hoặc giải mã mã số thành mô tả.

Lookup và enrichment là quá trình bổ sung thêm thông tin vào dữ liệu bằng cách kết hợp với các bảng hoặc nguồn dữ liệu bên ngoài. Ví dụ, trong hệ thống CVHT, có thể tìm thông tin CVHT từ bảng CVHT, hoặc enrich dữu liệu khóa với địa điểm. Việc tạo khóa thay thế (surrogate key) cũng là một bước chuyển đổi quan trọng trong kho dữ liệu. Trong bước này, các khóa nghiệp vụ (business key) sẽ được thay bằng các khóa sinh tự động để đảm bảo tính duy nhất và cải thiện hiệu suất khi thực hiện các phép nối dữ liệu.

Giai đoạn Load (Tải dữ liệu) chịu trách nhiệm ghi dữ liệu đã được xử lý vào hệ thống đích, thường là"kho dữ liệu."Có ba chiến lược tải dữ liệu chính: full load, incremental load và upsert. Full load sẽ xóa toàn bộ dữ liệu cũ trong bảng đích rồi nạp lại toàn bộ dữ liệu mới. Cách làm này đơn giản và đảm bảo bảng đích luôn phản ánh chính xác dữ liệu nguồn sau khi chuyển đổi, nhưng không phù hợp với những bảng có dung lượng lớn hoặc khi cần lưu lại lịch sử. Incremental load chỉ nạp những bản ghi mới hoặc đã thay đổi, giúp rút ngắn thời gian tải và giảm ảnh hưởng lên hệ thống đích. Phương pháp này đòi hỏi có cơ chế nhận diện bản ghi thay đổi, chẳng hạn như cột timestamp, kỹ thuật thu thập dữ liệu thay đổi (CDC), hoặc so sánh hash.

Upsert (Update + Insert) là chiến lược kết hợp cả việc chèn mới và cập nhật dữ liệu: các bản ghi mới sẽ được chèn vào, còn những bản ghi đã tồn tại thì được cập nhật lại. Việc triển khai upsert có thể thực hiện bằng câu lệnh MERGE trong SQL, hoặc dùng logic kiểm tra sự tồn tại của bản ghi rồi thực hiện INSERT hoặc UPDATE tương ứng. Trong môi trường kho dữ liệu, Slowly

Changing Dimensions (SCD) là một mẫu thiết kế quan trọng để xử lý dữ liệu chiều (dimension) thay đổi theo thời gian.

- SCD Type 1 ghi đè giá trị cũ.
- SCD Type 2 tạo phiên bản mới và lưu lại lịch sử thay đổi.
- SCD Type 3 lưu đồng thời giá trị cũ và giá trị mới ở các cột riêng biệt.

Tối ưu hiệu năng trong giai đoạn load là yếu tố then chốt, đặc biệt khi xử lý khối lượng dữ liệu lớn. Các kỹ thuật phổ biến để tăng tốc độ tải gồm:

- Bulk loading bằng các lệnh COPY hoặc API bulk insert thay cho việc chèn từng bản ghi.
- Tạm thời vô hiệu hóa chỉ mục và ràng buộc, sau khi tải xong mới xây dựng lại để giảm chi phí kiểm tra trong quá trình load.
- Parallel loading, chia dữ liệu thành nhiều phần nhỏ và tải song song để tận dụng tài nguyên hệ thống.
- Batch commit, gom nhiều thao tác chèn vào một lần commit thay vì commit sau mỗi bản ghi.
- Sử dụng bảng staging, nạp dữ liệu vào bảng tạm để kiểm tra và làm sạch trước khi chuyển sang bảng chính trong môi trường vận hành.

Việc xử lý lỗi và quản lý giao dịch trong giai đoạn load cần được thiết kế cẩn thận. Các chiến lược thường dùng bao gồm:

- Tải dữ liệu theo tính nguyên tử (atomic load) bằng cách sử dụng cơ chế giao dịch của cơ sở dữ liệu, đảm bảo hoặc toàn bộ dữ liệu được nạp thành công, hoặc toàn bộ bị rollback.
- Checkpoint và khôi phục (restart) cho phép tiếp tục quá trình tải từ vị trí bị gián đoạn thay vì phải chạy lại từ đầu.
- Ghi log lỗi, lưu lại các bản ghi không nạp được để xử lý sau.

- Đối soát dữ liệu (data reconciliation) nhằm kiểm tra và xác nhận rằng số lượng bản ghi đã tải khớp với số lượng bản ghi được xử lý.

ELT (Extract, Load, Transform) là một biến thể của ETL ngày càng phổ biến, đặc biệt trong các "hệ thống kho dữ liệu" trên nền tảng đám mây. Cách làm này tận dụng ưu điểm các kho dữ liệu nổi tiếng như Snowflake, BigQuery hay Redshift, thay vì phải dựng thêm các công cụ xử lý riêng.

Ưu điểm của ELT là rút ngắn thời gian khai thác dữ liệu vì dữ liệu thô có mặt ngay trong kho; cho phép linh hoạt thử nghiệm nhiều logic chuyển đổi khác nhau; và dễ dàng mở rộng nhờ "khả năng tính toán mạnh mẽ" của hạ tầng đám mây.

Trong bối cảnh hệ thống pipeline hỗ trợ công tác CVHT, quy trình ETL được thiết kế như sau:

- Extract: Dữ liệu được trích xuất từ các tệp Excel do người dùng tải lên MinIO Data Lake, gồm thông tin sinh viên, kết quả học tập từng kỳ và điểm rèn luyện. Python được sử dụng với thư viện pandas để đọc tệp Excel và boto3 để làm việc với MinIO qua API tương thích S3.
- Transform: Các bước chuyển đổi gồm chuẩn hóa định dạng mã sinh viên, kiểm tra giá trị GPA (0.0–4.0), xử lý các môn học thiếu điểm, tính GPA tích lũy, xác định tình trạng học vụ (tốt, cảnh báo, thử thách), và ánh xạ mã môn học sang thông tin khoa/bộ môn.
- Load: Dữ liệu sau khi chuyển đổi được nạp vào ClickHouse Data Warehouse theo mô hình star schema, sử dụng chiến lược incremental load dựa trên thông tin học kỳ và timestamp.

Toàn bộ quy trình ETL được điều phối bởi Apache Airflow với lịch chạy hằng ngày, log chi tiết và cơ chế cảnh báo khi có lỗi phát sinh.

CHƯƠNG II: PHÂN TÍCH VÀ XÂY DỰNG HỆ THỐNG

2.1. Phân tích yêu cầu bài toán

2.1.1. Phân tích nghiệp vụ cho công tác CVHT

2.1.1.1. Tổng quan nghiệp vụ CVHT trong quản lý sinh viên

CVHT là "lực lượng giữ vai trò quan trọng" trong công tác quản lý, theo dõi và "hỗ trợ sinh viên trong suốt quá trình học tập tại trường đại học". CVHT không chỉ là "cầu nối" giữa nhà trường và sinh viên mà còn là người "tiếp xúc", "nắm bắt" tình hình học tập, rèn luyện, tư tưởng và các khó khăn của sinh viên để kịp thời "tư vấn, định hướng và hỗ trợ".

Trong bối cảnh đào tạo theo "hệ thống tín chỉ", SV có quyền "chủ động" cao trong việc "đăng ký học phần", "xây dựng kế hoạch học tập cá nhân". Điều này đặt ra "yêu cầu" ngày càng lớn đối với CVHT trong việc:

- Theo dõi "tiến độ học tập" của từng sinh viên;
- Phát hiện sớm các "trường hợp sinh viên" có nguy cơ học vụ;
- Hỗ trợ sinh viên "điều chỉnh kế hoạch học tập" phù hợp;
- Theo dõi và "đánh giá kết quả rèn luyện" theo quy định của nhà trường;
- Tổng hợp "báo cáo phục vụ công tác quản lý lớp", khoa và nhà trường.

Tuy nhiên, trên thực tế, khối lượng SV lớn cùng với sự phân tán của dữ liệu học tập, rèn luyện và phản hồi giảng dạy khiến cho công tác CVHT gặp nhiều "khó khăn", đặc biệt trong việc "tổng hợp", "phân tích" và "đưa ra cảnh báo" kịp thời. Do đó, việc phân tích nghiệp vụ công tác CVHT là cơ sở quan trọng để xây dựng một hệ thống hỗ trợ dựa trên dữ liệu, giúp nâng cao hiệu quả quản lý sinh viên.

2.1.1.2. Các tác nhân tham gia trong nghiệp vụ CVHT

Nghiệp vụ công tác CVHT có sự tham gia của "nhiều tác nhân khác nhau", mỗi tác nhân đảm nhận "một vai trò nhất định" trong việc hình thành, xử lý và sử dụng thông tin liên quan đến sinh viên. Sinh viên là đối tượng trung tâm của nghiệp vụ, vừa là chủ thể tham gia các hoạt động học tập và rèn luyện, vừa là nguồn phát sinh dữ liệu cho hệ thống quản lý. Thông qua quá trình "học tập, tham gia" các hoạt động rèn luyện và thực hiện các khảo sát đánh giá, sinh viên tạo ra các dữ liệu phản ánh "kết quả học tập", "ý thức" và "thái độ" học tập của bản thân.

CVHT là tác nhân sử dụng "thông tin" để thực hiện các "nghiệp vụ quản lý" và hỗ trợ sinh viên. CVHT có trách nhiệm tiếp nhận và "phân tích dữ liệu học tập", rèn luyện và phản hồi từ giảng viên để đánh giá tình trạng của từng sinh viên và của cả lớp. Trên cơ sở đó, CVHT đưa ra các "biện pháp tư vấn", nhắc nhở hoặc hỗ trợ phù hợp. Ngoài ra, CVHT còn là người tổng hợp "báo cáo định kỳ" phục vụ "công tác quản lý" của khoa và nhà trường.

Giảng viên là tác nhân cung cấp dữ liệu học tập và các thông tin phản hồi mang tính chuyên môn. Thông qua việc giảng dạy và đánh giá học phần, giảng viên cập nhật điểm số, tình trạng chuyên cần và nhận xét về thái độ học tập của sinh viên. Những thông tin này có "ý nghĩa quan trọng trong việc đánh giá chất lượng học tập" và hỗ trợ CVHT trong việc nhận diện sớm các vấn đề phát sinh trong quá trình học tập của sinh viên.

2.1.1.3. Luồng thông tin trong công tác quản lý sinh viên

Luồng thông tin trong công tác quản lý sinh viên được hình thành từ "nhiều nguồn khác nhau" và phản ánh toàn diện quá trình học tập và rèn luyện của sinh viên. Trước hết, luồng thông tin học tập phát sinh từ hoạt động giảng dạy và đánh giá học phần, trong đó giảng viên cập nhật "điểm thành phần", "điểm kết thúc học phần" và "tình trạng chuyên cần của sinh viên vào các hệ

thống quản lý đào tạo. Các dữ liệu này sau đó được sử dụng để "tính toán các chỉ số" như điểm trung bình học kỳ, điểm trung bình tích lũy và số tín chỉ đạt hoặc không đạt".

Bên cạnh đó, luồng thông tin rèn luyện phản ánh "mức độ tham gia" của sinh viên vào "các hoạt động rèn luyện" theo quy định của nhà trường. Sinh viên thực hiện tự đánh giá, cung cấp minh chứng, sau đó các thông tin này được xác nhận và tổng hợp để hình thành điểm rèn luyện theo từng học kỳ. Luồng thông tin này đóng vai trò quan trọng trong việc đánh giá ý thức, thái độ và mức độ tuân thủ quy định của sinh viên.

Ngoài hai luồng thông tin trên, "luồng thông tin khảo sát và phản hồi cung cấp các dữ liệu mang tính định tính phục vụ công tác quản lý. Các phản hồi của giảng viên về tình hình lớp học và các đánh giá của sinh viên đối với giảng viên và chất lượng giảng dạy được thu thập thông qua các khảo sát định kỳ. Việc tổng hợp và phân tích các thông tin này giúp CVHT và lãnh đạo có cái nhìn đa chiều hơn về quá trình đào tạo".

2.1.1.4. Những hạn chế trong công tác CVHT

Mặc dù CVHT là "quan trọng trong việc hỗ trợ và quản lý sinh viên trong suốt quá trình đào tạo, song trên thực tế, công tác này tại nhiều cơ sở giáo dục đại học vẫn còn bộc lộ nhiều bất cập. Các hạn chế này không chỉ xuất phát từ cách thức tổ chức hoạt động mà còn liên quan đến hệ thống công cụ và phương pháp hỗ trợ CVHT trong quá trình thực hiện nhiệm vụ".

Trước hết, hệ thống dữ liệu phục vụ cho công tác CVHT hiện nay chưa được tích hợp và đồng bộ. Thông tin học tập của sinh viên thường được quản lý trong hệ thống đào tạo, trong khi dữ liệu về rèn luyện lại do các đơn vị khác phụ trách, còn kết quả khảo sát giảng viên và sinh viên thường được thu thập thông qua những công cụ riêng lẻ. Việc phân tán dữ liệu ở nhiều nguồn khác nhau gây khó khăn cho CVHT trong quá trình tổng hợp và khai thác thông tin,

đặc biệt khi cần đánh giá toàn diện tình hình học tập, rèn luyện cũng như thái độ của sinh viên.

Bên cạnh đó, hoạt động theo dõi kết quả học tập và rèn luyện của sinh viên hiện vẫn chủ yếu dựa vào các báo cáo thủ công hoặc bán tự động, được tổng hợp theo từng học kỳ hoặc từng giai đoạn nhất định. Cách tiếp cận này làm cho thông tin cung cấp cho CVHT thiếu tính kịp thời, không phản ánh đầy đủ những biến động diễn ra trong suốt quá trình học tập của sinh viên. Trong nhiều trường hợp, khi CVHT tiếp cận được thông tin thì sinh viên đã rơi vào tình trạng học vụ hoặc rèn luyện nghiêm trọng, khiến việc hỗ trợ và can thiệp gặp nhiều khó khăn.

Ngoài ra, công tác cảnh báo học vụ và rèn luyện hiện nay vẫn còn mang tính thụ động. Việc cảnh báo thường chỉ được kích hoạt khi sinh viên đã vi phạm các ngưỡng quy định cụ thể, chẳng hạn như điểm trung bình học tập thấp hoặc điểm rèn luyện không đạt yêu cầu. Điều này cho thấy chức năng cảnh báo sớm nhằm phòng ngừa rủi ro chưa được phát huy đúng mức. Việc thiếu các công cụ phân tích xu hướng và nhận diện sớm những dấu hiệu bất thường khiến CVHT khó chủ động xây dựng các biện pháp hỗ trợ phù hợp cho sinh viên.

Cuối cùng, "CVHT hiện chưa được trang bị đầy đủ các công cụ hỗ trợ phân tích và trực quan hóa dữ liệu. Việc đánh giá tình hình sinh viên chủ yếu dựa trên kinh nghiệm cá nhân kết hợp với các báo cáo tổng hợp đơn giản, chưa khai thác được các phân tích theo thời gian, theo nhóm đối tượng hay theo nhiều tiêu chí khác nhau. Hạn chế này ảnh hưởng đến khả năng đánh giá toàn diện và làm giảm hiệu quả của các quyết định hỗ trợ sinh viên".

2.1.1.5. Nhu cầu xây dựng hệ thống hỗ trợ CVHT dựa trên dữ liệu

Từ những nhu cầu thực tế của cố vấn học tập, việc xây dựng một hệ thống hỗ trợ cố vấn học tập dựa trên dữ liệu số là yêu cầu mang tính cấp thiết, nhằm nâng cao hiệu quả quản lý sinh viên cũng như chất lượng hoạt động CVHT. Hệ

thống này cần được định hướng phát triển theo mô hình tích hợp, cho phép thu thập và khai thác dữ liệu từ nhiều nguồn khác nhau như dữ liệu học tập, dữ liệu rèn luyện và dữ liệu khảo sát. Việc hình thành một nguồn dữ liệu tập trung và thống nhất sẽ tạo nền tảng quan trọng cho công tác phân tích, tổng hợp và báo cáo thông tin.

Trong đó, "việc áp dụng các giải pháp ETL (Extract – Transform – Load) kết hợp với kho dữ liệu giữ vai trò then chốt trong việc tự động hóa các khâu thu thập, xử lý, chuẩn hóa và lưu trữ dữ liệu. Cách tiếp cận này giúp đảm bảo dữ liệu được quản lý một cách chính xác, nhất quán và toàn vẹn, đồng thời tạo điều kiện thuận lợi cho việc phân tích thông tin theo nhiều chiều và ở các mức độ chi tiết khác nhau. Kho dữ liệu không chỉ đáp ứng nhu cầu báo cáo hiện tại mà còn đóng góp cho các hoạt động phân tích nâng cao trong tương lai".

Bên cạnh chức năng tích hợp và xử lý dữ liệu, hệ thống hỗ trợ CVHT cần được trang bị các công cụ phân tích và trực quan hóa thông tin thông qua các báo cáo và dashboard. Những công cụ này giúp CVHT theo dõi tình hình học tập và rèn luyện của sinh viên một cách trực quan, đồng thời hỗ trợ phát hiện sớm các trường hợp có nguy cơ để đưa ra cảnh báo kịp thời. Ngoài ra, hệ thống còn cung cấp thông tin tổng hợp phục vụ cho lãnh đạo khoa và nhà trường trong công tác quản lý và ra quyết định dựa trên dữ liệu.

Như vậy, việc xây dựng hệ thống hỗ trợ CVHT dựa trên dữ liệu không chỉ đáp ứng yêu cầu thực tiễn trong quản lý sinh viên mà còn phù hợp với xu hướng chuyển đổi số trong giáo dục đại học hiện nay. Phần phân tích nghiệp vụ công tác CVHT trong mục này đóng vai trò làm cơ sở cho việc xác định các yêu cầu chức năng và phi chức năng của hệ thống, đồng thời là tiền đề cho quá trình thiết kế kiến trúc, triển khai quy trình ETL và xây dựng các dashboard hỗ trợ CVHT được trình bày ở các chương tiếp theo của đề án.

2.1.2. Phân tích các use case chính của hệ thống

Trên cơ sở "kết quả phân tích nghiệp vụ công tác cố vấn học tập", hệ thống hỗ trợ CVHT được đề xuất nhằm "đáp ứng các nhu cầu theo dõi, phân tích và cảnh báo tình hình học tập cũng như rèn luyện của sinh viên. Các chức năng của hệ thống được biểu diễn thông qua mô hình use case, qua đó mô tả những tương tác chủ yếu giữa các tác nhân và hệ thống trong quá trình thực hiện các nghiệp vụ liên quan. Việc xây dựng và phân tích chi tiết các use case theo chuẩn tài liệu phân tích – thiết kế hệ thống cho phép xác định rõ ràng các yêu cầu chức năng, đồng thời tạo nền tảng cho các bước thiết kế tiếp theo, bao gồm thiết kế kiến trúc hệ thống, cơ sở dữ liệu và các mô-đun chức năng tương ứng".

2.1.2.1. Use case: Theo dõi tình trạng học tập và cảnh báo học vụ

Tên use case: Theo dõi tình trạng học tập và cảnh báo học vụ

Tác nhân chính: CVHT

Tác nhân phụ: Sinh viên, Giảng viên

Mô tả: Use case này cho phép hệ thống thu thập và phân tích dữ liệu học tập của sinh viên nhằm phát hiện sớm các trường hợp có nguy cơ học vụ. Hệ thống tổng hợp điểm số, số tín chỉ và tình trạng chuyên cần của sinh viên, từ đó xác định các trường hợp cần cảnh báo và cung cấp thông tin hỗ trợ cho CVHT.

Tiền điều kiện: Dữ liệu học tập của sinh viên đã được cập nhật đầy đủ từ hệ thống quản lý đào tạo; các ngưỡng cảnh báo học vụ đã được cấu hình theo quy định của nhà trường.

Hậu điều kiện: Danh sách sinh viên có nguy cơ học vụ được xác định và lưu trữ; thông tin cảnh báo được hiển thị trên hệ thống và sẵn sàng cho các use case tiếp theo.

Luồng chính:

- Hệ thống tiếp nhận dữ liệu điểm số, số tín chỉ và tình trạng chuyên cần của sinh viên.
- Hệ thống tổng hợp dữ liệu theo học kỳ và theo từng sinh viên.
- Hệ thống so sánh các chỉ số học tập với ngưỡng cảnh báo.
- Hệ thống phân loại mức độ rủi ro học vụ của sinh viên.
- Hệ thống hiển thị kết quả phân tích và danh sách sinh viên cần cảnh báo cho CVHT.

Luồng thay thế: Trường hợp dữ liệu học tập chưa đầy đủ, hệ thống ghi nhận trạng thái thiếu dữ liệu và không thực hiện phân tích cảnh báo đối với sinh viên tương ứng.

Dữ liệu vào: Điểm học phần, điểm trung bình học kỳ, số tín chỉ đạt/không đạt, số buổi nghỉ học.

Dữ liệu ra: Danh sách sinh viên có nguy cơ học vụ; các chỉ số học tập tổng hợp.

2.1.2.2. Use case: Theo dõi điểm rèn luyện và cảnh báo rèn luyện

Tên use case: Theo dõi điểm rèn luyện và cảnh báo rèn luyện

Tác nhân chính: CVHT

Tác nhân phụ: Sinh viên

Mô tả: Use case này hỗ trợ theo dõi và phân tích điểm rèn luyện của sinh viên theo từng học kỳ nhằm phát hiện các trường hợp chưa đạt yêu cầu hoặc thiếu tiêu chí rèn luyện.

Tiền điều kiện: Dữ liệu rèn luyện của sinh viên đã được cập nhật và xác nhận theo quy định.

Hậu điều kiện: Danh sách sinh viên chưa đạt yêu cầu rèn luyện được xác định và lưu trữ; trạng thái rèn luyện của sinh viên được cập nhật.

Luồng chính:

- Hệ thống tiếp nhận dữ liệu điểm rèn luyện theo từng tiêu chí.

- Hệ thống tổng hợp điểm rèn luyện theo học kỳ.
- Hệ thống kiểm tra các tiêu chí rèn luyện còn thiếu.
- Hệ thống xác định các trường hợp chưa đạt yêu cầu rèn luyện.
- Hệ thống hiển thị kết quả cho CVHT.

Luồng thay thế: Trường hợp dữ liệu rèn luyện chưa hoàn chỉnh, hệ thống ghi nhận trạng thái chờ hoàn thiện.

Dữ liệu vào: Điểm rèn luyện thành phần, minh chứng rèn luyện.

Dữ liệu ra: Danh sách sinh viên cần cảnh báo rèn luyện; báo cáo tổng hợp điểm rèn luyện.

2.1.2.3. Use case: Gửi email tự động khi có cảnh báo

Tên use case: Gửi email tự động khi có cảnh báo

Tác nhân chính: Hệ thống

Tác nhân phụ: Sinh viên, CVHT

Mô tả: Use case này thực hiện việc gửi email tự động khi phát sinh cảnh báo học vụ hoặc rèn luyện nhằm đảm bảo thông tin được truyền tải kịp thời và nhất quán.

Tiền điều kiện: Đã phát sinh cảnh báo học vụ hoặc rèn luyện; thông tin email của người nhận hợp lệ.

Hậu điều kiện: Email cảnh báo được gửi thành công và lịch sử gửi email được lưu trữ.

Luồng chính:

- Hệ thống nhận thông tin cảnh báo.
- Hệ thống tạo nội dung email theo mẫu.
- Hệ thống gửi email đến sinh viên và CVHT.
- Hệ thống lưu lịch sử cảnh báo và gửi email.

Luồng thay thế: Trường hợp gửi email không thành công, hệ thống ghi nhận lỗi và thực hiện gửi lại theo cấu hình.

Dữ liệu vào: Thông tin cảnh báo, thông tin người nhận.

Dữ liệu ra: Email cảnh báo; lịch sử gửi email.

2.1.2.4. Use case: Thống kê và báo cáo tình trạng lớp

Tên use case: Thống kê và báo cáo tình trạng lớp

Tác nhân chính: CVHT

Tác nhân phụ: Lãnh đạo khoa/trường

Mô tả: Use case này cho phép tổng hợp và trình bày các báo cáo về tình trạng học tập và rèn luyện của sinh viên theo lớp và theo thời gian.

Tiền điều kiện: Dữ liệu học tập và rèn luyện đã được tổng hợp trong hệ thống.

Hậu điều kiện: Các báo cáo và dashboard được tạo và sẵn sàng cho người dùng.

Luồng chính:

- Người dùng chọn phạm vi báo cáo.
- Hệ thống tổng hợp dữ liệu theo yêu cầu.
- Hệ thống hiển thị báo cáo và dashboard.
- Người dùng xuất báo cáo nếu cần.

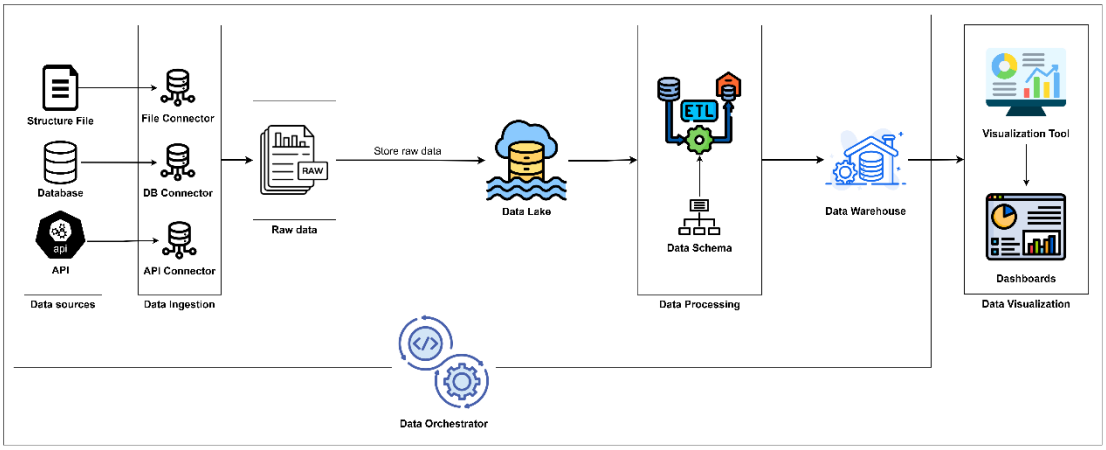
Dữ liệu vào: Tham số báo cáo (lớp, học kỳ, thời gian).

Dữ liệu ra: Báo cáo tổng hợp; dashboard trực quan.

2.2. Thiết kế và xây dựng hệ thống

2.2.1. Phân tích và thiết kế hệ thống

2.2.1.1. Kiến trúc hệ thống tổng quan



Hình 2. Tổng quan kiến trúc hệ thống

Hệ thống có kiến trúc hiện đại (Modern Data Architecture), bao gồm các tầng: Nguồn dữ liệu (Data Sources), Tiếp nhận dữ liệu (Data Ingestion), Lưu trữ dữ liệu thô (Data Lake), Xử lý – chuyển đổi dữ liệu (Data Processing / ETL), Kho dữ liệu (Data Warehouse) và Trực quan hóa dữ liệu (Data Visualization). Toàn bộ quy trình được điều phối bởi Data Orchestrator nhằm đảm bảo dữ liệu được di chuyển và xử lý tự động, nhất quán và hiệu quả.

- Hệ thống hỗ trợ đa nguồn dữ liệu:
  - File dữ liệu cấu trúc như CSV, Excel, JSON…
  - Cơ sở dữ liệu như MySQL, PostgreSQL, SQL Server…
  - API cung cấp dữ liệu theo thời gian thực hoặc theo yêu cầu.

Tầng này sử dụng các connector để thu thập dữ liệu từ từng loại nguồn:

- File Connector giúp đọc dữ liệu từ hệ thống file.
- DB Connector kết nối trực tiếp tới các hệ quản trị cơ sở dữ liệu.
- API Connector tiếp nhận dữ liệu thông qua REST API/HTTP.

Dữ liệu sau khi được ingest sẽ được lưu trữ trong Data Lake ở dạng nguyên bản. Data Lake đóng vai trò nơi tập trung dữ liệu, giúp hệ thống dễ dàng mở rộng và tái sử dụng.

Tầng xử lý thực hiện các nhiệm vụ:

- "Làm sạch dữ liệu" (cleaning).
- Biến đổi – chuẩn hóa dữ liệu (transformation).
- Ánh xạ vào mô hình dữ liệu (Data Schema) (không bắt buộc).
- Tải dữ liệu vào kho dữ liệu theo quy trình ETL/ELT.

Sau khi được chuẩn hóa, dữ liệu được lưu vào Data Warehouse, đóng vai trò:

- Lưu trữ dữ liệu sạch, có cấu trúc.
- "Tối ưu" cho truy vấn phân tích.
- Là nền tảng cho việc "xây dựng báo cáo" và dashboard.

Kho dữ liệu có thể theo mô hình star schema, snowflake, hoặc các mô hình hiện đại như Data Vault tùy theo yêu cầu hệ thống.

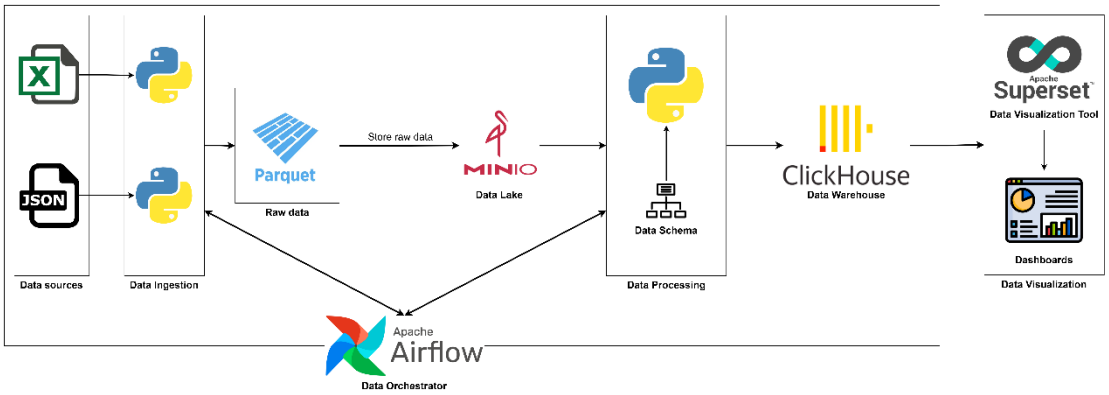
Tầng cuối cùng sử dụng các công cụ trực quan hóa để:

- Kết nối với Data Warehouse.
- Tạo dashboard.
- Hiển thị chỉ số và phân tích dữ liệu phục vụ nghiệp vụ và ra quyết định.

Công cụ Orchestrator đảm nhiệm:

- Lập lịch công việc (scheduler).
- Điều phối toàn bộ pipeline ingest → xử lý → load → kho dữ liệu.
- Giám sát luồng dữ liệu và xử lý lỗi.
- Tự động hóa toàn bộ hệ thống.

2.2.1.2. Thiết kế pipeline



Hình 3. Thiết kế pipeline chi tiết

Kiến trúc chi tiết của hệ thống đảm bảo khả năng thu thập, lưu trữ, xử lý và trực quan hóa dữ liệu một cách tự động, tập trung và mở rộng. Kiến trúc bao gồm các thành phần chính: nguồn dữ liệu, tầng tiếp nhận dữ liệu, hồ dữ liệu, tầng xử lý dữ liệu, kho dữ liệu, công cụ trực quan hóa và tầng điều phối dữ liệu. Các thành phần này liên kết với nhau thông qua một luồng xử lý khép kín, đảm bảo dữ liệu được chuyển đổi đúng chuẩn và sẵn sàng phục vụ phân tích.

Nguồn dữ liệu đầu vào của hệ thống chủ yếu bao gồm các tệp Excel và tệp JSON. Các tệp Excel thường chứa dữ liệu bảng được nhập thủ công hoặc xuất từ các hệ thống quản lý nội bộ, trong khi các tệp JSON đại diện cho dữ liệu bán cấu trúc thu thập từ API hoặc các ứng dụng lưu trữ key–value. Hai định dạng này phản ánh thực trạng phân tán và thiếu chuẩn hóa của dữ liệu hiện nay trong các đơn vị đào tạo, đồng thời là lý do cần xây dựng một pipeline dữ liệu thống nhất.

Quá trình tiếp nhận, thu thập dữ liệu được triển khai bằng các script Python, cho phép hệ thống tự động đọc, kiểm tra và chuẩn hóa dữ liệu đầu vào. Tại bước này, dữ liệu được xử lý sơ bộ như kiểm tra kiểu dữ liệu, loại bỏ lỗi định dạng, chuẩn hóa tên cột và cấu trúc. Sau khi hoàn tất các bước xử lý ban đầu, dữ liệu được chuyển đổi sang định dạng Parquet để tối ưu cho hệ thống

phân tích nhờ khả năng nén cột và giảm kích thước đáng kể so với CSV hoặc Excel.

Dữ liệu định dạng Parquet sau đó được lưu trữ tại Data Lake được xây dựng trên nền tảng MinIO. MinIO, với khả năng tương thích API S3, cho phép hệ thống lưu trữ dữ liệu phi cấu trúc và bán cấu trúc với khả năng mở rộng cao. Tại đây, dữ liệu được lưu giữ nguyên trạng (raw zone) nhằm đảm bảo tính toàn vẹn và cho phép tái xử lý khi cần thiết. Việc sử dụng MinIO là một lựa chọn phù hợp trong bối cảnh triển khai nội bộ, nhờ khả năng đơn giản, hiệu suất cao và dễ tích hợp với các công cụ xử lý dữ liệu.

Tầng "Xử lý dữ liệu (Data Processing)" được thực hiện bằng các chương trình Python đảm nhiệm vai trò ETL/ELT. Tại bước này, dữ liệu thô được trích xuất trực tiếp từ MinIO, sau đó trải qua các bước làm sạch, chuẩn hóa và mô hình hóa theo cấu trúc dữ liệu mục tiêu. Các thao tác bao gồm loại bỏ dữ liệu trùng lặp, xử lý giá trị thiếu, chuyển đổi định dạng, tính toán các chỉ số hoặc phân loại dữ liệu. Sau khi xử lý, dữ liệu được tổ chức lại theo schema phân tích và sẵn sàng được nạp vào kho dữ liệu.

Về kho dữ liệu, hệ thống sử dụng ClickHouse - cơ sở dữ liệu dạng cột có tốc độ xử lý truy vấn vượt trội làm và phù hợp cho các hệ thống phân tích thời gian thực và truy vấn tập dữ liệu lớn. Dữ liệu đã được chuẩn hóa từ tầng ETL được nạp vào ClickHouse dưới dạng các bảng phân tích, cho phép hệ thống thực hiện các truy vấn như tổng hợp, phân nhóm và phân tích đa chiều một cách nhanh chóng và chính xác.

Tầng trực quan hóa dữ liệu sử dụng Apache Superset, một công cụ mã nguồn mở hỗ trợ mạnh mẽ cho việc xây dựng dashboard tương tác. Superset kết nối trực tiếp với ClickHouse, cho phép hiển thị dữ liệu phân tích thông qua biểu đồ, bảng số liệu và các thành phần trực quan khác. Dashboard được sử

dụng cho nhiều mục tiêu như theo dõi kết quả học tập, cảnh báo học vụ hay đánh giá rèn luyện, hỗ trợ trực tiếp cho công tác CVHT.

Toàn bộ pipeline được điều phối tự động bởi Apache Airflow. Airflow quản lý lịch chạy, thứ tự thực thi và giám sát toàn bộ quá trình từ ingest, xử lý cho đến nạp dữ liệu vào kho phân tích. Thông qua các DAG (Directed Acyclic Graphs), hệ thống có khả năng tái thực thi từng bước khi lỗi xảy ra, đồng thời hỗ trợ lập lịch linh hoạt theo giờ, ngày hoặc tuần.

Tổng thể kiến trúc đảm bảo rằng dữ liệu từ các nguồn Excel và JSON được ingest → chuẩn hóa → lưu trữ → xử lý → phân tích → trực quan hóa theo một chu trình thống nhất, có tính tự động hóa cao và phù hợp với mô hình triển khai trong môi trường giáo dục đại học. Kiến trúc này không chỉ giải quyết bài toán phân tán dữ liệu mà còn đặt nền tảng cho việc phát triển các chức năng nâng cao như dự báo học vụ hay phân tích hành vi người học bằng AI trong tương lai.

2.2.1.3. Luồng dữ liệu

Luồng dữ liệu của hệ thống được thiết kế theo cấu trúc tuyến tính – phân tầng, đảm bảo dữ liệu được thu thập, lưu trữ, xử lý và trực quan hóa theo một quy trình khép kín. Toàn bộ luồng dữ liệu vận hành thống nhất từ nguồn đến dashboard, đồng thời được điều phối tự động bởi Apache Airflow, giúp hệ thống hoạt động ổn định và giảm thiểu thao tác thủ công. Luồng dữ liệu gồm các bước chính như sau.

Ở giai đoạn đầu, dữ liệu được thu thập từ hai nguồn chính là tệp Excel và tệp JSON. Đây là những loại dữ liệu phổ biến hiện nay trong môi trường giáo dục đại học, nơi thông tin thường được xuất từ các hệ thống quản lý hoặc tổng hợp thủ công từ nhiều bộ phận. Các tệp này được đưa vào hệ thống thông qua tầng tiếp nhận dữ liệu (Data Ingestion), nơi các chương trình Python thực hiện việc đọc dữ liệu, kiểm tra cấu trúc và chuẩn hóa ban đầu. Những xử lý sơ

bộ này nhằm đảm bảo dữ liệu đầu vào đạt mức chất lượng tối thiểu trước khi đưa vào pipeline.

Sau bước ingest, dữ liệu được chuyển đổi sang định dạng Parquet. Đây là định dạng lưu trữ tối ưu nhờ hỗ trợ nén theo cột, giúp giảm kích thước lưu trữ và tăng tốc độ truy vấn. Dữ liệu dạng Parquet được lưu trực tiếp vào Data Lake xây dựng trên nền tảng MinIO. Tại đây, MinIO đóng vai trò là vùng lưu trữ dữ liệu thô (Raw Zone), đảm bảo lưu giữ nguyên trạng dữ liệu nguồn, tránh mất mát thông tin và hỗ trợ tái xử lý khi cần thiết. Việc lưu trữ dữ liệu theo dạng phân lớp trong Data Lake giúp hệ thống dễ dàng kiểm soát các phiên bản và mức độ xử lý của dữ liệu.

Từ Data Lake, luồng dữ liệu chuyển sang giai đoạn xử lý (ETL/ELT). Các chương trình Python sẽ tải dữ liệu Parquet từ MinIO, sau đó thực hiện các bước làm sạch, chuẩn hóa và chuyển đổi theo mô hình dữ liệu đã thiết kế. Các phép xử lý bao gồm loại bỏ giá trị sai lệch, chuẩn hóa kiểu dữ liệu, tính toán thêm các trường mới và tổ chức lại dữ liệu theo dạng bảng phân tích. Đây là bước quan trọng nhằm đảm bảo dữ liệu đạt chất lượng và tính nhất quán trước khi nạp vào kho dữ liệu.

Dữ liệu sau xử lý được nạp vào ClickHouse – hệ quản trị cơ sở dữ liệu hướng cột có hiệu năng cao cho phân tích. Tại kho dữ liệu, các bảng phân tích được tổ chức theo cấu trúc tối ưu cho truy vấn tổng hợp, thống kê và phân tích đa chiều, giúp các truy vấn phân tích chạy nhanh và ổn định ngay cả với khối lượng dữ liệu lớn. ClickHouse đóng vai trò như tầng cung cấp dữ liệu cuối cùng cho các tác vụ trực quan hóa và phân tích.

Tiếp đó, Apache Superset được sử dụng làm công cụ trực quan hóa dữ liệu. Superset kết nối trực tiếp đến ClickHouse, cho phép người dùng xây dựng dashboard tương tác nhằm quan sát các chỉ số quan trọng như điểm trung bình, tín chỉ tích lũy, điểm rèn luyện hay tình trạng học vụ. Việc trực quan hóa giúp

đơn giản hóa quá trình phân tích, hỗ trợ CVHT và các bộ phận chuyên môn đưa ra quyết định chính xác hơn.

Toàn bộ luồng dữ liệu, từ ingest, chuyển đổi cho đến nạp dữ liệu và cập nhật dashboard, được quản lý và điều phối bởi Apache Airflow. Airflow sử dụng các DAG để đảm bảo pipeline hoạt động đúng trình tự, có khả năng tự động chạy định kỳ và xử lý lỗi hiệu quả.

2.2.2. Công nghệ sử dụng

2.2.2.1. Apache Airflow

Năm 2014, Apache Airflow được phát triển bởi Airbnb để lập lịch, giám sát và quản lý các workflow xử lý dữ liệu phức tạp [31]. Airflow cho phép định nghĩa các pipeline dữ liệu dưới dạng code Python thông qua khái niệm DAG (Directed Acyclic Graph), trong đó mỗi node đại diện cho một task và các edge thể hiện dependencies giữa các tasks. Kiến trúc của Airflow bao gồm bốn thành phần chính. Scheduler chịu trách nhiệm lập lịch thực thi các DAGs theo cấu hình định sẵn và trigger các tasks khi dependencies được thỏa mãn. Executor quản lý việc chạy các tasks, có thể là SequentialExecutor (chạy tuần tự), LocalExecutor (chạy song song trên local machine), hoặc CeleryExecutor/KubernetesExecutor cho thực thi phân tán. Web Server cung cấp giao diện người dùng để giám sát trạng thái DAG, xem logs, thực thi thủ công, và quản lý các kết nối. Metadata Database (thường là PostgreSQL hoặc MySQL) lưu trữ thông tin về các lần chạy DAG, task, kết nối, và các tham số. Ưu điểm nổi bật của Airflow là khả năng tạo pipeline động - DAGs được định nghĩa bằng Python code cho phép tạo pipelines động dựa trên tham số hoặc tập tin cấu hình. Airflow cung cấp khả năng theo dõi giám sát với giao diện web hiển thị chi tiết trạng thái từng task, thời gian chạy, tỉ lệ thành công/thất bại, và logs đầy đủ. Tính năng tự động chạy lại khi lỗi cho phép cấu hình số lần chạy lại và khoảng thời gian nghỉ giữa các lần thử khi lỗi. Hơn nữa, Airflow có một

hệ sinh thái phong phú với hàng trăm phương thức sẵn có để tích hợp với các hệ thống như cở sở dữ liệu, dịch vụ cloud, hàng đợi. Trong hệ thống pipeline phục vụ CVHT, Apache Airflow đóng vai trò điều phối toàn bộ quy trình ETL. DAG được thiết kế với các tasks: extract_from_minio để đọc Excel files từ Data Lake, transform_data thực hiện cleaning và calculation, validate_data kiểm tra data quality, load_to_clickhouse ghi dữ liệu vào Data Warehouse.

2.2.2.2. MinIO

MinIO là một object storage server mã nguồn mở được thiết kế để lưu trữ dữu liệu phi cấu trúc như ảnh, video, log,… MinIO được phát triển năm 2015 với mục tiêu cung cấp một giải pháp object storage tương thích với Amazon S3 API nhưng có thể triển khai on-premise hoặc trên bất kỳ cloud infrastructure nào [32]. Kiến trúc của MinIO được xây dựng trên nguyên tắc hệ thống phân tán với khả năng mở rộng theo chiều ngang bằng cách thêm các máy lưu trữ và mạng lưới sẵn có. MinIO hỗ trợ lưu trữ phiên bản dữ liệu cho phép giữ nhiều phiên bản của cùng một đối tượng. Hiệu năng là điểm mạnh của MinIO với khả năng đọc/ghi tốc độ cao nhờ thiết kế nhẹ và tối ưu cho phần cứng hiện đại. MinIO được viết bằng ngôn ngữ Go, cho phép triển khai dễ dàng không phụ thuộc vào thư viện bên ngoài. Tương thích với S3 nghĩa là các ứng dụng đã sử dụng AWS S3 SDK có thể chuyển sang MinIO mà không cần thay đổi code nhiều. MinIO cũng cung cấp giao dện web trực quan để quản lý buckets, truy xuất file, và cấu hình hệ thống. Trong đề tài này, MinIO được sử dụng làm Data Lake để lưu trữ các file Excel thô từ Phòng Đào tạo và Phòng Công tác Sinh viên. Dữ liệu được tổ chức theo cấu trúc bronze layer với naming convention: raw-data/<filename_timestamp.xlsx>. Ví dụ: raw-data/ student_grades_20240915_143022.xlsx.

2.2.2.3. Clickhouse

ClickHouse là một hệ quản trị cơ sở dữ liệu dạng columnar mã nguồn mở được phát triển bởi Yandex năm 2016, chuyên biệt hóa cho online analytical processing (OLAP). ClickHouse được thiết kế để xử lý hàng tỷ bản ghi và trả về kết quả trong thời gian thực, đạt thông lượng hàng triệu dòng trên giây trên một máy chủ. Kiến trúc lưu trữ columnar của ClickHouse là yếu tố then chốt mang lại hiệu năng cao. Dữ liệu được lưu trữ theo cột thay vì theo hàng như cơ sở dữ liệu truyền thống, cho phép đọc chỉ các cột cần thiết cho truy vấn, áp dụng điều kiện hiệu quả hơn, và tận dụng tốt CPU cache. ClickHouse sử dụng các thuật toán nén dữ liệu như LZ4, ZSTD đạt tỉ lệ nén 10:1 đến 100:1 tùy thuộc vào kiểu dữ liệu [33]. ClickHouse cung cấp nhiều engines phù hợp với các tình huống sử dụng khác nhau. MergeTree là engine phổ biến nhất, hỗ trợ khóa chính, phân vùng,… ReplacingMergeTree tự động gộp những bản ghi có khóa chính trùng nhau. SummingMergeTree tự động tính toán cột dạng số liệu khi kết hợp các phần của dữ liệu. Engine phân tán cho phép phân chia dữ liệu trên nhiều máy chủ. ClickHouse hỗ trợ cú pháp SQL với nhiều tiện ích mở rộng như mảng, các hàm bậc cao,… Materialized views trong ClickHouse cho phép tiền tính toán dữ liệu và tự động cập nhật khi dữ liệu mới được thêm vào, giúp tăng tốc truy vấn trên dữ liệu đã được tính toán. Sao y được thực hiện thông qua Apache ZooKeeper đảm bảo tính khả dụng và chịu lỗi. ClickHouse cũng cung cấp các API giao thức HTTP và TCP truyền thống, cho phép truy vấn trực tiếp qua REST API. Trong hệ thống Data Warehouse của đề tài, ClickHouse được lựa chọn để lưu trữ dữ liệu trong silver và gold layers. Lược đồ được thiết kế theo star schema với fact table fact_student_performance sử dụng MergeTree engine với partition by semester_id và order by (student_id, course_id). Dimension tables dim_student, dim_semester, dim_course, dim_date sử dụng ReplacingMergeTree để xử lý chuyển dữ liệu chậm.

Materialized views được tạo để tiền tính toán các tiêu chí như GPA theo lớp, GPA theo kỳ, số lượng sinh viên đang ở mức cảnh báo, giúp truy vấn chạy nhanh hơn 10-20 lần so với truy vấn trực tiếp trên fact table.

2.2.2.4. Apache Superset

Apache Superset là một modern data exploration và visualization platform mã nguồn mở được phát triển ban đầu tại Airbnb và trở thành Apache project năm 2017 [34]. Superset được thiết kế để dễ sử dụng cho cả người dùng không hiểu kỹ thuật và data analysts, cung cấp giao diện web động để tạo biểu đồ, dashboards và thực hiện các truy vấn kết hợp. Kiến trúc của Superset bao gồm Web Server chạy trên Flask framework (Python), Metadata Database lưu trữ cấu hình dashboard và quyền hạn người dùng, Cache Layer (Redis/Memcached) để cache kết quả truy vấn và cải thiện hiệu năng, và SQL Lab - một SQL IDE tích hợp cho phép viết và thực thi truy vấn trực tiếp. Superset kết nối với nguồn dữ liệu thông qua SQLAlchemy, hỗ trợ hơn 30 databases như PostgreSQL, MySQL, ClickHouse, Snowflake, BigQuery, … Superset cung cấp thư viện trực quan hóa phong phú với hơn 40 loại biểu đồ bao gồm biểu đồ cột, biểu đồ đường, biểu đồ điểm, bản đồ nhiệt, bản dồ địa lý, bảng đa chiều, và nhiều dạng nâng cao khác. Công cụ xây dựng truy vấn no-code cho phép người dùng tạo biểu đồ bằng cách kéo thả mà không cần viết code, chỉ cần chọn bộ dữ liệu, tiêu chí, chiều dữ liệu và loại biểu đồ. Công cụ xây dựng dashboard hỗ trợ tạo dashboard động với bộ lọc, lọc đa biểu đồ, và truy xuất dữ liệu được hiển thị trên biểu đồ. SQL Lab là một tính năng mạnh mẽ cung cấp công cụ truy vấn trên giao diện web với khả năng lưu lại lịch sử truy vấn, hỗ trợ hiển thị và viết cú pháp SQL, và khả năng “Lưu truy vấn dưới dạng bộ dữ liệu” để tái sử dụng khi vẽ biểu đồ hoặc truy vấn. Mô hình bảo mật của Superset cho phép bảo mật đến mức bản ghi, mức thuộc tính, và phân quyền theo vai trò (RBAC) để quản lý quyền chi tiết. Superset cũng hỗ trợ báo cáo

định kỳ và cảnh báo, cho phép gửi bản sao y dashboard qua email theo lịch hoặc thiết lập ngưỡng gửi khi tiêu chí nào đó vượt ngưỡng. Trong hệ thống presentation layer của đề tài, Apache Superset được sử dụng để xây dựng dashboard phục vụ CVHT. Dashboard tổng quan cung cấp cái nhìn khái quát với các thẻ số liệu như tổng số sinh viên, điểm GPA trung bình và tỷ lệ cảnh báo học vụ. Kèm theo đó là biểu đồ đường thể hiện xu hướng GPA theo từng học kỳ và biểu đồ cột so sánh GPA giữa các lớp. Dashboard thông tin sinh viên cho phép cố vấn tìm kiếm và xem chi tiết từng sinh viên, bao gồm lịch sử GPA, bảng điểm từng môn, biểu đồ điểm rèn luyện và các thông báo cảnh báo học tập. Bảng phân tích cung cấp góc nhìn chuyên sâu hơn với các biểu đồ phân bố (histogram phân bố GPA, phân bố điểm theo môn học), phân tích theo nhóm sinh viên (cohort analysis) và các chỉ số dự báo giúp nhận diện sinh viên có nguy cơ gặp vấn đề. Tất cả các dashboard đều cho phép lọc theo năm học, học kỳ, lớp và khoa, giúp cố vấn dễ dàng truy sâu vào từng nhóm dữ liệu cụ thể. Superset được kết nối trực tiếp với kho dữ liệu ClickHouse và sử dụng các materialized view để đảm bảo thời gian tải dưới 2 giây.

CHƯƠNG III: THỰC NGHIỆM VÀ ĐÁNH GIÁ HỆ THỐNG

3.1. Môi trường triển khai

Môi trường triển khai hệ thống pipeline xử lý dữ liệu phục vụ CVHT hướng tới tính ổn định, khả năng mở rộng và dễ dàng bảo trì. Việc lựa chọn công nghệ và cấu hình phần cứng phù hợp đóng vai trò quan trọng trong việc đảm bảo hiệu năng và độ tin cậy của hệ thống.

Thành phần	**Cấu hình tối thiểu**	**Cấu hình khuyến nghị**
CPU	12 vCPU	24 vCPU
RAM	8 GB	32 GB
Bộ nhớ	150 GB	Tùy theo lượng dữ liệu

Bảng 1. Cấu hình máy chủ thử nghiệm

Hệ điều hành: Ubuntu Server 22.04 LTS được chọn làm platform chính do tính ổn định, long-term support, và compatibility tốt với các công nghệ open-source. Containerization: Docker Engine và Docker Compose được sử dụng để đóng gói các dịch vụ, đảm bảo tính gọn nhẹ và dễ triển khai. Ngôn ngữ lập trình Python là ngôn ngữ chính cho ETL và xử lý dữ liệu.

3.2. Xây dựng các mô-đun chức năng

3.2.1. Mô-đun theo dõi học tập

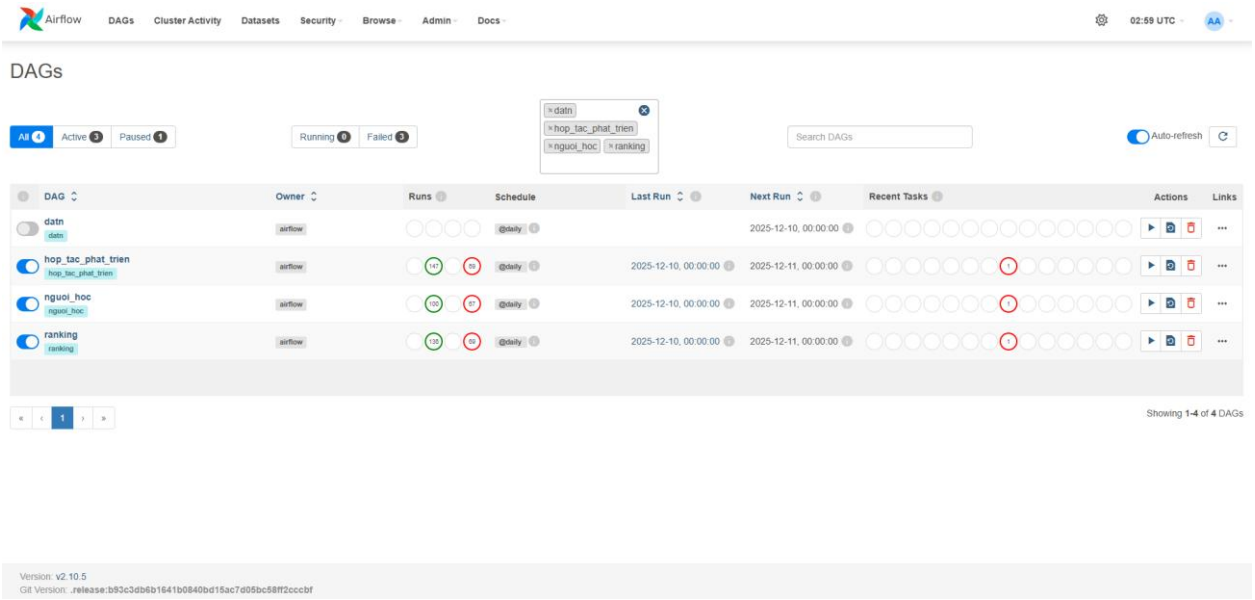
Mô-đun theo dõi học tập được xây dựng nhằm hỗ trợ CVHT trong việc giám sát tiến độ học tập của sinh viên một cách liên tục và có hệ thống. Mô-đun này thực hiện việc thu thập dữ liệu học tập từ các hệ thống quản lý đào tạo, bao gồm điểm học phần, điểm trung bình học kỳ, điểm trung bình tích lũy, số tín chỉ đăng ký, số tín chỉ đạt và số tín chỉ không đạt. Dữ liệu sau khi được trích xuất sẽ được xử lý và chuẩn hóa thông qua quy trình ETL trước khi lưu trữ vào kho dữ liệu phục vụ phân tích.

STT	Mã sinh viên	Tên	Lớp	Vắng CP	Vắng KP	Tổng điểm thưởng	Điểm chuyên cần	Điểm GK	Điểm nhóm
				0	0	0.5	9.5	9.3	6.5
				0	1	9	10	10	9.95
				0	0	5	10	10	10
				0	2	1	6	6	8.5
				0	0	7	10	10	9.95
				3	0	0	6	8.7	8.5
				0	0	7	10	10	8.8
				0	0	9.5	10	10	9
				0	1	1.5	8.5	7.5	6
				0	1	9	10	10	9.85
				0	2	9.5	10	10	8.75
				0	0	9.5	10	10	10
				0	0	1	10	8.4	6
				0	2	1	6	9.9	7
				0	0	9	10	10	10
				0	0	5.5	10	10	8.65
				0	0	9	10	10	10
				0	2	2	7	8.4	7
				1	0	1	9	7.8	6
				0	5	3	2	7.2	6
				0	0	4.5	10	10	8.35
				0	0	7	10	10	8.6
				0	3	2	5	8.9	7

Hình 4. Dữ liệu điểm danh học tập sinh viên

Số TT	Mã SV	Họ và tên	Lớp	Điểm CC	Điểm TBKT	Điểm TN-TH	Điểm BTTL	Ghi chú
Trọng số:				10	10	0	20	
				10.0	10.0		8.5	
				10.0	7.5		7.5	
				10.0	8.0		8.5	
				10.0	8.5		7.0	
				10.0	9.3		8.5	
				10.0	10.0		6.5	
				10.0	7.0		8.5	
				10.0	8.5		7.0	
				10.0	9.0		7.5	
				10.0	8.5		8.0	
				10.0	9.5		9.5	
				10.0	9.0		8.5	
				10.0	9.0		9.0	
				10.0	8.3		7.5	
				10.0	8.5		8.0	
				5.0	7.3		0.0	Không đủ ĐKDT
				10.0	7.5		9.0	
				10.0	9.3		7.5	
				10.0	8.3		7.5	
				10.0	9.0		8.5	
				6.0	7.8		7.5	

Hình 5. Dữ liệu điểm thành phần



Hình 6. Lập lịch xử lý dữ liệu

Trên cơ sở dữ liệu đã được xử lý, mô-đun theo dõi học tập tiến hành tính toán các chỉ số học tập quan trọng và so sánh với các ngưỡng cảnh báo theo quy định đào tạo của nhà trường. Kết quả phân tích cho phép xác định các sinh viên có dấu hiệu học tập yếu kém hoặc có nguy cơ bị cảnh báo học vụ. Thông tin này được hiển thị dưới dạng bảng tổng hợp và biểu đồ trực quan trên dashboard, giúp CVHT nhanh chóng nắm bắt tình hình học tập của từng sinh viên cũng như toàn bộ lớp học.

3.2.2. Mô-đun theo dõi rèn luyện

Mô-đun theo dõi rèn luyện được thiết kế nhằm quản lý và phân tích dữ liệu liên quan đến điểm rèn luyện của sinh viên theo từng học kỳ. Mô-đun này thu thập dữ liệu điểm rèn luyện theo các tiêu chí đánh giá, bao gồm kết quả tự đánh giá của sinh viên, xác nhận của cán bộ quản lý và tổng điểm rèn luyện cuối cùng.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1													
2													
3	Tiêu chí 1. Đánh giá về ý thức tham gia học tập												
4	Ý thức và thái độ trong học tập:												
5	Kết quả học tập trong kỳ học												
6	Ý thức chấp hành tốt nội quy về các kỳ thi	4	4	4	4	4	4	4	4	4	4	4	
7	Ý thức và thái độ tham gia các hoạt động ngoại khóa, các sự kiện liên quan đến nghiên cứu khoa học, học thuật, chuyên môn, Câu lạc bộ	2	1,5	1	2	1	2	1	1	0,5	0,5	0,5	
8	Tinh thần vượt khó, phấn đấu vươn lên trong học tập (có ĐTBCTL học kỳ sau lớn hơn học kỳ trước đó; đối với sinh viên năm thứ nhất, học kỳ 1 không có điểm dưới 2,5)												
9	Tiêu chí 2. Đánh giá về ý thức chấp hành nội quy, quy chế, quy định trong Học viện												
10	Thực hiện nghiêm túc các nội quy, quy chế, các quy định hiện hành trong Học viện.	15	15	15	15	15	15	15	15	15	15	15	
11	Thực hiện quy định về công tác nội trú, ngoại trú												
12	Thực hiện nghiêm túc các buổi họp lớp/ sinh hoạt đoàn thể do Học viện/Khoa/Viện, CVHT, Lớp/Chi đoàn tổ chức (tùy thuộc vào số buổi tổ chức sinh hoạt, họp)	2	2	2	2	2	2	2	2	2	2	2	
13	Tham gia các buổi hội thảo việc làm, định hướng nghề nghiệp do Học viện tổ chức	5	4	2	5	4	2	3	1	2	2	2	
14	Tiêu chí 3. Đánh giá về ý thức và kết quả tham gia hoạt động chính trị- xã hội, văn hóa, văn nghệ, thể thao, phòng chống tội phạm và các tệ nạn xã hội												
15	Tham gia đầy đủ các hoạt động chính trị, xã hội, các hoạt động văn hóa, văn nghệ, thể thao, phong trào tình nguyện, các buổi sinh hoạt chuyên đề do Học viện, lớp/chi đoàn, địa phương nơi cư trú tổ chức												
16	Tham gia công tác xã hội như: hiến máu nhân đạo, ủng hộ người nghèo gặp thiên tai lũ lụt và các công tác xã hội khác	4	4	4	2	1	1	0	1	0	0	0	
17	Tuyên truyền tích cực hình ảnh về Trường/Khoa trên các trang mạng xã hội	3	3	3	3	3	3	3	3	3	3	3	
18	Tích cực tham gia các hoạt động phòng, chống tội phạm, các tệ nạn xã hội, phát hiện và báo cáo kịp thời những hành vi có liên quan đến ma túy, các tệ nạn xã hội khác												
19	Đưa các thông tin sai lệch, thông tin chưa được kiểm chứng, đăng bình luận không chính xác, thiếu tính xây dựng về Học viện/Khoa/ngành đang học												

BỘ KHOA HỌC VÀ CÔNG NGHỆ
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

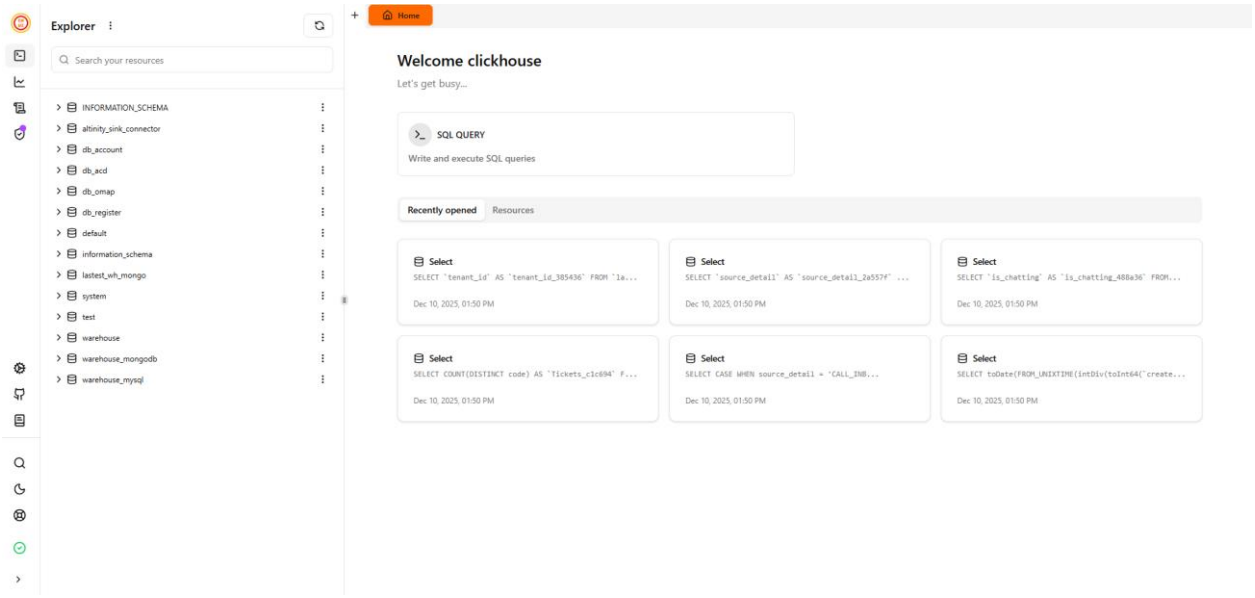
TỔNG HỢP KẾT QUẢ RÈN LUYỆN CỦA SINH VIÊN

Lớp: D24CQCC02-B **Khoa**: Viện Khoa học Kỹ thuật Bưu điện

Học kỳ: 2 **Năm học**: 2024

TT	Mã sinh viên	Họ đệm	Tên	Điểm đánh giá					Tổng điểm	Xếp loại rèn luyện
				ND1	ND2	ND3	ND4	ND5		
1	B24DCCC068	Nguyễn Minh	Đức	14.5	22	16	25		77.5	Khá
2	B24DCCC200	Nguyễn Đức	Minh	14	21	13	25		73	Khá
3	B24DCCC044	Lê Đức	Đại	14	22	13	25		74	Khá
4	B24DCCC182	Trịnh Đức	Lộc	15	22	17	25	4	83	Tốt
5	B24DCCC164	Trần Nam	Khánh	16	19	17	25		77	Khá
6	B24DCCC212	Tạ Nhất	Nam	12	20	16	25		73	Khá
7	B24DCCC242	Bùi Văn	Thắng	14.5	20	16	25		75.5	Khá
8	B24DCCC032	Lê Xuân	Bách	14.5	22	13	25		74.5	Khá
9	B24DCCC002	Đặng Thái	An	14	22	14	25		75	Khá
10	B24DCCC194	Phạm Thế	Mạnh	14	22	18	25		79	Khá
11	B24DCCC008	Cao Đức	Anh	4	10				14	Kém
12	B24DCCC248	Lương Lê Chí	Thanh	14	20	14	25		73	Khá
13	B24DCCC116	Giáp Văn	Hiếu	14	21	16	25		76	Khá
		Huyền								
21	B24DCCC218	Nguyễn Bảo	Ngọc	16.5	22	17	25		80.5	Tốt
22	B24DCCC284	Vương Sỹ	Tuyển	14.5	22	20	25		81.5	Tốt
23	B24DCCC290	Nguyễn Viết	Vinh	16.5	22	16	25	4	83.5	Tốt
24	B24DCCC236	Lê Thanh	Quảng	15	23	17	25	3	83	Tốt
25	B24DCCC038	Nguyễn Văn	Chí	14.5	20	12	20		66.5	Khá
26	B24DCCC260	Nguyễn Trường	Tín	14.5	20	19	25		78.5	Khá
27	B24DCCC158	Lê Trần Nam	Khánh	14	20	15	25		74	Khá
28	B24DCCC020	Nguyễn Lê	Anh	14	22	16	25		77	Khá
29	B24DCCC074	Trần Minh	Đức	15	22	13	25	4	79	Khá
30	B24DCCC206	Trần Hải	Minh	12	13	14	25		64	Trung bình
31	B24DCCC278	Nguyễn Đặng Anh	Tuấn	14	22	16	25		77	Khá
32	B24DCCC152	Hoàng Đình	Khải	14	22	16	25		77	Khá
33	B24DCCC056	Nguyễn Trọng	Đạt	14	14	14	25		67	Khá
34	B24DCCC134	Nguyễn Hữu	Hùng	16	20	16	25		77	Khá
35	B24DCCC092	Mai Đức	Duy	12	22	13	25		72	Khá
36	B24DCCC086	Trịnh Minh	Dũng	14.5	21	16	25		76.5	Khá

Hình 7. Dữ liệu điểm rèn luyện thu thập

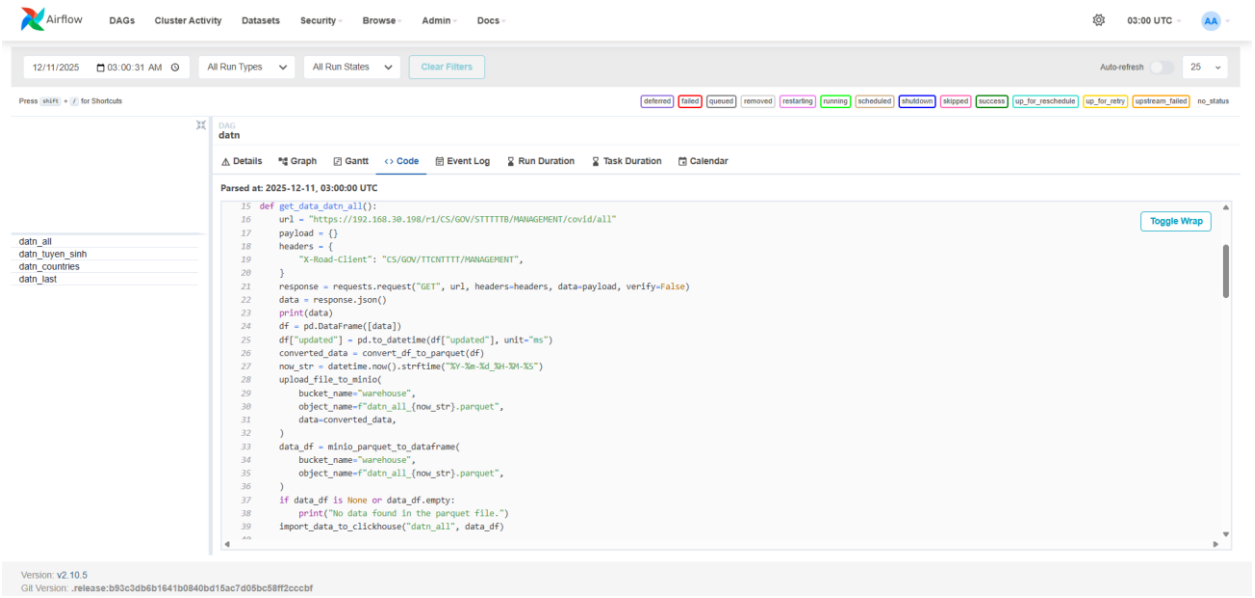


Hình 8. Kho dữ liệu lưu trữ dữ liệu đã được xử lý

Sau khi dữ liệu được tích hợp và lưu trữ trong kho dữ liệu, mô-đun tiến hành phân tích để phát hiện các trường hợp sinh viên chưa đạt điểm rèn luyện theo quy định hoặc còn thiếu các tiêu chí rèn luyện bắt buộc. Kết quả phân tích được cung cấp cho CVHT dưới dạng danh sách sinh viên cần theo dõi và hỗ trợ. Việc theo dõi rèn luyện không chỉ dừng lại ở một thời điểm mà còn cho phép phân tích xu hướng rèn luyện của sinh viên theo thời gian, qua đó hỗ trợ CVHT trong việc đánh giá toàn diện thái độ và ý thức học tập của sinh viên.

3.2.3. Mô-đun cảnh báo và gửi email tự động

Mô-đun cảnh báo và gửi email tự động đóng vai trò trung tâm trong việc tự động hóa công tác thông báo và nhắc nhở sinh viên. Mô-đun này nhận dữ liệu đầu vào từ các mô-đun theo dõi học tập và theo dõi rèn luyện, khi phát hiện sinh viên vượt ngưỡng cảnh báo, hệ thống sẽ tự động tạo thông báo cảnh báo tương ứng.



Hình 9. Xử lý dữ liệu và gửi email

To: Bui Van Tung M24KH02

Gửi sinh viên Bùi Văn Tùng,

Căn cứ vào kết quả tổng hợp điểm học tập và điểm rèn luyện của sinh viên trong học kỳ {{Học kỳ/Năm học}}, hệ thống ghi nhận:

- Điểm trung bình học tập (GPA): Trên 3.2, đạt mức xét học bổng theo quy định
- Điểm rèn luyện: 75 điểm

Theo quy định hiện hành của Nhà trường, điểm rèn luyện là một trong những tiêu chí bắt buộc khi xét học bổng. Do đó, với mức điểm rèn luyện hiện tại, sinh viên có nguy cơ không đủ điều kiện để được xét học bổng, mặc dù kết quả học tập đạt yêu cầu.

Đề nghị sinh viên:

- Kiểm tra lại các tiêu chí đánh giá điểm rèn luyện trong học kỳ;
- Khẩn trương hoàn thiện các nội dung còn thiếu (nếu có) theo hướng dẫn của lớp và Khoa;
- Chủ động liên hệ với Cố vấn học tập hoặc Ban cán sự lớp để được hỗ trợ và hướng dẫn kịp thời.

Nhà trường và Cố vấn học tập mong rằng sinh viên sẽ sớm khắc phục và cải thiện điểm rèn luyện nhằm đảm bảo quyền lợi chính đáng của bản thân.

Trân trọng thông báo.

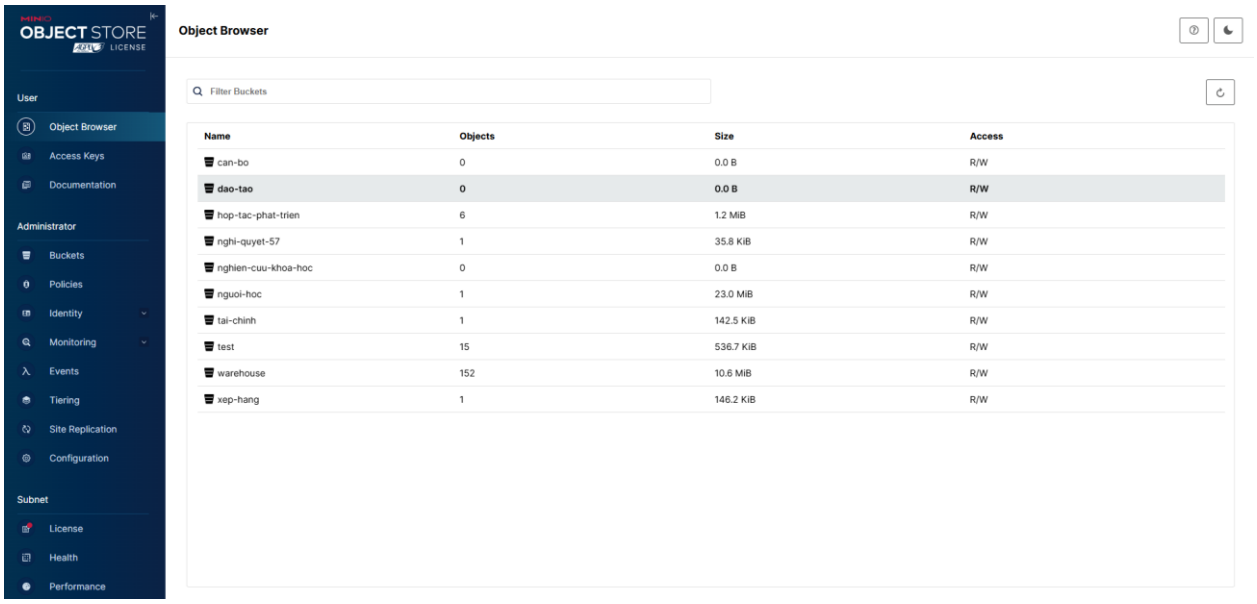
CỐ VẤN HỌC TẬP
Đỗ Hương Hà
Research Institute of Posts and Telecommunications (RIPT)
Posts and Telecommunications Institute of Technology (PTIT)
Address: 122 Hoang Quoc Viet, Nghia Do, Ha Noi

Hình 10. Email cảnh báo sinh viên

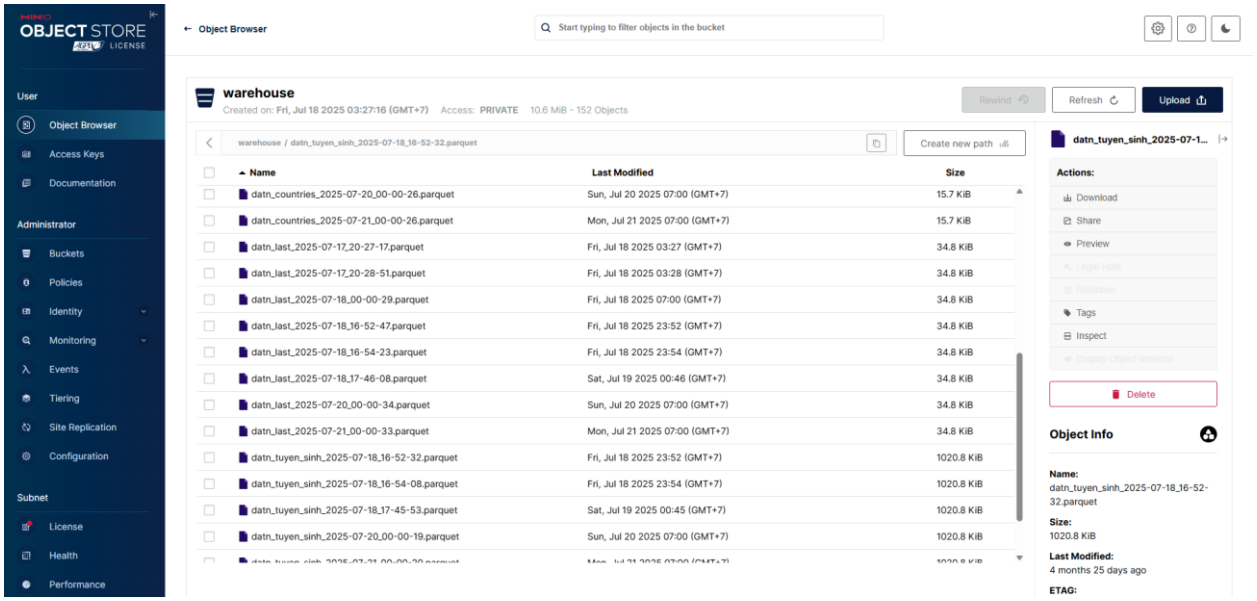
Nội dung cảnh báo được xây dựng theo các mẫu định sẵn, bao gồm thông tin sinh viên, loại cảnh báo, nguyên nhân và hướng dẫn xử lý. Hệ thống tự động gửi email cảnh báo đến sinh viên, đồng thời gửi thông báo tổng hợp đến CVHT để theo dõi và quản lý. Tất cả các cảnh báo và lịch sử gửi email đều được lưu trữ trong hệ thống nhằm phục vụ công tác truy vết, đánh giá hiệu quả cảnh báo và hỗ trợ công tác quản lý sau này.

3.2.4. Mô-đun báo cáo thống kê

Trên nền tảng kho dữ liệu, mô-đun báo cáo thống kê thực hiện việc xây dựng các báo cáo và dashboard phục vụ các nhóm người dùng khác nhau. Đối với CVHT, mô-đun cung cấp các báo cáo tổng hợp về tình hình học tập, điểm rèn luyện, danh sách sinh viên có nguy cơ và lịch sử cảnh báo theo từng lớp và từng học kỳ. Đối với lãnh đạo khoa và nhà trường, mô-đun cung cấp các báo cáo tổng hợp ở mức cao hơn, phản ánh xu hướng học tập và rèn luyện của sinh viên theo ngành, khoa và theo thời gian, từ đó hỗ trợ công tác quản lý và ra quyết định.

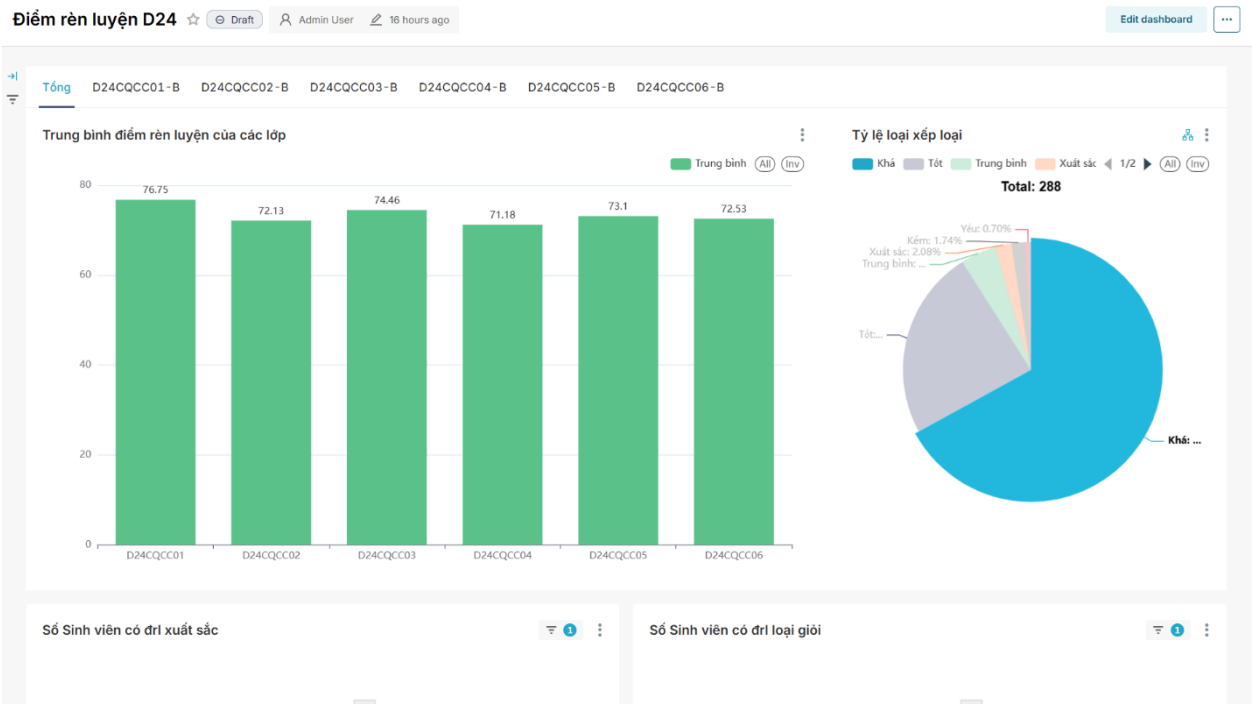


Hình 11. Hồ dữ liệu lưu trữ dữ liệu gốc

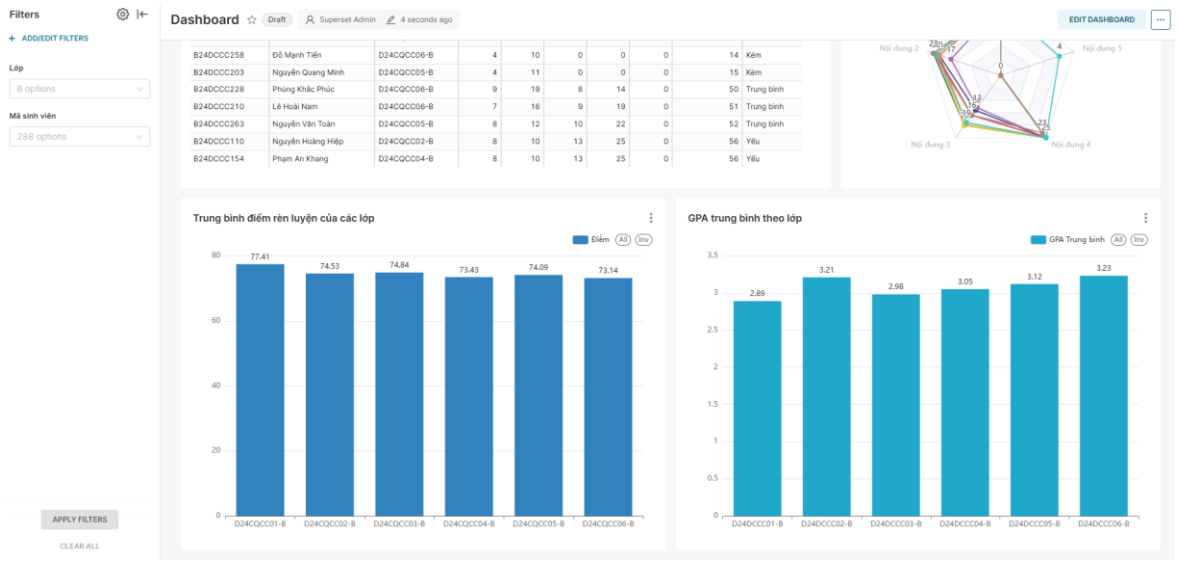


Hình 12. Dữ liệu lưu trữ trong hồ dữ liệu dưới dạng parquet

Các báo cáo được thiết kế dưới dạng bảng biểu, biểu đồ và dashboard trực quan, cho phép người dùng dễ dàng theo dõi các chỉ số quan trọng và nhận diện xu hướng biến động của dữ liệu. Hệ thống hỗ trợ lọc và phân tích dữ liệu theo nhiều tiêu chí khác nhau, giúp người dùng khai thác thông tin một cách linh hoạt mà không cần can thiệp trực tiếp vào dữ liệu gốc. Đồng thời, mô-đun cho phép xuất báo cáo ra các định dạng phổ biến nhằm phục vụ công tác báo cáo, lưu trữ và chia sẻ thông tin.



Hình 13. Dashboard điểm rèn luyện sinh viên



Hình 14.Dashboard điểm học tập sinh viên

3.3. Đánh giá hệ thống

3.3.1. Kịch bản thử nghiệm hệ thống

Để đánh giá một cách toàn diện tính khả thi và hiệu quả của hệ thống ETL Dashboard trong việc hỗ trợ công tác cố vấn học tập, một kịch bản thử nghiệm được xây dựng dựa trên dữ liệu giả lập nhưng bám sát thực tế triển khai

tại các cơ sở giáo dục đại học hiện nay. Kịch bản thử nghiệm tập trung kiểm tra khả năng xử lý dữ liệu nhiều nguồn, nhiều học kỳ, cũng như khả năng đáp ứng các use case đã được xác định trong giai đoạn phân tích yêu cầu.

Dữ liệu thử nghiệm bao gồm thông tin học tập và rèn luyện của 4 lớp sinh viên, với tổng số 170 sinh viên, trải dài trong 1 học kỳ gần nhất. Mỗi lớp có quy mô trung bình từ 40 đến 45 sinh viên, phản ánh quy mô lớp học phổ biến trong đào tạo đại học.

Trong mỗi học kỳ, hệ thống tiếp nhận các tập dữ liệu sau:

- Tệp điểm học tập: chứa thông tin mã sinh viên, mã học phần, số tín chỉ, điểm học phần và kết quả đạt/không đạt.
- Tệp điểm rèn luyện: chứa thông tin mã sinh viên và điểm rèn luyện học kỳ.
- Tệp thông tin sinh viên: bao gồm mã sinh viên, họ tên, lớp, khóa học và thông tin liên hệ.
- Tệp thông tin lớp học: mô tả mối quan hệ giữa sinh viên và lớp quản lý.

Các tệp dữ liệu được giả lập dưới định dạng CSV/Excel, tương ứng với cách thức trích xuất dữ liệu phổ biến từ các hệ thống quản lý đào tạo hiện nay. Dữ liệu giữa các học kỳ có sự thay đổi về số lượng học phần, điểm số và điểm rèn luyện, phản ánh sự biến động thực tế trong quá trình học tập của sinh viên.

Dựa trên quy chế đào tạo giả định của nhà trường, hệ thống được cấu hình các ngưỡng đánh giá và cảnh báo như sau:

- GPA học kỳ < 2.0: sinh viên thuộc diện cảnh báo học vụ.
- GPA tích lũy < 1.8: sinh viên thuộc diện cảnh báo học vụ mức cao.
- Điểm rèn luyện < 50: sinh viên thuộc diện cảnh báo rèn luyện.

Các ngưỡng này được cấu hình linh hoạt trong hệ thống, cho phép điều chỉnh khi quy định đào tạo thay đổi. Việc tách riêng logic ngưỡng cảnh báo

khỏi dữ liệu đầu vào giúp hệ thống dễ dàng mở rộng và tái sử dụng trong các bối cảnh khác nhau.

Quy trình thử nghiệm được thực hiện theo các bước sau:

- Thứ nhất, dữ liệu từ các tệp đầu vào được hệ thống ETL trích xuất và lưu trữ tại tầng Data Lake nhằm đảm bảo lưu giữ dữ liệu gốc phục vụ kiểm tra và đối soát.
- Thứ hai, hệ thống tiến hành làm sạch và chuẩn hóa dữ liệu, bao gồm kiểm tra tính đầy đủ của dữ liệu, chuẩn hóa định dạng điểm số và loại bỏ các bản ghi không hợp lệ. Sau đó, dữ liệu được chuyển đổi và nạp vào Data Warehouse theo mô hình phân tích đã thiết kế.
- Thứ ba, hệ thống tự động tính toán các chỉ số đánh giá như GPA học kỳ, GPA tích lũy, số tín chỉ đạt và điểm rèn luyện. Trên cơ sở các chỉ số này, hệ thống xác định danh sách sinh viên thuộc diện cảnh báo theo từng học kỳ.
- Thứ tư, kết quả xử lý được hiển thị trên các dashboard tương ứng, đồng thời danh sách sinh viên cảnh báo được sử dụng làm đầu vào cho mô-đun cảnh báo và gửi email tự động.

Thông qua kịch bản này, hệ thống được kiểm tra khả năng xử lý dữ liệu lớn theo chiều thời gian (nhiều học kỳ), khả năng tổng hợp dữ liệu từ nhiều nguồn và khả năng cập nhật kết quả phân tích một cách nhất quán.

3.3.2. Đánh giá hiệu quả hỗ trợ CVHT

Kết quả thử nghiệm cho thấy hệ thống ETL Dashboard đáp ứng đầy đủ và hiệu quả các use case đã được đề ra trong giai đoạn phân tích yêu cầu, đồng thời thể hiện rõ ưu thế so với phương pháp quản lý và tổng hợp dữ liệu thủ công.

Hệ thống cho phép theo dõi kết quả học tập của sinh viên một cách toàn diện thông qua các dashboard trực quan. Việc tự động tính toán GPA học kỳ và GPA tích lũy giúp loại bỏ sai sót do tính toán thủ công, đồng thời đảm bảo tính nhất quán giữa các học kỳ. CVHT có thể nhanh chóng xác định các sinh viên có kết quả học tập thấp và theo dõi xu hướng biến động GPA theo thời gian.

Thông qua dashboard điểm rèn luyện, hệ thống hỗ trợ CVHT đánh giá song song cả hai khía cạnh học tập và rèn luyện của sinh viên. Việc kết hợp dữ liệu rèn luyện với dữ liệu học tập trong cùng một hệ thống giúp CVHT có cái nhìn toàn diện hơn về tình trạng sinh viên, từ đó đưa ra các biện pháp hỗ trợ phù hợp.

Kết quả thử nghiệm cho thấy hệ thống có khả năng phát hiện sớm và chính xác các trường hợp sinh viên thuộc diện cảnh báo học vụ và rèn luyện. Việc tự động hóa quy trình xác định danh sách cảnh báo và hiển thị kết quả trên dashboard giúp giảm đáng kể thời gian xử lý của CVHT. Đồng thời, mô-đun gửi email tự động đảm bảo thông tin cảnh báo được truyền đạt kịp thời đến sinh viên và các bên liên quan.

Một trong những điểm nổi bật của hệ thống là khả năng tổng hợp dữ liệu từ nhiều file, nhiều học kỳ và nhiều nguồn dữ liệu khác nhau. Hệ thống vẫn đảm bảo hiệu năng và tính nhất quán khi quy mô dữ liệu tăng lên. Điều này cho thấy hệ thống có tiềm năng mở rộng và triển khai trong môi trường thực tế với quy mô lớn hơn.

Từ kết quả thử nghiệm, có thể khẳng định rằng hệ thống ETL Dashboard không chỉ đáp ứng yêu cầu kỹ thuật mà còn mang lại giá trị thực tiễn rõ rệt trong công tác cố vấn học tập. Hệ thống giúp CVHT chuyển từ phương pháp xử lý dữ liệu thủ công sang phương pháp hỗ trợ ra quyết định dựa trên dữ liệu, góp phần nâng cao hiệu quả quản lý và chất lượng đào tạo tại các cơ sở giáo dục đại học.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Thông qua đề tài này, chúng tôi đã thành công trong việc thiết kế và xây dựng một hệ thống Data Pipeline hoàn chỉnh, tích hợp các công nghệ hiện đại để tự động hóa quy trình xử lý dữ liệu sinh viên từ giai đoạn thu thập cho đến trực quan hóa. Hệ thống được xây dựng theo kiến trúc Medallion (Bronze-Silver-Gold), cho phép quản lý dữ liệu một cách hiệu quả qua các tầng xử lý khác nhau.

Về khía cạnh tính tự động hóa, Apache Airflow đã chứng tỏ là một lựa chọn xuất sắc làm trung tâm orchestration. Nhờ đó, cố vấn học tập không còn phải can thiệp thủ công vào việc thu thập và xử lý dữ liệu. Dữ liệu điểm rèn luyện và học tập được cập nhật định kỳ theo lịch biểu được lên lịch, và luôn sẵn sàng trên dashboard để sử dụng. Điều này đáng kể tiết kiệm thời gian, giảm khả năng sai sót do lỗi con người, và cho phép đội ngũ cố vấn tập trung vào việc phân tích và can thiệp hỗ trợ sinh viên.

Về khía cạnh hiệu năng truy vấn, ClickHouse tỏ ra là lựa chọn phù hợp cho use case này. Với khối lượng dữ liệu sinh viên thường từ vài nghìn đến vài chục nghìn records mỗi học kỳ, ClickHouse có khả năng xử lý các truy vấn phân tích phức tạp trong vài milliseconds. Điều này đảm bảo dashboard tải nhanh ngay cả khi nhiều cố vấn truy cập đồng thời, cung cấp trải nghiệm người dùng mượt mà và đáp ứng kịp thời.

Về khía cạnh khả năng mở rộng (scalability), kiến trúc phân tầng rõ ràng của hệ thống cho phép dễ dàng scale khi số lượng sinh viên tăng lên hoặc cần thêm các nguồn dữ liệu mới (chẳng hạn dữ liệu điểm danh, kết quả thi, hoặc dữ liệu từ các LMS khác). MinIO làm data lake có khả năng lưu trữ lượng lớn dữ liệu lịch sử với chi phí thấp, đáp ứng được nhu cầu tăng trưởng trong tương lai.

Về khía cạnh trực quan hóa dữ liệu, Apache Superset cung cấp một bộ công cụ đầy đủ với các loại biểu đồ đa dạng (line chart, bar chart, pie chart, table, heatmap, v.v.) để đáp ứng nhu cầu phân tích khác nhau. Khả năng drill-down và tương tác của Superset cho phép cố vấn dễ dàng chuyển đổi giữa các mức độ chi tiết, từ cái nhìn tổng quan đến các chi tiết cụ thể về từng sinh viên, hỗ trợ quá trình ra quyết định hiệu quả hơn.

Tuy nhiên, đề tài này vẫn còn một số hạn chế cần được cải thiện trong tương lai. Thứ nhất, hệ thống hiện tại được thử nghiệm với dữ liệu giả lập (dummy data) thay vì dữ liệu thực từ các hệ thống quản lý học tập (LMS) hiện hành. Trong bước tiếp theo, cần thiết kế và triển khai các connector để kết nối trực tiếp với các LMS phổ biến như Moodle, Canvas, hoặc các hệ thống nội bộ của trường, đảm bảo dữ liệu được cập nhật real-time từ các nguồn chính thức.

Thứ hai, hiện tại hệ thống cung cấp chủ yếu các chức năng báo cáo và trực quan hóa mô tả. Để tăng giá trị hơn nữa, có thể mở rộng hệ thống bằng cách tích hợp các mô hình Machine Learning để dự báo rủi ro học vụ, phân nhóm sinh viên có nhu cầu can thiệp, hoặc cá nhân hóa lộ trình đào tạo. Nền tảng dữ liệu đã được xây dựng là tiền đề tốt cho các ứng dụng nâng cao này.

Thứ ba, bảo mật dữ liệu và quản lý quyền hạn là những yếu tố cần được chú ý thêm. Mặc dù dữ liệu sinh viên là thông tin nhạy cảm, hệ thống hiện tại cần được bổ sung thêm các cơ chế xác thực nâng cao (authentication), phân quyền chi tiết (fine-grained authorization), mã hóa dữ liệu (encryption), và log audit để đảm bảo tuân thủ các quy định về bảo vệ dữ liệu cá nhân.

Cuối cùng, về mặt triển khai tại các trường đại học khác, cần phát triển một tài liệu hướng dẫn chi tiết, một template cấu hình có thể tái sử dụng, và các công cụ để dễ dàng triển khai hệ thống này tại các cơ sở giáo dục khác nhau.

Đồng thời, cần đào tạo đội ngũ kỹ thuật tại các trường để họ có thể duy trì, mở rộng, và tối ưu hóa hệ thống theo nhu cầu riêng của từng cơ sở.

Tóm lại, đề tài này đã chứng minh được khả năng của một kiến trúc Data Pipeline hiện đại trong việc hỗ trợ công tác cố vấn học tập tại các trường đại học. Hệ thống không chỉ giải quyết được các bài toán quản lý dữ liệu hiện tại mà còn tạo nền tảng vững chắc cho các phát triển tiếp theo. Với việc tiếp tục cải thiện và mở rộng, chúng tôi tin rằng giải pháp này sẽ đóng góp tích cực vào quá trình chuyển đổi số trong giáo dục, nâng cao chất lượng quản lý và hỗ trợ sinh viên tại các trường đại học Việt Nam.

TÀI LIỆU THAM KHẢO

[1] T. V. Nguyen and D. T. Tran, “Digital transformation in higher education: Opportunities and challenges,” *J. Educ. Sci.*, vol. 3, pp. 45–56, 2021.

[2] L. H. Pham, “Thực trạng quản lý dữ liệu sinh viên tại các cơ sở giáo dục đại học Việt Nam,” *Tạp Chí Khoa Học Giáo Dục*, vol. 19, no. 4, pp. 112–120, 2022.

[3] A. T. Win and P. M. Rahman, “Learning Analytics in higher education: A systematic review and future directions,” *Int. J. Emerg. Technol. Learn.*, vol. 17, no. 8, pp. 23–41, 2022.

[4] C. S. Kim and J. Park, “Improving academic advising through data analytics systems,” *J. Educ. Technol. Soc.*, vol. 25, no. 2, pp. 54–67, 2022.

[5] S. Ferguson, “Learning analytics: Drivers, developments and challenges,” *Int. J. Technol. Enhanc. Learn.*, vol. 6, no. 2, pp. 121–139, 2020.

[6] S. Khalil and M. Ebner, “Learning analytics in MOOCs: A review of the literature,” *IEEE Trans. Learn. Technol.*, vol. 12, no. 3, pp. 389–402, 2021.

[7] A. Khine and S. Wang, “Data pipeline: A comprehensive review of techniques and applications,” *IEEE Access*, vol. 9, pp. 145628–145643, 2021.

[8] M. Stonebraker and U. Çetintemel, “Requirements for science data bases and SciDB,” presented at the Proc. 4th Biennial Conf. Innovative Data Syst. Res., 2009, pp. 173–184.

[9] J. Kreps, “The log: What every software engineer should know about real-time data’s unifying abstraction.”

[10] M. Kleppmann, *Designing Data-Intensive Applications*. Sebastopol, CA, USA: O’Reilly Media, 2017.

[11] R. Kimball and M. Ross, *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*. Indianapolis, IN, USA: Wiley, 2013.

[12] W. H. Inmon, *Building the Data Warehouse*. Indianapolis, IN, USA: Wiley, 2005.

[13] S. Sagiroglu and D. Sinanc, “Big data: A review,” presented at the Proc. Int. Conf. Collaboration Technol. Syst., San Diego, CA, USA, 2013, pp. 42–47.

[14] W. H. Inmon and D. Linstedt, *Data Architecture: A Primer for the Data Scientist*. Boston, MA, USA: Morgan Kaufmann, 2014.

[15] T. Akidau and et al., “The dataflow model: A practical approach to balancing correctness, latency, and cost in massive-scale, unbounded, out-of-order data processing,” *Proc VLDB Endow.*, vol. 8, no. 12, pp. 1792–1803, 2015.

[16] D. Siemens and R. S. J. d. Baker, “Learning analytics and educational data mining: Towards communication and collaboration,” presented at the Proc. 2nd Int. Conf. Learning Analytics Knowledge, Vancouver, BC, Canada, 2012, pp. 252–254.

[17] H. V. Jagadish and et al., “Big data and its technical challenges,” *Commun ACM*, vol. 57, no. 7, pp. 86–94, 2014.

[18] J. Dean and S. Ghemawat, “MapReduce: Simplified data processing on large clusters,” *Commun ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[19] C. Batini, C. Cappiello, C. Francalanci, and A. Maurino, “Methodologies for data quality assessment and improvement,” *ACM Comput Surv*, vol. 41, no. 3, pp. 1–52, 2009.

[20] M. Zaharia and et al., “Apache Spark: A unified engine for big data processing,” *Commun ACM*, vol. 59, no. 11, pp. 56–65, 2016.

[21] P. Zikopoulos, C. Eaton, D. DeRoos, T. Deutsch, and G. Lapis, *Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data*. New York, NY, USA: McGraw-Hill, 2012.

[22] D. P. Reis and R. Housley, “End-to-end data governance,” *IBM J Res Dev.*, vol. 61, no. 4/5, p. 12:1-12:11, 2017.

[23] Y. Simmhan, B. Plale, and D. Gannon, “A survey of data provenance in e-science,” *ACM SIGMOD Rec.*, vol. 34, no. 3, pp. 31–36, 2005.

[24] A. Thusoo and et al., “Data warehousing and analytics infrastructure at Facebook,” presented at the Proc. ACM SIGMOD Int. Conf. Management Data, Indianapolis, IN, USA, 2010, pp. 1013–1020.

[25] B. Daniel, “Big Data and analytics in higher education: Opportunities and challenges,” *Br. J Educ Technol*, vol. 46, no. 5, pp. 904–920, 2015.

[26] N. Marz, “How to beat the CAP theorem.” [Online]. Available: http://nathanmarz.com/blog/how-to-beat-the-cap-theorem.html

[27] J. Kreps, “Questioning the Lambda Architecture.” [Online]. Available: https://www.oreilly.com/radar/questioning-the-lambda-architecture/

[28] M. Armbrust and et al., “Lakehouse: A new generation of open platforms that unify data warehousing and advanced analytics,” presented at the Proc. 11th Biennial Conf. Innovative Data Syst. Res., 2021, pp. 1–8.

[29] J. Dixon, “Pentaho, Hadoop, and Data Lakes.” [Online]. Available: https://jamesdixon.wordpress.com/2010/10/14/pentaho-hadoop-and-data-lakes/

[30] W. H. Inmon, *Building the Data Warehouse*. Indianapolis, IN, USA: Wiley, 2005.

[31] M. Lamberti, *Data Pipelines with Apache Airflow*. Shelter Island, NY, USA: Manning Publications, 2021.

[32] A. S. Tanenbaum and M. Van Steen, *Distributed Systems: Principles and Paradigms*. Upper Saddle River, NJ, USA: Prentice Hall, 2007.

[33] Y. Collet, “LZ4: Extremely fast compression algorithm.” [Online]. Available: https://lz4.github.io/lz4/

[34] “Apache Superset documentation.” [Online]. Available: https://superset.apache.org/docs/intro