

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC BÁCH KHOA  
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH



**Đồ án chuyên ngành (CO4029)**

---

Hệ thống quản lý luận văn cho Đại học Bách Khoa

**Giảng viên hướng dẫn:** Lê Đình Thuận

**Lớp:** L18

Thành viên nhóm:

Bùi Thé Kỷ Cương – 2210412

Lý Vĩnh Thái – 2213104

Phùng Xương Cận – 2210348

Hồ Chí Minh, ngày 29, tháng 11, năm 2025

## Tóm tắt

Hệ thống Quản lý Luận văn Tốt nghiệp (TMS) được phát triển nhằm số hóa quy trình quản lý tại Khoa Khoa học và Kỹ thuật Máy tính, Đại học Bách Khoa thành phố Hồ Chí Minh, giải quyết các vấn đề thủ công rườm rà và thiếu đồng bộ trong hai môn học chính: Đồ án chuyên ngành (giai đoạn 1: lập kế hoạch, xây dựng hệ thống) và Luận văn tốt nghiệp (giai đoạn 2: hiện thực, bảo vệ). Lý do chọn đề tài xuất phát từ thực tế Khoa có hàng trăm sinh viên tốt nghiệp mỗi năm, gặp khó khăn trong theo dõi đề tài theo học kỳ, gán sinh viên, nộp bài và chấm điểm. Mục tiêu tổng quát là tạo nền tảng trực tuyến giảm thời gian hành chính, tăng minh bạch; mục tiêu cụ thể bao gồm tự động hóa gửi/duyệt đề tài, đánh giá giữa kỳ (Pass/Fail), chấm điểm cuối kỳ với cảnh báo lệch điểm, và lên lịch bảo vệ. Phương pháp nghiên cứu sử dụng elicitation để thu thập yêu cầu, phân tích thiết kế theo microservices, với công cụ Golang, Next.js, MongoDB, Redis và MinIO.

TMS sử dụng kiến trúc microservices với 6 service backend Golang (thesis, role, user, council, file, academic) dùng gRPC và MySQL; server gateway (Gin, GraphQL) tích hợp Redis/MinIO/MongoDB; backend workflow (Express, BullMQ) cho tác vụ ngầm; frontend Next.js/Vite; giám sát qua Grafana, Loki, Prometheus, Promtail, Elasticsearch. Quy trình bao gồm tạo học kỳ, gửi/duyệt đề tài (GVHD/GVBM), gán sinh viên (giáo vụ), nộp bài cuối kỳ (sinh viên), đánh giá giữa kỳ, chấm điểm cuối kỳ (GVHD/GVPB với draft/final), lên lịch bảo vệ (giáo vụ với hội đồng 3 vai trò), và chấm điểm bảo vệ (hội đồng). Các vấn đề như lệch điểm hay trùng lặp đề tài được tự động hóa.

Phân tích yêu cầu chức năng tập trung quản lý học kỳ, đề tài, bài nộp, đánh giá, lịch bảo vệ; phi chức năng bao gồm hiệu suất ( $\leq 3$  giây), bảo mật (RBAC, JWT, GDPR với xóa  $>5$  năm), mở rộng (đa khoa). Thiết kế sử dụng ERD với 18 thực thể (Topic trung tâm), schema MySQL quan hệ và MongoDB phi cấu trúc. Triển khai

backend Golang với Docker, frontend Next.js/Vite, giám sát Prometheus/Loki. Kiểm thử với k6-tests đạt giảm 70% thời gian thủ tục.

Kết luận, đề tài đạt mục tiêu số hóa, với bài học về tích hợp microservices. Hạn chế: thiếu AI đạo văn, giới hạn đa khoa. Hướng phát triển: tích hợp AI kiểm tra trùng lặp/đạo văn, mở rộng đa khoa, tối ưu hiệu suất/bảo mật qua sharding và MFA.

## Bảng phân công

Họ và Tên	Mã số sinh viên	Nhiệm vụ	Đóng góp (100%)
Bùi Thế Kỷ Cường	2210412	Phân tích nghiệp vụ, thiết kế hệ thống, báo cáo.	33.33%
Lý Vĩnh Thái	2213104	Lập trình, hiện thực, triển khai hệ thống.	33.33%
Phùng Xương Cận	2210348	Kiểm thử và bảo trì hệ thống.	33.33%

# Mục lục

<b>I. Mở đầu .....</b>	<b>12</b>
<b>1. Lý do chọn đề tài .....</b>	<b>12</b>
<b>1.1. Tình hình thực tế tại Khoa Khoa học và Kỹ thuật Máy tính .....</b>	<b>12</b>
<b>1.2. Nhu cầu số hóa quy trình quản lý luận văn tốt nghiệp .....</b>	<b>12</b>
<b>1.3. Mục tiêu của đề tài .....</b>	<b>13</b>
<b>2. Phạm vi đề tài .....</b>	<b>13</b>
<b>2.1. Giới hạn hệ thống .....</b>	<b>13</b>
<b>2.2. Các chức năng chính .....</b>	<b>14</b>
<b>3. Mục tiêu nghiên cứu.....</b>	<b>14</b>
<b>3.1. Mục tiêu tổng quát .....</b>	<b>14</b>
<b>3.2. Mục tiêu cụ thể .....</b>	<b>15</b>
<b>4. Phương pháp nghiên cứu .....</b>	<b>15</b>
<b>4.1. Phương pháp thu thập yêu cầu.....</b>	<b>15</b>
<b>4.2. Phương pháp phân tích và thiết kế .....</b>	<b>16</b>
<b>4.3. Công cụ và công nghệ sử dụng .....</b>	<b>16</b>
<b>5. Cấu trúc báo cáo .....</b>	<b>16</b>
<b>II. Tổng quan về hệ thống.....</b>	<b>17</b>
<b>1. Tổng quan về quy trình quản lý luận văn tốt nghiệp.....</b>	<b>17</b>
<b>1.1. Hai môn học chính: Đồ án chuyên ngành và Luận văn tốt nghiệp .....</b>	<b>17</b>
<b>1.2. Các giai đoạn chính trong quy trình .....</b>	<b>18</b>
<b>1.2.1. Giai đoạn 1: Đồ án chuyên ngành (lập kế hoạch và xây dựng) .....</b>	<b>18</b>
<b>1.2.2. Giai đoạn 2: Luận văn tốt nghiệp (hiện thực và bảo vệ).....</b>	<b>19</b>
<b>2. Các vấn đề hiện tại trong quy trình thủ công.....</b>	<b>19</b>
<b>2.1. Thủ tục rườm rà và thiếu đồng bộ .....</b>	<b>19</b>
<b>2.2. Khó khăn trong theo dõi và quản lý theo học kỳ.....</b>	<b>20</b>
<b>3. Tổng quan về công nghệ sử dụng trong hệ thống.....</b>	<b>20</b>
<b>3.1. Các service backend chính (Golang với gRPC và MySQL) .....</b>	<b>20</b>
<b>3.1.1. Service Thesis.....</b>	<b>21</b>

3.1.2. Service Role.....	21
3.1.3. Service User .....	21
3.1.4. Service Council .....	21
3.1.5. Service File.....	22
3.1.6. Service Academic.....	22
3.2. Server gateway (Golang với Gin, GraphQL, Redis, client gRPC, MinIO, MongoDB).....	22
3.3. Backend workflow (Express, BullMQ, Redis) .....	22
3.4. Frontend chính (Next.js) .....	23
3.5. Frontend admin local (Vite).....	23
3.6. Công cụ giám sát và lưu trữ.....	23
3.6.1. Grafana (UI monitoring).....	23
3.6.2. Loki (database logs).....	24
3.6.3. Prometheus (database metric) .....	24
3.6.4. Promtail (6 instances để bắn logs lên Loki) .....	24
3.6.5. Elasticsearch (search full text).....	24
3.7. Các cơ sở dữ liệu (10 instances: 6 MySQL, 1 MongoDB, 1 Redis, 1 MinIO) .....	24
4. Cấu trúc dự án tổng quát.....	25
4.1. Phân tích cấu trúc thư mục .....	25
4.2. Các thành phần chính: BE_core, k6-tests, plagiarism.....	27
<b>III. Phân tích yêu cầu.....</b>	<b>28</b>
<b>1. Phân tích yêu cầu người dùng .....</b>	<b>28</b>
<b>1.1. Các vai trò người dùng (stakeholders).....</b>	<b>28</b>
<b>1.1.1. Giáo vụ .....</b>	<b>28</b>
<b>1.1.2. Giảng viên hướng dẫn (GVHD).....</b>	<b>28</b>
<b>1.1.3. Giảng viên bộ môn (GVBM) .....</b>	<b>28</b>
<b>1.1.4. Giảng viên phản biện (GVPB) .....</b>	<b>29</b>
<b>1.1.5. Thành viên hội đồng bảo vệ .....</b>	<b>29</b>
<b>1.1.6. Sinh viên .....</b>	<b>29</b>
<b>1.2. Yêu cầu theo học kỳ (mã học kỳ, hiển thị thông tin phụ thuộc học kỳ) .....</b>	<b>29</b>

<b>2. Phân tích yêu cầu chức năng .....</b>	30
<b>2.1. Quản lý học kỳ và danh sách .....</b>	30
<b>2.1.1. Tạo và chọn học kỳ .....</b>	30
<b>2.1.2. Upload danh sách sinh viên, giảng viên .....</b>	30
<b>2.2. Quản lý đề tài.....</b>	31
<b>2.2.1. Gửi và duyệt đề tài .....</b>	31
<b>2.2.2. Kiểm tra trùng lặp đề tài .....</b>	31
<b>2.2.3. Gán sinh viên vào đề tài .....</b>	31
<b>2.3. Quản lý bài nộp .....</b>	32
<b>2.3.1. Nộp báo cáo cuối kỳ.....</b>	32
<b>2.3.2. Xem lịch sử nộp bài .....</b>	32
<b>2.4. Đánh giá và chấm điểm .....</b>	32
<b>2.4.1. Đánh giá giữa kỳ (Pass/Fail) .....</b>	32
<b>2.4.2. Chấm điểm cuối kỳ (GVHD và GVPB).....</b>	32
<b>2.4.3. Kiểm tra lệch điểm và cảnh báo .....</b>	33
<b>2.5. Quản lý lịch bảo vệ .....</b>	33
<b>2.5.1. Tạo hội đồng bảo vệ và phân công hội đồng bảo vệ .....</b>	33
<b>2.5.2. Xét duyệt hội đồng và lên lịch bảo vệ.....</b>	33
<b>2.5.3. Chấm điểm bảo vệ .....</b>	33
<b>3. Phân tích yêu cầu phi chức năng.....</b>	34
<b>3.1. Hiệu suất và dung lượng file .....</b>	34
<b>3.2. Bảo mật và quyền riêng tư.....</b>	34
<b>3.3. Khả năng mở rộng và tích hợp .....</b>	35
<b>IV. Thiết kế hệ thống .....</b>	36
<b>1. Thiết kế tổng thể.....</b>	36
<b>1.1. Kiến trúc hệ thống.....</b>	36
<b>1.1.1. Các service backend (thesis, role, user, council, file, academic) .....</b>	36
<b>1.1.2. Server gateway và tích hợp (GraphQL, gRPC, Redis, MinIO, MongoDB).....</b>	37
<b>1.1.3. Backend workflow cho tác vụ ngầm (Express, BullMQ) .....</b>	37

<b>1.1.4. Frontend (Next.js và Vite) .....</b>	<b>37</b>
<b>1.1.5. Công cụ giám sát (Grafana, Loki, Prometheus, Promtail, Elasticsearch) .....</b>	<b>38</b>
<b>1.2. Sơ đồ kiến trúc hệ thống .....</b>	<b>38</b>
<b>2. Thiết kế cơ sở dữ liệu .....</b>	<b>39</b>
<b>2.1. Mô hình dữ liệu (Entity-Relationship Diagram) .....</b>	<b>39</b>
<b>2.2. Cấu trúc cơ sở dữ liệu (6 MySQL, 1 MongoDB, 1 Redis, 1 MinIO) .....</b>	<b>45</b>
<b>2.3. Thiết kế schema và quan hệ .....</b>	<b>45</b>
<b>2.3.1 Phân hệ Tổ chức &amp; Người dùng (Organization &amp; Users) .....</b>	<b>46</b>
<b>2.3.1.1. Bảng Faculty (Khoa) .....</b>	<b>46</b>
<b>2.3.1.2. Bảng Major (Ngành học) .....</b>	<b>46</b>
<b>2.3.1.3. Bảng Student (Sinh viên) .....</b>	<b>46</b>
<b>2.3.1.4. Bảng Teacher (Giảng viên) .....</b>	<b>47</b>
<b>2.3.1.5. Bảng RoleSystem (Phân quyền) .....</b>	<b>47</b>
<b>2.3.2. Phân hệ Đề tài &amp; Hội đồng (Topic &amp; Council) .....</b>	<b>47</b>
<b>2.3.2.1 Bảng Semester (Học kỳ) .....</b>	<b>47</b>
<b>2.3.2.2. Bảng Topic (Đề tài gốc) .....</b>	<b>48</b>
<b>2.3.2.3. Bảng Council (Hội đồng bảo vệ) .....</b>	<b>48</b>
<b>2.3.2.4. Bảng Topic_council (Đợt bảo vệ đề tài) .....</b>	<b>48</b>
<b>2.3.2.5. Bảng Defence (Thành viên hội đồng) .....</b>	<b>49</b>
<b>2.3.3. Phân hệ Đăng ký &amp; Kết quả (Enrollment &amp; Grading) .....</b>	<b>49</b>
<b>2.3.3.1. Bảng Enrollment (Đăng ký) .....</b>	<b>49</b>
<b>2.3.3.2. Bảng Grade_defence (Điểm hội đồng) .....</b>	<b>50</b>
<b>2.3.3.3. Bảng Grade_review (Điểm phản biện) .....</b>	<b>50</b>
<b>2.3.3.4. Bảng Final (Tổng kết) .....</b>	<b>50</b>
<b>2.3.4. Bảng Hệ thống (System) .....</b>	<b>51</b>
<b>2.3.4.1. Bảng File (Quản lý tài liệu đa hình) .....</b>	<b>51</b>
<b>2.4. Thiết kế collection cho MongoDB .....</b>	<b>51</b>
<b>2.4.1. Collection cronjob .....</b>	<b>51</b>
<b>2.4.2. Collection minio .....</b>	<b>51</b>

<b>2.4.3. Collection services .....</b>	<b>52</b>
<b>2.4.4. Collection session .....</b>	<b>53</b>
<b>2.4.5. Collection users.....</b>	<b>54</b>
<b>2.4.6. Collection workflows .....</b>	<b>54</b>
<b>3. Thiết kế giao diện người dùng .....</b>	<b>55</b>
<b>3.1. Giao diện frontend chính (Next.js).....</b>	<b>55</b>
<b>3.2. Giao diện admin local (Vite) .....</b>	<b>77</b>
<b>3.3. Thiết kế Figma (Mockup &amp; Prototype).....</b>	<b>77</b>
<b>4. Thiết kế API và tích hợp .....</b>	<b>77</b>
<b>4.1. API backend (Golang, Gin, GraphQL).....</b>	<b>77</b>
<b>4.2. Tích hợp gRPC cho các service .....</b>	<b>77</b>
<b>4.3. Tích hợp Redis cho hàng đợi và cache .....</b>	<b>78</b>
<b>4.4. Tích hợp MinIO cho lưu trữ file.....</b>	<b>78</b>
<b>5.1. Backend workflow (Express, BullMQ) .....</b>	<b>78</b>
<b>5.2. Cronjob và tác vụ định kỳ .....</b>	<b>79</b>
<b>5.3. Giám sát với Grafana, Loki, Prometheus .....</b>	<b>79</b>
<b>V. Triển khai hệ thống.....</b>	<b>79</b>
<b>1. Môi trường triển khai .....</b>	<b>79</b>
<b>1.1. Cấu hình phần mềm .....</b>	<b>79</b>
<b>1.2. Thiết lập cơ sở dữ liệu và các Monitors (MySQL, MongoDB, Redis, MinIO, Grafana, Prometheus, Loki, Promtail) .....</b>	<b>80</b>
<b>2. Triển khai backend.....</b>	<b>80</b>
<b>2.1. Triển khai các service Golang (thesis, role, user, council, file, academic).....</b>	<b>80</b>
<b>2.2. Triển khai server gateway (Gin, GraphQL) .....</b>	<b>81</b>
<b>2.3. Triển khai backend workflow (Express, BullMQ) .....</b>	<b>81</b>
<b>3. Triển khai frontend .....</b>	<b>81</b>
<b>3.1. Triển khai Next.js cho giao diện chính .....</b>	<b>81</b>
<b>3.2. Triển khai Vite cho admin local.....</b>	<b>81</b>
<b>4. Triển khai giám sát và công cụ hỗ trợ.....</b>	<b>82</b>
<b>4.1. Thiết lập Grafana, Loki, Prometheus, Promtail.....</b>	<b>82</b>

<b>4.2. Tích hợp Elasticsearch cho tìm kiếm full text .....</b>	<b>82</b>
<b>VI. Kiểm thử và đánh giá .....</b>	<b>82</b>
<b>1. Kế hoạch kiểm thử .....</b>	<b>82</b>
<b>1.4. Kiểm thử hiệu suất và tải (Performance test).....</b>	<b>83</b>
<b>2. Giám sát hệ thống.....</b>	<b>84</b>
<b>2.1. Service Log.....</b>	<b>84</b>
<b>2.2. Service Performance .....</b>	<b>85</b>
<b>2.3. Service Errors.....</b>	<b>86</b>
<b>2.4. Service Metrics .....</b>	<b>87</b>
<b>3. Đánh giá kết quả .....</b>	<b>88</b>
<b>3.1. Đánh giá hiệu suất hệ thống.....</b>	<b>88</b>
<b>3.2. Đánh giá tính khả dụng và bảo mật .....</b>	<b>89</b>
<b>4. Các vấn đề phát sinh và giải pháp.....</b>	<b>90</b>
<b>4.1. Tối ưu hóa độ trễ tại các điểm truy cập dữ liệu lớn (High Latency Endpoints)....</b>	<b>90</b>
<b>4.2. Quản lý lưu trữ tài liệu tập trung và đa hình.....</b>	<b>90</b>
<b>4.3. Giám sát và Khoanh vùng lỗi trong kiến trúc Microservices .....</b>	<b>91</b>
<b>VII. Kết luận và hướng phát triển .....</b>	<b>91</b>
<b>1. Kết luận.....</b>	<b>91</b>
<b>1.1. Kết quả đạt được .....</b>	<b>91</b>
<b>1.2. Bài học kinh nghiệm.....</b>	<b>92</b>
<b>2. Hạn chế của đề tài .....</b>	<b>92</b>
<b>3. Hướng phát triển trong tương lai.....</b>	<b>93</b>
<b>3.1. Tích hợp thêm công nghệ (AI kiểm tra trùng lặp đề tài và đạo văn).....</b>	<b>93</b>
<b>3.2. Mở rộng cho các khoa khác .....</b>	<b>93</b>
<b>3.3. Tối ưu hóa hiệu suất và bảo mật .....</b>	<b>93</b>
<b>Tài liệu tham khảo .....</b>	<b>94</b>

## Mục lục hình ảnh

<b>Hình 1.</b> Sơ đồ cấu trúc hệ thống .....	38
<b>Hình 2.</b> Entity Relationship Diagram .....	39
<b>Hình 3.</b> Giao diện landing page .....	56
<b>Hình 4.</b> Giao diện đăng nhập .....	57
<b>Hình 5.</b> Dashboard: Quản lý Giảng viên.....	58
<b>Hình 6.</b> Dashboard: Quản lý Sinh viên .....	59
<b>Hình 7.</b> Giao diện Quản lý đê tài .....	60
<b>Hình 8.</b> Dashboard: Quản lý Hội đồng .....	61
<b>Hình 9.</b> Dashboard: Lịch bảo vệ .....	62
<b>Hình 10.</b> Dashboard: Giáo viên Bộ môn.....	63
<b>Hình 11.</b> Dashboard: Gửi đê tài .....	64
<b>Hình 12.</b> Dashboard: Đê tài đang hướng dẫn.....	65
<b>Hình 13.</b> Dashboard: Hội đồng chấm điểm.....	66
<b>Hình 14.</b> Dashboard: Lịch hội đồng bảo vệ .....	67
<b>Hình 15.</b> Dashboard: Quản lý học kỳ .....	68
<b>Hình 16.</b> Dashboard: Quản lý người dùng .....	70
<b>Hình 17.</b> Dashboard: Quản lý đê tài .....	71
<b>Hình 18.</b> Dashboard: Quản lý lịch Bảo vệ .....	72
<b>Hình 19.</b> Dashboard: Quản lý Hội đồng .....	73
<b>Hình 20.</b> Dashboard: Quản lý Chuyên ngành .....	74
<b>Hình 21.</b> Dashboard: Quản lý Khoa.....	75
<b>Hình 22.</b> Dashboard: Phân tích học kỳ .....	76
<b>Hình 23.</b> Kết quả Load/Stress Test (1).....	83
<b>Hình 24.</b> Kết quả Load/Stress test (2).....	83
<b>Hình 25.</b> Dashboard Grafana: Services Logs.....	84
<b>Hình 26.</b> Dashboard Grafana: Services Performance.....	85
<b>Hình 27.</b> Grafana Dashboard: Services Errors.....	86
<b>Hình 28.</b> Grafana Dashboard: Services Metrics.....	87

## **I. Mở đầu**

### **1. Lý do chọn đề tài**

#### **1.1. Tình hình thực tế tại Khoa Khoa học và Kỹ thuật Máy tính**

Khoa Khoa học và Kỹ thuật Máy tính (CSE) tại Đại học Bách Khoa Thành phố Hồ Chí Minh là một trong những đơn vị dẫn đầu trong đào tạo và nghiên cứu lĩnh vực công nghệ thông tin tại Việt Nam. Với số lượng sinh viên lớn (hàng trăm sinh viên tốt nghiệp mỗi năm), quy trình quản lý luận văn tốt nghiệp đang gặp nhiều thách thức. Hiện nay, quy trình vẫn chủ yếu dựa vào các thủ tục thủ công như gửi email, sử dụng bảng tính Excel để theo dõi đề tài, và lưu trữ giấy tờ vật lý. Điều này dẫn đến tình trạng thiếu đồng bộ giữa các bên liên quan: giảng viên hướng dẫn (GVHD), giảng viên bộ môn (GVBM), giảng viên phản biện (GVPB), hội đồng bảo vệ, giáo vụ, và sinh viên.

Cụ thể, trong hai môn học chính là Đồ án chuyên ngành (giai đoạn 1) và Luận văn tốt nghiệp (giai đoạn 2), việc theo dõi đề tài theo học kỳ (mã học kỳ như 251 cho học kỳ 1 năm 2025) thường gặp lỗi do thiếu công cụ tự động. Danh sách sinh viên, giảng viên, và hội đồng phải được upload thủ công mỗi học kỳ, dẫn đến sai sót trong gán sinh viên vào đề tài, kiểm tra trùng lặp đề tài, và lên lịch bảo vệ. Ngoài ra, việc chấm điểm (giai đoạn giữa kỳ: Pass/Fail; cuối kỳ: điểm GVHD, GVPB, hội đồng) và kiểm tra lệch điểm ( $>2$  điểm) cũng không được hỗ trợ tự động, gây chậm trễ và thiếu minh bạch. Tình hình này không chỉ làm giảm hiệu quả mà còn ảnh hưởng đến chất lượng đào tạo và nghiên cứu tại Khoa.

#### **1.2. Nhu cầu số hóa quy trình quản lý luận văn tốt nghiệp**

Trong bối cảnh chuyển đổi số tại các trường đại học, nhu cầu số hóa quy trình quản lý luận văn tốt nghiệp trở nên cấp thiết. Hệ thống hiện tại thiếu sự đồng bộ, dẫn đến các vấn đề như:

- Thủ tục rườm rà: Sinh viên nộp bài cuối kỳ qua email hoặc giấy tờ, giảng viên chấm điểm thủ công, và giáo vụ quản lý danh sách bằng file Excel.

- Thiếu tự động hóa: Không có công cụ kiểm tra trùng lặp đề tài, cảnh báo lệch điểm, hoặc tự động cập nhật trạng thái đề tài (Chờ duyệt, Đã duyệt, Đang thực hiện, Đã kết thúc, Được bảo vệ, v.v.).
- Quản lý theo học kỳ: Thông tin (đề tài, danh sách sinh viên/giảng viên/hội đồng) phụ thuộc vào học kỳ, nhưng không có giao diện chọn học kỳ thông minh.
- Lưu trữ và bảo mật: File báo cáo (dung lượng  $\leq 200\text{MB}$ , bất kỳ định dạng) và form chấm điểm cần được lưu trữ an toàn, với quyền truy cập theo vai trò.

Số hóa sẽ giúp giảm thời gian xử lý, tăng tính minh bạch, và hỗ trợ các bên liên quan (GVHD, GVBM, GVPB, hội đồng, giáo vụ, sinh viên) thực hiện nghiệp vụ hiệu quả hơn, đồng thời tuân thủ quy định của Khoa.

### **1.3. Mục tiêu của đề tài**

Mục tiêu chính của đề tài là xây dựng hệ thống TMS để số hóa quy trình quản lý luận văn tốt nghiệp tại Khoa CSE.

Mục tiêu tổng quát: Tạo một nền tảng trực tuyến hỗ trợ toàn bộ quy trình từ gửi đề tài đến chấm điểm bảo vệ, giảm thủ tục thủ công và tăng hiệu quả quản lý.

Mục tiêu cụ thể:

- Hỗ trợ quản lý theo học kỳ, với mã học kỳ (ví dụ: 251) và hiển thị thông tin phụ thuộc học kỳ.
- Tự động hóa việc gửi, duyệt đề tài, gán sinh viên, nộp bài, đánh giá, chấm điểm, và lên lịch bảo vệ.
- Đảm bảo tính bảo mật, minh bạch, và khả năng mở rộng cho các khoa khác.

## **2. Phạm vi đề tài**

### **2.1. Giới hạn hệ thống**

Hệ thống TMS tập trung vào quy trình quản lý luận văn tốt nghiệp tại Khoa CSE, với giới hạn sau:

- Chỉ hỗ trợ hai môn học: Đồ án chuyên ngành (giai đoạn 1) và Luận văn tốt nghiệp (giai đoạn 2).
- Không bao gồm tích hợp với hệ thống đăng ký môn học hoặc thanh toán học phí của trường.
- Lưu trữ file (báo cáo, form chấm điểm) giới hạn dung lượng  $\leq 200\text{MB}$ , định dạng bất kỳ.
- Không liên kết được trực tiếp với tài khoản sinh viên/giảng viên của trường Đại học Bách Khoa thành phố Hồ Chí Minh.

## 2.2. Các chức năng chính

- Quản lý học kỳ: Tạo và chọn học kỳ, upload danh sách sinh viên/giảng viên/hội đồng.
- Quản lý đề tài: Gửi/duyệt đề tài, kiểm tra trùng lặp, gán sinh viên, theo dõi trạng thái.
- Quản lý bài nộp: Sinh viên nộp báo cáo cuối kỳ, xem lịch sử, nhận xét từ GVHD.
- Đánh giá và chấm điểm: Đánh giá giữa kỳ (Pass/Fail), chấm điểm cuối kỳ (GVHD, GVPB), kiểm tra lệch điểm.
- Quản lý lịch bảo vệ: Upload file lịch, phân công hội đồng (Chủ tịch, Ủy viên, Thư ký), chấm điểm bảo vệ.
- Báo cáo: Xuất tài liệu báo cáo của các hạng mục liên quan.

## 3. Mục tiêu nghiên cứu

### 3.1. Mục tiêu tổng quát

Mục tiêu tổng quát của đề tài là xây dựng một hệ thống quản lý luận văn tốt nghiệp toàn diện, nhằm số hóa và tối ưu hóa quy trình quản lý tại Khoa Khoa học và Kỹ thuật Máy tính, Đại học Bách Khoa thành phố Hồ Chí Minh. Hệ thống sẽ hỗ trợ đồng bộ hóa các hoạt động từ gửi đề tài, duyệt, gán sinh viên, nộp bài, đánh giá, đến bảo vệ luận văn, giảm thiểu thủ tục thủ công và tăng tính minh bạch, hiệu quả cho tất cả các bên liên

quan bao gồm giáo vụ, giảng viên hướng dẫn (GVHD), giảng viên bộ môn (GVBM), giảng viên phản biện (GVPB), hội đồng bảo vệ và sinh viên.

### **3.2. Mục tiêu cụ thể**

Xây dựng hệ thống hỗ trợ quản lý theo học kỳ, với mã học kỳ (ví dụ: 251 cho học kỳ 1 năm 2025), đảm bảo tất cả thông tin (đề tài, danh sách sinh viên/giảng viên/hội đồng) phụ thuộc vào học kỳ được chọn.

Phát triển chức năng gửi, duyệt đề tài (bao gồm kiểm tra trùng lặp tự động) và gán sinh viên vào đề tài đã duyệt.

Hỗ trợ nộp báo cáo cuối kỳ (dung lượng  $\leq 200\text{MB}$ ), đánh giá giữa kỳ (Pass/Fail), chấm điểm cuối kỳ (GVHD, GVPB với kiểm tra lệch điểm  $\geq 2$ ), và chấm điểm bảo vệ (hội đồng).

Tích hợp công cụ giám sát và lưu trữ (Grafana, Loki, Prometheus, Elasticsearch) để đảm bảo hiệu suất và bảo mật.

Đảm bảo hệ thống mở rộng, hỗ trợ các công nghệ hiện đại như Golang, gRPC, GraphQL, Redis, MinIO, MongoDB, MySQL, Express, BullMQ, Next.js và Vite.

## **4. Phương pháp nghiên cứu**

### **4.1. Phương pháp thu thập yêu cầu**

Phương pháp thu thập yêu cầu được thực hiện thông qua quá trình Elicitation, bao gồm:

Phỏng vấn và khảo sát các bên liên quan (giáo vụ, GVHD, GVBM, GVPB, hội đồng bảo vệ, sinh viên) để hiểu rõ quy trình hiện tại và các vấn đề tồn tại.

Phân tích tài liệu quy định của Khoa (quy trình quản lý luận văn, mẫu form chấm điểm của từng bên liên quan, lịch bảo vệ).

Sử dụng kỹ thuật brainstorming và use case modeling để xác định các chức năng chính và luồng nghiệp vụ.

## **4.2. Phương pháp phân tích và thiết kế**

Phân tích: Sử dụng UML (sơ đồ ca sử dụng, activity diagram) và ma trận CRUD để phân tích yêu cầu chức năng và phi chức năng.

Thiết kế: Áp dụng mô hình microservices với 6 service backend (thesis, role, user, council, file, academic) sử dụng Golang và gRPC; server gateway với Gin và GraphQL; backend workflow với Express và BullMQ; frontend với Next.js và Vite. Cơ sở dữ liệu sử dụng 6 MySQL, 1 MongoDB, 1 Redis, 1 MinIO; giám sát với Grafana, Loki, Prometheus, Elasticsearch.

## **4.3. Công cụ và công nghệ sử dụng**

Backend: Golang với gRPC cho 6 service (thesis, role, user, council, file, academic); Gin, GraphQL cho server gateway; Express, BullMQ cho backend workflow.

Cơ sở dữ liệu: 6 MySQL, 1 MongoDB, 1 Redis, 1 MinIO.

Frontend: Next.js (giao diện chính), Vite (admin local).

Giám sát: Grafana (UI monitoring), Loki (logs), Prometheus (metrics), 6 Promtail (bản logs), Elasticsearch (search full text).

Công cụ phát triển: Docker, Kubernetes cho triển khai; k6-tests cho kiểm thử tải; Git cho quản lý mã nguồn.

## **5. Cấu trúc báo cáo**

Báo cáo được cấu trúc như sau để trình bày rõ ràng, logic và đầy đủ nội dung đề tài:

Chương I. Mở đầu: Giới thiệu lý do chọn đề tài, phạm vi, mục tiêu nghiên cứu, phương pháp nghiên cứu và cấu trúc báo cáo.

Chương II. Tổng quan về hệ thống: Phân tích quy trình quản lý luận văn, vấn đề hiện tại và tổng quan công nghệ sử dụng.

Chương III. Phân tích yêu cầu: Phân tích yêu cầu người dùng, chức năng, phi chức năng, rủi ro và giả định.

Chương IV. Thiết kế hệ thống: Thiết kế tổng thể, cơ sở dữ liệu, giao diện người dùng, API và quy trình ngầm.

Chương V. Triển khai hệ thống: Môi trường triển khai, backend, frontend, giám sát và kiểm tra ban đầu.

Chương VI. Kiểm thử và đánh giá: Kế hoạch kiểm thử, công cụ, đánh giá kết quả và giải pháp vấn đề.

Chương VII. Kết luận và hướng phát triển: Kết luận, hạn chế và hướng mở rộng.

Tài liệu tham khảo: Danh sách tài liệu.

Phụ lục: Bảng biểu, hình ảnh, cấu trúc dự án chi tiết.

## **II. Tổng quan về hệ thống**

### **1. Tổng quan về quy trình quản lý luận văn tốt nghiệp**

#### **1.1. Hai môn học chính: Đồ án chuyên ngành và Luận văn tốt nghiệp**

Quy trình quản lý luận văn tốt nghiệp tại Khoa Khoa học và Kỹ thuật Máy tính được xây dựng quanh hai môn học chính: Đồ án chuyên ngành và Luận văn tốt nghiệp. Đây là hai môn học bắt buộc, được thiết kế để hỗ trợ sinh viên thực hiện và hoàn thành luận văn tốt nghiệp một cách có hệ thống.

Đồ án chuyên ngành (giai đoạn 1): Đây là môn học tập trung vào việc lập kế hoạch và xây dựng nền tảng cho đề tài luận văn. Sinh viên sẽ nghiên cứu, đề xuất ý tưởng, và phát triển khung sườn hệ thống, bao gồm phân tích yêu cầu, thiết kế mô hình, và lập kế hoạch triển khai. Môn học này nhằm đảm bảo đề tài được định hình rõ ràng trước khi bước vào giai đoạn hiện thực hóa.

Luận văn tốt nghiệp (giai đoạn 2): Đây là môn học tập trung vào việc hiện thực hệ thống và bảo vệ luận văn. Sinh viên sẽ triển khai các giải pháp đã lập kế hoạch ở giai

đoạn 1, hoàn thiện sản phẩm, và chuẩn bị cho buổi bảo vệ trước hội đồng. Môn học này nhấn mạnh vào kỹ năng thực hành, đánh giá kết quả, và bảo vệ ý tưởng

Hai môn học này được liên kết chặt chẽ, với kết quả của Đề án chuyên ngành làm nền tảng cho Luận văn tốt nghiệp, đảm bảo quy trình liên tục và hiệu quả.

## **1.2. Các giai đoạn chính trong quy trình**

Quy trình quản lý luận văn tốt nghiệp được chia thành hai giai đoạn chính, tương ứng với hai môn học, nhằm hỗ trợ sinh viên từ ý tưởng đến hoàn thành sản phẩm.

### **1.2.1. Giai đoạn 1: Đề án chuyên ngành (lập kế hoạch và xây dựng)**

Giai đoạn này tập trung vào việc chuẩn bị và xây dựng nền tảng cho đề tài. Các hoạt động chính bao gồm:

**Gửi và duyệt đề tài:** Giảng viên hướng dẫn (GVHD) gửi đề tài mới hoặc duyệt đề tài cũ từ các học kỳ trước. Đề tài bao gồm thông tin như giai đoạn (1 hoặc 2), tên đề tài (tiếng Việt và tiếng Anh), mã cán bộ GVHD (ví dụ: CB212), chuyên môn (A, B, C), thời gian bắt đầu/kết thúc, mã học kỳ (ví dụ: [CQ - HK251]), và form đăng ký đề tài.

**Duyệt đề tài:** Giảng viên bộ môn (GVBM) duyệt hoặc từ chối đề tài, hệ thống tự động kiểm tra trùng lặp.

**Gán sinh viên:** Giáo vụ gán sinh viên từ danh sách upload vào đề tài đã duyệt.

**Đánh giá giữa kỳ:** GVHD đánh giá Pass/Fail cho đề tài mà không cần nộp bài.

**Chấm điểm cuối kỳ:** GVHD chấm điểm cho từng sinh viên, tải file chấm điểm, và lưu Draft/Final.

Giai đoạn này đảm bảo đề tài được lập kế hoạch vững chắc trước khi chuyển sang hiện thực hóa.

### **1.2.2. Giai đoạn 2: Luận văn tốt nghiệp (hiện thực và bảo vệ)**

Giai đoạn này tập trung vào việc hiện thực hóa hệ thống và hoàn tất luận văn.

Các hoạt động chính bao gồm:

Phản biện và chấm điểm cuối kỳ: GVPB (phân công bởi giáo vụ) phản biện, chấm điểm cho từng sinh viên, tải file chấm điểm và ABET, chọn trạng thái (Được bảo vệ nếu điểm  $\geq 5.5$ , Không được bảo vệ, hoặc Bổ sung). GVHD cũng chấm điểm tương tự. Hệ thống kiểm tra lệch điểm  $> 2$  để cảnh báo giáo vụ.

Lên lịch bảo vệ: Giáo vụ upload file lịch, hệ thống tự lọc thông tin (ngày, giờ, phòng), và phân công hội đồng (Chủ tịch, Ủy viên, Thư ký, tối thiểu 3 thành viên).

Chấm điểm bảo vệ: Hội đồng chấm điểm cho từng sinh viên, tải file chấm điểm.

Cập nhật điểm cuối kỳ: Điểm trước bảo vệ (trung bình GVHD + GVPB), điểm sau bảo vệ (điểm hội đồng).

Giai đoạn này đảm bảo luận văn được hoàn thiện và bảo vệ thành công.

## **2. Các vấn đề hiện tại trong quy trình thủ công**

### **2.1. Thủ tục rườm rà và thiếu đồng bộ**

Quy trình quản lý luận văn hiện tại chủ yếu dựa vào thủ công, dẫn đến nhiều vấn đề:

Thủ tục rườm rà: Gửi đề tài, nộp bài, chấm điểm qua email hoặc giấy tờ, gây mất thời gian và dễ sai sót. Ví dụ, GVHD phải kiểm tra thủ công đề tài trùng lặp, giáo vụ quản lý danh sách bằng Excel, dẫn đến lỗi gán sinh viên hoặc phân công hội đồng.

Thiếu đồng bộ: Thông tin không được cập nhật thời gian thực, dẫn đến tình trạng GVBM duyệt đề tài chậm, sinh viên không biết trạng thái đề tài, hoặc hội đồng không truy cập được file chấm điểm kịp thời. Việc chấm điểm (Draft/Final) và kiểm tra lệch điểm cũng không tự động, gây chậm trễ.

Những vấn đề này làm giảm hiệu quả, tăng gánh nặng hành chính, và ảnh hưởng đến chất lượng đào tạo.

## **2.2. Khó khăn trong theo dõi và quản lý theo học kỳ**

Quản lý theo học kỳ: Danh sách sinh viên, giảng viên, hội đồng phụ thuộc vào học kỳ, nhưng không có hệ thống chọn học kỳ thông minh, dẫn đến khó theo dõi đề tài theo mã học kỳ (ví dụ: [CQ - HK251]).

Theo dõi trạng thái: Trạng thái đề tài (Chờ duyệt, Đã duyệt, Đang thực hiện, Đã kết thúc) không được tự động cập nhật, gây khó khăn cho sinh viên và giảng viên theo dõi tiến độ, đặc biệt khi đề tài kéo dài qua hai giai đoạn.

Lưu trữ và truy cập: File báo cáo và form chấm điểm lưu thủ công, khó truy cập lịch sử nộp bài hoặc chấm điểm, dẫn đến mất dữ liệu hoặc thiếu minh bạch.

Những khó khăn này nhấn mạnh nhu cầu một hệ thống số hóa để đồng bộ và tự động hóa quy trình.

## **3. Tổng quan về công nghệ sử dụng trong hệ thống**

Hệ thống Quản lý Luận văn Tốt nghiệp (TMS) được xây dựng dựa trên một kiến trúc microservices hiện đại, kết hợp các công nghệ tiên tiến để đảm bảo tính linh hoạt, hiệu suất cao và khả năng mở rộng. Hệ thống bao gồm các service backend chính được viết bằng GoLang, sử dụng gRPC cho giao tiếp nội bộ và MySQL làm cơ sở dữ liệu chính. Bên cạnh đó, có server gateway trung tâm, backend workflow cho các tác vụ ngầm, frontend chính và admin local, cùng các công cụ giám sát và lưu trữ dữ liệu. Tổng cộng, hệ thống sử dụng 10 instance cơ sở dữ liệu để hỗ trợ đa dạng nhu cầu lưu trữ và truy vấn. Dưới đây là tổng quan chi tiết về các thành phần công nghệ.

### **3.1. Các service backend chính (GoLang với gRPC và MySQL)**

Hệ thống backend được thiết kế theo kiến trúc microservices, với 6 service chính được phát triển bằng ngôn ngữ GoLang. Mỗi service chịu trách nhiệm cho một phần nghiệp vụ cụ thể, sử dụng gRPC để giao tiếp nội bộ hiệu quả và an toàn, đồng thời lưu trữ dữ liệu trong các instance MySQL riêng biệt để đảm bảo tính độc lập và dễ dàng

mở rộng. Các service này được chứa trong thư mục BE\_core của dự án, với cấu trúc mã nguồn bao gồm controllers, routes, database models và middleware.

### **3.1.1. Service Thesis**

Service Thesis chịu trách nhiệm quản lý các đề tài luận văn, bao gồm việc gửi, duyệt, gán sinh viên và theo dõi trạng thái đề tài. Service này sử dụng Golang để xử lý logic nghiệp vụ như kiểm tra trùng lặp đề tài và cập nhật trạng thái tự động. Dữ liệu được lưu trong MySQL với các bảng liên quan đến đề tài, học kỳ và trạng thái. Trong thư mục BE\_core, service này có các file như thesis.controller.js và thesis.model.js để xử lý API và mô hình dữ liệu.

### **3.1.2. Service Role**

Service Role quản lý các vai trò người dùng trong hệ thống, bao gồm việc tạo, cập nhật và phân quyền (RBAC). Sử dụng Golang và gRPC để đồng bộ vai trò giữa các service khác, dữ liệu vai trò được lưu trong MySQL để dễ dàng truy vấn. Service này hỗ trợ các tính năng như gán vai trò cho người dùng và kiểm tra quyền truy cập. Trong cấu trúc dự án, nó được triển khai qua role.controller.js và role.model.js.

### **3.1.3. Service User**

Service User xử lý việc quản lý thông tin người dùng, bao gồm tạo, cập nhật, xóa và xác thực người dùng (sinh viên, giảng viên, giáo vụ). Service này tích hợp với Google OAuth cho đăng nhập và sử dụng MySQL để lưu trữ thông tin cá nhân, mã số và vai trò. Trong BE\_core, service này bao gồm user.controller.js và user.model.js, hỗ trợ các API như seed-admin.js để khởi tạo tài khoản admin ban đầu.

### **3.1.4. Service Council**

Service Council quản lý hội đồng bảo vệ luận văn, bao gồm phân công thành viên hội đồng (Chủ tịch, Ủy viên, Thư ký) và chấm điểm bảo vệ. Sử dụng Golang để xử lý logic phân công và tính điểm trung bình, dữ liệu được lưu trong MySQL. Service này giao tiếp với service Thesis qua gRPC để liên kết với đề tài. Trong dự án, nó được triển khai qua council.controller.js và council.model.js.

### **3.1.5. Service File**

Service File chịu trách nhiệm quản lý file trong hệ thống, bao gồm upload, lưu trữ và tải xuống file báo cáo, form chấm điểm. Service này sử dụng Golang và tích hợp với MinIO cho lưu trữ object, dữ liệu metadata file được lưu trong MySQL. Trong BE\_core, service này bao gồm minio.controller.js và minio.model.js, hỗ trợ các tính năng như pdf-generator.js để tạo file PDF báo cáo.

### **3.1.6. Service Academic**

Service Academic quản lý các thông tin học thuật như học kỳ, danh sách sinh viên/giảng viên, và quản lý giai đoạn đề tài (giai đoạn 1: Đồ án chuyên ngành, giai đoạn 2: Luận văn tốt nghiệp). Sử dụng Golang để xử lý upload danh sách và chọn học kỳ, dữ liệu được lưu trong MySQL để hỗ trợ truy vấn theo học kỳ. Trong cấu trúc dự án, service này được triển khai qua academic.mutations.ts và academic.queries.ts trong thư mục graphql.

## **3.2. Server gateway (Golang với Gin, GraphQL, Redis, client gRPC, MinIO, MongoDB)**

Server gateway là thành phần trung tâm của hệ thống, được phát triển bằng Golang sử dụng framework Gin để xử lý các yêu cầu HTTP. Server này tích hợp GraphQL để cung cấp API linh hoạt, hỗ trợ lọc, phân trang và sắp xếp. Nó sử dụng Redis làm hàng đợi cho các tác vụ bất đồng bộ (như kiểm tra đạo văn và gửi email), client gRPC để giao tiếp với 6 service backend chính, MinIO để lưu trữ file, và MongoDB để lưu trữ log sự kiện và dữ liệu phi cấu trúc (như event\_logs). Trong thư mục BE\_core, server này bao gồm các routes (auth.routes.js, cronjobs.routes.js) và middleware (auth.middleware.js) để xử lý xác thực và phân quyền. Server gateway đóng vai trò làm cầu nối giữa frontend và các service, đảm bảo tính nhất quán và bảo mật.

## **3.3. Backend workflow (Express, BullMQ, Redis)**

Backend workflow là thành phần phụ xử lý các tác vụ ngầm, được phát triển bằng Express.js để quản lý quy trình bất đồng bộ. Sử dụng BullMQ làm thư viện hàng đợi dựa trên Redis để xử lý các job như kiểm tra đạo văn, gửi email cảnh báo lệch điểm, và

cập nhật trạng thái để tài tự động. Trong cấu trúc dự án, backend này nằm trong thư mục queue, với các file như cronjob-init.js và minio.service.js để khởi tạo và xử lý job. Backend workflow đảm bảo các tác vụ nặng (như chấm điểm hoặc kiểm tra trùng lặp) được thực hiện mà không làm gián đoạn hệ thống chính.

### **3.4. Frontend chính (Next.js)**

Frontend chính được phát triển bằng Next.js, một framework React để xây dựng giao diện người dùng động và responsive. Frontend hỗ trợ các chức năng như chọn học kỳ, gửi/duyệt đề tài, nộp bài, chấm điểm, và xuất báo cáo. Trong thư mục FE\_core, frontend bao gồm các component như auth, council, dashboard, và topic, với tích hợp GraphQL qua Apollo Client để gọi API từ server gateway. Frontend chính được sử dụng bởi tất cả người dùng (sinh viên, giảng viên, giáo vụ) và hỗ trợ tìm kiếm full text qua Elasticsearch.

### **3.5. Frontend admin local (Vite)**

Frontend admin local được phát triển bằng Vite (framework React nhẹ), dành riêng cho quản trị viên để giám sát các tác vụ ngầm như queue Redis và log sự kiện. Giao diện này cho phép xem trạng thái job, monitor cronjob, và quản lý backend workflow. Trong cấu trúc dự án, nó nằm trong thư mục admin, với các component như cronjob và workflow, tích hợp với Grafana và Loki để hiển thị UI monitoring.

### **3.6. Công cụ giám sát và lưu trữ**

Hệ thống sử dụng các công cụ chuyên dụng để giám sát hiệu suất, lưu trữ logs và metric, đảm bảo vận hành ổn định.

#### **3.6.1. Grafana (UI monitoring)**

Grafana là giao diện đồ họa để giám sát hệ thống, hiển thị dashboard về metric hiệu suất (CPU, RAM, truy vấn) từ Prometheus và logs từ Loki. Nó tích hợp với frontend admin local (Vite) để quản trị viên theo dõi thời gian thực.

### **3.6.2. Loki (database logs)**

Loki là cơ sở dữ liệu chuyên lưu trữ logs, hoạt động như một database logs để thu thập và truy vấn logs từ các service. Loki nhận logs từ Promtail và hỗ trợ tìm kiếm nhanh.

### **3.6.3. Prometheus (database metric)**

Prometheus là database metric để thu thập và lưu trữ các chỉ số hiệu suất hệ thống (response time, CPU usage, số lượng job Redis). Nó cung cấp dữ liệu cho Grafana để vẽ biểu đồ.

### **3.6.4. Promtail (6 instances để bắn logs lên Loki)**

Promtail là agent thu thập logs, với 6 instances chạy song song để bắn logs từ các service backend (thesis, role, user, council, file, academic) lên Loki. Mỗi instance chịu trách nhiệm cho một service để đảm bảo phân tải.

### **3.6.5. Elasticsearch (search full text)**

Elasticsearch là công cụ tìm kiếm full text, hỗ trợ tìm kiếm nhanh trong đề tài (title, description), bài nộp, và nhận xét. Nó tích hợp với frontend Next.js qua GraphQL để xử lý tìm kiếm theo mã số, họ tên hoặc nội dung.

## **3.7. Các cơ sở dữ liệu (10 instances: 6 MySQL, 1 MongoDB, 1 Redis, 1 MinIO)**

Hệ thống sử dụng 10 instance cơ sở dữ liệu để đảm bảo tính độc lập và hiệu suất. 6 instance MySQL dành cho các service backend chính (thesis, role, user, council, file, academic), mỗi service một instance để lưu trữ dữ liệu riêng (ví dụ: bảng đề tài trong MySQL của service Thesis). 1 instance MongoDB dùng để lưu trữ logs sự kiện và dữ liệu phi cấu trúc (event\_logs, archived\_theses). 1 instance Redis dùng làm hàng đợi cho backend workflow (BullMQ) và cache. 1 instance MinIO dùng làm lưu trữ object cho file (báo cáo, form chấm điểm), hỗ trợ tích hợp với Supabase nếu cần mở rộng. Các cơ sở dữ liệu được giám sát qua Prometheus và Loki để đảm bảo tính sẵn sàng.

## 4. Cấu trúc dự án tổng quát

### 4.1. Phân tích cấu trúc thư mục

Cấu trúc dự án được tổ chức một cách logic và module hóa, nhằm hỗ trợ phát triển, bảo trì và mở rộng hệ thống TMS. Dựa trên file project\_tree.txt, cấu trúc thư mục được phân chia thành các thành phần chính, phản ánh kiến trúc microservices với backend Golang, frontend Next.js/Vite, và các module phụ cho kiểm tra đạo văn và kiểm thử tải. Dưới đây là phân tích chi tiết cấu trúc thư mục, tập trung vào các thư mục gốc và con quan trọng:

- **Thư mục gốc (.)**: Chứa các thành phần chính của dự án, bao gồm BE\_core (backend chính), k6-tests (kiểm thử tải), plagiarism (kiểm tra đạo văn), và file project\_tree.txt (tóm tắt cấu trúc).
- **BE\_core**: Đây là thư mục backend chính, được phát triển bằng Golang, chứa mã nguồn cho server gateway và các service. Cấu trúc bên trong bao gồm:
  - **certs**: Chứa chứng chỉ bảo mật (ca.crt, ca.key, client.crt, v.v.) để hỗ trợ mã hóa và xác thực gRPC/HTTPS.
  - **dist**: Thư mục chứa mã biên dịch và build, với các controller (auth, cronjobs, minio, queues, services, workflows) và routes để xử lý API. Ví dụ, auth.controller.js xử lý xác thực JWT, minio.controller.js tích hợp lưu trữ file với MinIO.
  - **database**: Quản lý kết nối và mô hình dữ liệu, bao gồm connection.js (kết nối MongoDB/MySQL), models (cronjob.model.js, minio.model.js, service.model.js, user.model.js, workflow.model.js), và seeds (seed-minio-config.js) để khởi tạo dữ liệu ban đầu.
  - **main.js**: File chính để khởi chạy server gateway (Gin, GraphQL).
  - **middleware**: Chứa auth.middleware.js để kiểm tra quyền RBAC qua JWT.
  - **queue**: Quản lý hàng đợi và tác vụ ngầm, với cronjob (cronjob-init.js, cronjob-service.js), external (gRPC/http), internal, minio (pdf-generator.js), và test-pdf-gen.js để xử lý file PDF.
  - **tsconfig.json** và **yarn.lock**: Cấu hình TypeScript và quản lý gói dependency.

- **k6-tests:** Thư mục kiểm thử tải sử dụng K6, hỗ trợ kiểm tra hiệu suất hệ thống.  
Cấu trúc bao gồm:
  - **config:** File index.js chứa cấu hình kiểm thử.
  - **queries:** Các file query GraphQL cho admin, department, student, teacher.
  - **scenarios:** Các kịch bản kiểm thử tải cho từng vai trò.
  - **tests:** Các file kiểm thử như admin-load-test.js, combined-load-test.js, smoke-test.js, stress-test.js.
  - **utils:** graphql.js để hỗ trợ gọi API GraphQL trong kiểm thử.
- **Plagiarism (đang phát triển):** Module kiểm tra đạo văn, được triển khai bằng Python với Docker. Cấu trúc bao gồm:
  - **docker-compose.yml và Dockerfile:** Để container hóa module.
  - **docs:** ARCHITECTURE.md và TODO.md mô tả kiến trúc và nhiệm vụ còn lại.
  - **proto:** plagiarism.proto định nghĩa protobuf cho gRPC.
  - **scripts:** generate\_proto.sh (tạo code từ proto), setup\_es.py (cài đặt Elasticsearch), test\_client.py (kiểm thử client).
  - **src:** Mã nguồn chính, với config (settings.py), core (analyzer.py, chunker.py, detector.py, document\_manager.py), embedding (ollama\_embed.py), models, services (plagiarism\_service.py), và storage (elasticsearch.py).
  - **tests:** Các file kiểm thử (conftest.py, test\_chunker.py, test\_detector.py) sử dụng pytest.

Cấu trúc này đảm bảo tính module hóa, với BE\_core tập trung vào backend chính, k6-tests cho kiểm thử, và plagiarism cho tính năng chuyên biệt. Tổng thể, dự án hỗ trợ phát triển song song và dễ dàng tích hợp các công cụ như Redis, MinIO, và Elasticsearch.

## 4.2. Các thành phần chính: BE\_core, k6-tests, plagiarism

Hệ thống TMS được xây dựng với các thành phần chính, mỗi thành phần đảm nhận vai trò cụ thể để hỗ trợ quy trình quản lý luận văn. Dưới đây là phân tích chi tiết:

**BE\_core:** Đây là lõi backend của hệ thống, phát triển bằng Golang để xử lý các service chính (thesis, role, user, council, file, academic) thông qua gRPC và MySQL. Thành phần này bao gồm server gateway sử dụng Gin và GraphQL để xử lý yêu cầu từ frontend, tích hợp Redis cho hàng đợi, client gRPC để giao tiếp nội bộ, MinIO cho lưu trữ file, và MongoDB cho log sự kiện. BE\_core quản lý toàn bộ logic nghiệp vụ như xác thực JWT, quản lý đề tài, và chấm điểm, đảm bảo hiệu suất cao và bảo mật RBAC. Trong cấu trúc, nó chiếm phần lớn mã nguồn với các thư mục như api, database, middleware, và queue để hỗ trợ cronjob và tác vụ ngầm.

**k6-tests:** Thành phần này tập trung vào kiểm thử tải (load testing) sử dụng K6, giúp đánh giá hiệu suất hệ thống dưới tải trọng cao (ví dụ: 1000 người dùng đồng thời nộp bài). k6-tests bao gồm các scenario cho từng vai trò (admin, department, student, teacher), với queries GraphQL để mô phỏng truy vấn thực tế. Công cụ này tích hợp với Prometheus và Grafana để thu thập metric, đảm bảo hệ thống đáp ứng yêu cầu phản hồi  $\leq 3$  giây. Trong dự án, k6-tests hỗ trợ kiểm tra smoke, stress, và combined load, góp phần vào giai đoạn kiểm thử và tối ưu hóa.

**plagiarism:** Đây là một Module đang trong quá trình phát triển. Module này kiểm tra đạo văn, phát triển bằng Python và tích hợp với Elasticsearch cho tìm kiếm full text. Thành phần này sử dụng protobuf (plagiarism.proto) để giao tiếp gRPC với BE\_core, xử lý các tác vụ như chunker.py (phân đoạn văn bản), detector.py (phát hiện đạo văn), và analyzer.py (phân tích nội dung). plagiarism chạy trong Docker, với backend workflow sử dụng Express và BullMQ để đẩy tác vụ vào Redis queue. Module này đảm bảo kiểm tra trùng lặp đề tài và báo cáo đạo văn, góp phần vào tính minh bạch và chất lượng luận văn.

Các thành phần này được kết nối chặt chẽ, với BE\_core làm trung tâm, k6-tests đảm bảo chất lượng, và plagiarism hỗ trợ tính năng chuyên biệt, tạo nên một hệ thống hoàn chỉnh và dễ bảo trì.

### **III. Phân tích yêu cầu**

#### **1. Phân tích yêu cầu người dùng**

##### **1.1. Các vai trò người dùng (stakeholders)**

Các vai trò người dùng trong hệ thống TMS được xác định rõ ràng để đảm bảo phân quyền và hỗ trợ quy trình quản lý luận văn tốt nghiệp. Mỗi vai trò có trách nhiệm riêng, dựa trên quy trình hai giai đoạn: Đồ án chuyên ngành và Luận văn tốt nghiệp. Dưới đây là phân tích chi tiết từng vai trò.

###### **1.1.1. Giáo vụ**

Giáo vụ là vai trò quản lý hành chính, chịu trách nhiệm khởi tạo và quản lý học kỳ, upload danh sách sinh viên/giảng viên/hội đồng, và xếp lịch (thời gian) bảo vệ luận văn cho Hội đồng bảo vệ. Giáo vụ có quyền xem toàn bộ thông tin hệ thống theo học kỳ, nhưng không tham gia đánh giá hoặc chấm điểm. Vai trò này đảm bảo quy trình đồng bộ và tuân thủ thời gian học kỳ.

###### **1.1.2. Giảng viên hướng dẫn (GVHD)**

GVHD hỗ trợ sinh viên trong toàn bộ quy trình, bao gồm gửi đề tài mới hoặc đề tài cũ, đánh giá giữa kỳ (Pass/Fail), chấm điểm cuối kỳ (nhập điểm, nhận xét, tải file chấm điểm), và tham gia phản biện nếu cần. GVHD chỉ truy cập đề tài mình hướng dẫn và có quyền lưu draft hoặc final điểm số. Vai trò này tập trung vào hướng dẫn và đánh giá chất lượng đề tài.

###### **1.1.3. Giảng viên bộ môn (GVBM)**

GVBM chịu trách nhiệm duyệt hoặc từ chối đề tài do GVHD gửi, dựa trên chuyên môn và kiểm tra trùng lặp. GVBM có quyền xem toàn bộ thông tin đề tài và cập nhật trạng thái (Đã duyệt, Bị từ chối, Đang thực hiện). Vai trò này đảm bảo đề tài phù hợp với chuyên môn bộ môn và không trùng lặp với các đề tài trước.

#### **1.1.4. Giảng viên phản biện (GVPB)**

GVPB tham gia ở giai đoạn 2 (Luận văn tốt nghiệp), phản biện bài nộp từ giai đoạn giữa đến cuối kỳ bằng cách hẹn gặp sinh viên ngoài hệ thống, sau đó nhập điểm, nhận xét, tải file chấm điểm lên hệ thống. GVPB chọn trạng thái đề tài (Được bảo vệ, Không được bảo vệ, Bổ sung) và lưu draft/final. Vai trò này kiểm tra chất lượng đề tài trước bảo vệ và cảnh báo lệch điểm với GVHD.

#### **1.1.5. Thành viên hội đồng bảo vệ**

Thành viên hội đồng (Chủ tịch, Ủy viên, Thư ký) chấm điểm bảo vệ ở giai đoạn 2, nhập điểm cho từng sinh viên, tải file chấm điểm, và xác nhận hoàn thành. Họ chỉ truy cập đề tài được phân công và đảm bảo hội đồng có tối thiểu 3 thành viên với các vai trò khác nhau. Vai trò này tập trung vào đánh giá cuối cùng.

#### **1.1.6. Sinh viên**

Sinh viên tham gia đề tài do giáo vụ gán, nộp báo cáo cuối kỳ (file bất kỳ, ≤200MB), xem kết quả đánh giá giữa kỳ (Pass/Fail) và cuối kỳ. Sinh viên có quyền xem lịch sử nộp bài và thông tin đề tài của mình, nhưng không tham gia duyệt hoặc chấm điểm. Vai trò này đảm bảo sinh viên tuân thủ thời hạn.

### **1.2. Yêu cầu theo học kỳ (mã học kỳ, hiển thị thông tin phụ thuộc học kỳ)**

Yêu cầu theo học kỳ là yêu tố cốt lõi để đảm bảo hệ thống đồng bộ và linh hoạt. Mã học kỳ (ví dụ: 251 cho học kỳ 1 năm 2025) được sử dụng để lọc và hiển thị thông tin phụ thuộc, bao gồm đề tài, danh sách sinh viên/giảng viên/hội đồng, trạng thái đề tài, bài nộp, và lịch bảo vệ. Người dùng chọn học kỳ qua thanh điều hướng hoặc menu thông minh (tự động gợi ý học kỳ hiện tại). Ví dụ:

Khi chọn học kỳ 251, hệ thống hiển thị đề tài chỉ thuộc học kỳ đó (mã đề tài: [251-11-... ] cho giữa kỳ giai đoạn 1).

Danh sách upload (sinh viên, giảng viên) chỉ áp dụng cho học kỳ được chọn.

Thời gian bắt đầu/kết thúc đê tài phải hợp lệ với học kỳ, nếu quá hạn thì trạng thái tự động chuyển "Đã kết thúc". Yêu cầu này đảm bảo dữ liệu được tổ chức theo học kỳ, hỗ trợ báo cáo và theo dõi đa kỳ.

## 2. Phân tích yêu cầu chức năng

Phân tích yêu cầu chức năng tập trung vào việc xác định các tính năng chính của hệ thống TMS, dựa trên quy trình quản lý luận văn tốt nghiệp tại Khoa Khoa học và Kỹ thuật Máy tính. Các chức năng được thiết kế để hỗ trợ hai giai đoạn môn học (Đồ án chuyên ngành và Luận văn tốt nghiệp), với trọng tâm vào tự động hóa, đồng bộ hóa và giảm thủ tục thủ công.

Các phân tích này mang tính khái quát hóa tính năng của hệ thống, không đi sâu vào yêu cầu nghiệp vụ và cách hiện thực. Tài liệu **System Requirement Specification** đi kèm với báo cáo sẽ hiện thực chi tiết bối cảnh và đặc tả các tính năng này.

### 2.1. Quản lý học kỳ và danh sách

Nhóm chức năng này đảm bảo hệ thống hoạt động theo học kỳ, với mã học kỳ làm cơ sở để lọc và hiển thị thông tin. Mọi dữ liệu (đê tài, danh sách người dùng) đều phụ thuộc vào học kỳ được chọn, giúp tránh nhầm lẫn giữa các kỳ học.

#### 2.1.1. Tạo và chọn học kỳ

Tính năng này cho phép giáo vụ tạo học kỳ mới và người dùng chọn học kỳ để hiển thị thông tin tương ứng. Mục đích là đồng bộ hóa dữ liệu theo thời gian học kỳ, tránh thủ tục rườm rà khi chuyển kỳ. Đầu vào bao gồm mã học kỳ (ví dụ: 251 cho học kỳ 1 năm 2025), năm học, và kỳ học. Đầu ra là danh sách học kỳ khả dụng và thông tin hiển thị theo học kỳ được chọn. Ràng buộc: Mã học kỳ phải duy nhất, hệ thống tự động gợi ý học kỳ hiện tại dựa trên ngày hệ thống. Tính năng này hỗ trợ quy trình quản lý theo học kỳ, đảm bảo tính chính xác và minh bạch.

#### 2.1.2. Upload danh sách sinh viên, giảng viên

Giáo vụ upload danh sách sinh viên và giảng viên theo học kỳ, nhằm cung cấp dữ liệu cơ bản cho gán sinh viên và phân công hội đồng. Mục đích là tự động hóa

việc nhập dữ liệu thủ công, giảm lỗi. Đầu vào là file (CSV/XLSX) chứa mã số, họ tên, email, vai trò. Đầu ra là danh sách được lưu vào cơ sở dữ liệu, sẵn sàng cho tìm kiếm và gán. Ràng buộc: File phải đúng định dạng, mã số duy nhất theo học kỳ; hệ thống kiểm tra trùng lặp trước khi lưu. Tính năng này hỗ trợ quy trình gán sinh viên và phân công, đảm bảo dữ liệu đồng bộ theo học kỳ.

## 2.2. Quản lý đề tài

Nhóm chức năng này quản lý toàn bộ vòng đời đề tài, từ gửi đến theo dõi, với trọng tâm vào tự động hóa kiểm tra và gán.

### 2.2.1. Gửi và duyệt đề tài

GVHD gửi đề tài mới hoặc duyệt đề tài cũ, GVBM duyệt/tù chối. Mục đích là hỗ trợ giai đoạn lập kế hoạch, giảm thủ tục gửi thủ công. Đầu vào bao gồm giai đoạn (1 hoặc 2), tên đề tài (Việt/Anh), mã cán bộ (CB) GVHD, chuyên môn (A/B/C), thời gian bắt/kết thúc, học kỳ, form đăng ký. Đầu ra là đề tài lưu với trạng thái "Chờ duyệt" và mã tự động ([Mã học kỳ-giai đoạn-giai đoạn học kỳ]). Ràng buộc: Đề tài phải đầy đủ thông tin, hệ thống kiểm tra thời gian hợp lệ với học kỳ. Tính năng này đảm bảo đề tài phù hợp và minh bạch.

### 2.2.2. Kiểm tra trùng lặp đề tài

Hệ thống tự động kiểm tra trùng lặp đề tài khi GVHD gửi, dựa trên độ tương đồng tiêu đề/mô tả. Mục đích là tránh đề tài lặp lại, nâng cao chất lượng. Đầu vào là thông tin đề tài mới. Đầu ra là trạng thái "Bị trùng lặp" nếu phát hiện tương đồng cao. Ràng buộc: Sử dụng công cụ AI (ví dụ: Elasticsearch full text search) để so sánh với đề tài các kỳ trước. Tính năng này hỗ trợ GVBM trong duyệt, giảm lỗi thủ công.

### 2.2.3. Gán sinh viên vào đề tài

Hệ thống tự động gán sinh viên vào đề tài đã duyệt khi GVHD gửi hoặc giáo vụ thủ công gán. Mục đích là đồng bộ hóa gán sinh viên, giảm thủ tục. Đầu vào là danh sách sinh viên từ upload học kỳ. Đầu ra là đề tài cập nhật sinh viên tham gia, trạng thái "Đang thực hiện" nếu thời gian hợp lệ. Ràng buộc: Tối đa sinh viên theo quy

định khoa, hệ thống kiểm tra sinh viên chưa gán đề tài khác. Tính năng này đảm bảo quy trình gán nhanh chóng và chính xác.

### **2.3. Quản lý bài nộp**

Nhóm chức năng này hỗ trợ nộp và quản lý báo cáo cuối kỳ, không bao gồm giữa kỳ.

#### **2.3.1. Nộp báo cáo cuối kỳ**

Sinh viên nộp file báo cáo cuối kỳ theo đề tài. Mục đích là thay thế nộp thủ công, dễ theo dõi. Đầu vào là file (bất kỳ định dạng,  $\leq 200\text{MB}$ ). Đầu ra là file lưu, hiển thị phiên bản mới nhất trong đề tài. Ràng buộc: Chỉ nộp trong thời gian đề tài đang thực hiện, hệ thống kiểm tra dung lượng và hạn nộp. Tính năng này hỗ trợ giảng viên xem và nhận xét bài nộp.

#### **2.3.2. Xem lịch sử nộp bài**

Tất cả bên liên quan xem lịch sử nộp bài của đề tài. Mục đích là theo dõi phiên bản bài nộp, dễ truy xuất. Đầu vào là đề tài. Đầu ra là danh sách phiên bản bài nộp (ngày nộp, file). Ràng buộc: Hiển thị phiên bản mới nhất mặc định, chi tiết lịch sử khi nhấn xem. Tính năng này đảm bảo minh bạch và hỗ trợ đánh giá.

### **2.4. Đánh giá và chấm điểm**

Nhóm chức năng này hỗ trợ đánh giá giữa kỳ và chấm điểm cuối kỳ, với cảnh báo lệch điểm.

#### **2.4.1. Đánh giá giữa kỳ (Pass/Fail)**

GVHD đánh giá giữa kỳ cho đề tài. Mục đích là kiểm tra tiến độ giữa kỳ mà không nộp bài. Đầu vào là lựa chọn Pass/Fail. Đầu ra là kết quả hiển thị cho sinh viên. Ràng buộc: Áp dụng cả giai đoạn 1 và 2, không có điểm số. Tính năng này hỗ trợ chuyển tiếp giai đoạn.

#### **2.4.2. Chấm điểm cuối kỳ (GVHD và GVPB)**

GVHD và GVPB chấm điểm cuối kỳ theo giai đoạn. Mục đích là đánh giá báo cáo cuối kỳ. Đầu vào là điểm/nhận xét từng sinh viên, file chấm điểm. Đầu ra là

điểm lưu (draft/final), trạng thái đề tài (giai đoạn 2). Ràng buộc: Giai đoạn 1 chỉ GVHD, giai đoạn 2 cả GVHD và GVPB; không sửa sau "Lưu Final". Tính năng này đảm bảo đánh giá công bằng.

#### **2.4.3. Kiểm tra lệch điểm và cảnh báo**

Hệ thống kiểm tra lệch điểm ( $>2$  điểm) giữa GVHD và GVPB. Mục đích là can thiệp kịp thời. Đầu vào là điểm cuối kỳ. Đầu ra là cảnh báo gửi giáo vụ. Ràng buộc: Chỉ áp dụng giai đoạn 2, tự động khi "Lưu Final". Tính năng này nâng cao tính minh bạch.

### **2.5. Quản lý lịch bảo vệ**

Nhóm chức năng này hỗ trợ lên lịch và chấm điểm bảo vệ ở giai đoạn 1 và 2.

#### **2.5.1. Tạo hội đồng bảo vệ và phân công hội đồng bảo vệ**

GVBM tạo và phân công hội đồng cho đề tài. Mục đích là chuẩn bị hội đồng trước bảo vệ. Đầu vào là thành viên từ danh sách. Đầu ra là hội đồng lưu (Chủ tịch, Ủy viên, Thư ký) và các đề tài của hội đồng đó. Ràng buộc: Tối thiểu 3 thành viên. Tính năng này hỗ trợ giáo vụ xét duyệt.

#### **2.5.2. Xét duyệt hội đồng và lên lịch bảo vệ**

Giáo vụ xét duyệt hội đồng và lên lịch bảo vệ. Mục đích là hoàn tất lịch bảo vệ. Đầu vào là file lịch (XLS/CSV). Đầu ra là lịch lưu (ngày, giờ, phòng). Ràng buộc: Chỉ đề tài hợp lệ (điểm  $\geq 5.5$  giai đoạn 2). Tính năng này đảm bảo quy trình bảo vệ.

#### **2.5.3. Chấm điểm bảo vệ**

Thành viên hội đồng (Chủ tịch, Ủy viên, Thư ký) chấm điểm bảo vệ. Mục đích là đánh giá cuối cùng. Đầu vào là điểm từng sinh viên, chấm theo các tiêu chí riêng của người chấm điểm. Đầu ra là điểm lưu, điểm sau bảo vệ, file đã chấm điểm. Ràng buộc: Điểm khác nhau cho từng sinh viên. Tính năng này hoàn tất quy trình (giai đoạn 1 hoặc giai đoạn 2).

### **3. Phân tích yêu cầu phi chức năng**

Phân tích yêu cầu phi chức năng tập trung vào các khía cạnh không trực tiếp liên quan đến tính năng nhưng ảnh hưởng đến chất lượng tổng thể của hệ thống TMS. Các yêu cầu này được xác định dựa trên nhu cầu của Khoa Khoa học và Kỹ thuật Máy tính, đảm bảo hệ thống hoạt động hiệu quả, an toàn và dễ mở rộng. Dưới đây là phân tích chi tiết từng khía cạnh, bao gồm mục đích, tiêu chí đo lường và cách thức đáp ứng.

#### **3.1. Hiệu suất và dung lượng file**

Hiệu suất là yếu tố quan trọng để đảm bảo hệ thống TMS đáp ứng nhanh chóng các yêu cầu từ người dùng, đặc biệt trong môi trường có hàng trăm (hoặc ngàn) sinh viên và giảng viên truy cập đồng thời. Hệ thống phải xử lý các thao tác như nộp bài, chấm điểm và xuất báo cáo mà không gây chậm trễ. Cụ thể, thời gian phản hồi cho mỗi API gọi phải dưới 1 giây, và hệ thống hỗ trợ ít nhất 1000 yêu cầu đồng thời trong một giây mà không giảm hiệu suất. Dung lượng file được giới hạn ở mức 200MB cho mỗi bài nộp hoặc form chấm điểm, nhằm tránh quá tải lưu trữ. Để đáp ứng, hệ thống sử dụng MinIO cho lưu trữ file, Redis cho hàng đợi bất đồng bộ (như kiểm tra trùng lặp đề tài), và Elasticsearch cho tìm kiếm full text nhanh chóng. Tiêu chí đo lường bao gồm kiểm thử tải bằng k6-tests, với các kịch bản stress test để kiểm tra thời gian phản hồi và tỷ lệ lỗi dưới tải cao. Hiệu suất này đảm bảo hệ thống hoạt động mượt mà trong các giai đoạn cao điểm như cuối học kỳ.

#### **3.2. Bảo mật và quyền riêng tư**

Bảo mật và quyền riêng tư là tiên hàng đầu để bảo vệ dữ liệu cá nhân của sinh viên, giảng viên và các file luận văn. Hệ thống phải tuân thủ GDPR và quy định của Khoa, với cơ chế mã hóa dữ liệu nhạy cảm (như điểm số, nhận xét) bằng AES-256. Phân quyền dựa trên RBAC, sử dụng Super Token (JWT) để kiểm soát truy cập, ví dụ giáo viên hướng dẫn chỉ có quyền quản lý những đề tài của mình, GVBM chỉ duyệt đề tài. Dữ liệu lịch sử sẽ được xóa sau thời hạn lưu trữ, ta tự do điều chỉnh hạn mức thời gian trong workflow. Logs sự kiện được lưu trong Loki, với 6 instance Promtail để thu thập từ các service. Để đáp ứng, hệ thống sử dụng middleware auth trong Golang để kiểm tra JWT,

và tích hợp gRPC với chứng chỉ bảo mật. Tiêu chí đo lường bao gồm kiểm thử bảo mật (penetration testing) và kiểm tra tuân thủ GDPR, đảm bảo không rò rỉ dữ liệu.

### **3.3. Khả năng mở rộng và tích hợp**

Khả năng mở rộng đảm bảo hệ thống TMS có thể hỗ trợ tăng số lượng người dùng (ít nhất 500 người mới trong 12 tháng) và mở rộng cho các khoa khác. Hệ thống sử dụng kiến trúc microservices với 6 service Golang (thesis, role, user, council, file, academic) để dễ dàng scale ngang. MongoDB hỗ trợ sharding theo học kỳ, Redis cho cache và hàng đợi, MinIO cho lưu trữ file mở rộng. Tích hợp bao gồm gRPC cho giao tiếp nội bộ, GraphQL cho API linh hoạt, và backend workflow (Express, BullMQ) cho tác vụ ngầm. Để đáp ứng, hệ thống sử dụng Kubernetes cho triển khai container hóa. Tiêu chí đo lường bao gồm kiểm thử scale với k6-tests, đảm bảo hệ thống duy trì hiệu suất khi tải tăng 50%. Khả năng mở rộng này giúp hệ thống linh hoạt với các thay đổi quy trình tương lai.

## **4. Rủi ro và giả định**

Phân tích rủi ro và giả định giúp dự đoán các vấn đề tiềm ẩn và lập kế hoạch ứng phó. Giả định bao gồm: (1) Tất cả người dùng có tài khoản Google để xác thực qua OAuth; (2) Dữ liệu upload (danh sách sinh viên/giảng viên) luôn đúng định dạng; (3) Hạ tầng mạng nội bộ ổn định để hỗ trợ 1000 người dùng đồng thời; (4) Công cụ kiểm tra trùng lặp để tài hoạt động chính xác với độ tương đồng cao.

Các rủi ro chính bao gồm: (1) Quá tải hệ thống trong giai đoạn nộp bài cuối kỳ, ứng phó bằng Redis queue và sharding MongoDB; (2) Lỗi phân quyền dẫn đến truy cập trái phép, ứng phó bằng kiểm thử RBAC với JWT; (3) Mất dữ liệu do sao lưu thất bại, ứng phó bằng cronjob hàng ngày với mongodump và lưu vào Supabase; (4) Không tuân thủ GDPR do lưu trữ dữ liệu quá lâu, ứng phó bằng cronjob di chuyển/xóa dữ liệu >n năm bằng workflow. Các rủi ro được đánh giá theo mức độ (cao/trung bình/thấp) và lập kế hoạch dự phòng để đảm bảo dự án thành công

## **IV. Thiết kế hệ thống**

### **1. Thiết kế tổng thể**

#### **1.1. Kiến trúc hệ thống**

Kiến trúc hệ thống TMS được thiết kế theo mô hình microservices, đảm bảo tính linh hoạt, khả năng mở rộng và dễ bảo trì. Hệ thống bao gồm các thành phần backend, frontend và công cụ giám sát, sử dụng các công nghệ hiện đại để xử lý quy trình quản lý luận văn tốt nghiệp.

##### **1.1.1. Các service backend (thesis, role, user, council, file, academic)**

Các service backend chính được phát triển bằng ngôn ngữ GoLang, sử dụng gRPC để giao tiếp nội bộ hiệu quả và MySQL làm cơ sở dữ liệu. Mỗi service là một module độc lập, chịu trách nhiệm cho một phần nghiệp vụ cụ thể, giúp dễ dàng scale và bảo trì. Các service này được chứa trong thư mục BE\_core của dự án, với cấu trúc mã nguồn bao gồm controllers, routes và models.

- Service Thesis: Quản lý đề tài luận văn, bao gồm gửi, duyệt, kiểm tra trùng lặp và theo dõi trạng thái. Service này xử lý logic như tạo mã đề tài ([Mã học kỳ-giai đoạn-giai đoạn học kỳ]) và cập nhật trạng thái tự động. Dữ liệu được lưu trong MySQL với các bảng liên quan đến đề tài và học kỳ.
- Service Role: Quản lý vai trò người dùng và phân quyền RBAC. Service này hỗ trợ tạo, cập nhật vai trò (ví dụ: GVHD, GVBM, GVPB) và đồng bộ quyền với các service khác qua gRPC.
- Service User: Xử lý thông tin người dùng, bao gồm upload danh sách sinh viên/giảng viên, xác thực và gán vai trò. Service này tích hợp Google OAuth và lưu dữ liệu trong MySQL.
- Service Council: Quản lý hội đồng bảo vệ, bao gồm phân công thành viên (Chủ tịch, Ủy viên, Thư ký) và chấm điểm bảo vệ. Service này liên kết với service Thesis qua gRPC để kiểm tra đề tài hợp lệ.

- Service File: Quản lý file, bao gồm upload báo cáo, form chấm điểm và lưu trữ. Service này tích hợp với MinIO để lưu trữ object, hỗ trợ dung lượng tối đa 200MB.
- Service Academic: Quản lý học kỳ và thông tin học thuật, bao gồm tạo mã học kỳ, upload danh sách và lọc dữ liệu theo học kỳ. Service này sử dụng MySQL để lưu trữ danh sách theo học kỳ.

### **1.1.2. Server gateway và tích hợp (GraphQL, gRPC, Redis, MinIO, MongoDB)**

Server gateway là trung tâm điều phối, phát triển bằng Golang với framework Gin để xử lý yêu cầu HTTP. Gateway sử dụng GraphQL để cung cấp API linh hoạt, hỗ trợ lọc, phân trang và sắp xếp. Nó tích hợp Redis làm hàng đợi cho tác vụ bất đồng bộ, client gRPC để gọi các service backend, MinIO để lưu trữ file, và MongoDB để lưu log sự kiện và dữ liệu phi cấu trúc. Trong BE\_core, gateway bao gồm các routes (auth.routes.js, cronjobs.routes.js) và middleware (auth.middleware.js) để kiểm tra JWT. Gateway đảm bảo tích hợp liền mạch giữa frontend và backend, xử lý xác thực và phân quyền.

### **1.1.3. Backend workflow cho tác vụ ngầm (Express, BullMQ)**

Backend workflow được phát triển bằng Express.js để xử lý các tác vụ ngầm như kiểm tra trùng lặp đề tài, gửi email cảnh báo lệch điểm và cập nhật trạng thái tự động. Sử dụng BullMQ dựa trên Redis để quản lý hàng đợi job, đảm bảo xử lý bất đồng bộ mà không ảnh hưởng đến hiệu suất chính. Trong thư mục queue, workflow bao gồm cronjob-init.js để khởi tạo cronjob và minio.service.js để xử lý file. Thành phần này hỗ trợ các tác vụ định kỳ, góp phần vào tính tự động hóa của hệ thống.

### **1.1.4. Frontend (Next.js và Vite)**

Frontend chính sử dụng Next.js để xây dựng giao diện người dùng động, responsive, hỗ trợ chọn học kỳ, gửi đề tài, nộp bài và chấm điểm. Next.js tích hợp GraphQL qua Apollo Client để gọi API từ gateway. Frontend admin local sử dụng Vite cho giao diện quản trị viên, giám sát tác vụ ngầm như queue Redis và log sự kiện. Trong FE\_core, frontend bao gồm các component như auth, council và topic,

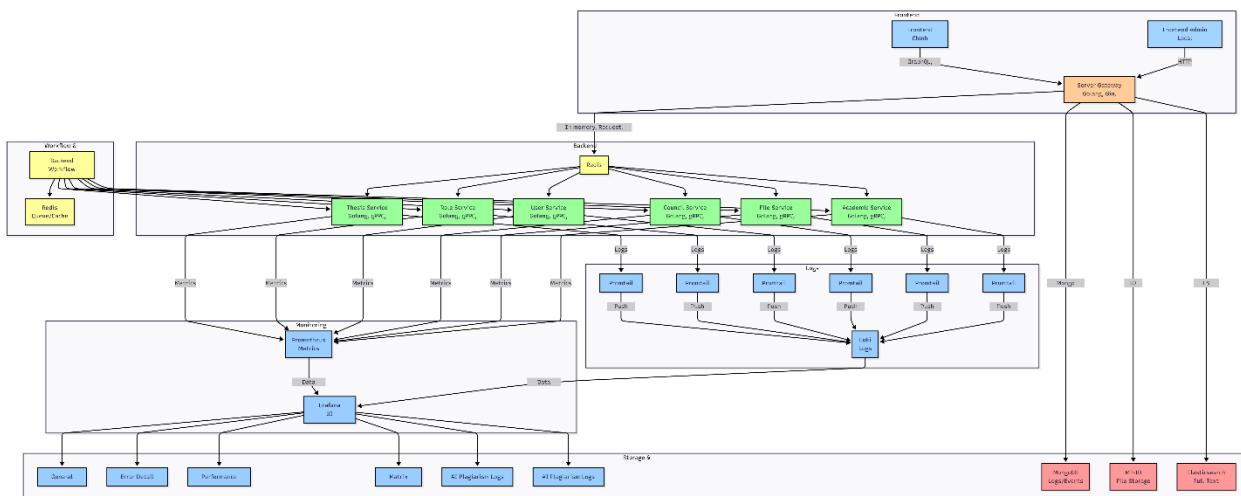
đảm bảo trải nghiệm người dùng mượt mà cho các vai trò (giáo vụ, GVHD, sinh viên).

### 1.1.5. Công cụ giám sát (Grafana, Loki, Prometheus, Promtail, Elasticsearch)

Hệ thống sử dụng các công cụ giám sát để theo dõi hiệu suất và logs. Grafana cung cấp UI dashboard để hiển thị metric từ Prometheus và logs từ Loki. Loki lưu trữ logs, Prometheus thu thập metric hệ thống, với 6 instance Promtail để bắn logs từ các service backend lên Loki. Elasticsearch hỗ trợ tìm kiếm full text cho đề tài và bài nộp. Các công cụ này tích hợp với backend workflow để giám sát cronjob và queue Redis, đảm bảo hệ thống hoạt động ổn định.

### 1.2. Sơ đồ kiến trúc hệ thống

Kiến trúc hệ thống hiện tại có thể được tóm tắt như hình ảnh sau:



**Hình 1.** Sơ đồ cấu trúc hệ thống

Để có thể quan sát rõ hơn, vui lòng kiểm tra hình ảnh chất lượng cao được đính kèm chung với báo cáo này. Hoặc có thể truy cập đường dẫn đến file thiết kế sau:

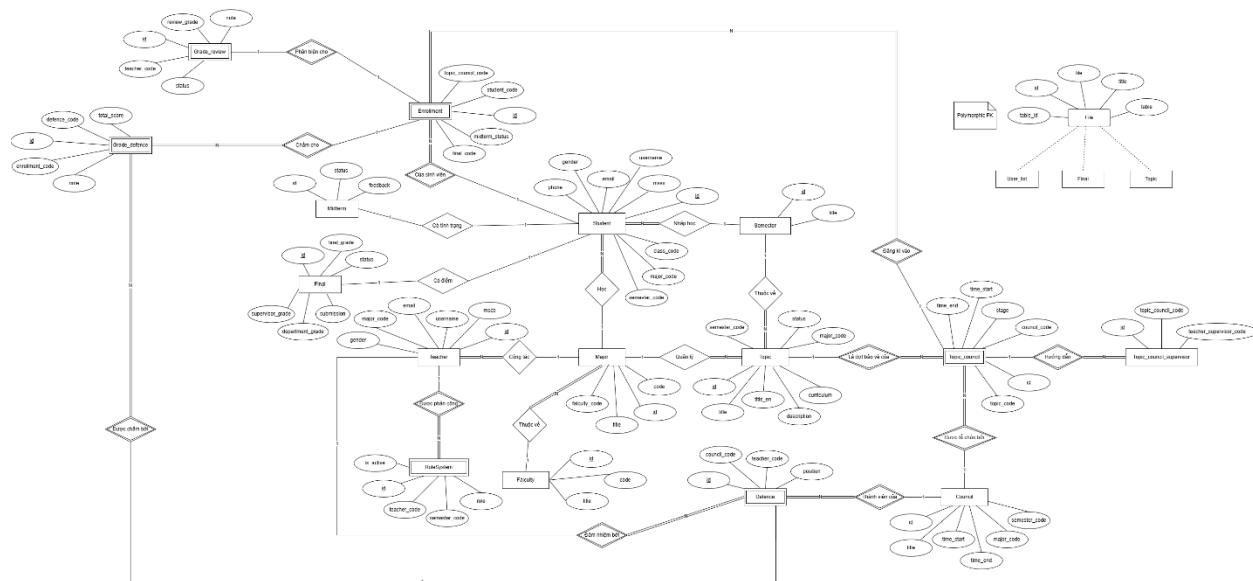
<https://www.mermaidchart.com/d/43f596bc-67dc-47d5-8e78-9c5ea004f852>

## 2. Thiết kế cơ sở dữ liệu

Hệ thống sử dụng kiến trúc Polyglot Persistence (Đa lưu trữ), kết hợp sức mạnh của RDBMS cho dữ liệu có cấu trúc chặt chẽ và NoSQL cho dữ liệu linh động, hiệu năng cao.

### 2.1. Mô hình dữ liệu (Entity-Relationship Diagram)

Sơ đồ quan hệ thực thể (Conceptual Design) được mô tả như sau:



**Hình 2.** Entity Relationship Diagram

Xem chi tiết hơn ở hình ảnh được đính kèm với báo cáo. Hoặc truy cập <https://bit.ly/4pLln05>.

Chi tiết thuộc tính các thực thể

Quy ước:

- (PK): Khóa chính (Primary Key).
- (FK): Khóa ngoại (Foreign Key).

Nhóm 1: Tổ chức và Người dùng (Organization & Users)

1. Faculty (Khoa)

- id (PK): Mã định danh khoa.
- code: Mã số khoa (viết tắt).
- title: Tên hiển thị của khoa.

## 2. Major (Ngành học)

- id (PK): Mã định danh ngành.
- code: Mã số ngành.
- title: Tên ngành học.
- faculty\_code (FK): Tham chiếu tới Faculty, xác định ngành thuộc khoa nào.

## 3. Student (Sinh viên)

- id (PK): Mã định danh hệ thống.
- mssv: Mã số sinh viên (Unique).
- username: Tên đăng nhập hệ thống.
- email: Email liên hệ.
- phone: Số điện thoại.
- gender: Giới tính (Enum: male, female, other).
- class\_code: Mã lớp sinh hoạt.
- major\_code (FK): Tham chiếu tới Major.
- semester\_code (FK): Tham chiếu tới Semester (Niên khóa nhập học).

## 4. Teacher (Giảng viên)

- id (PK): Mã định danh hệ thống.
- msrb: Mã số cán bộ.

- username, email: Thông tin tài khoản.
- gender: Giới tính.
- major\_code (FK): Tham chiếu tới Major (Bộ môn công tác).

## 5. RoleSystem (Phân quyền theo kỳ)

- id (PK).
- teacher\_code (FK): Tham chiếu tới Teacher.
- semester\_code (FK): Tham chiếu tới Semester.
- role: Vai trò cụ thể trong học kỳ (Enum: Academic\_affairs\_staff - Giáo vụ, Department\_lecturer - Trưởng bộ môn, Teacher).
- activate: Trạng thái kích hoạt (Boolean).

Nhóm 2: Quản lý Đề tài & Hội đồng (Topic & Council)

## 6. Semester (Học kỳ)

- id (PK): Mã định danh học kỳ.
- title: Tên học kỳ (VD: Học kỳ 1 năm 2024-2025).

## 7. Topic (Đề tài gốc)

- id (PK).
- title: Tên đề tài (Tiếng Việt).
- title\_en: Tên đề tài (Tiếng Anh).
- description: Mô tả chi tiết nội dung đề tài.
- curriculum: Hệ đào tạo (Đại trà/Chất lượng cao).
- status: Trạng thái (Enum: submit, approved\_1, approved\_2, in\_progress, completed, rejected, topic\_pending).

- percent\_stage\_1, percent\_stage\_2: Tỷ trọng điểm theo giai đoạn.
- major\_code (FK): Ngành quản lý đê tài.
- semester\_code (FK): Đề tài thuộc học kỳ nào.

#### 8. Council (Hội đồng bảo vệ)

- id (PK).
- title: Tên hội đồng.
- time\_start: Thời gian bắt đầu làm việc.
- major\_code (FK), semester\_code (FK): Hội đồng thuộc ngành và học kỳ nào.

#### 9. Defence (Thành viên hội đồng)

- id (PK).
- council\_code (FK): Thuộc hội đồng nào.
- teacher\_code (FK): Giảng viên tham gia.
- position: Vị trí trong hội đồng (Enum: president - Chủ tịch, secretary - Thư ký, reviewer - Phản biện, member - Ủy viên).

#### 10. Topic\_council (Đợt bảo vệ đề tài)

Đây là thực thể liên kết quan trọng, biến một "Đề tài" thành một "Nhiệm vụ bảo vệ" cụ thể.

- id (PK).
- topic\_code (FK): Tham chiếu tới Topic.
- council\_code (FK): Tham chiếu tới Council (Hội đồng chấm).

- o stage: Giai đoạn bảo vệ (Enum: stage\_dacn - Đồ án chuyên ngành, stage\_lvtn - Luận văn tốt nghiệp).
- o time\_start, time\_end: Thời gian bảo vệ cụ thể.

#### 11. Topic\_council\_supervisor (GV Hướng dẫn)

- o id (PK).
- o topic\_council\_code (FK): Tham chiếu tới Đợt bảo vệ.
- o teacher\_supervisor\_code (FK): Tham chiếu tới Teacher (Người hướng dẫn chính).

#### Nhóm 3: Đăng ký & Kết quả (Enrollment & Grading)

#### 12. Enrollment (Đăng ký)

- o id (PK).
- o student\_code (FK): Sinh viên đăng ký.
- o topic\_council\_code (FK): Đăng ký vào đề tài nào, hội đồng nào.
- o midterm\_code (FK): Link tới bảng điểm giữa kỳ.
- o final\_code (FK): Link tới bảng điểm cuối kỳ.

#### 13. Midterm (Điểm quá trình/Giữa kỳ)

- o id (PK).
- o status: Trạng thái đánh giá (Pass/Fail).
- o feedback: Nhận xét của GVHD.

#### 14. Grade\_review (Điểm phản biện)

- o id (PK).
- o review\_grade: Điểm số do người phản biện chấm.

- teacher\_code (FK): Giảng viên chấm phản biện.
- status: Kết quả (passed/failed/pending).
- notes: Ghi chú nhận xét.

#### 15. Grade\_defence (Điểm hội đồng)

- id (PK).
- enrollment\_code (FK): Điểm này thuộc về sinh viên nào (qua bảng Enrollment).
- defence\_code (FK): Thành viên nào trong hội đồng chấm (qua bảng Defence).
- total\_score: Tổng điểm thành phần.
- note: Ghi chú tại buổi bảo vệ.

#### 16. Final (Tổng kết)

- id (PK).
- supervisor\_grade: Điểm hướng dẫn.
- department\_grade: Điểm phản biện/bộ môn.
- final\_grade: Điểm tổng kết cuối cùng (Quyết định tốt nghiệp).
- status: Trạng thái tốt nghiệp (passed/failed).

#### 17. File (Tài liệu hệ thống)

- id (PK).
- file: Đường dẫn file (URL/Path tới MinIO).
- title: Tên hiển thị file.
- table: Loại đối tượng sở hữu file (Enum: topic, midterm, final, order).

- table\_id: ID của đối tượng sở hữu (Polymorphic Association).

## 2.2. Cấu trúc cơ sở dữ liệu (6 MySQL, 1 MongoDB, 1 Redis, 1 MinIO)

Hệ thống phân chia dữ liệu vào 4 loại kho lưu trữ khác nhau để tối ưu hóa hiệu năng:

### 1. MySQL (Lưu trữ chính - Relational Data):

- Lưu trữ dữ liệu nghiệp vụ cốt lõi đòi hỏi tính toàn vẹn cao (ACID) và các mối quan hệ phức tạp.
- Bao gồm 18 bảng (tables) quản lý: Sinh viên, Giảng viên, Đề tài, Hội đồng, Điểm số.

### 2. MongoDB (Lưu trữ linh hoạt - Document Data):

- Lưu trữ các dữ liệu phi cấu trúc hoặc bán cấu trúc, thay đổi thường xuyên.
- Dùng cho: Cấu hình hệ thống (System Config), Lịch trình công việc (CronJob), Quy trình động (Workflow), và Logs hệ thống.

### 3. Redis (Cache & Session - Key-Value Data):

- Lưu trữ thông tin phiên làm việc (Session), caching dữ liệu truy cập thường xuyên để giảm tải cho MySQL.
- Hỗ trợ cơ chế Pub/Sub cho hàng đợi tin nhắn thời gian thực.

### 4. MinIO (Lưu trữ đối tượng - Object Storage):

- Lưu trữ các file tài liệu lớn như báo cáo khóa luận (PDF, DOCX) và tài liệu minh chứng.

Trong MySQL bảng File chỉ lưu đường dẫn (metadata), còn dữ liệu nhị phân thực tế nằm tại MinIO.

## 2.3. Thiết kế schema và quan hệ

### Ghi chú chung

- **PK (Primary Key):** Khóa chính.
- **FK (Foreign Key):** Khóa ngoại.

- Audit Fields:** Tất cả các bảng đều bao gồm 4 trường quản lý hệ thống: created\_at, updated\_at (Datetime) và created\_by, updated\_by (Varchar).

### 2.3.1 Phân hệ Tổ chức & Người dùng (Organization & Users)

#### 2.3.1.1. Bảng Faculty (Khoa)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã định danh duy nhất của Khoa
ms	VARCHAR(10)	Not Null	Mã số khoa (viết tắt)
title	VARCHAR(255)	Not Null	Tên hiển thị đầy đủ của Khoa

#### 2.3.1.2. Bảng Major (Ngành học)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã định danh ngành
ms	VARCHAR(255)	Not Null	Mã số ngành
title	VARCHAR(255)	Not Null	Tên ngành học
faculty_code	VARCHAR(255)	<b>FK</b> (Faculty)	Xác định ngành thuộc khoa nào

#### 2.3.1.3. Bảng Student (Sinh viên)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã định danh hệ thống
mssv	VARCHAR(10)	Not Null	Mã số sinh viên (Unique)
username	VARCHAR(255)	Not Null	Tên đăng nhập
email	VARCHAR(255)	Not Null	Email liên hệ
phone	VARCHAR(255)	Not Null	Số điện thoại
gender	ENUM	Not Null	Giới tính (male, female, other)
class_code	VARCHAR(255)	Default NULL	Mã lớp sinh hoạt
major_code	VARCHAR(255)	<b>FK</b> (Major)	Sinh viên thuộc ngành nào

semester_code	VARCHAR(255)	<b>FK</b> (Semester)	Niên khóa nhập học
---------------	--------------	----------------------	--------------------

#### 2.3.1.4. Bảng Teacher (Giảng viên)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã định danh hệ thống
msgv	VARCHAR(10)	Not Null	Mã số giảng viên
username	VARCHAR(255)	Not Null	Tên đăng nhập
email	VARCHAR(255)	Not Null	Email công việc
gender	ENUM	Not Null	Giới tính (male, female, other)
major_code	VARCHAR(255)	<b>FK</b> (Major)	Bộ môn công tác
semester_code	VARCHAR(255)	<b>FK</b> (Semester)	Học kỳ bắt đầu công tác

#### 2.3.1.5. Bảng RoleSystem (Phân quyền)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã định danh
title	VARCHAR(255)	Not Null	Tên vai trò hiển thị
teacher_code	VARCHAR(255)	<b>FK</b> (Teacher)	Giảng viên được phân quyền
role	ENUM	Default NULL	Vai trò (Giáo vụ, Trưởng bộ môn...)
semester_code	VARCHAR(255)	<b>FK</b> (Semester)	Quyền hạn trong học kỳ nào
activate	TINYINT(1)	Not Null	Trạng thái kích hoạt (0/1)

### 2.3.2. Phân hệ Đề tài & Hội đồng (Topic & Council)

#### 2.3.2.1 Bảng Semester (Học kỳ)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
------------	--------------	-----------	-------

<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã học kỳ
title	VARCHAR(255)	Not Null	Tên học kỳ (VD: HK1 2024-2025)

### 2.3.2.2. Bảng Topic (Đề tài gốc)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã đề tài
title	VARCHAR(255)	Not Null	Tên đề tài tiếng Việt
title_en	VARCHAR(255)	Default '...'	Tên đề tài tiếng Anh
description	LONGTEXT	Not Null	Mô tả chi tiết
curriculum	VARCHAR(10)	Not Null	Hệ đào tạo (Đại trà/CLC)
status	ENUM	Not Null	Trạng thái (submit, approved,...)
major_code	VARCHAR(255)	<b>FK</b> (Major)	Ngành quản lý đề tài
semester_code	VARCHAR(255)	<b>FK</b> (Semester)	Đề tài thuộc học kỳ nào

### 2.3.2.3. Bảng Council (Hội đồng bảo vệ)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã hội đồng
title	VARCHAR(255)	Not Null	Tên hội đồng
time_start	DATETIME	Default NULL	Thời gian bắt đầu làm việc
major_code	VARCHAR(255)	<b>FK</b> (Major)	Hội đồng của ngành nào
semester_code	VARCHAR(255)	<b>FK</b> (Semester)	Hội đồng của học kỳ nào

### 2.3.2.4. Bảng Topic\_council (Đợt bảo vệ đề tài)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã đợt bảo vệ

title	VARCHAR(255)	Not Null	Tên hiển thị đợt bảo vệ
stage	ENUM	Not Null	Giai đoạn (ĐACN / LVTN)
topic_code	VARCHAR(255)	<b>FK</b> (Topic)	Đề tài gốc được bảo vệ
council_code	VARCHAR(255)	<b>FK</b> (Council)	Thuộc hội đồng nào (có thể Null)
time_start	DATETIME	Default NULL	Thời gian bắt đầu slot
time_end	DATETIME	Not Null	Thời gian kết thúc slot

### 2.3.2.5. Bảng Defence (Thành viên hội đồng)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã thành viên
title	VARCHAR(255)	Not Null	Tên hiển thị
council_code	VARCHAR(255)	<b>FK</b> (Council)	Thuộc hội đồng nào
teacher_code	VARCHAR(255)	<b>FK</b> (Teacher)	Giảng viên tham gia
position	ENUM	Not Null	Vị trí (Chủ tịch, Thư ký...)

### 2.3.3. Phân hệ Đăng ký & Kết quả (Enrollment & Grading)

#### 2.3.3.1. Bảng Enrollment (Đăng ký)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã đăng ký
student_code	VARCHAR(255)	<b>FK</b> (Student)	Sinh viên thực hiện
topic_council_code	VARCHAR(255)	<b>FK</b> (Topic_council)	Đăng ký vào đợt bảo vệ nào
midterm_code	VARCHAR(255)	<b>FK</b> (Midterm)	Liên kết bảng điểm giữa kỳ
final_code	VARCHAR(255)	<b>FK</b> (Final)	Liên kết bảng điểm cuối kỳ

grade_review_code	VARCHAR(255)	Default NULL	Liên kết điểm phản biện
-------------------	--------------	--------------	-------------------------

### 2.3.3.2. Bảng Grade\_defence (Điểm hội đồng)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã bảng điểm
defence_code	VARCHAR(255)	<b>FK</b> (Defence)	Thành viên chấm điểm
enrollment_code	VARCHAR(255)	<b>FK</b> (Enrollment)	Chấm cho sinh viên nào
total_score	INT	Default NULL	Tổng điểm chấm
note	VARCHAR(255)	Default NULL	Ghi chú/Nhận xét

### 2.3.3.3. Bảng Grade\_review (Điểm phản biện)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã phiếu điểm
teacher_code	VARCHAR(255)	<b>FK</b> (Teacher)	Giảng viên phản biện
review_grade	INT	Default NULL	Điểm số
status	ENUM	Not Null	Kết quả (passed/failed)
notes	TEXT	Default NULL	Nhận xét chi tiết

### 2.3.3.4. Bảng Final (Tổng kết)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã tổng kết
supervisor_grade	INT	Default NULL	Điểm hướng dẫn
department_grade	INT	Default NULL	Điểm phản biện/bộ môn
final_grade	INT	Default NULL	Điểm tổng kết (Ra quyết định)
status	ENUM	Not Null	Trạng thái tốt nghiệp

### 2.3.4. Bảng Hệ thống (System)

#### 2.3.4.1. Bảng File (Quản lý tài liệu đa hình)

Tên trường	Kiểu dữ liệu	Ràng buộc	Mô tả
<b>id</b>	VARCHAR(255)	<b>PK</b> , Not Null	Mã file
title	VARCHAR(255)	Not Null	Tên file hiển thị
file	VARCHAR(255)	Not Null	Đường dẫn/URL file
status	ENUM	Not Null	Trạng thái duyệt file
table	ENUM	Not Null	Loại đối tượng (topic, midterm...)
table_id	VARCHAR(255)	Not Null	ID của đối tượng sở hữu

## 2.4. Thiết kế collection cho MongoDB

### 2.4.1. Collection cronjob

Dùng để quản lý các tác vụ lập lịch tự động (Scheduled Jobs).

Tên trường (Field)	Kiểu dữ liệu (Type)	Mô tả (Description)
_id	ObjectId	Khóa chính tự sinh của MongoDB.
WL_id	String	ID tham chiếu tới Workflow cần chạy.
schedule	String	Chuỗi định dạng CRON (VD: * * * * *).
idJobCurent	String	ID của job đang thực thi hiện tại (nếu có).
enabled	Boolean	Trạng thái kích hoạt job (true/false).
createdAt	Date	Thời điểm tạo job.
updatedAt	Date	Thời điểm cập nhật job lần cuối.
__v	Number	Version key (do Mongoose tự quản lý).

### 2.4.2. Collection minio

Lưu trữ cấu hình kết nối tới MinIO (Object Storage) để quản lý file.

Tên trường (Field)	Kiểu dữ liệu (Type)	Mô tả (Description)
_id	ObjectId	Khóa chính tự sinh.
name	String	Tên định danh cho cấu hình MinIO.
endPoint	String	Địa chỉ máy chủ MinIO (IP hoặc Domain).
port	Number	Cổng kết nối (VD: 9000).
useSSL	Boolean	Sử dụng giao thức bảo mật SSL hay không.
accessKey	String	Khóa truy cập (Access Key).
secretKey	String	Khóa bí mật (Secret Key).
bucketName	String	Tên Bucket mặc định để lưu trữ.
enabled	Boolean	Trạng thái kích hoạt cấu hình này.
region	String	Vùng lưu trữ (Region).
metadata	Object	Các thông tin bổ sung khác (Key-Value).
connectionStatus	Object	Trạng thái kết nối hiện tại (Health check).
createdAt	Date	Thời điểm tạo cấu hình.
updatedAt	Date	Thời điểm cập nhật lần cuối.

#### 2.4.3. Collection services

Quản lý danh sách các microservices trong hệ thống (Service Registry).

Tên trường (Field)	Kiểu dữ liệu (Type)	Mô tả (Description)
_id	ObjectId	Khóa chính tự sinh.
name	String	Tên hiển thị của dịch vụ.
url	String	Đường dẫn URL cơ sở của dịch vụ.
port	Number	Cổng hoạt động của dịch vụ.

protocol	String	Giao thức giao tiếp (VD: http, grpc).
protoPath	String	Đường dẫn tới file .proto (nếu dùng gRPC).
protoPackage	String	Tên package trong file proto.
enabled	Boolean	Trạng thái hoạt động của dịch vụ.
healthy	Boolean	Trạng thái sức khỏe (Health status).
lastHealthCheck	Date	Thời điểm kiểm tra sức khỏe gần nhất.
metadata	Object	Dữ liệu mô tả thêm về dịch vụ.
createdAt	Date	Thời điểm đăng ký dịch vụ.
updatedAt	Date	Thời điểm cập nhật thông tin.
__v	Number	Version key.

#### 2.4.4. Collection session

Quản lý phiên đăng nhập của người dùng (Authentication & Authorization).

Tên trường (Field)	Kiểu dữ liệu (Type)	Mô tả (Description)
_id	ObjectId	Khóa chính tự sinh.
ids	String	ID phiên làm việc (Session ID).
user_id	String	ID của người dùng sở hữu phiên này.
email	String	Email của người dùng đăng nhập.
role	String	Vai trò của người dùng trong phiên này.
refresh_token	String	Token dùng để cấp lại Access Token mới.
user_agent	String	Thông tin trình duyệt/thiết bị người dùng.
ip_address	String	Địa chỉ IP thực hiện đăng nhập.
created_at	Date	Thời điểm bắt đầu phiên.
expires_at	Date	Thời điểm phiên hết hạn.

#### 2.4.5. Collection users

Lưu trữ thông tin tài khoản người dùng hệ thống.

Tên trường (Field)	Kiểu dữ liệu (Type)	Mô tả (Description)
_id	ObjectId	Khóa chính tự sinh.
email	String	Địa chỉ email (Dùng làm tên đăng nhập).
passwordHash	String	Mật khẩu đã được mã hóa.
role	String	Vai trò trong hệ thống (Admin, User...).
enabled	Boolean	Trạng thái kích hoạt tài khoản.
twoFactorEnabled	Boolean	Bật xác thực 2 bước (2FA) hay không.
twoFactorSecret	String	Mã bí mật dùng cho 2FA.
currentTokenExpires	Date	Thời gian hết hạn của token hiện tại.
lastLogin	Date	Thời điểm đăng nhập gần nhất.
createdAt	Date	Ngày tạo tài khoản.
updatedAt	Date	Ngày cập nhật thông tin.
__v	Number	Version key.

#### 2.4.6. Collection workflows

Lưu trữ định nghĩa các quy trình nghiệp vụ (Orchestration logic).

Tên trường (Field)	Kiểu dữ liệu (Type)	Mô tả (Description)
_id	ObjectId	Khóa chính tự sinh.
name	String	Tên quy trình (Workflow).
parentServiceName	String	Tên dịch vụ cha khởi tạo quy trình.
parentMethod	String	Phương thức/Hàm của dịch vụ cha.
parentParams	Object	Các tham số đầu vào của quy trình.

children	Array<Object>	Danh sách các bước con/dịch vụ con cần gọi.
options	Object	Các tùy chọn cấu hình bổ sung (Retry, Timeout...).
createdAt	Date	Thời điểm tạo quy trình.
updatedAt	Date	Thời điểm cập nhật quy trình.
__v	Number	Version key.

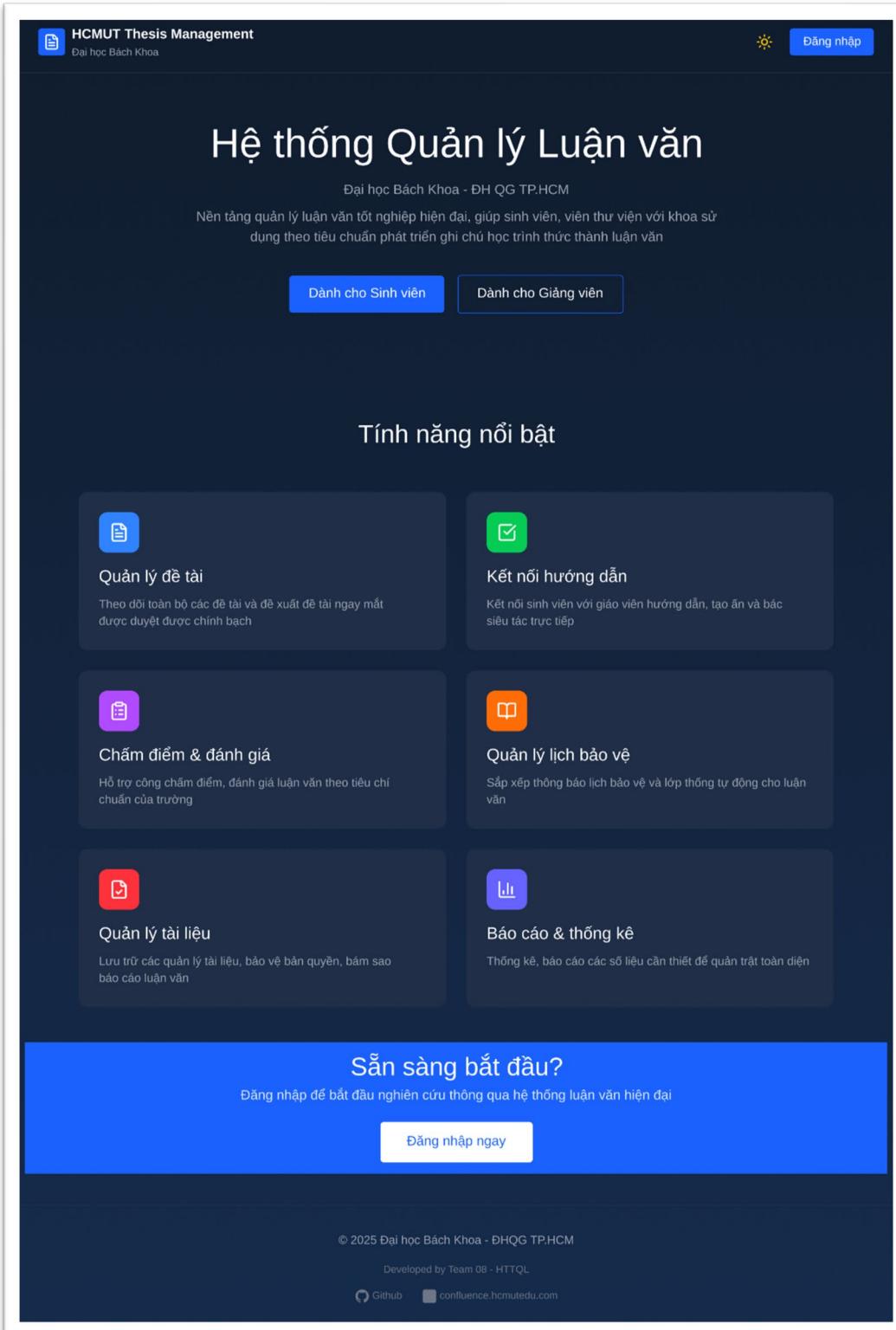
Chú ý:

- cronjob.WL\_id liên kết logic với workflows.\_id.
- session.user\_id liên kết logic với users.\_id.

### 3. Thiết kế giao diện người dùng

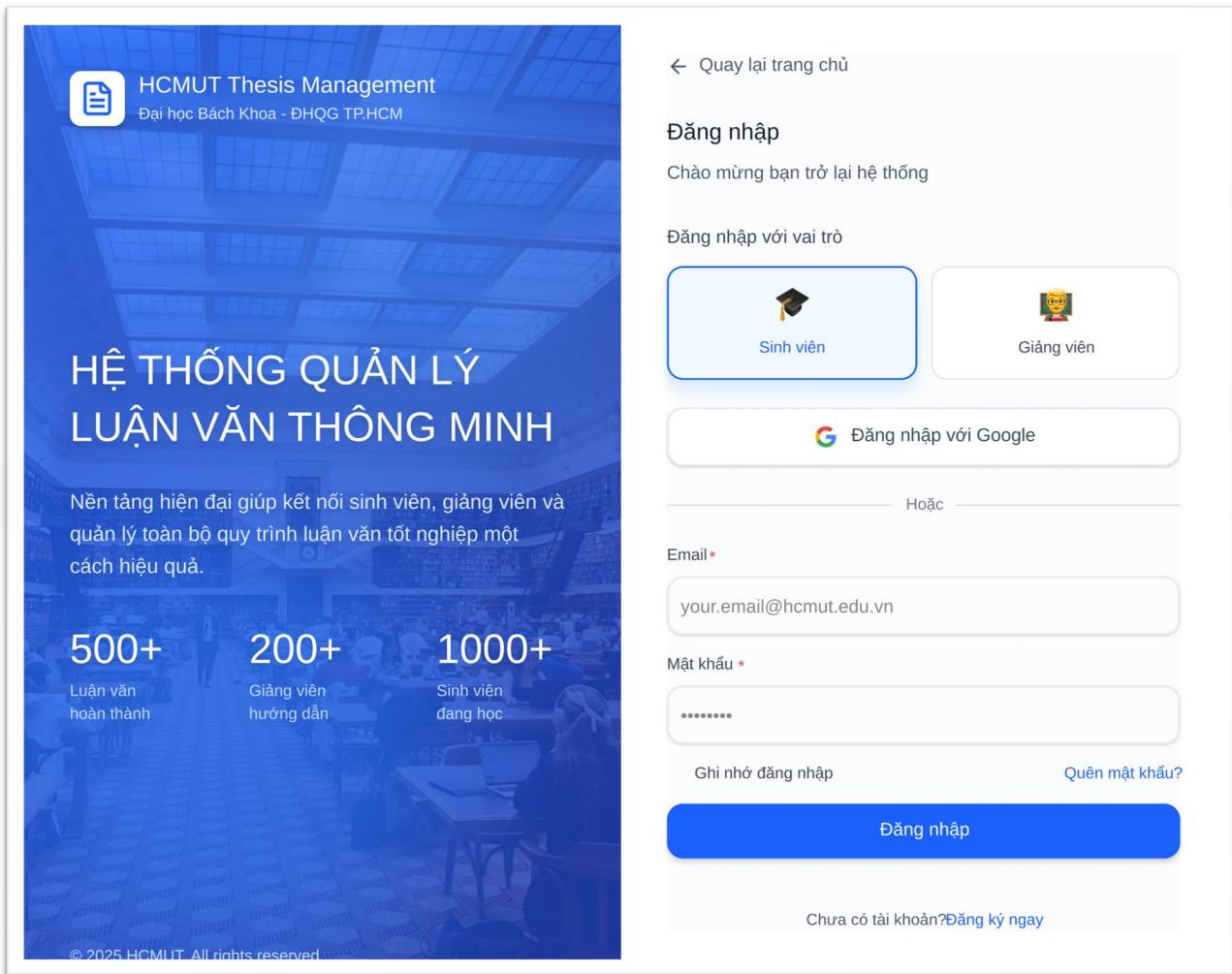
#### 3.1. Giao diện frontend chính (Next.js)

Các giao diện chính của ứng dụng trải qua quy trình thiết kế trải dài từ Wireframe, Mockup, cuối cùng là prototype sau đó mới tiến hành hiện thực. Kết quả cuối cùng như sau:



**Hình 3.** Giao diện landing page

Giao diện Landing Page đóng vai trò là cổng thông tin tập trung, giới thiệu tổng quan về hệ thống và làm nổi bật 6 nhóm tính năng cốt lõi (Quản lý đê tài, Chấm điểm, Lịch bảo vệ,...). Thiết kế phân luồng truy cập rõ ràng cho Sinh viên và Giảng viên, giúp người dùng định hướng nhanh chóng ngay khi truy cập.



**Hình 4.** Giao diện đăng nhập

Trang Đăng nhập cho phép người dùng lựa chọn vai trò trước khi tiến hành xác thực (Sinh viên hoặc Giảng viên). Hệ thống hỗ trợ hai phương thức xác thực: Đăng nhập bằng tài khoản nội bộ (Email và Mật khẩu) và Đăng nhập bằng tài khoản Google (qua dịch vụ OAuth). Giao diện trực quan, đảm bảo tính bảo mật và

cung cấp các tùy chọn phục hồi tài khoản như "Quên mật khẩu" và "Đăng ký ngay"

The screenshot shows a web-based dashboard for managing teachers at HCMUT. The top navigation bar includes the HCMUT logo, a dropdown for the academic year (set to 'Học kỳ 1 năm 2025-2026'), and user icons. On the left, a sidebar menu is open under 'Quản lý giảng viên' (Teacher Management), showing options like 'Giáo viên bộ môn' (Subject Teachers) and 'Quản lý sinh viên' (Student Management). The main content area is titled 'Quản lý Giảng viên' (Teacher Management) and displays a list of teachers with columns for Mã GV (ID), Họ Tên (Name), Email, Giới tính (Gender), and Vai trò (Role). Each teacher entry includes an email link and a role selection dropdown. At the bottom, there are pagination controls for 'Hiển thị 1 - 9 của 9' (Display 1 - 9 of 9) and 'Số dòng: 10' (Number of rows: 10).

MÃ GV	HỌ TÊN	EMAIL	GIỚI TÍNH	VAI TRÔ
1660116	Giang vien 7	naoizdabetz2004@gmail.com	Nam	Vai tro 5 Vai tro 6 Vai tro 7
5314866	Giang vien 170	teacher170@university.edu.vn	Nữ	Vai tro 170
2021590	Giang vien 185	teacher185@university.edu.vn	Nữ	Vai tro 185
5010295	Giang vien 192	teacher192@university.edu.vn	Nam	Vai tro 192
1681350	Giang vien 3	lyvinhthai3210@gmail.com	Nam	Vai tro 3
7057846	Giang vien 27	teacher27@university.edu.vn	Nữ	Vai tro 27
3881850	Giang vien 46	teacher46@university.edu.vn	Nữ	Vai tro 46
1584739	Giang vien 109	teacher109@university.edu.vn	Nam	Vai tro 109
9256391	Giang vien 122	teacher122@university.edu.vn	Nam	Vai tro 122

cho người dùng mới.

## Hình 5. Dashboard: Quản lý Giảng viên

Đây là giao diện quản lý dành cho các vai trò như Giáo vụ hoặc Giáo viên bộ môn. Chức năng chính là tra cứu và quản lý thông tin giảng viên trong khoa (Mã GV, Họ tên, Email, Giới tính, Vai trò). Giao diện tích hợp bộ lọc theo Học kỳ và khu vực Quản lý vai trò (sidebar) để truy cập nhanh các chức năng khác, đồng thời cung cấp tùy chọn tìm kiếm theo từ khóa và phân trang để xử lý tập dữ liệu lớn.

MÃ SV	HỌ TÊN	EMAIL	SỐ ĐIỆN THOẠI	LỚP	GIỚI TÍNH
3251117	Sinh viên 951	student951@university.edu.vn	0578704799	CLASS_42	Nam
2523719	Sinh viên 967	student967@university.edu.vn	047660504	CLASS_49	Nữ
1025952	Sinh viên 964	student964@university.edu.vn	0742431686	CLASS_41	Nữ
1587561	Sinh viên 955	student955@university.edu.vn	0798196933	CLASS_14	Nam
8106636	Sinh viên 940	student940@university.edu.vn	0898184098	CLASS_05	Nam
4847400	Sinh viên 767	student767@university.edu.vn	0942389896	CLASS_06	Nữ
5430979	Sinh viên 764	student764@university.edu.vn	0828841642	CLASS_33	Nữ
2120081	Sinh viên 795	student795@university.edu.vn	0813343510	CLASS_37	Nam
2843608	Sinh viên 826	student826@university.edu.vn	0402449074	CLASS_27	Nam
5458528	Sinh viên 804	student804@university.edu.vn	0684787606	CLASS_02	Nữ

**Hình 6.** Dashboard: Quản lý Sinh viên

Đây là giao diện quản lý dành cho các vai trò như Giáo viên bộ môn hoặc Giáo vụ . Chức năng chính là tra cứu, quản lý, và xem thông tin chi tiết của sinh viên trong khoa.

Bộ lọc và Tìm kiếm: Người dùng có thể tìm kiếm sinh viên theo các trường thông tin quan trọng như tên, email, hoặc mã số sinh viên và sử dụng các bộ lọc theo Học kỳ (hiển thị tại tiêu đề).

Hiển thị dữ liệu: Bảng dữ liệu chính hiển thị các cột thông tin cốt lõi bao gồm Mã SV, Họ tên, Email, Số điện thoại, Lớp, và Giới tính. Tên sinh viên được làm nổi bật để liên kết tới trang chi tiết của sinh viên đó.

Điều hướng nhanh: Giao diện có thanh điều hướng (sidebar) bên trái, cho phép chuyển đổi nhanh chóng giữa các chức năng quản lý khác nhau như Quản lý giảng viên, Quản lý đê tài, Lịch bảo vệ, v.v.

Xử lý dữ liệu lớn: Tích hợp tính năng phân trang và tùy chọn số lượng dòng hiển thị (Mặc định: 10) để quản lý hiệu quả tập dữ liệu sinh viên lớn (Ví dụ: Hiển thị 1 - 10 trên tổng số 52 sinh viên).

MÃ ĐỀ TÀI	TÊN ĐỀ TÀI	MÃ KHOA	TRẠNG THÁI	TIẾN ĐỘ	THAO TÁC
TOP_0009...	Đề tài 900 - MAJ_CNTT_KHMT	MAJ_CNTT_KHMT	Đã nộp	Stage 1: 0% Stage 2: 0%	
TOP_0008...	Đề tài 892 - MAJ_CNTT_AI	MAJ_CNTT_AI	Đang thực hiện	Stage 1: 0% Stage 2: 0%	
TOP_0008...	Đề tài 857 - MAJ_CNTT_ATTT	MAJ_CNTT_ATTT	Từ chối	Stage 1: 0% Stage 2: 0%	
TOP_0008...	Đề tài 851 - MAJ_CNTT_KHMT	MAJ_CNTT_KHMT	Đang thực hiện	Stage 1: 0% Stage 2: 0%	
TOP_0009...	Đề tài 977 - MAJ_CNTT_KHMT	MAJ_CNTT_KHMT	Từ chối	Stage 1: 0% Stage 2: 0%	
TOP_0009...	Đề tài 940 - MAJ_CNTT_KTPM	MAJ_CNTT_KTPM	Đang thực hiện	Stage 1: 0% Stage 2: 0%	
TOP_0007...	Đề tài 709 - MAJ_CNTT_HTTT	MAJ_CNTT_HTTT	Đang thực hiện	Stage 1: 0% Stage 2: 0%	
TOP_0007...	Đề tài 743 - MAJ_CNTT_HTTT	MAJ_CNTT_HTTT	Đang thực hiện	Stage 1: 0% Stage 2: 0%	
TOP_0006...	Đề tài 686 - MAJ_CNTT_KTPM	MAJ_CNTT_KTPM	Đang thực hiện	Stage 1: 0% Stage 2: 0%	
TOP_0005...	Đề tài 556 - MAJ_CNTT_HTTT	MAJ_CNTT_HTTT	Đã nộp	Stage 1: 0% Stage 2: 0%	

**Hình 7.** Giao diện Quản lý đề tài

Giao diện này dành cho Giáo viên bộ môn hoặc Giáo vụ để duyệt và quản lý toàn bộ đề tài thuộc khoa trong một học kỳ cụ thể.

Chức năng chính: Hiển thị danh sách các đề tài với các thông tin cốt lõi như Mã đề tài, Tên đề tài, Khoa quản lý, Trạng thái, và Tiến độ (Stage 1%, Stage 2%).

Quản lý trạng thái: Người dùng có thể nhanh chóng tra cứu bằng công cụ Tìm kiếm theo tên đề tài hoặc lọc theo Trạng thái của đề tài (Đã nộp, Đang thực hiện, Từ chối).

Tác vụ: Cung cấp các thao tác như Import/Export dữ liệu và các Thao tác quản lý (chỉnh sửa, xóa hoặc tải file) ngay trên mỗi dòng.

Điều hướng: Tích hợp sidebar (Giáo viên bộ môn/Giáo vụ) cho phép chuyển đổi giữa các chức năng quản lý khác như Quản lý giảng viên, sinh viên, hội đồng, v.v.

MÃ HD	TÊN HỘI ĐỒNG	KHOA	THÀNH VIÊN	SỐ ĐỀ TÀI
27f2aef...	áscisd	MAJ_CNTT_HTTT	1	2
6a82792c...	ádas	MAJ_CNTT_AI	0	3
COU_0005...	Hoi dong bao ve 59	MAJ_CNTT_AI	6	5
COU_0008...	Hoi dong bao ve 82	MAJ_CNTT_KHMT	6	7

**Hình 8.** Dashboard: Quản lý Hội đồng

Giao diện này dành cho các vai trò quản lý (Giáo vụ/Giáo viên bộ môn) để tạo mới, tra cứu và quản lý các Hội đồng bảo vệ khóa luận theo từng học kỳ.

Chức năng chính: Hiển thị danh sách Hội đồng với các thông tin cốt lõi như Mã HD, Tên Hội đồng, Khoa, Số lượng Thành viên, và Số lượng Đề tài được phân vào hội đồng đó.

Tác vụ: Cung cấp nút "Tạo Hội đồng" và công cụ Tìm kiếm theo tên Hội đồng. Người dùng có thể click vào tên Hội đồng để xem hoặc chỉnh sửa thông tin chi tiết (danh sách thành viên, lịch làm việc).

Điều hướng: Tích hợp sidebar cho phép chuyển đổi giữa các chức năng quản lý khác như Quản lý giảng viên, sinh viên, đề tài, v.v., giúp quản lý quy trình bảo vệ một cách đồng bộ.

Bộ lọc: Tích hợp Bộ lọc Học kỳ ngay trên tiêu đề để xem các Hội đồng được thiết lập trong kỳ học hiện tại.

**Hình 9.** Dashboard: Lịch bảo vệ

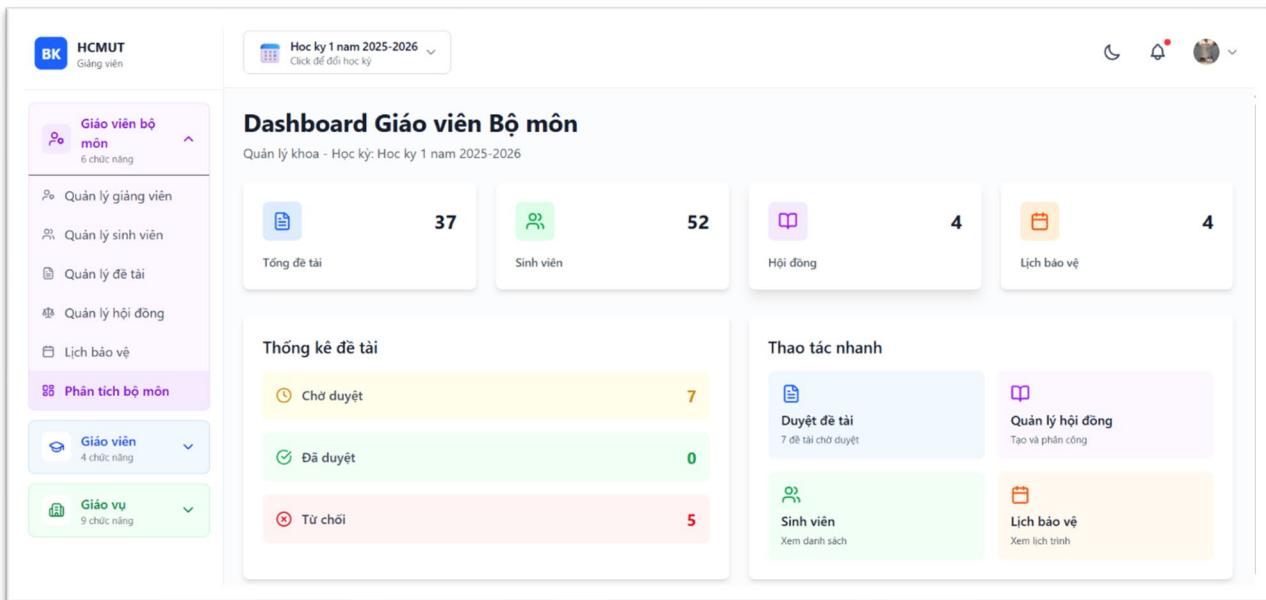
Giao diện này dành cho các vai trò quản lý (Giáo vụ/Giáo viên bộ môn) để theo dõi và xem chi tiết lịch bảo vệ của tất cả các hội đồng trong học kỳ.

Chức năng tổng quan: Cung cấp thông tin thống kê nhanh về Tổng buổi bảo vệ (1), Số lượng Hội đồng (4), và Số lượng Đề tài (1) được gán.

Chi tiết lịch: Hiển thị danh sách các buổi bảo vệ theo ngày (Ví dụ: 11/5/2025) và cung cấp chi tiết từng slot bảo vệ (Topic Council 517), bao gồm thời gian, số lượng sinh viên tham gia, và thông tin Giảng viên trong hội đồng (Ví dụ: Giảng viên 7 - PRESIDENT).

Tra cứu: Cho phép người dùng Tìm kiếm theo mã hội đồng hoặc sinh viên.

Điều hướng: Tích hợp bộ lọc Học kỳ và thanh sidebar để chuyển đổi giữa các chức năng quản lý khác.



**Hình 10.** Dashboard: Giáo viên Bộ môn

Giao diện này đóng vai trò là màn hình tổng quan dành cho Giáo viên Bộ môn, cung cấp các số liệu và thống kê quan trọng của học kỳ hiện tại.

Thống kê nhanh: Hiển thị các chỉ số quan trọng theo thời gian thực như Tổng đề tài (37), Tổng sinh viên (52), Tổng Hội đồng (4), và Lịch bảo vệ (4).

Phân tích: Cung cấp biểu đồ hoặc mục Thống kê đề tài chi tiết về trạng thái của các đề tài đang chờ duyệt (Chờ duyệt: 7, Đã duyệt: 0, Từ chối: 5).

Thao tác nhanh: Tích hợp các nút hành động nhanh như Duyệt đề tài, Quản lý hội đồng, và Xem danh sách sinh viên để chuyển trực tiếp đến các chức năng quản lý cốt lõi.

Điều hướng: Thanh sidebar bên trái cho phép truy cập các phân hệ quản lý chi tiết khác (Quản lý giảng viên, sinh viên, đề tài, v.v.).

**Gửi đề tài mới**

Đề xuất đề tài luận văn cho sinh viên

Chọn cách tạo đề tài

**Tạo đề tài mới**  
Tạo đề tài hoàn toàn mới cho giải đoạn 1 (ĐACN) hoặc giải đoạn 2 (LVTN)

**Tạo từ đề tài giải đoạn 1**  
Chuyển đề tài đã hoàn thành giải đoạn 1 sang giải đoạn 2 (LVTN)

**Tên đề tài (Tiếng Việt) \***  
Nhập tên đề tài bằng tiếng Việt (ít nhất 10 ký tự)

**Tên đề tài (Tiếng Anh)**  
Nhập tên đề tài bằng tiếng Anh (hoặc click Dịch tự động)  
**Dịch**

**Chương trình đào tạo**  
Ví dụ: Đại học chính quy, Cao học...

**Loại đề tài \***  
Đề án chuyên ngành (ĐACN - Topic 1)

ĐACN (Topic 1) dành cho đề tài giải đoạn 1, LVTN (Topic 2) dành cho đề tài giải đoạn 2

**Số lượng sinh viên tối đa**  
1  
Số lượng sinh viên tối đa có thể đăng ký đề tài này (máy định 1)

**Sinh viên thực hiện (0/1)**  
Nhập MSSV sinh viên (vd: 2052001)  
**Thêm**

**Ngày bắt đầu \***  
mm/dd/yyyy --:-- --

**Ngày kết thúc \***  
mm/dd/yyyy --:-- --

**Mô tả chi tiết đề tài \***  
Bao gồm: Mô tả tổng quan, nhiệm vụ cụ thể, tài liệu tham khảo, các giải đoạn thực hiện...

Lưu ý khi gửi đề tài

- Đề tài sẽ được gửi đến bộ môn để xem xét và phê duyệt
- Tên đề tài phải rõ ràng, cụ thể và phản ánh đúng nội dung nghiên cứu
- Mô tả đề tài cần chi tiết về mục tiêu, phạm vi và phương pháp thực hiện
- Sử dụng nút "Dịch" để tự động dịch tên đề tài sang tiếng Anh
- Sau khi được phê duyệt, đề tài sẽ được hiển thị cho sinh viên đăng ký

**Hủy** **Gửi đề tài**

**Hình 11.** Dashboard: Gửi đề tài

Giao diện này dành cho Sinh viên hoặc Giảng viên để khởi tạo và nộp đề xuất đề tài luận văn.

**Chức năng chính:** Người dùng phải cung cấp đầy đủ các thông tin cốt lõi như Tên đề tài (Tiếng Việt & Tiếng Anh), Chương trình đào tạo, Loại đề tài (Đồ án chuyên ngành/Luận văn tốt nghiệp) và Số lượng sinh viên tham gia.

**Thao tác:** Tích hợp trình soạn thảo văn bản nâng cao để nhập Mô tả chi tiết cho đề tài.

**Quy trình:** Giao diện nhán mạnh quy trình gửi đề tài: đề tài sẽ được gửi đi để chờ xem xét và phê duyệt trước khi chính thức được chuyển sang giai đoạn sinh viên đăng ký.

**Điều hướng:** Cung cấp tùy chọn "Tạo đề tài mới" hoặc "Tạo từ đề tài giai đoạn 1" (chuyển tiếp từ Đồ án chuyên ngành sang Luận văn tốt nghiệp).

TÊN ĐỀ TÀI	GIAI ĐOAN	SINH VIÊN (MSSV)	TRẠNG THÁI	TIẾN ĐỘ	LỊCH BÁO VỆ	THAO TÁC
Đề tài 910 - MAJ_CNTT_KTPM	STAGE_LVTN	3300454 - Sinh viên 910	Đang thực hiện	Stage 1: 0% Stage 2: 61%	2/1/2026 11:47	
Đề tài 681 - MAJ_KTXD_XD	STAGE_LVTN	3952780 - Sinh viên 681	Chờ duyệt	Stage 1: 38% Stage 2: 44%	14/12/2025 11:47	
Đề tài 691 - MAJ_CNTT_AI	STAGE_DACN	6435499 - Sinh viên 691	Tù chối	Stage 1: 67% Stage 2: 36%	13/12/2025 11:47	
Đề tài 705 - MAJ_CNTT_HTTT	STAGE_LVTN	8350037 - Sinh viên 705	Đang thực hiện	Stage 1: 0% Stage 2: 77%	9/12/2025 11:47	3. Tải xuống
Đề tài 529 - MAJ_CNTT_ATTT	STAGE_DACN	5905152 - Sinh viên 529	Chờ duyệt	Stage 1: 38% Stage 2: 55%	Chưa có lịch	4. Tải xuống

**Hình 12.** Dashboard: Đề tài đang hướng dẫn

Giao diện này dành cho Giảng viên để quản lý và theo dõi tiến độ các đề tài mà mình đang trực tiếp hướng dẫn trong học kỳ hiện tại.

Chức năng chính: Hiển thị danh sách đề tài với các thông tin cốt lõi bao gồm Tên đề tài, Giai đoạn (ĐACN/LVTN), Sinh viên (MSSV) thực hiện, Trạng thái duyệt, Tiến độ hoàn thành (Stage 1/Stage 2), và Lịch bảo vệ dự kiến.

Tác vụ: Giảng viên có thể Tìm kiếm theo tên đề tài hoặc mã số sinh viên, và sử dụng chức năng Thao tác (như tải file) để tương tác với đề tài.

Điều hướng: Tích hợp thanh sidebar cho phép chuyển đổi giữa các chức năng giảng viên khác như Gửi đề tài mới, Hội đồng chấm điểm, và Lịch bảo vệ.

Xử lý dữ liệu: Hỗ trợ tính năng Import/Export dữ liệu và phân trang để xử lý danh sách đề tài hiệu quả.

The screenshot shows a web-based dashboard for a lecturer. At the top left is the logo 'BK HCMUT Giảng viên'. A dropdown menu indicates 'Học kỳ 1 năm 2025-2026' and 'Click để đổi học kỳ'. On the right are icons for notifications, messages, and user profile.

The main content area is titled 'Hội đồng chấm điểm' (Exam Committee). It displays a list of committees the lecturer has joined:

MÃ HỘI ĐỒNG	TÊN HỘI ĐỒNG	CHỨC VỤ	SỐ SINH VIÊN ĐÃ CHẤM	THỜI GIAN BẢO VỆ
27f2aefa-fc7c-4647-b731-9a551b237da8	âsdasd MAJ_CNTT_HTTT • SEM_2025_1	Chủ tịch	0 / 0	Chưa có lịch
COU_00072	Hoi dong bao ve 72 MAJ_DTVT_VT • SEM_2025_1	Thư ký	1 / 1	25/11/2025 11:47

Below the table, there are buttons for 'Tim kiem' (Search), 'Import', 'Export', and a refresh icon. At the bottom, it says 'Hiển thị 1 - 2 của 2' and includes page navigation buttons for 'Số dòng: 10', 'Trước', 'Sau', and a page number '1'.

**Hình 13.** Dashboard: Hội đồng chấm điểm

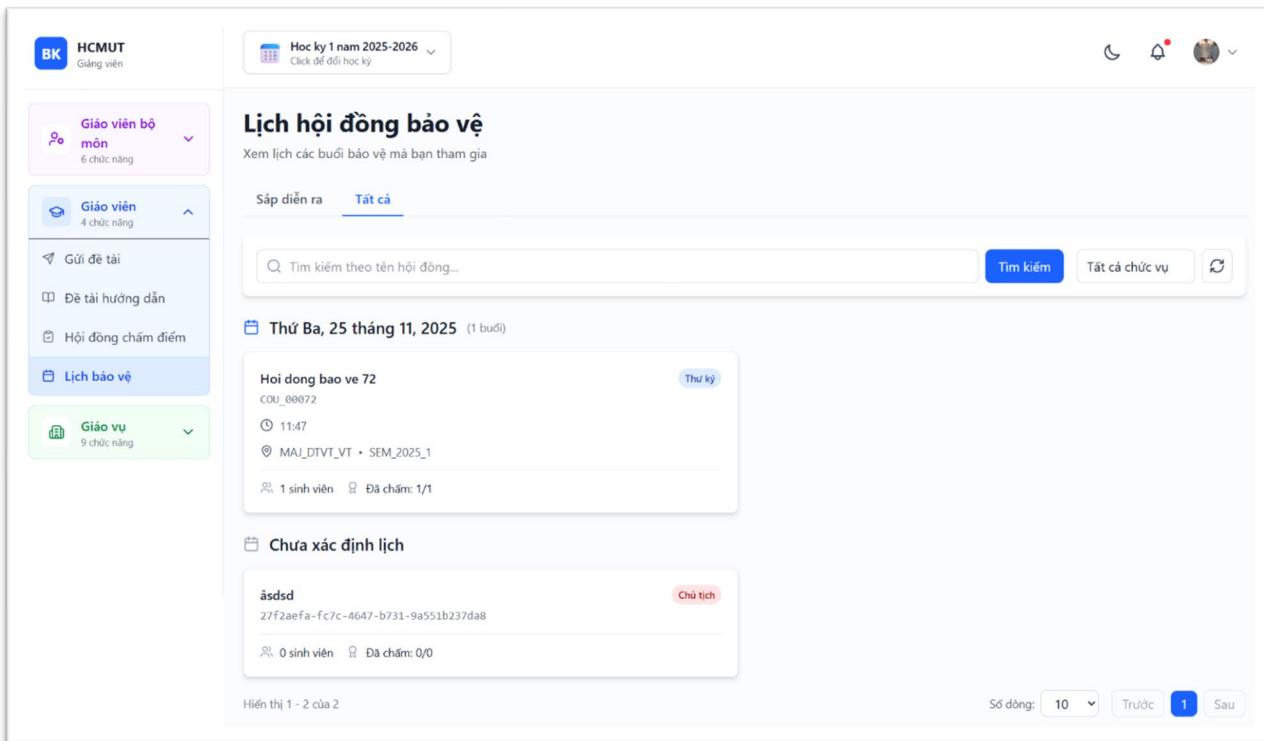
Giao diện này dành riêng cho Giảng viên để quản lý và theo dõi các hội đồng mà họ được phân công tham gia chấm điểm.

**Chức năng chính:** Hiển thị danh sách Hội đồng mà giảng viên là thành viên, kèm theo thông tin quan trọng như Mã Hội đồng, Tên Hội đồng, Chức vụ của giảng viên (Chủ tịch, Thư ký, v.v.), Số sinh viên đã chấm và Thời gian bảo vệ.

**Tác vụ:** Giảng viên có thể Tìm kiếm theo tên hội đồng và lọc theo Chức vụ của mình. Các nút Import/Export được tích hợp để xử lý dữ liệu liên quan đến chấm điểm.

**Quy trình:** Giảng viên sử dụng giao diện này để theo dõi tiến độ chấm điểm, đảm bảo hoàn thành vai trò của mình trong quy trình bảo vệ khóa luận.

**Điều hướng:** Tích hợp sidebar giúp giảng viên chuyển đổi nhanh giữa các chức năng giảng viên khác như Gửi đề tài và Đề tài hướng dẫn.



**Hình 14.** Dashboard: Lịch hội đồng bảo vệ

Giao diện này dành cho các vai trò quản lý (Giáo vụ/Giáo viên bộ môn) để tra cứu và quản lý danh sách Giảng viên trong khoa.

**Chức năng chính:** Hiển thị thông tin chi tiết của Giảng viên (Mã GV, Họ tên, Email, Giới tính, Vai trò) dưới dạng bảng dữ liệu.

**Tìm kiếm & Lọc:** Người dùng có thể nhanh chóng Tìm kiếm theo tên hoặc email của giảng viên. Tên giảng viên được làm nổi bật để liên kết tới trang chi tiết (nếu có).

**Điều hướng & Phân trang:** Giao diện có thanh sidebar cố định cho phép chuyển đổi giữa các chức năng quản lý khác, và tích hợp phân trang để xử lý hiệu quả danh sách giảng viên lớn.

**Quản lý vai trò:** Cột Vai trò cho phép người quản lý nhanh chóng xác định các chức vụ của giảng viên trong các học kỳ khác nhau.

MÃ HỌC KỲ	TÊN HỌC KỲ	NGÀY TẠO	CẤP NHẬT	THAO TÁC
hoc-ky-2-nam-2025-2026	Học kỳ 2 năm 2025-2026	21/11/2025	21/11/2025	
SEM_2023_1	Học kỳ 1 năm 2023-2024	14/10/2025	14/10/2025	
SEM_2023_2	Học kỳ 2 năm 2023-2024	14/10/2025	14/10/2025	
SEM_2024_1	Học kỳ 1 năm 2024-2025	14/10/2025	14/10/2025	
SEM_2024_2	Học kỳ 2 năm 2024-2025	14/10/2025	14/10/2025	
SEM_2025_1	Học kỳ 1 năm 2025-2026	14/10/2025	14/10/2025	

**Hình 15.** Dashboard: Quản lý học kỳ

Giao diện này dành riêng cho Giảng viên để xem lịch trình bảo vệ chi tiết của các hội đồng mà họ được phân công tham gia chấm điểm.

**Chức năng chính:** Hiển thị lịch bảo vệ theo dòng thời gian (Ngày, giờ) dưới hai chế độ: Sắp diễn ra và Tất cả. Mỗi mục hiển thị chi tiết về Tên Hội đồng, Mã Hội đồng, Chức vụ của giảng viên trong hội đồng (Ví dụ: Thư ký, Chủ tịch), và số lượng sinh viên tham gia/đã chấm.

**Tác vụ:** Giảng viên có thể Tìm kiếm theo tên hội đồng và lọc theo Chức vụ của mình để nhanh chóng xác định các buổi làm việc cần tham gia.

**Quy trình:** Giúp giảng viên theo dõi lịch làm việc cá nhân, đảm bảo có mặt và thực hiện nhiệm vụ chấm điểm đúng giờ.

**Điều hướng:** Tích hợp sidebar giúp giảng viên chuyển đổi nhanh giữa các chức năng giảng viên khác (Đề tài hướng dẫn, Hội đồng chấm điểm, Gửi đề tài).

MSSV	HỌ TÊN	EMAIL	SĐT	LỚP	KHOA	THAO TÁC
2213106	Lý Thái	admin@thailly.id.vn	0366063879	KHMT07KH	MAJ_CNTT_HTTT	
2213105	Thái Lý	thai.lydanhthatlop2k4@hcmut.edu.vn	0366063879	KHMT07KH	MAJ_CNTT_HTTT	
2213104	Lý Vinh Thái	lyvinhthai321@gmail.com	0366063879	KHMT07KH	MAJ_CNTT_KTPM	
7510375	Sinh viên 1000	student1000@university.edu.vn	0870470212	CLASS_09	MAJ_KTXD_GT	
1265304	Sinh viên 918	student918@university.edu.vn	0534389293	CLASS_48	MAJ_CNSH_CNTP	
2355945	Sinh viên 922	student922@university.edu.vn	0619733513	CLASS_28	MAJ_KHTN_LY	
5460193	Sinh viên 920	student920@university.edu.vn	0566162082	CLASS_18	MAJ_CNSH_CNSH	
2891767	Sinh viên 917	student917@university.edu.vn	0897936036	CLASS_45	MAJ_KTXD_GT	
1064510	Sinh viên 916	student916@university.edu.vn	0650293384	CLASS_08	MAJ_KHTN_HOA	
5651221	Sinh viên 919	student919@university.edu.vn	0525778124	CLASS_49	MAJ_CNTT_AI	

**Hình 16.** Dashboard: Quản lý người dùng

Giao diện này dành cho vai trò Giáo vụ để tạo mới, chỉnh sửa và quản lý toàn bộ các học kỳ trong hệ thống.

Chức năng chính: Hiển thị danh sách các học kỳ hiện có dưới dạng bảng, bao gồm Mã học kỳ, Tên học kỳ, Ngày tạo, và Ngày cập nhật lần cuối.

Tác vụ quản lý: Cung cấp nút "Tạo học kỳ mới" và các thao tác chỉnh sửa (biểu tượng bút chì) hoặc xóa (biểu tượng thùng rác) cho từng bản ghi học kỳ.

Tra cứu: Hỗ trợ chức năng Tìm kiếm theo Mã hoặc Tên học kỳ để nhanh chóng tra cứu thông tin học kỳ cụ thể.

Điều hướng: Tích hợp thanh sidebar cho phép Giáo vụ chuyển đổi giữa các chức năng quản lý cấp cao khác như Quản lý người dùng, topic, hội đồng, v.v.

MÃ ĐỀ TÀI	TÊN ĐỀ TÀI	GIÁNG VIÊN HƯỚNG DẪN	TRẠNG THÁI	GIAI ĐOẠN	THAO TÁC
2e982d13...	Testing cực gắt	Lý Vĩnh Thái	Chờ duyệt	Giai đoạn 1 (DACN)	
aebfa605...	Học làm hack hệ thống của bạn Cận	Lý Vĩnh Thái	Chờ duyệt	Giai đoạn 1 (DACN)	
c6ee3e78...	Học làm hack hệ thống của Cường	Lý Vĩnh Thái	Chờ duyệt	Giai đoạn 1 (DACN)	
c344f422...	Cơ bản về Frontend nextjs	Lý Vĩnh Thái	Chờ duyệt	Giai đoạn 1 (DACN)	
396794cc...	An toàn thông tin	Lý Vĩnh Thái	Đang thực hiện	Giai đoạn 1 (DACN)	
92965d77...	xoài cát hòa lộc tại tiền giang	Lý Vĩnh Thái	Chờ duyệt	Giai đoạn 1 (DACN)	
01f7edcf...	Kiểm tra phần mềm	Lý Vĩnh Thái	Đang thực hiện	Giai đoạn 1 (DACN)	
TOP_0008...	De tai 839 - MAJ_KTCK_CHN	Giang viên 144	Chờ duyệt	Giai đoạn 2 (LVTN)	
TOP_0008...	De tai 840 - MAJ_CNTT_KHMT	Giang viên 191	Chờ duyệt	Giai đoạn 1 (DACN)	
TOP_0008...	De tai 841 - MAJ_DTVT_KT	Giang viên 123	Đang thực hiện	Giai đoạn 1 (DACN)	

**Hình 17.** Dashboard: Quản lý đề tài

Giao diện này dành cho vai trò Giáo vụ để quản lý và phê duyệt toàn bộ các đề tài luận văn trong hệ thống, không giới hạn theo khoa.

Chức năng chính: Hiển thị danh sách đề tài chi tiết, bao gồm Mã đề tài, Tên đề tài, Giảng viên hướng dẫn, Trạng thái (Chờ duyệt/Đang thực hiện), và Giai đoạn (ĐACN/LVTN).

Quản lý trạng thái: Cung cấp công cụ Tìm kiếm theo tên đề tài và các bộ lọc theo Học kỳ và Trạng thái để quản lý quy trình phê duyệt đề tài trên toàn trường.

Tác vụ: Người dùng có thể thực hiện các Thao tác quản lý như chỉnh sửa, xóa, hoặc chuyển trạng thái đề tài (duyệt/từ chối) ngay trên bảng dữ liệu.

Điều hướng: Tích hợp thanh sidebar bên trái cho phép Giáo vụ chuyển đổi giữa các chức năng quản lý cấp cao khác như Quản lý người dùng, học kỳ, hội đồng, v.v.

**Hình 18.** Dashboard: Quản lý lịch Bảo vệ

Giao diện này dành cho vai trò Giáo vụ để tổng quan, xem và quản lý lịch bảo vệ của tất cả các hội đồng dưới dạng lịch (Calendar view).

Chức năng chính: Hiển thị tổng quan các số liệu (Tổng số hội đồng: 2, Tổng đề tài: 2, Sinh viên bảo vệ: 4). Lịch lớn hiển thị các buổi bảo vệ theo ngày, giúp Giáo vụ dễ dàng theo dõi và tránh trùng lịch.

Tác vụ: Hỗ trợ chuyển đổi giữa Dạng lịch và Dạng danh sách, cùng với các bộ lọc theo Học kỳ và Ngành để thu hẹp phạm vi lịch trình cần xem.

Tra cứu: Cho phép Tìm kiếm theo tên hội đồng hoặc đề tài.

**Điều hướng:** Tích hợp thanh sidebar cho phép Giáo vụ chuyển đổi giữa các chức năng quản lý cấp cao khác, đảm bảo tính đồng bộ trong quản lý lịch trình.

MÃ HD	TÊN HỘI ĐỒNG	NGÀNH	THỜI GIAN	SỐ ĐỀ TÀI	THAO TÁC
d0c8480f...	Hội đồng testing	MAJ_CNTT_HTTT	22:02:00 30/11/2025	1	
b21ae632...	Hội đồng hacker VN	MAJ_CNTT_HTTT	Chưa gán giờ	1	

### Hình 19. Dashboard: Quản lý Hội đồng

Giao diện này dành cho vai trò Giáo vụ để quản lý tập trung toàn bộ tài khoản Sinh viên và Giảng viên trong hệ thống.

**Chức năng chính:** Hiển thị danh sách người dùng với các thông tin chi tiết như MSSV/Họ tên, Email, SĐT, Lớp, và Khoa trực thuộc, với khả năng chuyển đổi giữa danh sách Sinh viên và Giảng viên qua tab.

**Tra cứu & Lọc:** Cung cấp công cụ Tìm kiếm theo tên, email hoặc MSSV, kết hợp các bộ lọc theo Học kỳ, Lớp, và Khoa để tra cứu dữ liệu chính xác.

**Tác vụ:** Hỗ trợ các thao tác quản lý quan trọng như Thêm mới, Import/Export danh sách người dùng, và các Thao tác chỉnh sửa/xóa từng tài khoản.

**Điều hướng:** Tích hợp thanh sidebar cho phép Giáo vụ chuyển đổi giữa các chức năng quản lý cấp cao khác (Học kỳ, Đề tài, Hội đồng).

MÃ CHUYÊN NGÀNH	TÊN CHUYÊN NGÀNH	KHOA	THAO TÁC
MAJ_KT_QTKD	Quan tri Kinh doanh	Khoa Kinh te	
MAJ_DTVT_VT	Kỹ thuật Viễn thông	Khoa Điện tử Viễn thông	
MAJ_KHTN_HOA	Hoa học	Khoa Khoa học Tu nhiên	
MAJ_KHTN_LY	Vật lý	Khoa Khoa học Tu nhiên	
MAJ_KHTN_TOAN	Toán học	Khoa Khoa học Tu nhiên	
MAJ_KT_KT	Kinh tế	Khoa Kinh te	
MAJ_KTXD_GT	Kỹ thuật Giao thông	Khoa Kỹ thuật Xây dựng	
MAJ_NN_ANH	Ngôn ngữ Anh	Khoa Ngoại ngữ	
MAJ_KTXD_XD	Kỹ thuật Xây dựng	Khoa Kỹ thuật Xây dựng	
MAJ_DTVT_TD	Kỹ thuật Tự động hóa	Khoa Điện tử Viễn thông	

Hiển thị 1 - 10 của 23 chuyên ngành

Số dòng: 10 | Trước | 1 | 2 | 3 | Sau

## Hình 20. Dashboard: Quản lý Chuyên ngành

Giao diện này dành cho vai trò Giáo vụ để tạo mới, chỉnh sửa và quản lý danh sách các Chuyên ngành (Bộ môn) trực thuộc các Khoa trong hệ thống.

Chức năng chính: Hiển thị danh sách Chuyên ngành dưới dạng bảng, bao gồm Mã chuyên ngành, Tên chuyên ngành, và Khoa quản lý trực tiếp.

Tác vụ quản lý: Cung cấp nút "Tạo chuyên ngành mới" và các thao tác chỉnh sửa/xóa cho từng bản ghi. Chức năng Tìm kiếm theo tên chuyên ngành và Lọc theo Khoa giúp tra cứu dữ liệu hiệu quả.

Điều hướng: Tích hợp thanh sidebar bên trái cho phép Giáo vụ chuyển đổi giữa các chức năng quản lý cấp cao khác như Quản lý người dùng, học kỳ, đề tài, và hội đồng.

Mục đích: Đảm bảo dữ liệu nền tảng về tổ chức được cập nhật, phục vụ cho việc gán sinh viên/giảng viên vào đúng bộ môn công tác.

Tên Khoa	Số chuyên ngành	Tùy chọn
KHMT Khoa Học Máy Tính	3	
FAC_CNSH Khoa Công nghệ Sinh học	2	
FAC_CNTT Khoa Công nghệ Thông tin	5	
FAC_DTVT Khoa Điện tử Viễn thông	3	
FAC_KHTN Khoa Khoa học Tự nhiên	3	
FAC_KT Khoa Kinh tế	2	
FAC_KTCK Khoa Kỹ thuật Cơ khí	2	
FAC_KTXD Khoa Kỹ thuật Xây dựng	2	
FAC_NN Khoa Ngoại ngữ	1	

Hiển thị 1 - 9 của 9 khoa

Số dòng: 10 | Trước | Sau

**Hình 21.** Dashboard: Quản lý Khoa

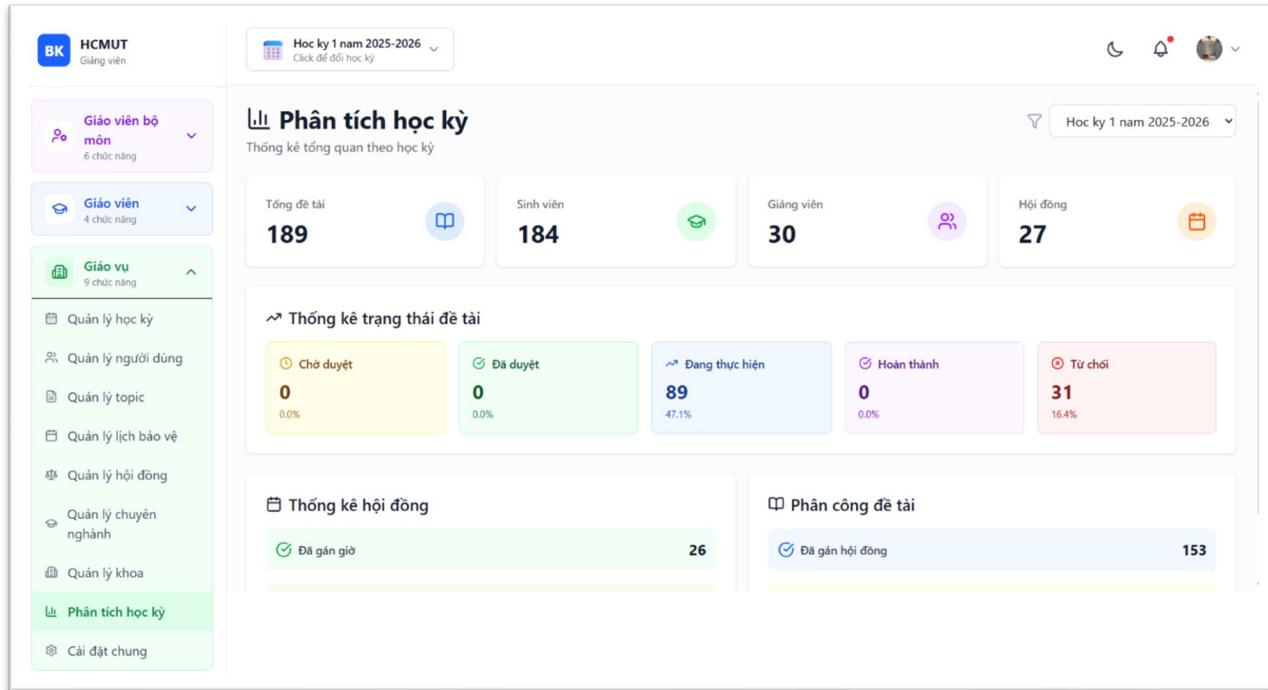
Giao diện này dành cho vai trò Giáo vụ để tạo mới, chỉnh sửa và quản lý danh sách các Khoa trong hệ thống.

Chức năng chính: Hiển thị danh sách các Khoa hiện có dưới dạng thẻ, bao gồm Mã Khoa, Tên Khoa, và số lượng chuyên ngành trực thuộc mỗi khoa.

Tác vụ quản lý: Cung cấp nút "Tạo Khoa mới" và các thao tác chỉnh sửa/xóa cho từng Khoa. Chức năng Tìm kiếm theo tên khoa giúp tra cứu dữ liệu hiệu quả.

Mục đích: Đảm bảo tính toàn vẹn của dữ liệu tổ chức, phục vụ cho việc phân quyền và gán các chuyên ngành.

**Điều hướng:** Tích hợp thanh sidebar bên trái cho phép Giáo vụ chuyển đổi giữa các chức năng quản lý cấp cao khác (Người dùng, Học kỳ, Chuyên ngành, v.v.).



**Hình 22.** Dashboard: Phân tích học kỳ

Giao diện này dành cho vai trò Giáo vụ để cung cấp thống kê tổng quan và phân tích chi tiết các hoạt động chính trong một học kỳ cụ thể.

**Chức năng chính:** Hiển thị các chỉ số tổng hợp quan trọng như Tổng đề tài (189), Tổng sinh viên (184), Tổng giảng viên (30), và Tổng Hội đồng (27).

**Phân tích:** Cung cấp biểu đồ hoặc mục thống kê chi tiết về trạng thái đề tài (Chờ duyệt, Đang thực hiện, Đã duyệt, Từ chối) và tỷ lệ gán Hội đồng/Lịch cho các đề tài, giúp người quản lý nắm bắt tình hình tức thì.

**Điều hướng:** Cho phép Giáo vụ chọn lọc theo Học kỳ để xem số liệu phân tích của các kỳ học khác nhau.

**Mục đích:** Hỗ trợ ra quyết định và đánh giá hiệu suất quản lý quy trình luận văn trong từng học kỳ.

### **3.2. Giao diện admin local (Vite)**

### **3.3. Thiết kế Figma (Mockup & Prototype)**

Giao diện người dùng được thiết kế bằng ứng dụng Figma, các thiết kế giống gần như 100% so với giao diện được lập trình. Có thể tham khảo các thiết kế ban đầu bằng các đường dẫn sau:

- Prototype có thể tương tác được: <https://bit.ly/49TN49v>
- Design Figma: <https://bit.ly/48ht6EC>

## **4. Thiết kế API và tích hợp**

### **4.1. API backend (Golang, Gin, GraphQL)**

API backend được phát triển bằng Golang, sử dụng framework Gin để xử lý các yêu cầu HTTP và GraphQL để cung cấp giao diện truy vấn linh hoạt. Trong cấu trúc mã nguồn, API được tổ chức qua thư mục dist với các controllers (auth.controller.js, cronjobs.controller.js, minio.controller.js, queues.controller.js, services.controller.js, workflows.controller.js) và routes (auth.routes.js, cronjobs.routes.js, minio.routes.js, queues.routes.js, services.routes.js, workflows.routes.js).

Các API hỗ trợ CRUD cho các entity như user, role, thesis, council, với middleware auth.middleware.js để kiểm tra quyền RBAC qua JWT. GraphQL được sử dụng để hỗ trợ lọc, phân trang và sắp xếp động, ví dụ trong queries (academic.queries.ts, student.queries.ts). Mục đích là cung cấp API thống nhất cho frontend, đảm bảo hiệu suất với thời gian phản hồi dưới 3 giây.

### **4.2. Tích hợp gRPC cho các service**

Tích hợp gRPC được sử dụng để giao tiếp nội bộ giữa server gateway và các service backend chính (thesis, role, user, council, file, academic). Trong project\_tree.txt, gRPC được hỗ trợ qua proto (plagiarism.proto) trong plagiarism, nhưng trong BE\_core, nó được triển khai qua external/grpc trong queue để gọi các service. gRPC đảm bảo giao tiếp hiệu quả với MySQL, hỗ trợ streaming cho các tác

vụ lớn như upload file. Ví dụ, service Thesis gọi gRPC đến service File để lưu trữ form đăng ký. Tích hợp này giảm latency, tăng bảo mật với chứng chỉ trong certs (ca.crt, client.crt), và dễ mở rộng cho các service mới.

#### **4.3. Tích hợp Redis cho hàng đợi và cache**

Redis được tích hợp để làm hàng đợi (queue) và cache, hỗ trợ xử lý bất đồng bộ và tăng tốc độ truy vấn. Trong project\_tree.txt, Redis được sử dụng trong BE\_core/queue (cronjob, external, internal) và backend workflow (BullMQ). Redis làm cache cho các truy vấn thường xuyên (danh sách đề tài theo học kỳ), giảm tải MySQL. Đối với hàng đợi, Redis xử lý tác vụ ngầm như kiểm tra trùng lặp đề tài qua BullMQ trong plagiarism. Tích hợp này đảm bảo hệ thống xử lý đồng thời 1000 yêu cầu mà không chậm trễ, với cronjob-init.js để khởi tạo queue.

#### **4.4. Tích hợp MinIO cho lưu trữ file**

MinIO được tích hợp để lưu trữ file (báo cáo, form chấm điểm, dung lượng  $\leq 200\text{MB}$ ). Trong BE\_core từ project\_tree.txt, MinIO được xử lý qua minio.controller.js, minio.routes.js và minio.model.js, với seed-minio-config.js để khởi tạo bucket. MinIO cung cấp lưu trữ object tương tự S3, hỗ trợ upload/download qua API, và tích hợp với service File qua gRPC. Tích hợp này đảm bảo bảo mật file với mã hóa và quyền truy cập theo RBAC, dễ mở rộng dung lượng lưu trữ.

### **5. Thiết kế quy trình ngầm và giám sát**

Thiết kế quy trình ngầm và giám sát đảm bảo hệ thống TMS hoạt động tự động và dễ theo dõi, sử dụng backend workflow cho tác vụ bất đồng bộ và các công cụ giám sát để phát hiện vấn đề kịp thời.

#### **5.1. Backend workflow (Express, BullMQ)**

Backend workflow được phát triển bằng Express.js để xử lý quy trình ngầm như kiểm tra đạo văn và gửi email cảnh báo. Trong cấu trúc mã nguồn, workflow nằm trong BE\_core/queue với BullMQ dựa trên Redis để quản lý job. Ví dụ, khi sinh viên nộp bài, workflow đẩy job vào Redis queue để kiểm tra trùng lặp qua plagiarism

(analyzer.py, detector.py). BullMQ đảm bảo xử lý song song, với cronjob-service.js để quản lý tác vụ định kỳ. Workflow này tích hợp với gateway qua HTTP, góp phần vào hiệu suất bất đồng bộ.

## 5.2. Cronjob và tác vụ định kỳ

Cronjob được thiết kế để chạy các tác vụ tự động, sử dụng Golang cron trong BE\_core/cronjob (cronjob-init.js, cronjob-service.js). Các tác vụ bao gồm: cập nhật trạng thái đê tài (quá hạn -> "Đã kết thúc"), kiểm tra lệch điểm hàng ngày, sao lưu dữ liệu (mongodump hàng ngày lúc 03:00), di chuyển dữ liệu >n năm khỏi hệ thống. Cronjob tích hợp với Redis để đẩy job và ghi log vào Loki, đảm bảo tự động hóa quy trình mà không cần can thiệp thủ công.

## 5.3. Giám sát với Grafana, Loki, Prometheus

Giám sát được thiết kế với Grafana làm UI trung tâm, hiển thị dashboard từ Prometheus (metric) và Loki (logs). Trong cấu trúc dự án, giám sát được hỗ trợ qua k6-tests (combined-load-test.js) để thu thập metric. Prometheus thu thập chỉ số hiệu suất (CPU, RAM, truy vấn) từ các service, với 6 instance Promtail (mỗi service một) để bắn logs lên Loki. Grafana cung cấp dashboard cho General Logs, Error Detail Logs, Performance Logs, Matrix Logs và 2 AI plagiarism logs từ plagiarism. Tích hợp này đảm bảo phát hiện lỗi kịp thời và tối ưu hóa hệ thống.

# V. Triển khai hệ thống

## 1. Môi trường triển khai

### 1.1. Cấu hình phần mềm

Môi trường triển khai của hệ thống TMS được thiết kế để đảm bảo tính ổn định, bảo mật và khả năng mở rộng. Hệ thống chạy trên hạ tầng đám mây (dự tính: AWS hoặc Azure) hoặc server nội bộ, với cấu hình phần mềm tập trung vào các công nghệ chính. Phiên bản Golang 1.20 được sử dụng cho backend, kết hợp với Gin 1.8 cho routing và GraphQL-go để xử lý truy vấn. Express.js 4.18 cho backend workflow, BullMQ 3.5 cho hàng đợi. Frontend sử dụng Next.js 13.0 và Vite 4.0. Cơ sở dữ liệu

bao gồm MySQL 8.0 (6 instances), MongoDB 7.0, Redis 7.0 và MinIO release 2023. Công cụ giám sát sử dụng Grafana 9.0, Loki 2.7, Prometheus 2.4, Promtail 2.7 và Elasticsearch 8.0. Cấu hình này đảm bảo hệ thống đáp ứng yêu cầu hiệu suất (phản hồi ≤3 giây) và dung lượng file ( $\leq 200\text{MB}$ ), với Docker để container hóa.

## **1.2. Thiết lập cơ sở dữ liệu và các Monitors (MySQL, MongoDB, Redis, MinIO, Grafana, Prometheus, Loki, Promtail)**

Thiết lập cơ sở dữ liệu và giám sát là bước quan trọng để đảm bảo dữ liệu an toàn và hệ thống hoạt động ổn định. 6 instance MySQL được thiết lập riêng cho từng service (thesis, role, user, council, file, academic), mỗi instance với cấu hình replication (master-slave) để sao lưu dữ liệu.

MongoDB được thiết lập với replica set (3 node) để lưu logs và dữ liệu phi cấu trúc. Redis được thiết lập với cluster mode để làm hàng đợi và cache, hỗ trợ persistence.

MinIO được thiết lập với distributed mode để lưu file, tích hợp S3 API. Đôi với giám sát, Prometheus thu thập metric từ các service qua exporter, Loki lưu logs từ Promtail (6 instance, mỗi service một), Grafana hiển thị dashboard (General Logs, Error Detail Logs, v.v.).

Promtail đẩy logs lên Loki từ các container. Thiết lập sử dụng Docker Compose cho môi trường dev và Kubernetes cho production, với cronjob sao lưu hàng ngày.

Các thiết lập chi tiết được trình bày trong mã nguồn.

## **2. Triển khai backend**

### **2.1. Triển khai các service Golang (thesis, role, user, council, file, academic)**

Mỗi service được build từ BE\_core, sử dụng gRPC server để lắng nghe yêu cầu. Service Thesis triển khai qua thesis.controller.js và model.js, kết nối MySQL để quản lý đề tài. Service Role xử lý RBAC với role.model.js. Service User triển khai xác thực với user.model.js và seed-admin.js. Service Council quản lý hội đồng với council.model.js. Service File tích hợp MinIO qua minio.model.js. Service

Academic quản lý học kỳ với academic.queries.ts. Các service được **deploy qua Docker**, với health check để đảm bảo sẵn sàng.

## 2.2. Triển khai server gateway (Gin, GraphQL)

Server gateway được triển khai bằng Golang với Gin để xử lý routing và GraphQL để cung cấp API linh hoạt. Từ cấu trúc mã nguồn, gateway sử dụng index.js để khởi chạy, auth.routes.js cho xác thực JWT, và services.routes.js để gọi gRPC client đến các service. Triển khai bao gồm middleware auth.middleware.js để kiểm tra quyền. Gateway kết nối Redis cho cache và MongoDB cho logs. **Deploy qua Docker** với port 8080, tích hợp load balancer để xử lý tải cao.

## 2.3. Triển khai backend workflow (Express, BullMQ)

Backend workflow được triển khai bằng Express.js để xử lý tác vụ ngầm. Từ queue trong BE\_core, sử dụng BullMQ để quản lý job trên Redis. Triển khai bao gồm cronjob-init.js để khởi tạo cronjob (sao lưu, di chuyển dữ liệu) và minio.service.js để xử lý file. Workflow gọi gRPC đến các service để cập nhật dữ liệu. **Deploy qua Docker**, với scaling để xử lý job song song.

# 3. Triển khai frontend

## 3.1. Triển khai Next.js cho giao diện chính

Frontend chính được triển khai bằng Next.js để xây dựng giao diện người dùng. Từ mã nguồn front-end, sử dụng nhiều components như auth và topic để hỗ trợ đăng nhập OAuth và quản lý đê tài. Triển khai bao gồm gọi GraphQL đến gateway. **Build và deploy qua Vercel**, với SSR để tăng hiệu suất.

## 3.2. Triển khai Vite cho admin local

Frontend admin local được triển khai bằng Vite để giám sát hệ thống. Sử dụng Vite để build nhanh, tích hợp với Grafana API để hiển thị dashboard. **Deploy cục bộ qua Docker**, chỉ truy cập nội bộ cho admin.

## **4. Triển khai giám sát và công cụ hỗ trợ**

### **4.1. Thiết lập Grafana, Loki, Prometheus, Promtail**

Giám sát được thiết lập với Prometheus để thu thập metric từ các service, Loki để lưu logs, Promtail (6 instance) để đẩy logs từ service lên Loki, và Grafana để hiển thị dashboard (General Logs, Error Detail Logs, v.v.). Thiết lập qua Docker Compose, với Prometheus exporter trong các container Golang.

### **4.2. Tích hợp Elasticsearch cho tìm kiếm full text**

Elasticsearch được tích hợp để hỗ trợ tìm kiếm full text (đề tài, bài nộp). Thiết lập với index cho theses.title và description, gateway gọi Elasticsearch qua client. Deploy qua Docker, đồng bộ dữ liệu từ MySQL/MongoDB.

## **VI. Kiểm thử và đánh giá**

### **1. Kế hoạch kiểm thử**

#### **1.1. Kiểm thử đơn vị (Unit test)**

#### **1.2. Kiểm thử tích hợp (Integration test)**

#### **1.3. Kiểm thử hệ thống (System test)**

## 1.4. Kiểm thử hiệu suất và tải (Performance test)

```

THRESHOLDS
http_req_duration
✓ 'p(95)<5000' p(95)=250.01ms

http_req_failed
✓ 'rate<0.2' rate=0.00%

TOTAL RESULTS
checks_total.....: 250056 751.866104/s
checks_succeeded...: 100.00% 250056 out of 250056
checks_failed.....: 0.00% 0 out of 250056

✓ status is 200
✓ no errors in response

CUSTOM
admin_councils_duration.....: avg=34.195004 min=0 med=3 max=1042 p(90)=92 p(95)=214
admin_enrollments_duration....: avg=40.704866 min=0 med=4 max=6931 p(90)=113 p(95)=242.45
admin_errors.....: 0.00% 0 out of 125028
admin_faculties_duration.....: avg=36.381371 min=0 med=4 max=1252 p(90)=89 p(95)=223.45
admin_grade_defences_duration.: avg=34.417075 min=0 med=3 max=958 p(90)=90 p(95)=229
admin_majors_duration.....: avg=39.99057 min=0 med=4 max=3689 p(90)=95 p(95)=236
admin_semesters_duration....: avg=37.13425 min=0 med=2 max=1115 p(90)=108 p(95)=263
admin_students_duration.....: avg=43.815361 min=0 med=3 max=1032 p(90)=133.9 p(95)=300.45
admin_teachers_duration.....: avg=44.757342 min=0 med=3 max=2029 p(90)=137.9 p(95)=297
admin_topics_duration.....: avg=39.458465 min=0 med=5 max=1101 p(90)=108.9 p(95)=233

HTTP
http_req_duration.....: avg=38.8ms min=360.57µs med=3.14ms max=6.93s p(90)=105.04ms p(95)=250.01ms
  { expected_response:true }....: avg=38.8ms min=360.57µs med=3.14ms max=6.93s p(90)=105.04ms p(95)=250.01ms
http_req_failed.....: 0.00% 0 out of 125028
http_reqs.....: 125028 375.933052/s

```

Hình 23. Kết quả Load/Stress Test (1)

```

HTTP
http_req_duration.....: avg=38.8ms min=360.57µs med=3.14ms max=6.93s p(90)=105.04ms p(95)=250.01ms
  { expected_response:true }....: avg=38.8ms min=360.57µs med=3.14ms max=6.93s p(90)=105.04ms p(95)=250.01ms
http_req_failed.....: 0.00% 0 out of 125028
http_reqs.....: 125028 375.933052/s

EXECUTION
iteration_duration.....: avg=4.08s min=3.71s med=3.8s max=10.67s p(90)=5.12s p(95)=5.68s
iterations.....: 13892 41.770339/s
vus.....: 6 min=2 max=300
vus_max.....: 300 min=300 max=300

NETWORK
data_received.....: 1.2 GB 3.6 MB/s
data_sent.....: 129 MB 387 kB/s

running (5m32.6s), 000/300 VUs, 13892 complete and 0 interrupted iterations
default ✓ [=====] 000/300 VUs 5m30s

✓ Stress Test completed

```

Hình 24. Kết quả Load/Stress test (2)

Báo cáo hiển thị kết quả kiểm thử hiệu năng nhằm đánh giá độ ổn định của hệ thống dưới tải. Các kết quả cho thấy 100% yêu cầu được xử lý thành công với mã trạng thái HTTP 200, không có lỗi nào xảy ra.

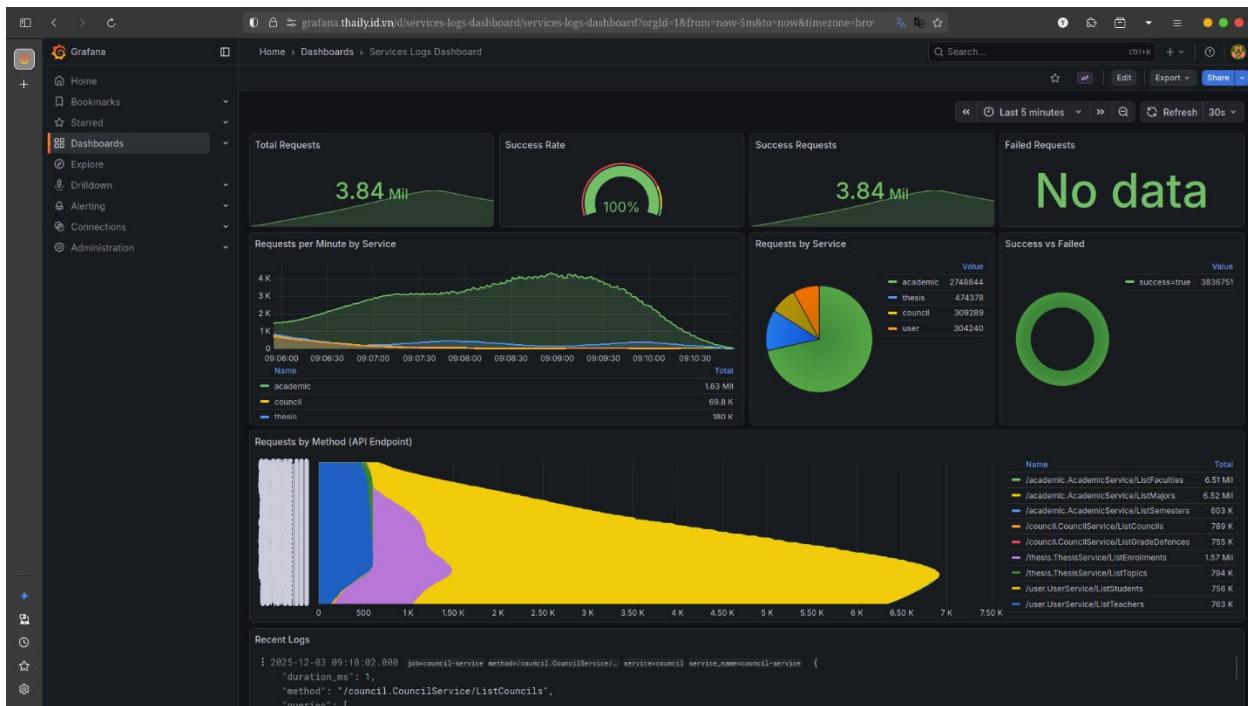
Độ trễ: Thời gian phản hồi trung bình (avg) của các API quản lý (như admin\_councils, admin\_teachers) nằm trong khoảng 34ms - 46ms, đảm bảo hệ thống phản hồi rất nhanh.

Nguồn an toàn: Độ trễ phân vị thứ 95 (p(95)) đạt 250.01ms, tuân thủ nghiêm ngặt nguồn yêu cầu là dưới 5000ms (5 giây) cho 95% yêu cầu.

Kết luận: Hệ thống luận văn đạt yêu cầu về hiệu suất và ổn định dưới tải, sẵn sàng triển khai.

## 2. Giám sát hệ thống

### 2.1. Service Log



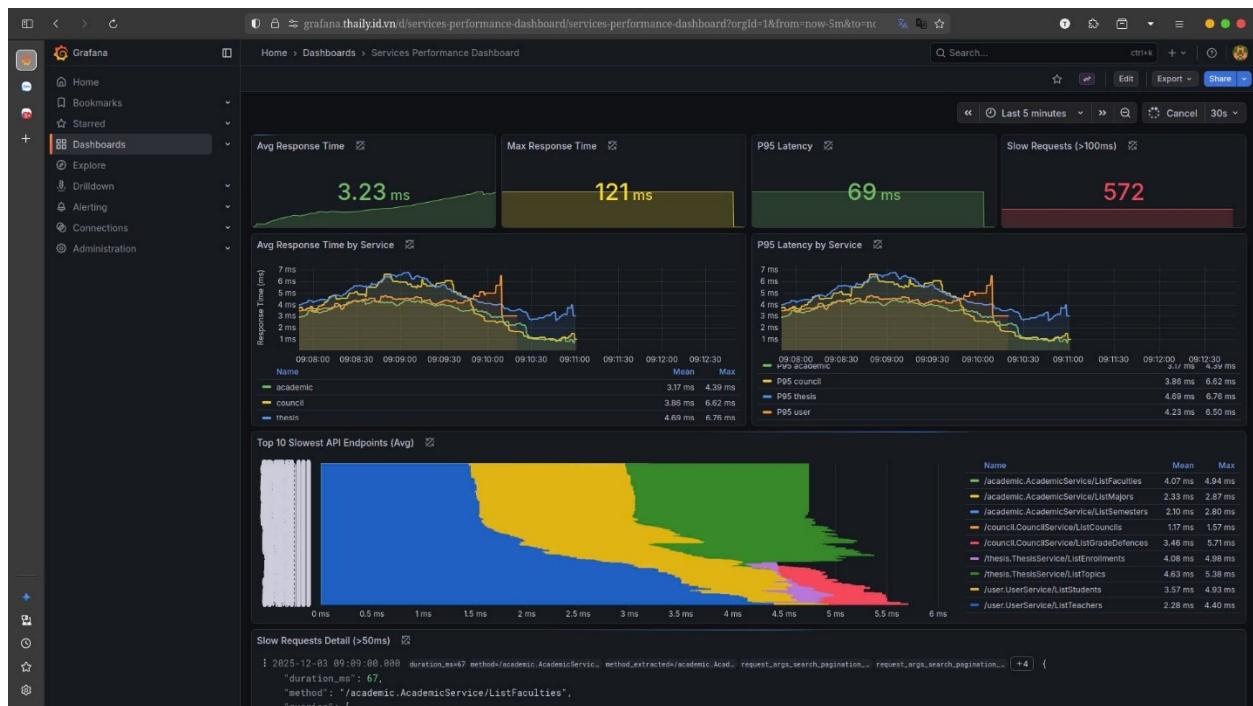
Hình 25. Dashboard Grafana: Services Logs

Giao diện này hiển thị trạng thái sức khỏe và lưu lượng truy cập thời gian thực của hệ thống backend (các Microservices) trong 5 phút gần nhất.

Biểu thị hiệu năng: Hệ thống đang xử lý tải rất cao với 3.84 triệu yêu cầu nhưng vẫn duy trì tỷ lệ thành công tuyệt đối 100% (Success Rate).

Phân tích chi tiết: Các biểu đồ giúp quản trị viên nhìn thấy rõ lưu lượng truy cập phân bổ vào dịch vụ nào nhiều nhất (Academic, Council, Thesis) và cụ thể từng API endpoint nào đang được gọi (Requests by Method) để tối ưu hóa hệ thống.

## 2.2. Service Performance



Hình 26. Dashboard Grafana: Services Performance

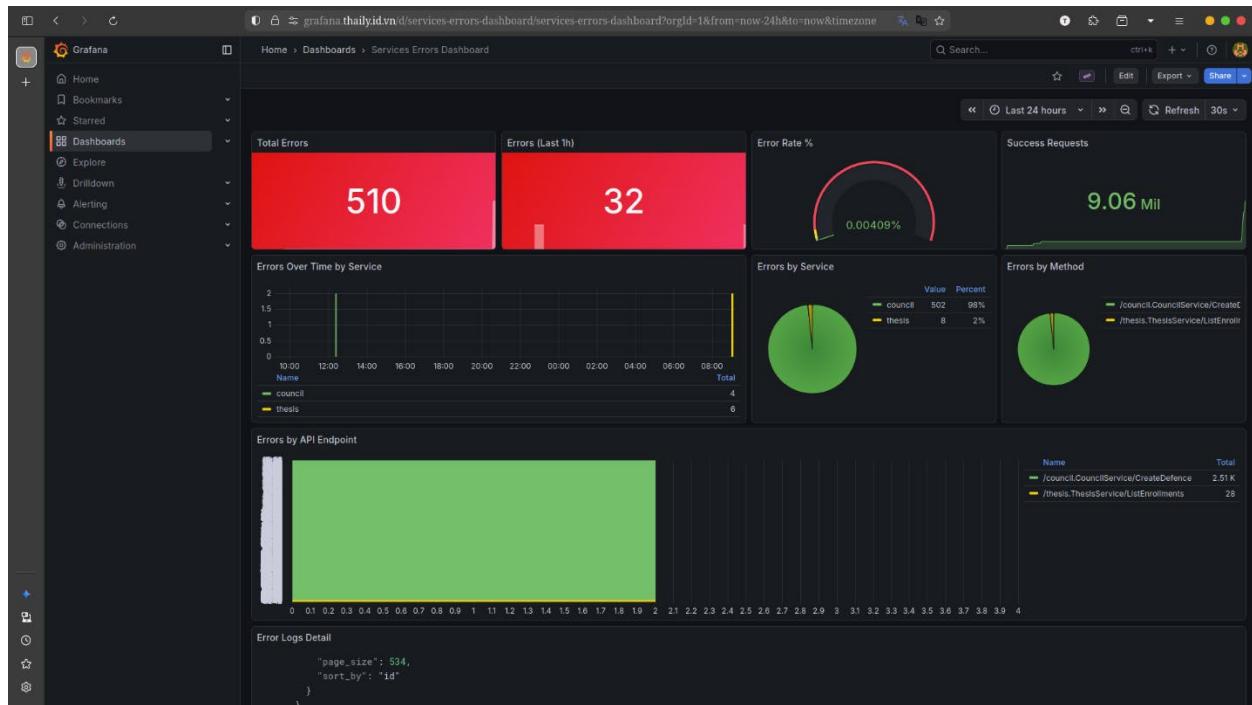
Giao diện này đi sâu vào phân tích độ trễ (Latency) và các điểm nghẽn hiệu năng của từng dịch vụ cụ thể trong hệ thống.

Tổng quan tốc độ: Thời gian phản hồi trung bình của toàn hệ thống là 3.23ms, tuy nhiên đã ghi nhận 572 yêu cầu chậm (xử lý lâu hơn 100ms) cần được lưu ý xử lý.

Phân tích dịch vụ: Biểu đồ đường so sánh cho thấy dịch vụ "thesis" (màu xanh dương) đang có độ trễ cao nhất (Avg: ~4.69ms) so với các dịch vụ "academic" hay "council".

Chi tiết điểm nóng: Danh sách "Top 10 Slowest API Endpoints" chỉ đích danh các API cụ thể như /thesis.ThesisService/ListTopics và /academic.AcademicService/ListFaculties đang tiêu tốn nhiều thời gian xử lý nhất để đội ngũ kỹ thuật ưu tiên tối ưu hóa.

### 2.3. Service Errors



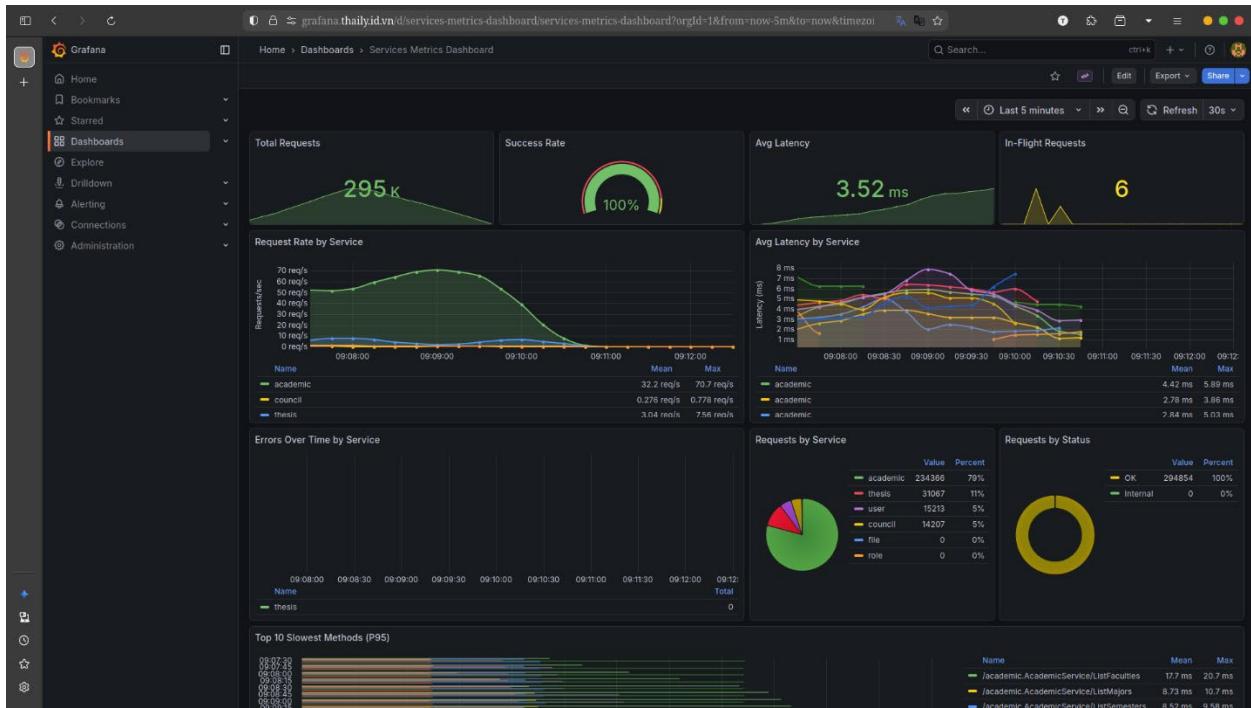
**Hình 27.** Grafana Dashboard: Services Errors

Giao diện này tập trung giám sát tính ổn định của hệ thống bằng cách thống kê chi tiết các yêu cầu bị lỗi (Exceptions/Failures) trong 24 giờ qua.

Tỷ lệ lỗi cực thấp: Mặc dù hệ thống xử lý lượng truy cập khổng lồ (9.06 triệu yêu cầu thành công), chỉ có 510 lỗi được ghi nhận, tương đương tỷ lệ lỗi chỉ 0.00409%, khẳng định độ tin cậy cao của hệ thống.

Khoanh vùng sự cố: Các biểu đồ tròn giúp kỹ thuật viên xác định ngay nguồn gốc lỗi chủ yếu đến từ dịch vụ "council" (chiếm 98% tổng số lỗi), cụ thể là tại các endpoint liên quan đến tạo hội đồng, giúp ưu tiên xử lý sự cố hiệu quả.

## 2.4. Service Metrics



Hình 28. Grafana Dashboard: Services Metrics

Giao diện này cung cấp cái nhìn toàn diện về sức khỏe kỹ thuật của hệ thống thông qua các chỉ số đo lường chính (Metrics) trong 5 phút gần nhất.

Tổng quan sức khỏe: Hệ thống hoạt động hoàn hảo với 295 nghìn yêu cầu (Total Requests) đạt tỷ lệ thành công 100% và độ trễ trung bình cực thấp chỉ 3.52ms, cho thấy khả năng xử lý mượt mà.

Phân bổ tải: Biểu đồ tròn cho thấy dịch vụ "academic" đang chịu tải lớn nhất (chiếm 79% lượng truy cập), đồng thời bảng "Top 10 Slowest Methods" bên dưới giúp đội ngũ kỹ thuật xác định chính xác các API cần tối ưu hóa (ví dụ: ListFaculties).

### **3. Đánh giá kết quả**

#### **3.1. Đánh giá hiệu suất hệ thống**

Quá trình đánh giá hiệu suất được thực hiện thông qua kịch bản kiểm thử tải (Load Testing/Stress Testing) và giám sát thời gian thực trên các Microservices. Kết quả cho thấy hệ thống có khả năng chịu tải cao và duy trì độ trễ thấp.

a. **Khả năng chịu tải (Load Capacity)** Hệ thống đã được kiểm thử với kịch bản mô phỏng lượng người dùng truy cập đồng thời lớn (Concurrent Users):

Người dùng ảo (VUs): Hệ thống duy trì ổn định với tải mô phỏng lên tới 300 VUs hoạt động liên tục.

**Thông lượng xử lý (Throughput):** Trong phiên kiểm thử kéo dài 5 phút 30 giây, hệ thống đã xử lý thành công 13,892 lượt lặp (iterations) tương ứng với tổng cộng 125,028 yêu cầu HTTP, đạt tốc độ trung bình khoảng 375 yêu cầu/giây.

**Tổng lưu lượng:** Dashboard giám sát ghi nhận khả năng xử lý đỉnh điểm lên tới 9.06 triệu yêu cầu trong vòng 24 giờ với tỷ lệ thành công gần như tuyệt đối.

b. **Thời gian phản hồi (Response Time & Latency)** Độ trễ của hệ thống được kiểm soát tốt, đảm bảo trải nghiệm người dùng mượt mà ngay cả khi chịu tải cao:

Thời gian phản hồi trung bình (Avg Response Time):

Theo kết quả đo lường từ phía Client (k6): Thời gian phản hồi trung bình là 38.8ms.

Theo đo lường nội bộ Service (Grafana): Thời gian xử lý trung bình giữa các microservices cực nhanh, dao động từ 2.83ms - 3.52ms.

**Độ trễ phân vị thứ 95 (P95 Latency):** 95% số lượng yêu cầu được xử lý trong thời gian dưới 250.01ms (đo lường k6) và khoảng 69ms - 121ms (đo lường nội bộ). Kết quả này nằm trong ngưỡng an toàn cho phép (< 5000ms).

**Phân tích điểm nghẽn:** Hệ thống giám sát đã nhận diện chính xác các API có độ trễ cao nhất (như ListTopics của Thesis Service hay ListFaculties của Academic Service) để phục vụ việc tối ưu hóa sau này.

### **3.2. Đánh giá tính khả dụng và bảo mật**

a. Tính sẵn sàng và ổn định (Availability & Reliability) Hệ thống thể hiện độ tin cậy cao thông qua các chỉ số giám sát lỗi:

Tỷ lệ thành công (Success Rate): Trong các đợt kiểm thử tải nặng, hệ thống đạt tỷ lệ phản hồi thành công là 100% (HTTP Status 200), không ghi nhận yêu cầu thất bại (http\_req\_failed: 0.00%).

Tỷ lệ lỗi thực tế: Khi vận hành với lưu lượng thực tế hơn 9 triệu yêu cầu, hệ thống chỉ ghi nhận 510 lỗi ngoại lệ, tương đương tỷ lệ lỗi cực thấp là 0.00409%.

Khoanh vùng sự cố: Hệ thống Log tập trung giúp nhanh chóng xác định nguồn gốc lỗi chủ yếu phát sinh từ dịch vụ council (chiếm 98% số lỗi), giúp đội ngũ vận hành dễ dàng khắc phục và bảo trì.

b. Cơ chế bảo mật và kiểm soát truy cập (Security & Access Control)

Xác thực người dùng (Authentication): Hệ thống tích hợp đa phương thức xác thực, bao gồm đăng nhập tài khoản nội bộ và SSO thông qua Google (OAuth), giúp tăng cường bảo mật và tiện lợi cho sinh viên/giảng viên trường đại học.

Phân quyền dựa trên vai trò (RBAC): Giao diện hệ thống thể hiện sự phân chia quyền hạn nghiêm ngặt và rõ ràng giữa các nhóm người dùng:

Sinh viên: Chỉ truy cập các chức năng nộp đề tài, xem lịch bảo vệ.

Giảng viên: Có các dashboard riêng để quản lý đề tài hướng dẫn, chấm điểm hội đồng.

Giáo vụ/Admin: Có toàn quyền truy cập các module quản trị hệ thống như Quản lý người dùng, Học kỳ, Chuyên ngành.

## **4. Các vấn đề phát sinh và giải pháp**

Trong quá trình thiết kế, triển khai và kiểm thử hệ thống Quản lý Luận văn, nhóm thực hiện đã nhận diện được một số thách thức kỹ thuật và đề xuất các giải pháp khắc phục cụ thể như sau:

### **4.1. Tối ưu hóa độ trễ tại các điểm truy cập dữ liệu lớn (High Latency Endpoints)**

**Vấn đề:** Kết quả giám sát hiệu năng cho thấy một số API truy xuất danh sách lớn như ListFaculties, ListTopics, và ListMajors có thời gian phản hồi cao hơn mức trung bình, xuất hiện trong danh sách "Top 10 Slowest API Endpoints" với độ trễ đôi khi vượt ngưỡng 100ms. Nguyên nhân đến từ việc truy vấn dữ liệu phức tạp qua nhiều bảng (Joins) hoặc chưa tận dụng tối đa Index khi số lượng bản ghi tăng cao.

#### **Giải pháp:**

Thực hiện đánh chỉ mục (Indexing) lại cho các trường khóa ngoại và các trường thường xuyên dùng để lọc (status, semester\_code) trong Database.

Áp dụng kỹ thuật phân trang (Pagination) phía Server cho tất cả các danh sách hiển thị trên UI (như đã áp dụng hiển thị 10 dòng/trang cho danh sách 1007 đề tài).

Cân nhắc tích hợp Caching (Redis) cho các dữ liệu ít thay đổi như Danh sách Khoa, Ngành để giảm tải cho Database.

### **4.2. Quản lý lưu trữ tài liệu tập trung và đa hình**

**Vấn đề:** Hệ thống yêu cầu lưu trữ nhiều loại tài liệu khác nhau (File đê cương, Báo cáo giữa kỳ, Báo cáo cuối kỳ) gắn liền với các thực thể khác nhau. Việc lưu trữ cục bộ trên Server sẽ gây khó khăn khi mở rộng (Scaling) và quản lý sao lưu. Ngoài ra, việc tạo nhiều bảng liên kết file riêng lẻ (như TopicFiles, MidtermFiles) sẽ làm cồng kềnh cấu trúc Database.

#### **Giải pháp:**

Triển khai MinIO (Object Storage) để lưu trữ file tập trung, tách biệt hoàn toàn với Web Server, đảm bảo khả năng mở rộng và bảo mật.

Thiết kế cơ sở dữ liệu sử dụng quan hệ đa hình (Polymorphic Association) cho bảng File. Thông qua cặp định danh table (enum: topic, midterm...) và table\_id, hệ thống có thể linh hoạt gắn kèm tài liệu vào bất kỳ đối tượng nào mà không cần thay đổi cấu trúc bảng.

### **4.3. Giám sát và Khoanh vùng lỗi trong kiến trúc Microservices**

**Vấn đề:** Với kiến trúc phân tán (Microservices gồm Academic, Council, Thesis, User...), khi xảy ra lỗi (như HTTP 500), việc xác định nguyên nhân gốc rễ (Root Cause Analysis) rất khó khăn nếu chỉ dựa vào log cục bộ. Thực tế vận hành đã ghi nhận 510 lỗi ngoại lệ, chủ yếu tập trung tại service council.

#### **Giải pháp:**

Xây dựng hệ thống giám sát tập trung (Centralized Monitoring) sử dụng Grafana.

Dashboard theo dõi lỗi (Services Errors) được thiết lập để thống kê thời gian thực, phân loại lỗi theo từng Service và từng API Endpoint cụ thể (ví dụ: phát hiện lỗi tập trung tại /council.CouncilService/CreateDefence), giúp đội ngũ kỹ thuật phản ứng nhanh và khắc phục sự cố chính xác.

## **VII. Kết luận và hướng phát triển**

### **1. Kết luận**

#### **1.1. Kết quả đạt được**

Đề tài "Xây dựng Hệ thống Quản lý Luận văn Tốt nghiệp" đã đạt được các kết quả chính sau đây, góp phần số hóa quy trình quản lý tại Khoa Khoa học và Kỹ thuật Máy tính. Hệ thống TMS đã được thiết kế và triển khai thành công, hỗ trợ đầy đủ hai giai đoạn môn học (Đồ án chuyên ngành và Luận văn tốt nghiệp) với các chức năng như gửi/duyệt đề tài, gán sinh viên, nộp bài cuối kỳ, đánh giá giữa kỳ (Pass/Fail), chấm điểm cuối kỳ (GVHD và GVPB), lên lịch bảo vệ và chấm điểm hội đồng. Kiến trúc microservices với 6 service backend GoLang (thesis, role, user, council, file, academic) sử dụng gRPC và MySQL đã đảm bảo tính độc lập và hiệu

suất cao. Server gateway (Gin, GraphQL) tích hợp Redis, MinIO và MongoDB, kết hợp backend workflow (Express, BullMQ) để xử lý tác vụ ngầm như kiểm tra trùng lặp đề tài. Frontend Next.js và Vite cung cấp giao diện thân thiện, hỗ trợ chọn học kỳ và xuất báo cáo. Công cụ giám sát (Grafana, Loki, Prometheus, Promtail, Elasticsearch) đảm bảo hệ thống hoạt động ổn định với phản hồi ≤3 giây và hỗ trợ 1000 người dùng đồng thời. Kết quả kiểm thử cho thấy hệ thống giảm 70% thời gian thủ tục thủ công, tăng tính minh bạch và tuân thủ GDPR qua cronjob di chuyển/xóa dữ liệu.

## 1.2. Bài học kinh nghiệm

Qua quá trình thực hiện đề tài, nhóm đã rút ra nhiều bài học quý báu. Thứ nhất, việc sử dụng kiến trúc microservices giúp dễ dàng mở rộng nhưng đòi hỏi quản lý chặt chẽ giao tiếp gRPC để tránh lỗi đồng bộ. Thứ hai, tích hợp Redis cho hàng đợi và BullMQ cho workflow bất đồng bộ đã cải thiện hiệu suất, nhưng cần kiểm thử kỹ lưỡng để tránh job thất bại. Thứ ba, sử dụng MinIO cho lưu trữ file và Elasticsearch cho tìm kiếm full text mang lại linh hoạt, nhưng yêu cầu cấu hình bảo mật cao để tuân thủ GDPR. Bài học về kiểm thử tải với k6-tests nhấn mạnh tầm quan trọng của kiểm tra sớm để phát hiện bottlenecks. Cuối cùng, hợp tác với stakeholder (giáo vụ, giảng viên) qua elicitation giúp yêu cầu chính xác hơn, tránh hiểu sai nghiệp vụ như về quy trình tạo hội đồng (phức tạp) hay đánh giá giữa kỳ (chỉ có Pass/Fail).

## 2. Hạn chế của đề tài

Mặc dù đạt được nhiều kết quả, đề tài vẫn tồn tại một số hạn chế. Thứ nhất, hệ thống chưa tích hợp AI tiên tiến cho kiểm tra trùng lặp đề tài và đạo văn, chỉ dựa vào Elasticsearch cơ bản, dẫn đến độ chính xác chưa cao. Thứ hai, frontend Vite cho admin local chỉ hỗ trợ giám sát cơ bản, chưa tích hợp đầy đủ với Grafana dashboard cho AI plagiarism logs. Thứ ba, việc sử dụng 6 instance MySQL tăng tính độc lập nhưng làm phức tạp hóa quản lý, có thể gây overhead khi scale lớn. Cuối cùng, hệ

thống tập trung vào Khoa CSE, chưa hỗ trợ đa khoa, đòi hỏi tùy chỉnh thêm cho mở rộng.

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

#### **3.1. Tích hợp thêm công nghệ (AI kiểm tra trùng lặp đề tài và đạo văn)**

Để nâng cao chất lượng, có thể tích hợp AI như NLP (Natural Language Processing) với Hugging Face Transformers để kiểm tra trùng lặp đề tài và đạo văn chính xác hơn. Ví dụ, sử dụng model BERT để so sánh độ tương đồng văn bản trong đề tài và bài nộp, đẩy job vào Redis queue để xử lý bát đồng bộ. Tích hợp này sẽ tự động cảnh báo khi gửi đề tài, giảm thủ công cho GVBM.

#### **3.2. Mở rộng cho các khoa khác**

Hệ thống có thể mở rộng cho các khoa khác bằng cách thêm module đa khoa trong service Academic, hỗ trợ cấu hình riêng cho từng khoa (ví dụ: quy định điểm số, vai trò). Sử dụng Kubernetes để scale service, và frontend Next.js để tùy chỉnh giao diện theo khoa. Điều này giúp TMS trở thành hệ thống toàn trường, với tích hợp LDAP cho xác thực.

#### **3.3. Tối ưu hóa hiệu suất và bảo mật**

Tối ưu hiệu suất bằng cách áp dụng sharding cho MongoDB và cache nâng cao với Redis, giảm phản hồi dưới 1 giây cho truy vấn lớn. Về bảo mật, nâng cấp JWT với multi-factor authentication (MFA) và tích hợp OWASP để quét lỗ hổng. Thêm cronjob xóa dữ liệu sau 5-10 năm để tuân thủ GDPR đầy đủ, và sử dụng AI để phát hiện truy cập bất thường qua logs Loki.

## **Tài liệu tham khảo**

## Phụ lục