Đại học Thái Nguyên

Trường Đại học Công nghệ Thông tin và Truyền thông

Khoa Công nghệ thông tin

GIÁO TRÌNH

CÔNG NGHỆ PHẦN MỀM CHO CÁC HỆ THỐNG THÔNG MINH

Tác giả: **Nguyễn Thế Vịnh**
Nguyễn Thị Dung
Đoàn Ngọc Phương
Nguyễn Thu Phương

Thái Nguyên – 2026

Đại học Thái Nguyên

Trường Đại học Công nghệ Thông tin và Truyền thông

GIÁO TRÌNH

CÔNG NGHỆ PHẦN MỀM CHO CÁC HỆ THỐNG THÔNG MINH

Đơn vị phụ trách: Khoa Công nghệ thông tin

Tác giả: Nguyễn Thế Vịnh

Nguyễn Thị Dung

Đoàn Ngọc Phương

Nguyễn Thu Phương

Thái Nguyên, năm 2026

Thông tin biên soạn

Họ và tên	Vai trò
Nguyễn Thế Vịnh	Tác giả
Nguyễn Thị Dung	Tác giả
Đoàn Ngọc Phương	Tác giả
Nguyễn Thu Phương	Tác giả

Lời nói đầu

Giáo trình *Công nghệ phần mềm cho các hệ thống thông minh* được biên soạn theo Hợp đồng số 04.2026/BSGT-ĐHCNTT&TT ngày 06/02/2026, phục vụ học phần thuộc định hướng công nghệ phần mềm trong bối cảnh trí tuệ nhân tạo đang được tích hợp ngày càng sâu vào sản phẩm số. Mục tiêu của giáo trình không phải biến sinh viên thành nhà nghiên cứu mô hình AI, mà giúp sinh viên biết xây dựng, tích hợp, kiểm thử, triển khai và vận hành các thành phần AI như một phần của hệ thống phần mềm có trách nhiệm.

Nội dung giáo trình được tổ chức thành ba mảng chính. Mảng thứ nhất trình bày nền tảng AI Engineering: nhận diện hệ thống thông minh, mô tả dữ liệu, xác định tác vụ AI, đọc model card và lựa chọn mô hình đã huấn luyện. Mảng thứ hai tập trung vào tích hợp AI vào sản phẩm phần mềm: thiết kế yêu cầu, kiến trúc, API suy luận, ứng dụng web, mobile/edge, AI tạo sinh, RAG và agent workflow. Mảng thứ ba đi vào MLOps, kiểm thử, đánh giá chất lượng, triển khai, bảo mật, quản trị rủi ro và hồ sơ bàn giao.

Giáo trình được thiết kế theo hướng thực hành. Sinh viên không chỉ đọc khái niệm mà phải cài công cụ, lấy dữ liệu hoặc mã nguồn từ các nguồn thật như Kaggle, GitHub và Hugging Face, chạy lệnh, sinh file kết quả, phân tích lỗi và viết báo cáo thực hành. Các case study trong giáo trình sử dụng những nguồn công khai như Kaggle Titanic, Hugging Face model card, FastAPI template, TensorFlow Lite examples, Hugging Face cookbook, MLflow examples, OpenAI Evals và BentoML để sinh viên thấy rõ cách một ý tưởng AI đi từ dữ liệu, mô hình, API, đánh giá đến triển khai.

Trong học phần này, các bài kiểm tra thường xuyên và thi cuối kỳ đều hướng tới sản phẩm báo cáo thực hành. Vì vậy, mỗi chương đều nhấn mạnh artefact cụ thể như product brief, data note, model note, API contract, latency report, RAG evaluation packet, experiment log, evaluation report và deployment runbook. Một báo cáo đạt yêu cầu phải cho thấy nhóm đã chạy được lệnh, có output thật, có smoke test sau deploy, biết giải thích giới hạn của hệ thống và đưa ra quyết định kỹ thuật dựa trên bằng chứng.

Nhóm tác giả mong rằng giáo trình sẽ giúp sinh viên tiếp cận AI bằng tư duy kỹ nghệ phần mềm: cụ thể, có kiểm chứng, có trách nhiệm và có khả năng bàn giao. Trong

quá trình sử dụng, giáo trình chắc chắn còn cần tiếp tục cập nhật theo sự thay đổi nhanh của công cụ, mô hình và quy trình phát triển AI. Nhóm tác giả trân trọng tiếp nhận các góp ý của giảng viên, sinh viên và đồng nghiệp để hoàn thiện các phiên bản tiếp theo.

MỤC LỤC

Thông tin biên soạn **i**

Lời nói đầu **ii**

Danh mục hình ảnh **vii**

Danh mục bảng biểu **ix**

Danh mục từ viết tắt **xv**

Hướng dẫn cài đặt và chạy lab **1**

Bộ công cụ bắt buộc 1
Cấu trúc thư mục nộp bài 2
Lệnh chạy cụ thể theo chương 4
Một lab mẫu chạy từ đầu đến cuối 6
Cách ghi bằng chứng chạy 7

I Nền tảng kỹ nghệ hệ thống thông minh **8**

1 Tổng quan về hệ thống thông minh và AI Engineering **9**

1.1 Hệ thống thông minh và sản phẩm phần mềm có AI 11
1.2 Vòng đời, yêu cầu và kiến trúc AI Engineering 17
1.3 Case study và thực hành bài tập lớn 27
1.4 Case study từ internet: Kaggle Titanic như một sản phẩm AI nhỏ 41
Câu hỏi ôn tập 43

2 Dữ liệu, tác vụ AI, mô hình đã huấn luyện và tệp mô hình **44**

2.1 Dữ liệu và tác vụ AI trong sản phẩm phần mềm 46
2.2 Mô hình đã huấn luyện và artefact mô hình 54
2.3 Đọc model card, chạy thử inference và lập model note 60

2.4 Case study: model card Hugging Face 71
Câu hỏi ôn tập 73

3 Yêu cầu, kiến trúc và thiết kế sản phẩm AI 74
3.1 Đặc tả bài toán và yêu cầu hệ thống AI 77
3.2 Thiết kế kiến trúc và hợp đồng kỹ thuật 82
3.3 Metric, rủi ro và rà soát thiết kế 89
3.4 Case study: kiến trúc FastAPI 101
Câu hỏi ôn tập 103

II Tích hợp AI vào sản phẩm phần mềm 104

4 Xây dựng dịch vụ suy luận AI cho ứng dụng web 105
4.1 Thiết kế dịch vụ suy luận AI 108
4.2 Xây dựng API suy luận cho ứng dụng web 112
4.3 Tích hợp frontend, kiểm thử và quản lý phiên bản 116
4.4 Case study: API suy luận từ Transformers 132
Câu hỏi ôn tập 134

5 Tích hợp AI vào ứng dụng mobile và edge 135
5.1 Lựa chọn phương án tích hợp AI 138
5.2 Ràng buộc thiết bị và tối ưu mô hình 141
5.3 Đánh giá, kiểm thử và phối hợp người-AI 144
5.4 Case study: TensorFlow Lite Android 164
Câu hỏi ôn tập 165

6 AI tạo sinh, RAG và tác nhân phần mềm 166
6.1 Thiết kế ứng dụng AI tạo sinh 169
6.2 RAG và kiểm soát nguồn dữ liệu 172
6.3 Tool calling, agent workflow và bảo vệ hệ thống 177
6.4 Case study từ internet: Hugging Face cookbook về RAG evaluation . . . 195
Câu hỏi ôn tập 196

III MLOps, chất lượng và vận hành 198

7 Công cụ phát triển, quản lý phiên bản và thực nghiệm 199

7.1 Môi trường phát triển và quản lý mã nguồn . . . 202
7.2 Dữ liệu, mô hình và thực nghiệm . . . 204
7.3 Cấu trúc repo, tài liệu và phối hợp người-AI . . . 208
7.4 Case study từ internet: MLflow examples để quản lý thực nghiệm . . . 226
Câu hỏi ôn tập . . . 227

8 Kiểm thử, đánh giá và đảm bảo chất lượng hệ thống AI 229
8.1 Kiểm thử các lớp của hệ thống AI . . . 232
8.2 Metric, đánh giá mô hình và hành vi LLM/RAG . . . 236
8.3 Monitoring, báo cáo đánh giá và phối hợp người-AI . . . 240
8.4 Case study từ internet: OpenAI Evals và bộ đánh giá có thể chạy lại . . . 257
Câu hỏi ôn tập . . . 259

9 Triển khai, vận hành, bảo mật và quản trị hệ thống AI 260
9.1 Đóng gói, CI/CD và triển khai . . . 263
9.2 Vận hành, rollback và bảo mật . . . 268
9.3 Quản trị rủi ro và hồ sơ bàn giao . . . 271
9.4 Case study từ internet: Docker, GitHub Actions và BentoML quickstart . 289
Câu hỏi ôn tập . . . 291

Tài liệu tham khảo 292

DANH MỤC HÌNH ẢNH

Hình 1.1. Hệ thống thông minh là một sản phẩm phần mềm có thành phần AI, không phải chỉ là mô hình AI 12
Hình 1.2. Kiến trúc khái quát của hệ thống phần mềm có thành phần AI 15
Hình 1.3. Vòng đời kỹ nghệ của hệ thống AI 18
Hình 1.4. Ba trụ cột AI Engineering ở mức nhập môn 18
Hình 1.5. Vòng lặp triển khai, giám sát và cải tiến trong hệ thống AI . . 20
Hình 1.6. Dữ liệu cần được xem như một phần của yêu cầu hệ thống AI 24
Hình 1.7. Luồng xử lý khái quát của trợ lý RAG hỏi đáp tài liệu môn học 27
Hình 1.8. Bộ hồ sơ tối thiểu đi cùng sản phẩm AI trong bài tập lớn . . . 31
Hình 1.9. Vòng lặp cộng tác người-AI khi lập hồ sơ sản phẩm AI 31
Hình 1.10. Luồng thực hành tổng hợp lập AI product brief 33

Hình 2.1. Vòng đời dữ liệu trong sản phẩm phần mềm có thành phần AI 47
Hình 2.2. Các nhóm tác vụ AI phổ biến trong sản phẩm phần mềm . . . 52
Hình 2.3. Luồng xử lý nhập môn cho bài toán phân loại phản hồi sinh viên 54
Hình 2.4. Các artefact thường gặp trong thư mục mô hình đã huấn luyện 57
Hình 2.5. Các mục cần đọc khi đánh giá model card 61
Hình 2.6. Luồng smoke test cho một mô hình đã huấn luyện 63
Hình 2.7. Các thành phần nên có trong model note 66
Hình 2.8. Quy trình dùng AI hỗ trợ đọc tài liệu mô hình nhưng vẫn kiểm chứng thủ công 68

Hình 3.1. Từ ý tưởng sản phẩm AI đến yêu cầu, kiến trúc, API, metric và rủi ro 77
Hình 3.2. Kiến trúc tham khảo cho trợ lý cố vấn học tập có human review và log 85

Hình 4.1. Vòng đời request trong dịch vụ suy luận AI cho ứng dụng web 108
Hình 4.2. Các trạng thái giao diện cần có khi frontend gọi endpoint AI . 118

Hình 4.3. Các lớp kiểm thử cần có cho endpoint AI 120

Hình 5.1. Ba phương án tích hợp AI cho mobile và edge 138
Hình 5.2. Vòng lặp tối ưu, triển khai, đo và cập nhật mô hình trên thiết bị 148

Hình 6.1. Pipeline RAG từ tài liệu đến câu trả lời có nguồn và đánh giá 169
Hình 6.2. Workflow agent có validation, xác nhận và audit log trước khi gọi tool 178

Hình 7.1. Cấu trúc repo và luồng tái lập trong dự án AI 201
Hình 7.2. Từ experiment run đến model registry, lựa chọn model và rollback 207

Hình 8.1. Các lớp đảm bảo chất lượng cho hệ thống AI 232
Hình 8.2. Luồng lập evaluation report từ test set đến quyết định rủi ro . 241

Hình 9.1. Vòng đời triển khai và vận hành hệ thống AI 263
Hình 9.2. Các cổng kiểm soát bảo mật và quản trị trong hệ thống AI . . 271

DANH MỤC BẢNG BIỂU

Bảng 3. Tool và file output bắt buộc theo chương 4

Bảng 1.1. Phân biệt AI, ML, DL, GenAI và hệ thống dựa trên luật . . . 14
Bảng 1.2. Metric mô hình, metric sản phẩm và metric vận hành 17
Bảng 1.3. Phần mềm truyền thống và phần mềm có thành phần AI . . . 19
Bảng 1.4. Ví dụ chuyển yêu cầu mơ hồ thành yêu cầu AI có thể kiểm thử 21
Bảng 1.5. Ba nhóm tiêu chí chấp nhận cho user story AI 23
Bảng 1.6. Các điểm có thể đưa human-in-the-loop vào hệ thống AI . . . 25
Bảng 1.7. Một số lựa chọn kiến trúc ban đầu cho tính năng AI 26
Bảng 1.8. AI product brief rút gọn cho trợ lý RAG hỏi đáp tài liệu môn học 28
Bảng 1.9. Mẫu AI product brief một trang cho bài tập lớn 30
Bảng 1.10. Kịch bản thực hành tổng hợp lập AI product brief 34
Bảng 1.11. Rubric đánh giá AI product brief 35
Bảng 1.12. Lỗi thường gặp khi lập kế hoạch sản phẩm AI 36
Bảng 1.13. Gợi ý đề tài bài tập lớn 38
Bảng 1.14. Ba mức tham vọng cho đề tài trợ lý RAG hỏi đáp tài liệu . . . 39
Bảng 1.15. Mẫu bảng metric và rủi ro ban đầu 40
Bảng 1.16. Cách biến Kaggle Titanic thành case study kỹ nghệ AI 42

Bảng 2.1. Các loại dữ liệu cần nhận diện trong sản phẩm AI 48
Bảng 2.2. Mẫu data note ngắn cho bài tập lớn 49
Bảng 2.3. Ví dụ data contract cho phản hồi sinh viên 51
Bảng 2.4. Tác vụ AI, đầu ra và metric nhập môn 53
Bảng 2.5. Ba cách dùng mô hình có sẵn ở mức nhập môn 55
Bảng 2.6. Mẫu bảng so sánh mô hình ứng viên 56
Bảng 2.7. Artefact mô hình và câu hỏi kiểm tra 58
Bảng 2.8. Định dạng và môi trường chạy mô hình 59
Bảng 2.9. Lỗi thường gặp với artefact mô hình 60
Bảng 2.10. Mẫu đọc model card ở mức nhập môn 62

Bảng 2.11. Bộ ví dụ tối thiểu cho smoke test 65
Bảng 2.12. Mẫu bảng ghi kiểm chứng thông tin do AI hỗ trợ trích xuất . . 69
Bảng 2.13. Lỗi cần tránh khi chọn dữ liệu và mô hình 70

Bảng 3.1. Từ ý tưởng mơ hồ đến bài toán sản phẩm rõ hơn 78
Bảng 3.2. Mẫu yêu cầu cho tính năng AI 80
Bảng 3.3. Ví dụ user story và acceptance criteria cho sản phẩm AI . . . 82
Bảng 3.4. So sánh các kiểu kiến trúc AI thường gặp 84
Bảng 3.5. Mẫu API contract cho endpoint phân loại phản hồi 86
Bảng 3.6. Thiết kế giao diện cho kết quả AI 89
Bảng 3.7. Ba lớp metric cần theo dõi 90
Bảng 3.8. Mẫu risk register cho sản phẩm AI 91
Bảng 3.9. Hồ sơ thiết kế tối thiểu cho case study trợ lý cố vấn học tập . . 95
Bảng 3.10. Mẫu design decision record cho bài tập lớn 96
Bảng 3.11. Rubric rà soát thiết kế sản phẩm AI nhập môn 96
Bảng 3.12. Các lỗi thiết kế hay gặp trước khi cài đặt 98
Bảng 3.13. Worksheet thiết kế sản phẩm AI trước khi code 98
Bảng 3.14. Mẫu phiếu review chéo thiết kế 100
Bảng 3.15. Gợi ý response schema cho mini case dashboard hài lòng . . . 101
Bảng 3.16. Bài thực hành kiến trúc từ FastAPI template 102

Bảng 4.1. Trách nhiệm của các thành phần trong ứng dụng web có AI . 109
Bảng 4.2. Chọn kiểu xử lý cho endpoint AI 110
Bảng 4.3. Các lỗi thường gặp của endpoint AI 114
Bảng 4.4. Trạng thái UI cần có khi tích hợp endpoint AI 117
Bảng 4.5. Backlog gợi ý cho endpoint phân loại phản hồi 121
Bảng 4.6. Phân loại field trong response của endpoint AI 122
Bảng 4.7. Test matrix nhập môn cho endpoint AI 123
Bảng 4.8. Mapping lỗi endpoint AI sang hành vi giao diện 124
Bảng 4.9. Nội dung cần có trong tài liệu endpoint AI 125
Bảng 4.10. Phiếu tự chấm endpoint AI trước demo 126
Bảng 4.11. Các bước chuyển notebook thành service 126
Bảng 4.12. Runbook debug endpoint AI 127
Bảng 4.13. Đối chiếu mock model với rủi ro cần kiểm thử 129
Bảng 4.14. Mẫu báo cáo kiểm thử endpoint AI 129
Bảng 4.15. Kịch bản demo endpoint AI trong lớp 131

Bảng 5.1. So sánh ba phương án tích hợp AI cho mobile và edge 139
Bảng 5.2. Ma trận quyết định phương án tích hợp 140
Bảng 5.3. Ràng buộc thiết bị cần đo khi chạy AI on-device 142
Bảng 5.4. Các bước tối ưu mô hình cho mobile/edge 143
Bảng 5.5. Mẫu bảng đo phương án mobile/edge 145
Bảng 5.6. So sánh casebook nhận diện thiết bị phòng thực hành 150
Bảng 5.7. Worksheet đo hiệu năng mobile/edge 150
Bảng 5.8. Các bước xử lý ảnh trước khi inference 151
Bảng 5.9. Rà soát riêng tư cho AI mobile/edge 152
Bảng 5.10. Các rủi ro khi cập nhật model on-device 153
Bảng 5.11. Bộ test ảnh cho ứng dụng nhận diện thiết bị 154
Bảng 5.12. Mẫu báo cáo lựa chọn phương án mobile/edge 154
Bảng 5.13. Lỗi thường gặp và cách sửa trong mobile/edge AI 156
Bảng 5.14. Kế hoạch đo end-to-end latency 156
Bảng 5.15. Bảng so sánh model trước và sau tối ưu 157
Bảng 5.16. Fallback theo trạng thái mobile/edge 158
Bảng 5.17. Phiếu privacy review cho dữ liệu camera 159
Bảng 5.18. Mẫu release note cho model mobile/edge 159
Bảng 5.19. Câu hỏi tự phản biện cho phương án mobile/edge 160
Bảng 5.20. Rubric chấm phương án tích hợp AI mobile/edge 161
Bảng 5.21. Mẫu nhật ký đo thiết bị cho mobile/edge AI 162
Bảng 5.22. Câu hỏi phản biện nhanh cho phần trình bày mobile/edge . . . 163
Bảng 5.23. Thực nghiệm cần làm với case TensorFlow Lite Android . . . 164

Bảng 6.1. Các thành phần của một request LLM có kiểm soát 170
Bảng 6.2. Các artefact cần quản lý trong pipeline RAG 173
Bảng 6.3. Quy trình kiểm soát câu trả lời RAG 175
Bảng 6.4. Thiết kế tool an toàn cho LLM 177
Bảng 6.5. Bộ test an toàn tối thiểu cho LLM/RAG/agent 180
Bảng 6.6. Backlog cho trợ lý RAG tài liệu môn học 181
Bảng 6.7. Mẫu data card cho nguồn RAG 182
Bảng 6.8. Cấu trúc bộ câu hỏi đánh giá RAG 182
Bảng 6.9. Rubric 0–2 điểm cho câu trả lời RAG 183
Bảng 6.10. Phân loại lỗi RAG và hướng sửa 183
Bảng 6.11. Ví dụ tool schema nhập môn cho trợ lý học tập 184
Bảng 6.12. Red-team test cho agent có tool 185

Bảng 6.13. Mẫu báo cáo đánh giá LLM/RAG/agent . . . 186
Bảng 6.14. Lỗi thường gặp và cách sửa trong LLM/RAG/agent . . . 187
Bảng 6.15. Bảng thử nghiệm chunking cho RAG . . . 187
Bảng 6.16. Mẫu phiếu citation audit . . . 188
Bảng 6.17. Mẫu prompt changelog . . . 188
Bảng 6.18. Permission matrix nhập môn cho agent . . . 189
Bảng 6.19. Mẫu incident note cho LLM/RAG . . . 189
Bảng 6.20. Thành phần UI cho câu trả lời RAG . . . 190
Bảng 6.21. Rubric chấm đồ án LLM/RAG/agent . . . 191
Bảng 6.22. Phân tích mini case citation đúng nhưng trả lời sai . . . 192
Bảng 6.23. Thành phần của eval packet cho RAG/agent . . . 193
Bảng 6.24. Cách xử lý câu hỏi mơ hồ trong RAG . . . 194
Bảng 6.25. Cách chuyển notebook RAG evaluation thành bài thực hành . 195

Bảng 7.1. Các thành phần môi trường cần ghi lại . . . 203
Bảng 7.2. Metadata tối thiểu cho dữ liệu và mô hình . . . 205
Bảng 7.3. Mẫu experiment log ngắn . . . 206
Bảng 7.4. rà soát tái lập dự án AI . . . 211
Bảng 7.5. Clean clone protocol cho bài tập lớn . . . 211
Bảng 7.6. Quy ước đặt tên artefact gợi ý . . . 212
Bảng 7.7. Phiếu review experiment . . . 213
Bảng 7.8. Mẫu model registry nhập môn . . . 214
Bảng 7.9. Mẫu release note cho artefact AI . . . 214
Bảng 7.10. Lỗi quản lý dự án AI và cách sửa . . . 215
Bảng 7.11. Git workflow gợi ý cho nhóm làm dự án AI . . . 216
Bảng 7.12. Tư duy DVC có thể áp dụng ngay cả khi chưa dùng công cụ . 217
Bảng 7.13. Trạng thái model registry nhập môn . . . 217
Bảng 7.14. Mẫu reproducibility report . . . 218
Bảng 7.15. Template README cho dự án AI . . . 218
Bảng 7.16. Rubric khả năng tái lập dự án AI . . . 219
Bảng 7.17. Cách xử lý khi metric không tái lập được . . . 220
Bảng 7.18. Vai trò tối thiểu trong nhóm dự án AI . . . 221
Bảng 7.19. Mẫu issue cho lỗi reproducibility . . . 221
Bảng 7.20. rà soát nộp bài về quản lý phiên bản và thực nghiệm . . . 222
Bảng 7.21. Runbook rollback artefact AI . . . 223
Bảng 7.22. Mẫu postmortem nhẹ sau demo AI . . . 224

Bảng 7.23. Mẫu bảng rà quyền truy cập artefact . . . 225
Bảng 7.24. Artefact cần quan sát trong case MLflow . . . 226

Bảng 8.1. Các lớp test trong hệ thống AI . . . 233
Bảng 8.2. Metric gợi ý theo tác vụ AI . . . 237
Bảng 8.3. Metric phi chức năng tối thiểu . . . 238
Bảng 8.4. Rubric đánh giá câu trả lời RAG . . . 239
Bảng 8.5. Dashboard tối thiểu cho hệ thống AI nhập môn . . . 240
Bảng 8.6. Từ rủi ro đến test và metric . . . 242
Bảng 8.7. Data validation suite cho dự án AI nhập môn . . . 243
Bảng 8.8. Thành phần của model evaluation packet . . . 244
Bảng 8.9. Phân tích lỗi từ confusion matrix . . . 244
Bảng 8.10. Safety suite tối thiểu cho LLM/RAG . . . 245
Bảng 8.11. Kế hoạch test phi chức năng . . . 246
Bảng 8.12. Scenario monitoring và hành động . . . 246
Bảng 8.13. Rubric chấm evaluation report . . . 247
Bảng 8.14. Mẫu test case catalog cho hệ thống AI . . . 248
Bảng 8.15. Bug triage cho lỗi AI . . . 248
Bảng 8.16. Acceptance gate cho hệ thống AI nhập môn . . . 249
Bảng 8.17. Mẫu red-team log cho LLM/RAG . . . 250
Bảng 8.18. Evaluation packet cho bài tập lớn . . . 250
Bảng 8.19. Hướng xử lý khi RAG đúng nhưng chậm . . . 251
Bảng 8.20. QA runbook cho lỗi hệ thống AI . . . 252
Bảng 8.21. Phân mức độ nghiêm trọng cho lỗi AI . . . 253
Bảng 8.22. Mẫu regression suite sau khi sửa lỗi . . . 253
Bảng 8.23. Dashboard QA một trang . . . 254
Bảng 8.24. Ba loại quyết định sau đánh giá . . . 255
Bảng 8.25. Mẫu biên bản QA review . . . 255
Bảng 8.26. Acceptance bổ sung cho RAG sau mini case . . . 256
Bảng 8.27. Mẫu sign-off cho evaluation report . . . 257
Bảng 8.28. Áp dụng tư duy OpenAI Evals vào bài báo cáo thực hành . . . 258

Bảng 9.1. Thành phần cần kiểm tra khi đóng gói hệ thống AI . . . 265
Bảng 9.2. Pipeline CI/CD nhập môn cho ứng dụng AI . . . 266
Bảng 9.3. Chỉ số vận hành cần theo dõi . . . 268
Bảng 9.4. Mẫu incident note cho hệ thống AI . . . 269

Bảng 9.5. Rủi ro bảo mật và biện pháp giảm thiểu 270
Bảng 9.6. Hồ sơ bàn giao tối thiểu 273
Bảng 9.7. Deployment runbook cho hệ thống AI nhập môn 274
Bảng 9.8. Environment matrix cho triển khai AI 275
Bảng 9.9. Health/readiness check cho AI service 276
Bảng 9.10. Dashboard vận hành tối thiểu cho hệ thống AI 276
Bảng 9.11. Severity cho incident AI 277
Bảng 9.12. Rollback matrix cho hệ thống AI 277
Bảng 9.13. Rà soát bảo mật cho AI service 278
Bảng 9.14. Governance packet cho sản phẩm AI 279
Bảng 9.15. Agenda handover meeting cho sản phẩm AI 279
Bảng 9.16. Rubric triển khai, vận hành và quản trị 280
Bảng 9.17. Phân tích mini incident agent gửi draft sai nhóm 281
Bảng 9.18. Runbook xử lý lộ secret 281
Bảng 9.19. Quy trình kiểm soát chi phí AI API 282
Bảng 9.20. Chính sách retention nhập môn 283
Bảng 9.21. Kịch bản diễn tập rollback 283
Bảng 9.22. Rà soát human oversight 284
Bảng 9.23. Sign-off bàn giao hệ thống AI 284
Bảng 9.24. Retry policy an toàn hơn cho AI API 285
Bảng 9.25. Release readiness gate cho hệ thống AI 286
Bảng 9.26. Quyết định go/no-go ở góc độ vận hành 286
Bảng 9.27. Mẫu incident report hoàn chỉnh 287
Bảng 9.28. Hồ sơ nộp cuối cho sản phẩm AI 287
Bảng 9.29. Bảng ký xác nhận release readiness 289
Bảng 9.30. Bằng chứng triển khai cần có sau case Docker/GitHub Actions 290

Danh mục từ viết tắt

Từ viết tắt	Diễn giải
AI	Artificial Intelligence, trí tuệ nhân tạo
API	Application Programming Interface, giao diện lập trình ứng dụng
AUC	Area Under the Curve, diện tích dưới đường cong đánh giá mô hình
BERT	Bidirectional Encoder Representations from Transformers, họ mô hình ngôn ngữ dựa trên Transformer
CD	Continuous Delivery/Deployment, chuyển giao hoặc triển khai liên tục
CI	Continuous Integration, tích hợp liên tục
CI/CD	Continuous Integration/Continuous Delivery, tích hợp và chuyển giao/triển khai liên tục
CLI	Command Line Interface, giao diện dòng lệnh
CLO	Chuẩn đầu ra học phần
CPU	Central Processing Unit, bộ xử lý trung tâm
CRUD	Create, Read, Update, Delete; bốn thao tác dữ liệu cơ bản
CSV	Comma-Separated Values, tệp dữ liệu dạng giá trị phân tách bằng dấu phẩy
DVC	Data Version Control, công cụ quản lý phiên bản dữ liệu và pipeline
F1	F1-score, trung bình điều hòa giữa precision và recall
FAQ	Frequently Asked Questions, các câu hỏi thường gặp
GenAI	Generative AI, trí tuệ nhân tạo tạo sinh
GPU	Graphics Processing Unit, bộ xử lý đồ họa
HTTP/HTTPS	HyperText Transfer Protocol; HTTPS là bản có mã hóa

IDE	Integrated Development Environment, môi trường phát triển tích hợp
JSON	JavaScript Object Notation, định dạng dữ liệu có cấu trúc phổ biến trong API
JSONL	JSON Lines, định dạng mỗi dòng là một đối tượng JSON
LLM	Large Language Model, mô hình ngôn ngữ lớn
MAE	Mean Absolute Error, sai số tuyệt đối trung bình
MAPE	Mean Absolute Percentage Error, sai số phần trăm tuyệt đối trung bình
ML	Machine Learning, học máy
MLflow	Nền tảng theo dõi thực nghiệm, quản lý model và MLOps
MLOps	Machine Learning Operations, thực hành vận hành vòng đời hệ thống học máy
MRR	Mean Reciprocal Rank, chỉ số đánh giá thứ hạng trong truy xuất thông tin
NIST	National Institute of Standards and Technology, Viện Tiêu chuẩn và Công nghệ Quốc gia Hoa Kỳ
NLP	Natural Language Processing, xử lý ngôn ngữ tự nhiên
NPU	Neural Processing Unit, bộ xử lý chuyên dụng cho tác vụ AI
OCR	Optical Character Recognition, nhận dạng ký tự quang học
ONNX	Open Neural Network Exchange, định dạng mở để trao đổi mô hình học máy
OS	Operating System, hệ điều hành
OWASP	Open Worldwide Application Security Project, tổ chức và bộ khuyến nghị bảo mật ứng dụng
PDF	Portable Document Format, định dạng tài liệu di động
PI	Chỉ báo năng lực
PII	Personally Identifiable Information, thông tin định danh cá nhân
PLO	Chuẩn đầu ra chương trình đào tạo
QA	Quality Assurance, đảm bảo chất lượng
RAG	Retrieval-Augmented Generation, sinh nội dung có bổ sung truy xuất nguồn
RAM	Random Access Memory, bộ nhớ truy cập ngẫu nhiên

REST	Representational State Transfer, kiểu kiến trúc API phổ biến trên web
RMF	Risk Management Framework, khung quản lý rủi ro
RMSE	Root Mean Squared Error, căn bậc hai của sai số bình phương trung bình
SDK	Software Development Kit, bộ công cụ phát triển phần mềm
SEI	Software Engineering Institute, Viện Kỹ nghệ Phần mềm thuộc Carnegie Mellon University
SLA	Service Level Agreement, cam kết mức dịch vụ
SST-2	Stanford Sentiment Treebank 2, bộ dữ liệu phân loại cảm xúc câu tiếng Anh
SVN	Subversion, hệ thống quản lý phiên bản mã nguồn
TF-IDF	Term Frequency–Inverse Document Frequency, phương pháp biểu diễn văn bản theo tần suất từ và độ hiếm của từ
UI	User Interface, giao diện người dùng
URL	Uniform Resource Locator, địa chỉ tài nguyên trên web
UX	User Experience, trải nghiệm người dùng
YAML	YAML Ain’t Markup Language, định dạng cấu hình thường dùng trong pipeline và CI/CD

Hướng dẫn cài đặt và chạy lab

Phần này quy định môi trường thực hành tối thiểu cho toàn bộ giáo trình. Sinh viên không chỉ đọc case study; mỗi chương phải chạy được ít nhất một lệnh, sinh ra một file kết quả và đưa file đó vào báo cáo. Nếu máy cá nhân yếu, nhóm có thể dùng Google Colab, Kaggle Notebook hoặc máy lab, nhưng vẫn phải ghi rõ môi trường đã chạy.

1. Bộ công cụ bắt buộc

Máy thực hành nên cài các công cụ sau trước tuần học đầu tiên: Python 3.11 hoặc 3.12, Git, Visual Studio Code, Docker Desktop, Kaggle CLI, JupyterLab, và trình duyệt Chrome/Edge. Riêng chương mobile/edge cần Android Studio nếu nhóm muốn build app Android; nếu chưa có thiết bị, nhóm vẫn phải đọc repo TensorFlow Lite và chạy script đo latency mô phỏng.

```
python --version
git --version
docker --version
code --version
```

Mã nguồn 1. *Kiểm tra công cụ nền tảng trên máy*

Nếu một lệnh báo không tìm thấy chương trình, sinh viên phải cài công cụ đó trước khi vào lab. Trên Windows nên dùng PowerShell hoặc Git Bash. Trên macOS/Linux dùng Terminal. Khi dùng Python, mọi lab nên chạy trong virtual environment riêng để tránh lỗi thư viện giữa các chương.

```
mkdir se4ai-labs
cd se4ai-labs
python -m venv .venv

# macOS/Linux
```

```
source .venv/bin/activate

# Windows PowerShell
.venv\Scripts\Activate.ps1

python -m pip install --upgrade pip
pip install pandas scikit-learn transformers torch fastapi
   uvicorn pytest jupyterlab mlflow kaggle
```

Mã nguồn 2. *Tạo môi trường Python dùng chung cho các lab*

Với Kaggle, sinh viên cần tài khoản Kaggle và file token `kaggle.json`. Vào trang Kaggle, mở Account, tạo API token, tải file `kaggle.json`, sau đó đặt vào thư mục cấu hình.

```
# macOS/Linux
mkdir -p ~/.kaggle
cp kaggle.json ~/.kaggle/kaggle.json
chmod 600 ~/.kaggle/kaggle.json

# Windows PowerShell
mkdir $env:USERPROFILE\.kaggle
copy kaggle.json $env:USERPROFILE\.kaggle\kaggle.json

kaggle --version
```

Mã nguồn 3. *Cấu hình Kaggle CLI*

2. Cấu trúc thư mục nộp bài

Mỗi nhóm nên dùng một repo hoặc một thư mục chung theo cấu trúc dưới đây. Giảng viên có thể mở repo, chạy lệnh và thấy kết quả nằm trong `reports/`. Nếu báo cáo chỉ có ảnh chụp màn hình mà không có file output, bài chưa chứng minh được đã chạy thực nghiệm.

```
se4ai-project/
  README.md
  requirements.txt
```

```
  data/
    raw/
    processed/
  scripts/
    ch01_titanic_baseline.py
    ch02_hf_smoke_test.py
    ch04_fastapi_app.py
    ch05_measure_latency.py
    ch06_mini_retrieval.py
    ch08_run_eval.py
    ch09_deploy_smoke_test.py
  reports/
    ch01_metrics.json
    ch02_smoke_outputs.json
    ch05_latency.csv
    ch06_retrieval_topk.txt
    ch08_evaluation_summary.txt
    ch09_smoke_result.json
  docs/
    product_brief.md
    model_note.md
    evaluation_report.md
    deployment_runbook.md
```

Mã nguồn 4. *Cấu trúc thư mục thực hành đề nghị*

File `requirements.txt` tối thiểu có thể bắt đầu như sau. Nhóm chỉ thêm thư viện thật sự dùng trong dự án.

```
pandas
scikit-learn
transformers
torch
fastapi
uvicorn
pytest
jupyterlab
mlflow
kaggle
```

Mã nguồn 5. *requirements.txt tối thiểu cho các lab Python*

3. Lệnh chạy cụ thể theo chương

Bảng dưới đây là đường chạy tối thiểu. Sinh viên có thể làm sâu hơn, nhưng không được nộp thấp hơn mức này. Mỗi chương cần có lệnh chạy, file output và đoạn nhận xét dựa trên output thật. Các lệnh cụ thể được đặt ngay sau bảng để sinh viên có thể copy vào terminal.

Bảng 3. *Tool và file output bắt buộc theo chương*

Chương	Tool chính	File phải có trong báo cáo
1	Kaggle CLI, pandas, scikit-learn	`reports/ch01_metrics.json`, confusion matrix hoặc bảng lỗi mẫu.
2	Hugging Face Transformers	Model note, output JSON, latency, quyết định dùng/không dùng model.
3	GitHub, VS Code, FastAPI template	`docs/design_decision.md`, sơ đồ endpoint, API contract.
4	FastAPI, Uvicorn, curl, pytest	Response JSON thật, log lỗi, test report.
5	TensorFlow Lite repo, Android Studio hoặc script latency	`reports/ch05_latency.csv`, p50/p95, nhận xét thiết bị.
6	scikit-learn, JupyterLab, Hugging Face cookbook	Top-k retrieval, 30 câu hỏi eval, lỗi retrieval điển hình.
7	Git, MLflow, pytest	Run id, params, metrics, artefact path, clean clone note.
8	pandas, scikit-learn, pytest	Metric, confusion matrix, error samples, go/no-go decision.
9	Docker, Docker Compose, GitHub Actions	Docker image tag, smoke result, log, rollback note.

```
# Chapter 1: Kaggle Titanic baseline
```

```
kaggle competitions download -c titanic -p data/raw
python scripts/ch01_titanic_baseline.py

# Chapter 2: Hugging Face model smoke test
python scripts/ch02_hf_smoke_test.py > reports/
   ch02_smoke_outputs.json

# Chapter 3: Read a real FastAPI architecture template
git clone https://github.com/fastapi/full-stack-fastapi-
   template.git

# Chapter 4: Run an AI inference API
uvicorn scripts.ch04_fastapi_app:app --reload
curl -X POST http://127.0.0.1:8000/ai/classify \
  -H "Content-Type: application/json" \
  -d '{"text":"The demo is useful but the latency is high."}'
pytest

# Chapter 5: Inspect TensorFlow Lite and measure latency
git clone https://github.com/tensorflow/examples.git
python scripts/ch05_measure_latency.py

# Chapter 6: Mini retrieval and RAG evaluation notebook
python scripts/ch06_mini_retrieval.py > reports/
   ch06_retrieval_topk.txt
git clone https://github.com/huggingface/cookbook.git
jupyter lab cookbook/notebooks/en/rag_evaluation.ipynb

# Chapter 7: Experiment tracking
mlflow ui --host 127.0.0.1 --port 5000
pytest
python scripts/smoke_test.py

# Chapter 8: Evaluation report
python scripts/ch08_run_eval.py > reports/
   ch08_evaluation_summary.txt

# Chapter 9: Containerized deployment and smoke test
```

```
docker compose up --build -d
python scripts/ch09_deploy_smoke_test.py
```

Mã nguồn 6. *Lệnh chạy tối thiểu cho từng chương*

4. Một lab mẫu chạy từ đầu đến cuối

Đoạn dưới đây là mẫu tối thiểu cho Chương 2. Sinh viên có thể dùng nguyên cấu trúc này để kiểm tra môi trường trước khi làm bài lớn.

```
mkdir -p scripts reports

cat > scripts/ch02_hf_smoke_test.py <<'PY'
from transformers import pipeline
import json, time

model_id = "distilbert/distilbert-base-uncased-finetuned-sst
   -2-english"
clf = pipeline("text-classification", model=model_id)
inputs = [
    "The demo is clear and useful.",
    "The service is slow and confusing.",
    "This sentence is neutral and only describes a fact."
]

rows = []
for text in inputs:
    start = time.perf_counter()
    output = clf(text)[0]
    rows.append({
        "model_id": model_id,
        "input": text,
        "label": output["label"],
        "score": round(output["score"], 4),
        "latency_ms": round((time.perf_counter() - start) *
   1000, 2)
    })
```

```
print(json.dumps(rows, ensure_ascii=False, indent=2))
PY

python scripts/ch02_hf_smoke_test.py > reports/
   ch02_smoke_outputs.json
cat reports/ch02_smoke_outputs.json
```

Mã nguồn 7. *Tạo script smoke test Hugging Face và chạy ra file report*

Nếu chạy thành công, file `reports/ch02_smoke_outputs.json` phải có ba dòng kết quả, mỗi dòng có `model_id`, `input`, `label`, `score` và `latency_ms`. Báo cáo không chỉ chép file này; sinh viên phải nhận xét: input nào model xử lý tốt, input nào mơ hồ, score có đủ cao không, model tiếng Anh này có phù hợp với dữ liệu tiếng Việt của nhóm không.

5. Cách ghi bằng chứng chạy

Mỗi báo cáo thực hành cần ghi bốn loại bằng chứng: lệnh đã chạy, môi trường chạy, file output và kết luận kỹ thuật. Ví dụ: “Python 3.11, transformers, torch, chạy trên laptop CPU; lệnh chạy là `python scripts/ch02_hf_smoke_test.py`; output nằm ở `reports/ch02_smoke_outputs.json`; kết luận là model chạy được nhưng chưa phù hợp dữ liệu tiếng Việt”. Cách ghi này giúp người chấm biết nhóm thật sự đã chạy lab, không chỉ mô tả công cụ.

Phần I

Nền tảng kỹ nghệ hệ thống thông minh

CHƯƠNG 1

Tổng quan về hệ thống thông minh và AI Engineering

Nội dung chính

Người học bắt đầu bằng một việc cụ thể: viết bản *AI product brief* đầu tiên cho đề tài của nhóm. Để làm được việc đó, nhóm phải tách mô hình AI khỏi sản phẩm phần mềm hoàn chỉnh, nhận diện người dùng, dữ liệu, API, mô hình, kiểm thử, giám sát và trách nhiệm vận hành. Phần lý thuyết chỉ được dùng để trả lời câu hỏi thực hành: sản phẩm định làm là gì, AI nằm ở đâu, đo bằng gì và rủi ro nào phải kiểm soát trước khi code.

Mục tiêu cần đạt

1. Phân biệt được phần mềm truyền thống, phần mềm có thành phần AI và hệ thống thông minh.
2. Giải thích được quan hệ giữa AI, ML, DL, GenAI, hệ thống dựa trên luật và AI-enabled software.
3. Mô tả được các thành phần chính của một hệ thống AI ở mức kiến trúc sản phẩm.
4. Trình bày được vòng đời AI Engineering từ đặt vấn đề, dữ liệu, mô hình, tích hợp, đánh giá, triển khai đến giám sát.
5. Nhận diện được các khác biệt kỹ nghệ quan trọng: đầu ra xác suất, phụ thuộc dữ liệu, drift, metric đa lớp và human-in-the-loop.
6. Lập được AI product brief một trang cho đề tài bài tập lớn của nhóm.

Sản phẩm thực hành phải nộp

Nhóm nộp một *AI product brief* một trang cho đề tài của mình. Brief phải có: người dùng chính, vấn đề cần giải quyết, workflow hiện tại, điểm AI can thiệp, dữ liệu đầu vào, output dự kiến, metric kiểm tra, ba rủi ro lớn nhất và quyết định human-in-the-loop. Sản phẩm đạt yêu cầu khi nhóm khác đọc brief và mô tả lại được hệ thống sẽ làm gì, không làm gì và kiểm tra bằng cách nào.

Dạy và thực hành trong bối cảnh AI hiện đại

Giảng viên dạy trọng tâm: AI là một thành phần trong sản phẩm phần mềm, không phải toàn bộ sản phẩm. Người học cần nhìn thấy người dùng, client, backend, model service, dữ liệu, log, monitoring, quyền truy cập, human oversight và rủi ro vận hành trong cùng một hệ thống.

Sinh viên thực hành: Chọn ba sản phẩm có AI, vẽ sơ đồ thành phần và chỉ ra AI nằm ở đâu, dữ liệu đi qua đâu, ai kiểm tra khi AI không chắc, metric nào chứng minh sản phẩm hữu ích. Sau đó viết product brief cho đề tài bài tập lớn.

Tiêu chí đạt: Brief không được dừng ở câu "ứng dụng AI hỗ trợ học tập". Người đọc phải biết sản phẩm phục vụ ai, quyết định nào được AI hỗ trợ, input/output là gì, đo chất lượng bằng gì và rủi ro nào phải kiểm soát.

Tình huống học tập

Một nhóm sinh viên xây dựng chức năng hỏi đáp tài liệu môn học. Nhóm dùng mô hình ngôn ngữ lớn để trả lời câu hỏi từ đề cương, slide và tài liệu tham khảo. Bản thử nghiệm ban đầu trông rất thuyết phục: giao diện đẹp, câu trả lời trôi chảy, thời gian phản hồi nhanh. Tuy nhiên, khi giảng viên kiểm tra, hệ thống đôi khi trả lời sai quy định đánh giá, trích dẫn nhầm đoạn tài liệu, và vẫn trả lời tự tin khi không tìm thấy nguồn phù hợp. Vấn đề không nằm ở một dòng code duy nhất. Nhóm chưa mô tả rõ bài toán, chưa kiểm soát dữ liệu đầu vào, chưa có metric đánh giá groundedness, chưa thiết kế fallback, chưa lưu log để phân tích lỗi, và chưa xác định ai duyệt nội dung rủi ro. Tình huống đó cho thấy một tính năng AI chỉ trở thành sản phẩm đáng dùng khi được kỹ nghệ hóa như một hệ thống phần mềm hoàn chỉnh.

1.1. Hệ thống thông minh và sản phẩm phần mềm có AI

1.1.1. AI là thành phần, không phải toàn bộ sản phẩm

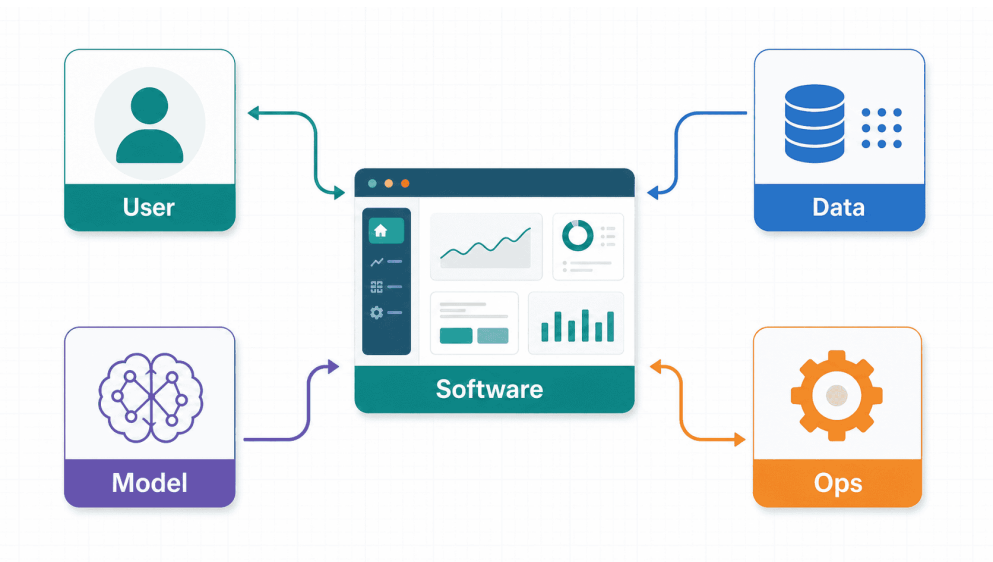
Trong giáo trình này, *hệ thống thông minh* được hiểu là một hệ thống phần mềm sử dụng dữ liệu, mô hình AI hoặc quy tắc suy luận để hỗ trợ nhận biết, dự đoán, gợi ý, tạo nội dung hoặc tự động hóa một phần quyết định. Cách định nghĩa này đặt AI vào đúng vị trí của môn học: AI cung cấp một phần năng lực thông minh, còn sản phẩm thực tế vẫn phải

được thiết kế, lập trình, kiểm thử, triển khai và vận hành như mọi hệ thống phần mềm nghiêm túc khác.

Một ứng dụng phân loại phản hồi sinh viên không chỉ gồm mô hình phân loại văn bản. Nó còn cần giao diện nhập phản hồi, API nhận dữ liệu, lớp kiểm tra quyền truy cập, cơ sở dữ liệu lưu kết quả, màn hình cho cố vấn học tập xem thống kê, cơ chế sửa nhãn sai và nhật ký vận hành. Tương tự, một chatbot hỏi đáp tài liệu không chỉ là lời nhắc gửi đến mô hình ngôn ngữ lớn. Nó còn cần kho tài liệu, bước tách đoạn, embedding, tìm kiếm, trích dẫn nguồn, kiểm soát câu trả lời, quản lý phiên bản tài liệu và cơ chế báo lỗi.

Không đồng nhất mô hình AI với hệ thống AI

Mô hình AI chỉ là một thành phần tạo năng lực dự đoán hoặc sinh nội dung. Hệ thống AI còn bao gồm dữ liệu, phần mềm, giao diện, API, hạ tầng, kiểm thử, vận hành, bảo mật và con người tham gia giám sát. Nếu chỉ đánh giá mô hình mà bỏ qua phần còn lại, sản phẩm có thể vẫn thất bại khi đưa vào sử dụng.



Hình 1.1. *Hệ thống thông minh là một sản phẩm phần mềm có thành phần AI, không phải chỉ là mô hình AI*

1.1.2. Các thuật ngữ nền tảng

AI là một lĩnh vực rộng, bao gồm nhiều cách làm khác nhau để máy tính thực hiện nhiệm vụ thường cần đến năng lực nhận thức của con người, chẳng hạn nhận dạng, dự đoán, lập

luận, lập kế hoạch, giao tiếp hoặc tạo nội dung. Machine Learning (ML) là một nhánh của AI, trong đó hệ thống học mẫu từ dữ liệu thay vì chỉ dựa vào quy tắc được lập trình thủ công. Deep Learning (DL) là một nhóm phương pháp ML dùng mạng nơ-ron nhiều lớp để học biểu diễn phức tạp từ dữ liệu. Generative AI (GenAI) là nhóm hệ thống có khả năng sinh văn bản, ảnh, âm thanh, mã nguồn hoặc dữ liệu mới dựa trên mẫu đã học.

Tuy nhiên, không phải hệ thống thông minh nào cũng cần ML hoặc DL. Một hệ thống dựa trên luật có thể rất hữu ích nếu bài toán có quy tắc rõ, dễ kiểm chứng và chi phí sai sót cao. Ví dụ, điều kiện "điểm chuyên cần dưới ngưỡng thì cảnh báo" có thể chỉ cần quy tắc. Ngược lại, phân loại cảm xúc trong phản hồi tự do của sinh viên có thể cần ML hoặc GenAI vì ngôn ngữ tự nhiên đa dạng và khó mô tả đầy đủ bằng luật.

Bảng 1.1. *Phân biệt AI, ML, DL, GenAI và hệ thống dựa trên luật*

Khái niệm	Ý chính	Ví dụ trong sản phẩm phần mềm
Rule-based system	Hành vi được đặc tả bằng luật, điều kiện và quy tắc nghiệp vụ.	Cảnh báo học vụ nếu số buổi vắng vượt ngưỡng; kiểm tra định dạng mã sinh viên.
AI	Khái niệm rộng về hệ thống thực hiện nhiệm vụ cần năng lực thông minh.	Trợ lý học tập, hệ thống gợi ý tài liệu, phát hiện gian lận, nhận dạng ảnh.
ML	Mô hình học từ dữ liệu để dự đoán, phân loại hoặc phát hiện mẫu.	Phân loại phản hồi sinh viên thành nhóm "hài lòng", "cần hỗ trợ", "khiếu nại".
DL	ML dùng mạng nơ-ron sâu, thường hiệu quả với ảnh, âm thanh, văn bản lớn.	Nhận dạng chữ viết tay, phát hiện đối tượng trong ảnh phòng thí nghiệm.
GenAI	Mô hình sinh nội dung mới như văn bản, ảnh, mã nguồn hoặc tóm tắt.	Chatbot hỏi đáp tài liệu môn học, sinh bản nháp email phản hồi sinh viên.
AI-enabled software	Phần mềm có một hoặc nhiều thành phần AI được tích hợp vào workflow sản phẩm.	Hệ thống quản lý học tập có gợi ý tài liệu, phân tích phản hồi và trợ lý hỏi đáp.

1.1.3. Khi nào nên dùng AI?

AI không phải câu trả lời mặc định cho mọi bài toán. Một nguyên tắc kỹ nghệ thực dụng là chỉ dùng AI khi nó tạo thêm giá trị so với giải pháp phần mềm truyền thống, và khi nhóm có đủ dữ liệu, metric, quy trình kiểm thử và cơ chế vận hành để kiểm soát rủi ro. Nếu bài toán có luật rõ, dữ liệu ít biến động, yêu cầu giải thích tuyệt đối và hậu quả sai cao, giải pháp rule-based hoặc quy trình con người duyệt có thể phù hợp hơn. Nếu bài

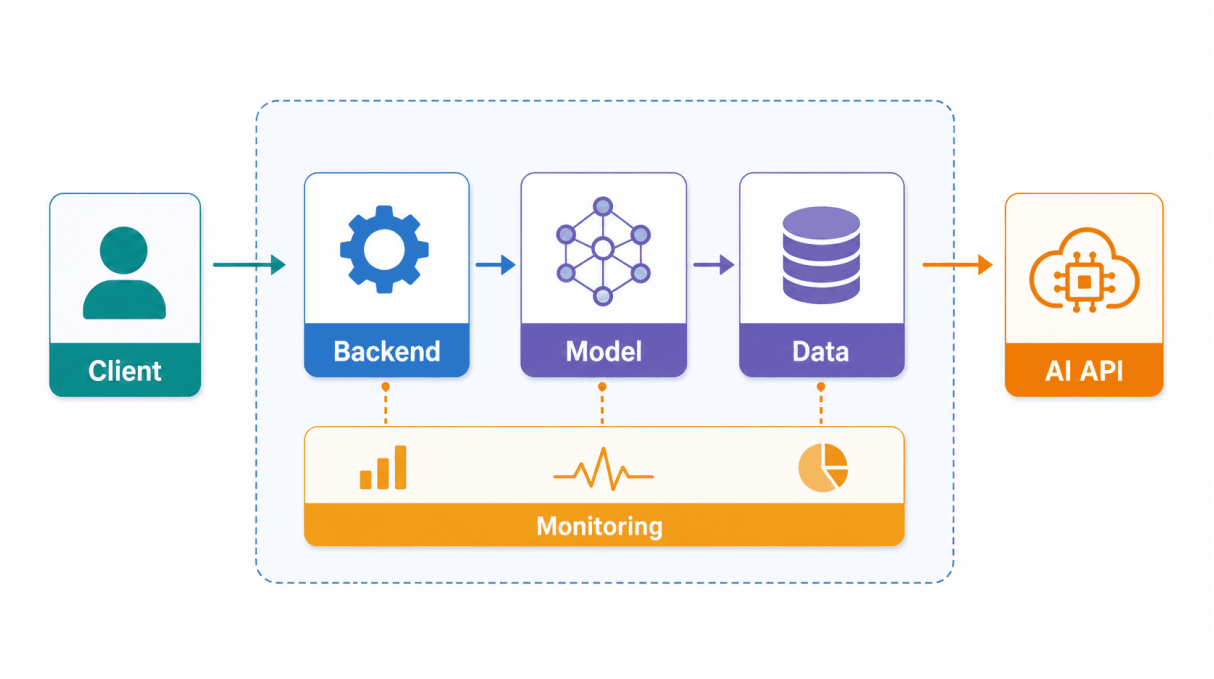
toán liên quan đến văn bản tự nhiên, ảnh, âm thanh, dự đoán từ dữ liệu lịch sử hoặc gợi ý cá nhân hóa, AI có thể đem lại lợi ích rõ rệt.

Kỹ sư phần mềm cần đặt câu hỏi ở mức sản phẩm: AI giúp người dùng hoàn thành việc gì nhanh hơn, chính xác hơn hoặc an toàn hơn? Sai số nào chấp nhận được? Khi AI không chắc, hệ thống nên hỏi thêm, từ chối, chuyển sang con người hay dùng quy tắc dự phòng? Câu trả lời cho các câu hỏi này thường quan trọng không kém việc chọn mô hình.

1.1.4. Thành phần của hệ thống phần mềm có AI

Từ người dùng đến mô hình.

Ở mức kiến trúc, một hệ thống AI thường bắt đầu từ người dùng và nhiệm vụ của họ, không bắt đầu từ mô hình. Người dùng tương tác qua web, mobile, desktop, thiết bị IoT hoặc một hệ thống khác gọi API. Backend tiếp nhận yêu cầu, kiểm tra quyền truy cập, chuẩn hóa dữ liệu, gọi model service hoặc AI API, xử lý hậu kỳ, ghi log và trả kết quả. Dữ liệu được lưu trong database, object storage, feature store hoặc vector store tùy bài toán. Monitoring theo dõi lỗi phần mềm, hiệu năng, chi phí và chất lượng AI sau triển khai.



Hình 1.2. *Kiến trúc khái quát của hệ thống phần mềm có thành phần AI*

Mỗi thành phần trong Hình 1.2 có trách nhiệm riêng. Client không nên tự quyết

định các chính sách bảo mật quan trọng. Backend không nên gửi dữ liệu nhạy cảm sang AI API nếu chưa có căn cứ pháp lý và cơ chế che giấu phù hợp. Model service không nên chỉ trả về văn bản tự do nếu hệ thống cần lưu nhãn, confidence, nguồn trích dẫn hoặc lý do từ chối. Monitoring không nên chỉ đo CPU và RAM, mà còn cần theo dõi lỗi suy luận, tỷ lệ fallback, độ trễ, chi phí và phản hồi của người dùng.

Dữ liệu, mô hình và hợp đồng API.

Trong phần mềm truyền thống, hợp đồng API thường mô tả endpoint, schema đầu vào, schema đầu ra, mã lỗi và quy tắc xác thực. Với hệ thống AI, hợp đồng này cần thêm thông tin về phiên bản mô hình, ngưỡng confidence, giới hạn dữ liệu đầu vào, cảnh báo rủi ro và cách diễn giải kết quả. Nếu không có hợp đồng rõ, client có thể hiển thị kết quả AI như một sự thật tuyệt đối, backend khó kiểm thử hồi quy, và nhóm vận hành khó truy vết khi có sự cố.

Ví dụ dưới đây minh họa phản hồi JSON từ một dịch vụ phân loại phản hồi sinh viên. Dù đây chưa phải API hoàn chỉnh, nó cho thấy một nguyên tắc quan trọng: kết quả AI nên có cấu trúc để phần mềm có thể xử lý, lưu trữ, kiểm thử và giám sát.

```
{
  "request_id": "fb-2026-00031",
  "model_version": "feedback-cls-v1",
  "label": "needs_support",
  "confidence": 0.86,
  "action_hint": "route_to_advisor",
  "warnings": ["low_context"]
}
```

Mã nguồn 1.1. *Ví dụ phản hồi JSON từ dịch vụ suy luận*

Ở mức nhập môn, điều quan trọng là hiểu vì sao output của AI phải được đặt vào một giao diện phần mềm có kiểm soát. Một kết quả có cấu trúc giúp nhóm dễ hiển thị, lưu vết, kiểm thử thủ công và giải thích khi người dùng hỏi lại.

Ba lớp chất lượng.

Chất lượng của hệ thống AI không thể được rút gọn thành một con số accuracy. Một mô hình có F1 cao vẫn có thể gây thất vọng nếu API phản hồi quá chậm. Một chatbot trả lời đúng ở dữ liệu thử nghiệm vẫn có thể nguy hiểm nếu nó tự bịa nguồn khi không tìm thấy tài liệu. Một tính năng gợi ý cá nhân hóa có thể tăng tương tác nhưng vi phạm quyền riêng tư nếu thu thập dữ liệu quá mức.

Vì vậy, cần đánh giá ở ít nhất ba lớp: metric mô hình, metric sản phẩm và metric vận hành. Ba lớp này liên hệ với nhau nhưng không thay thế cho nhau.

Bảng 1.2. *Metric mô hình, metric sản phẩm và metric vận hành*

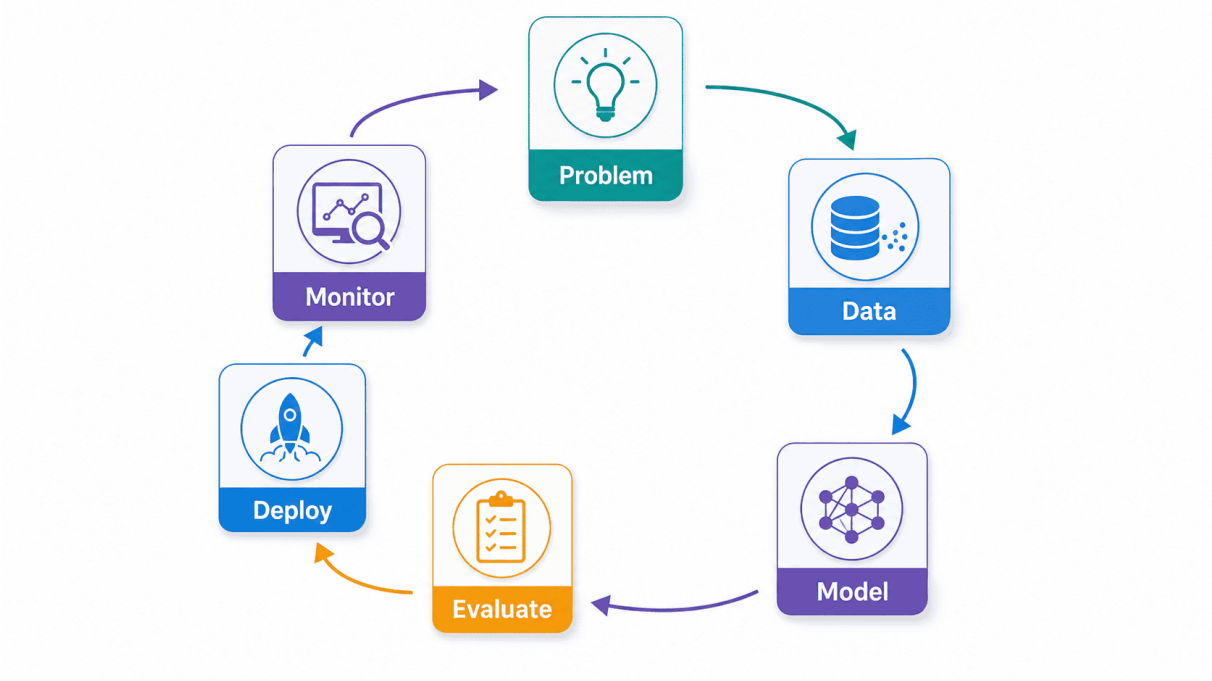
Lớp metric	Ví dụ	Câu hỏi kỹ nghệ
Mô hình	Accuracy, precision, recall, F1, MAE, groundedness, toxicity rate.	Mô hình có đủ tốt trên dữ liệu giống bối cảnh sử dụng thực tế không?
Sản phẩm	Tỷ lệ hoàn thành tác vụ, thời gian xử lý công việc, mức hài lòng, số lần người dùng phải sửa kết quả.	AI có giúp người dùng đạt mục tiêu tốt hơn cách làm cũ không?
Vận hành	Latency, throughput, error rate, availability, chi phí mỗi request, tỷ lệ fallback, drift signal.	Dịch vụ có ổn định, kiểm soát được chi phí và phát hiện được suy giảm chất lượng không?

1.2. Vòng đời, yêu cầu và kiến trúc AI Engineering

1.2.1. Vòng đời AI Engineering từ đặt vấn đề đến giám sát

AI Engineering được SEI mô tả như sự kết hợp giữa software engineering, systems engineering, computer science và human-centered design để đưa AI vào bối cảnh thực tế [1, 2]. Vì vậy, vòng đời của hệ thống AI không bắt đầu bằng câu hỏi "dùng model nào", mà bắt đầu bằng câu hỏi "bài toán sản phẩm là gì, ai chịu ảnh hưởng, và tiêu chí thành công là gì".

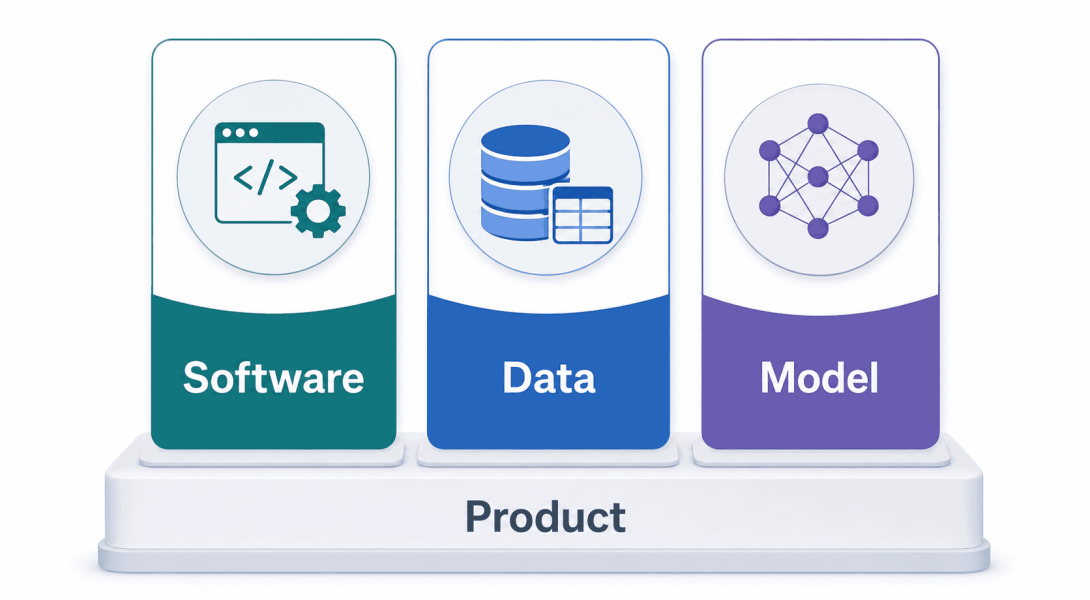
Một vòng đời nhập môn có thể gồm bảy bước: đặt vấn đề, chuẩn bị dữ liệu, chọn hoặc huấn luyện mô hình, tích hợp vào phần mềm, đánh giá, triển khai, giám sát và cải tiến. Các bước này không phải đường thẳng một chiều. Khi đánh giá phát hiện dữ liệu thử nghiệm không đại diện, nhóm phải quay lại bước dữ liệu. Khi triển khai thấy latency quá cao, nhóm phải xem lại kiến trúc hoặc chọn mô hình nhẹ hơn. Khi monitoring phát hiện drift, nhóm phải cập nhật dữ liệu, điều chỉnh ngưỡng hoặc huấn luyện lại.



Hình 1.3. *Vòng đời kỹ nghệ của hệ thống AI*

Ba trụ cột của AI Engineering.

Một hệ thống AI tốt không chỉ cần chính xác. Nó cần phục vụ con người, hoạt động tin cậy và mở rộng được trong môi trường triển khai. Có thể tóm tắt định hướng này bằng ba trụ cột: human-centered, robust and secure, scalable.



Hình 1.4. *Ba trụ cột AI Engineering ở mức nhập môn*

Human-centered nhắc nhóm phát triển rằng AI không thay thế việc hiểu người

dùng. Người dùng cần biết khi nào kết quả là gợi ý, khi nào cần kiểm chứng, và ai chịu trách nhiệm cuối cùng. Robust and secure nhấn mạnh hệ thống phải chịu được dữ liệu nhiễu, đầu vào bất thường, prompt độc hại, thay đổi phân phối dữ liệu và rủi ro bảo mật. Scalable yêu cầu hệ thống có thể phục vụ nhiều người dùng, kiểm soát chi phí, quản lý phiên bản và vận hành được sau ngày demo.

Khác biệt với phần mềm truyền thống.

Phần mềm truyền thống và phần mềm có AI chia sẻ nhiều nguyên tắc: yêu cầu rõ, thiết kế kiến trúc, quản lý mã nguồn, kiểm thử, CI/CD, bảo mật và tài liệu. Điểm khác là một phần hành vi của hệ thống AI được học từ dữ liệu hoặc đến từ mô hình bên ngoài. Điều này làm thay đổi cách viết yêu cầu, kiểm thử và vận hành.

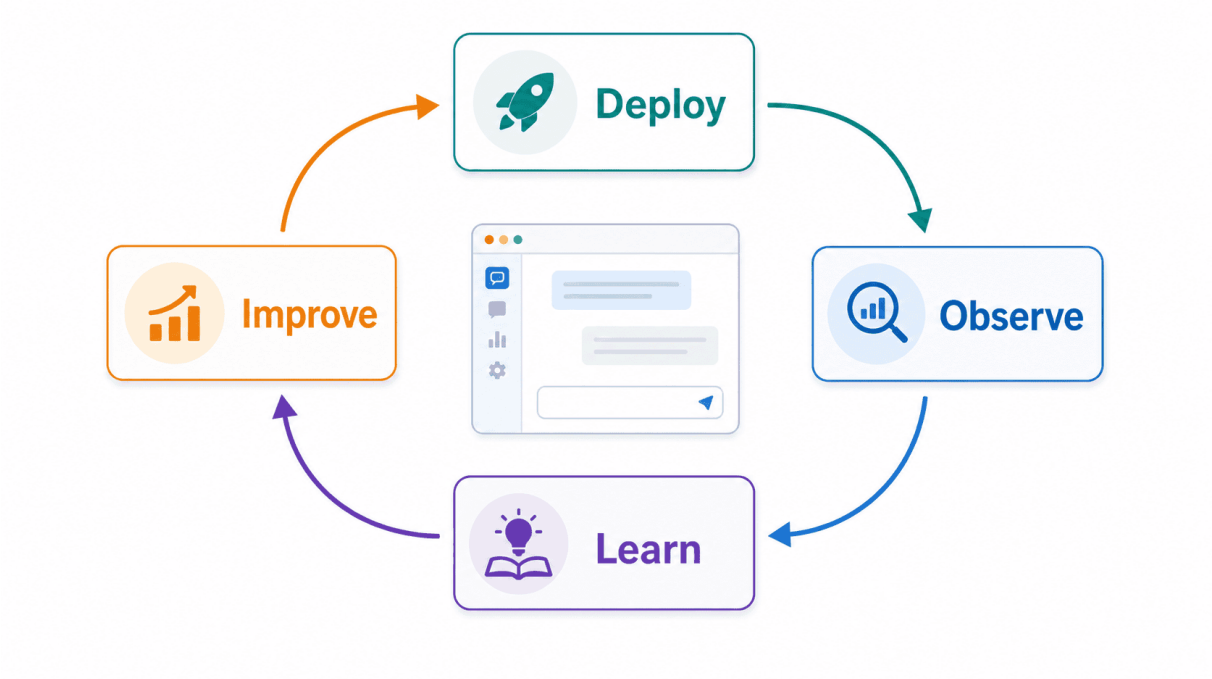
Bảng 1.3. *Phần mềm truyền thống và phần mềm có thành phần AI*

Tiêu chí	Phần mềm truyền thống	Phần mềm có thành phần AI
Logic xử lý	Chủ yếu do lập trình viên đặc tả bằng quy tắc, thuật toán và điều kiện rõ ràng.	Một phần hành vi đến từ mô hình học từ dữ liệu hoặc mô hình đã huấn luyện trước.
Đầu ra	Thường xác định nếu đầu vào và trạng thái hệ thống giống nhau.	Có thể mang tính xác suất, phụ thuộc ngưỡng, dữ liệu, prompt và phiên bản mô hình.
Yêu cầu	Có thể mô tả bằng chức năng, quy tắc nghiệp vụ và luồng xử lý.	Cần mô tả thêm dữ liệu, metric, sai số chấp nhận được, fallback và giám sát.
Kiểm thử	Tập trung vào unit, integration, UI, hiệu năng và bảo mật.	Cần thêm kiểm thử dữ liệu, mô hình, bias, drift, robustness và chất lượng sau triển khai.
Vận hành	Chủ yếu theo dõi lỗi phần mềm, tài nguyên và hành vi người dùng.	Cần theo dõi cả chất lượng dự đoán, phân phối dữ liệu, chi phí suy luận và rủi ro AI.

Sculley và cộng sự gọi nhiều vấn đề trong hệ thống ML là "nợ kỹ thuật ẩn" vì chúng không chỉ nằm trong code, mà còn nằm trong dữ liệu, pipeline, cấu hình, phụ thuộc mô hình và cách hệ thống thay đổi theo thời gian [3]. Các rubric kiểm tra mức sẵn sàng sản xuất của ML cũng nhấn mạnh dữ liệu, kiểm thử, monitoring và quy trình triển khai chứ không chỉ metric offline [4].

Vòng lặp triển khai, giám sát và cải tiến.

Trong hệ thống AI, triển khai không phải điểm kết thúc. Sau triển khai, nhóm cần thu thập log phù hợp, theo dõi metric, nhận phản hồi người dùng, phân tích lỗi và cập nhật hệ thống. Với bài tập lớn, sinh viên có thể chưa vận hành ở quy mô thật, nhưng vẫn cần mô phỏng tư duy này bằng evaluation report, log mẫu và kế hoạch monitoring.



Hình 1.5. *Vòng lặp triển khai, giám sát và cải tiến trong hệ thống AI*

Thực hành

Nhiệm vụ: Phân tích ba sản phẩm có thành phần AI bằng quy trình thủ công.
Sản phẩm: Một bảng gồm ba dòng, mỗi dòng là một sản phẩm AI. Với mỗi sản phẩm, ghi người dùng, nhiệm vụ AI, dữ liệu đầu vào, kết quả đầu ra, metric cần quan tâm và rủi ro nếu AI sai. Cuối bảng, chọn một sản phẩm phù hợp để phát triển thành bài tập lớn của nhóm.

1.2.2. Đặc tả yêu cầu cho tính năng AI

Từ yêu cầu chức năng đến yêu cầu chất lượng AI.

Trong các môn kỹ nghệ phần mềm cơ bản, yêu cầu chức năng thường được viết

dưới dạng hệ thống phải làm gì: đăng nhập, tìm kiếm, thêm bản ghi, xuất báo cáo, gửi thông báo. Với hệ thống có AI, cách viết này vẫn cần thiết nhưng chưa đủ. Một yêu cầu kiểu "hệ thống trả lời câu hỏi của sinh viên" quá mơ hồ, vì nó không nói câu hỏi nào nằm trong phạm vi, nguồn nào được phép dùng, câu trả lời phải có trích dẫn hay không, độ trễ tối đa là bao nhiêu, khi không chắc thì xử lý thế nào, và câu trả lời sai có hậu quả gì.

Yêu cầu cho tính năng AI cần mô tả cả hành vi mong muốn và biên an toàn. Ví dụ tốt hơn là: "Khi sinh viên hỏi về nội dung có trong đề cương, slide hoặc FAQ đã duyệt, hệ thống trả lời bằng tiếng Việt trong tối đa 8 giây, kèm ít nhất một nguồn trích dẫn. Nếu không tìm thấy nguồn phù hợp, hệ thống phải nói rõ chưa đủ căn cứ và gợi ý liên hệ giảng viên hoặc trợ giảng." Yêu cầu này chưa nói đến thuật toán, nhưng đã định hình hành vi sản phẩm, dữ liệu được phép dùng, ngưỡng chất lượng và fallback.

Bảng 1.4. *Ví dụ chuyển yêu cầu mơ hồ thành yêu cầu AI có thể kiểm thử*

Mức mô tả	Cách viết yếu	Cách viết tốt hơn
Chức năng	Hệ thống dùng AI để trả lời câu hỏi.	Hệ thống trả lời câu hỏi về tài liệu môn học đã được đưa vào kho tri thức.
Nguồn dữ liệu	Dùng tài liệu môn học.	Chỉ dùng đề cương, slide, FAQ và hướng dẫn bài tập lớn đã được giảng viên duyệt.
Chất lượng	Trả lời chính xác.	Câu trả lời phải có trích dẫn nguồn; nếu không có nguồn, hệ thống phải từ chối trả lời chắc chắn.
Hiệu năng	Phản hồi nhanh.	95% request thông thường phản hồi dưới 8 giây trong điều kiện demo lớp học.
Rủi ro	Tránh trả lời sai.	Với câu hỏi về điểm số, kỷ luật hoặc quyết định chính thức, hệ thống chỉ cung cấp thông tin tham khảo và gợi ý liên hệ người phụ trách.

Điểm quan trọng là yêu cầu AI phải kiểm thử được. "Thông minh", "tốt", "tự

nhiên", "chính xác cao" là các từ dễ gây hiểu nhầm nếu không kèm tiêu chí đo. Trong bài tập lớn, nhóm không cần đạt chất lượng công nghiệp, nhưng cần chứng minh rằng mình biết biến kỳ vọng mơ hồ thành tiêu chí có thể quan sát, thử nghiệm và tranh luận.

User story cho tính năng AI.

User story vẫn hữu ích khi làm sản phẩm AI, miễn là nhóm bổ sung tiêu chí chấp nhận phù hợp. Một user story nhập môn có dạng: "Là sinh viên, tôi muốn hỏi hệ thống về hạn nộp bài tập lớn để biết mình cần chuẩn bị gì." Tiêu chí chấp nhận có thể gồm: câu trả lời phải lấy từ tài liệu đã duyệt, hiển thị nguồn, nói rõ nếu tài liệu không có hạn nộp, và không bịa ngày.

Ví dụ user story

User story: Là sinh viên đang đọc đề cương, tôi muốn hỏi trợ lý AI về yêu cầu bài tập lớn để nhanh chóng nắm sản phẩm cần nộp.
Acceptance criteria: Hệ thống chỉ trả lời dựa trên tài liệu đã duyệt; câu trả lời có ít nhất một trích dẫn; nếu tài liệu không có thông tin, hệ thống phải nói chưa tìm thấy căn cứ; câu trả lời không được yêu cầu sinh viên cung cấp dữ liệu cá nhân không cần thiết.

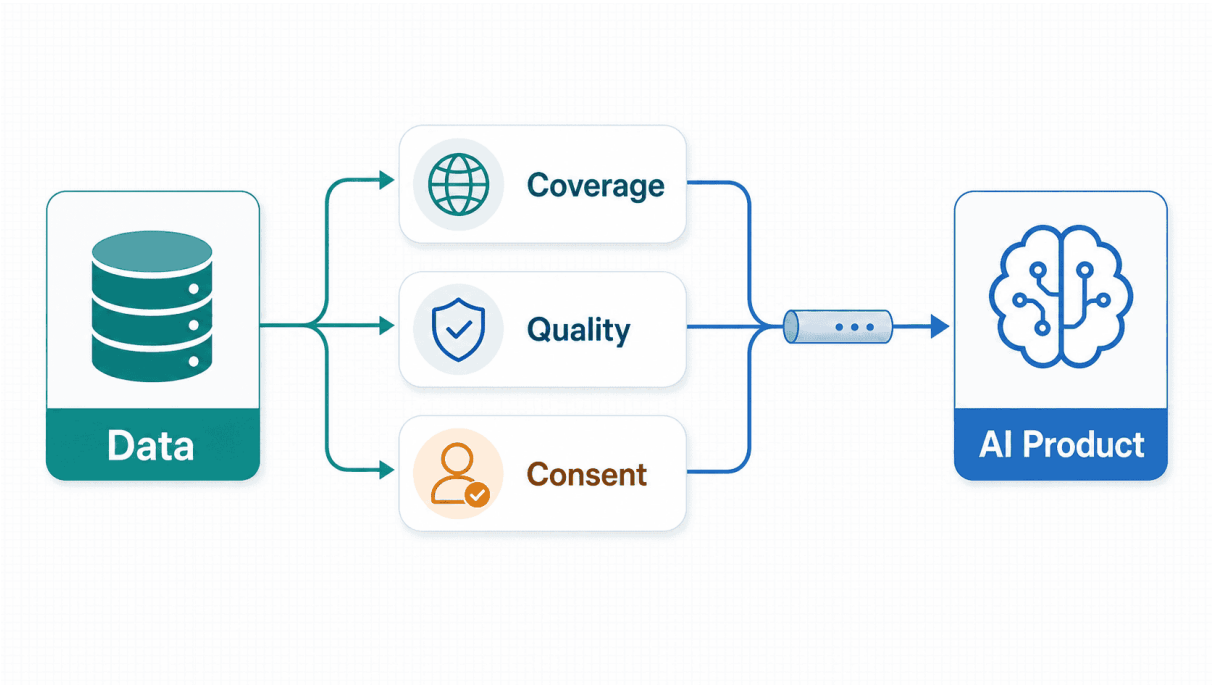
Khi viết user story cho AI, nhóm nên thêm ba loại tiêu chí: tiêu chí chất lượng kết quả, tiêu chí an toàn và tiêu chí vận hành. Tiêu chí chất lượng mô tả đầu ra đúng là gì. Tiêu chí an toàn mô tả hệ thống không được làm gì. Tiêu chí vận hành mô tả phản hồi, log, fallback hoặc thông báo lỗi.

Bảng 1.5. *Ba nhóm tiêu chí chấp nhận cho user story AI*

Nhóm tiêu chí	Câu hỏi kiểm tra	Ví dụ
Chất lượng kết quả	Kết quả đúng, đủ, có căn cứ và hữu ích ở mức nào?	Câu trả lời phải kèm nguồn; nhãn phân loại phải có confidence; kết quả tóm tắt không bỏ qua điều kiện quan trọng.
An toàn và trách nhiệm	Hệ thống không được làm gì, hoặc khi nào phải từ chối/chuyển người?	Không suy đoán điểm cá nhân; không tiết lộ thông tin sinh viên khác; từ chối câu hỏi ngoài phạm vi tài liệu.
Vận hành	Hệ thống phản hồi, ghi log và xử lý lỗi ra sao?	Lỗi AI API phải trả thông báo thân thiện; request phải có request ID; kết quả confidence thấp phải được gắn cảnh báo.

Dữ liệu là một phần của yêu cầu.

Trong hệ thống AI, dữ liệu không chỉ là tài nguyên kỹ thuật mà còn là một phần của yêu cầu. Cùng một mô hình phân loại văn bản có thể hoạt động tốt trên phản hồi tiếng Anh nhưng kém trên phản hồi tiếng Việt. Một chatbot RAG có thể trả lời đúng khi tài liệu gọn, nhưng trả lời sai khi tài liệu trùng lặp, lỗi thời hoặc có nhiều phiên bản mâu thuẫn. Vì vậy, khi mô tả yêu cầu, nhóm cần mô tả nguồn dữ liệu, quyền sử dụng, chất lượng, định dạng, tần suất cập nhật và dữ liệu không được xử lý.



Hình 1.6. *Dữ liệu cần được xem như một phần của yêu cầu hệ thống AI*

Trong bài tập lớn, mô tả dữ liệu tối thiểu nên trả lời: dữ liệu đến từ đâu, có được phép dùng không, có dữ liệu cá nhân không, định dạng là gì, kích thước khoảng bao nhiêu, ai chịu trách nhiệm cập nhật, và dữ liệu nào phải loại trừ. Nếu nhóm dùng dữ liệu tự tạo hoặc dữ liệu mẫu, cần nói rõ giới hạn đó khi đánh giá.

Human-in-the-loop.

Human-in-the-loop là thiết kế trong đó con người tham gia ở một hoặc nhiều điểm của quy trình AI: duyệt dữ liệu, sửa nhãn, kiểm tra kết quả, phê duyệt hành động rủi ro hoặc xử lý trường hợp hệ thống không chắc chắn. Đây không phải dấu hiệu sản phẩm yếu. Ngược lại, trong nhiều bối cảnh, human-in-the-loop là cách làm có trách nhiệm vì AI chỉ hỗ trợ, không tự ý quyết định các vấn đề ảnh hưởng trực tiếp đến con người.

Ví dụ, hệ thống phân loại phản hồi sinh viên có thể tự động gom phản hồi thành nhóm chủ đề, nhưng phản hồi liên quan đến sức khỏe tâm lý, khiếu nại nghiêm trọng hoặc thông tin cá nhân cần được chuyển cho người phụ trách. Hệ thống gợi ý điểm rủi ro học vụ có thể giúp cố vấn ưu tiên hỗ trợ, nhưng không nên tự động đưa ra quyết định kỷ luật hoặc loại sinh viên khỏi lớp. Trợ lý RAG có thể trả lời câu hỏi thông thường, nhưng các câu hỏi về điểm số chính thức nên được đánh dấu là cần xác nhận.

Bảng 1.6. *Các điểm có thể đưa human-in-the-loop vào hệ thống AI*

Điểm can thiệp	Khi nào cần	Ví dụ
Trước suy luận	Dữ liệu đầu vào nhạy cảm, thiếu quyền sử dụng hoặc cần làm sạch.	Ẩn thông tin cá nhân trước khi gửi văn bản sang AI API.
Trong đánh giá	Cần nhãn chuẩn, đánh giá chất lượng hoặc so sánh phương án.	Giảng viên chấm mẫu 50 câu trả lời RAG để tính groundedness.
Sau suy luận	Confidence thấp, kết quả tác động đến con người hoặc có dấu hiệu rủi ro.	Chuyển phản hồi "cần hỗ trợ khẩn cấp" cho cố vấn học tập xem trước.
Sau triển khai	Người dùng báo lỗi, dữ liệu thay đổi hoặc metric suy giảm.	Phân tích log câu hỏi không trả lời được để cập nhật FAQ.

Thiết kế human-in-the-loop cũng cần thực tế. Nếu mọi câu trả lời đều phải duyệt thủ công, hệ thống có thể mất lợi ích tốc độ. Nếu không câu nào được duyệt, rủi ro có thể quá cao. Nhiệm vụ của kỹ sư là xác định đúng ngưỡng: việc nào AI được tự động hóa, việc nào cần cảnh báo, việc nào phải chuyển sang con người.

Quyết định kiến trúc ở mức nhập môn.

Ngay từ bản mô tả ban đầu, nhóm đã có thể nhận diện một số quyết định kiến trúc lớn. Quyết định đầu tiên là AI chạy ở đâu: trong ứng dụng, trong service riêng, qua API bên ngoài, hay trên thiết bị người dùng. Quyết định thứ hai là dữ liệu được lưu và truy xuất như thế nào. Quyết định thứ ba là hệ thống kiểm soát lỗi ra sao khi AI không chắc, dịch vụ bên ngoài chậm hoặc dữ liệu không hợp lệ.

Bảng 1.7. *Một số lựa chọn kiến trúc ban đầu cho tính năng AI*

Lựa chọn	Phù hợp khi	Trade-off chính
AI API bên ngoài	Cần thử nghiệm nhanh, không có GPU, dùng mô hình lớn hoặc GenAI.	Phụ thuộc nhà cung cấp, chi phí theo request, rủi ro dữ liệu và thay đổi API.
Model service riêng	Cần kiểm soát phiên bản, endpoint, scaling và monitoring.	Cần triển khai, bảo trì, đóng gói model và quản lý tài nguyên.
Embedded model	Mô hình nhỏ, logic đơn giản, muốn giảm độ trễ gọi mạng.	Khó cập nhật độc lập, tăng kích thước ứng dụng, giới hạn tài nguyên.
On-device AI	Cần offline, quyền riêng tư hoặc độ trễ thấp trên mobile/edge.	Mô hình phải nhẹ, phụ thuộc phần cứng, khó quan sát lỗi sau triển khai.
Human-assisted workflow	Rủi ro cao hoặc dữ liệu cần chuyên gia xác nhận.	Tốn thời gian con người, cần giao diện duyệt và quy trình phân quyền.

Không có lựa chọn nào luôn đúng. Trong bài tập lớn, API bên ngoài có thể giúp nhóm tập trung vào kỹ nghệ tích hợp, nhưng nhóm vẫn phải mô tả cách bảo vệ dữ liệu, kiểm soát chi phí và xử lý lỗi. Với bài toán phân loại đơn giản, model service riêng bằng scikit-learn hoặc Transformers có thể phù hợp hơn vì dễ kiểm thử và dễ triển khai local. Với bài toán mobile, on-device AI có thể là hướng tốt nếu dữ liệu nhạy cảm hoặc cần chạy offline.

Thực hành

Nhiệm vụ: Viết đặc tả yêu cầu ngắn cho một tính năng AI.
Sản phẩm: Một trang gồm một user story, 5-7 acceptance criteria, mô tả nguồn dữ liệu, một điểm human-in-the-loop và một lựa chọn kiến trúc ban đầu. Với mỗi acceptance criterion, đánh dấu nó thuộc nhóm chất lượng kết quả, an toàn hay vận hành.

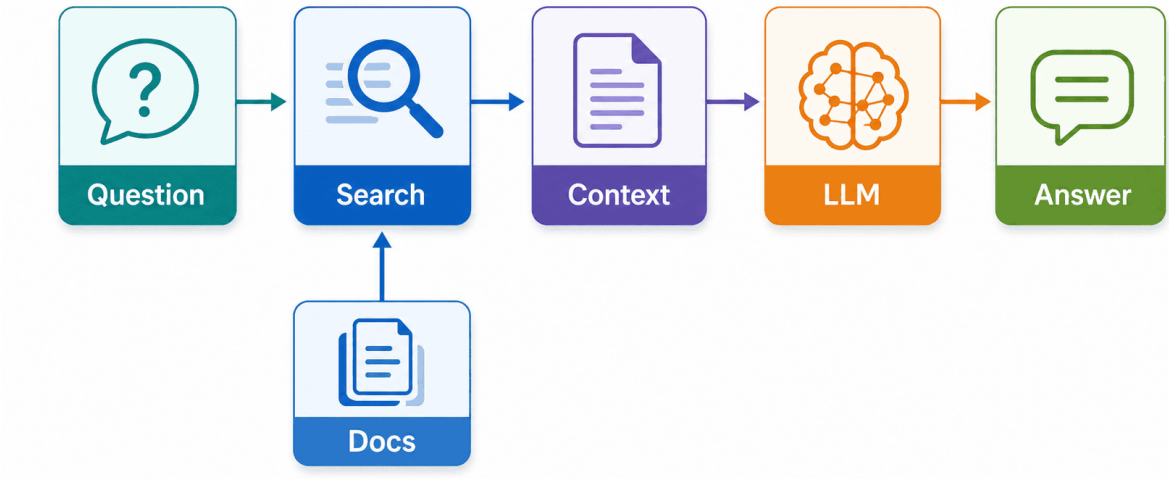
1.3. Case study và thực hành bài tập lớn

1.3.1. Case study: trợ lý RAG hỏi đáp tài liệu môn học

Retrieval-Augmented Generation (RAG) là một cách xây dựng ứng dụng hỏi đáp trong đó hệ thống tìm các đoạn tài liệu liên quan trước, sau đó đưa chúng vào ngữ cảnh để mô hình ngôn ngữ sinh câu trả lời. Với bối cảnh môn học, nhóm sinh viên có thể xây dựng trợ lý hỏi đáp dựa trên đề cương, slide, quy định đánh giá, hướng dẫn bài tập lớn và tài liệu tham khảo. Người dùng chính là sinh viên cần tra cứu nhanh và giảng viên cần giảm số câu hỏi lặp lại.

Sản phẩm nghe có vẻ đơn giản, nhưng quyết định kỹ nghệ xuất hiện ở mọi bước. Tài liệu nào được phép đưa vào kho? Khi tài liệu cập nhật, vector store có cập nhật theo không? Câu trả lời có trích dẫn nguồn không? Nếu câu hỏi nằm ngoài tài liệu, hệ thống có từ chối không? Log có lưu nội dung câu hỏi nhạy cảm không? Có cần giảng viên duyệt câu trả lời cho các câu hỏi về điểm số, kỷ luật hoặc quy định chính thức không?

Luồng xử lý RAG ở mức hệ thống.



Hình 1.7. *Luồng xử lý khái quát của trợ lý RAG hỏi đáp tài liệu môn học*

Trong luồng này, mô hình ngôn ngữ lớn chỉ là một bước. Nếu bước tách đoạn tài liệu kém, hệ thống có thể không tìm đúng nguồn. Nếu prompt không yêu cầu trích dẫn,

câu trả lời có thể nghe hợp lý nhưng không kiểm chứng được. Nếu không có guardrail, hệ thống có thể trả lời cả những câu hỏi nằm ngoài phạm vi. Nếu không ghi log, nhóm không biết người dùng thường hỏi gì và lỗi xuất hiện ở đâu.

Product brief mẫu cho case study.

Bảng 1.8. *AI product brief rút gọn cho trợ lý RAG hỏi đáp tài liệu môn học*

Mục	Nội dung minh họa
Bài toán	Hỗ trợ sinh viên tra cứu nhanh thông tin trong tài liệu môn học và giảm câu hỏi lặp lại cho giảng viên.
Người dùng	Sinh viên, giảng viên, trợ giảng; mức độ tin cậy cần cao với thông tin về lịch học, điểm số và yêu cầu bài tập.
Đầu vào	Câu hỏi tiếng Việt, đề cương môn học, slide, hướng dẫn bài tập lớn, FAQ đã duyệt.
Đầu ra	Câu trả lời ngắn, trích dẫn nguồn, mức cảnh báo khi thiếu căn cứ, gợi ý liên hệ giảng viên nếu cần.
Thành phần AI	Embedding để tìm đoạn liên quan; mô hình ngôn ngữ lớn để sinh câu trả lời có căn cứ.
Metric	Groundedness, tỷ lệ câu trả lời có nguồn, latency, tỷ lệ fallback đúng, mức hài lòng của người dùng.
Rủi ro	Hallucination, tài liệu lỗi thời, lộ dữ liệu cá nhân, người dùng hiểu nhầm câu trả lời AI là quyết định chính thức.

Ví dụ product brief cũng có thể được biểu diễn dưới dạng JSON để nhóm dùng trong issue tracker, API mock hoặc tài liệu kỹ thuật.

```
{
  "project": "course_document_qa",
  "users": ["student", "lecturer", "teaching_assistant"],
  "ai_task": "answer_questions_with_citations",
  "inputs": ["pdf", "docx", "slides", "user_question"],
  "outputs": ["answer", "citations", "confidence_note"],
  "success_metrics": ["groundedness", "latency", "
   fallback_rate"],
  "risks": ["hallucination", "outdated_documents", "
   private_data_leakage"]
```

```
}
```

Mã nguồn 1.2. *Product brief ngắn ở dạng JSON*

Thực hành

Nhiệm vụ: Vẽ sơ đồ thành phần cho một tính năng AI.
Sản phẩm: Chọn một đề tài bài tập lớn tiềm năng và vẽ sơ đồ gồm ít nhất năm thành phần: client, backend, model service hoặc AI API, nơi lưu dữ liệu, và logging/monitoring. Dưới sơ đồ, viết 5-7 dòng giải thích dữ liệu đi qua hệ thống như thế nào từ lúc người dùng nhập yêu cầu đến lúc nhận kết quả.

1.3.2. Hồ sơ bài tập lớn và phối hợp người-AI

AI product brief một trang.

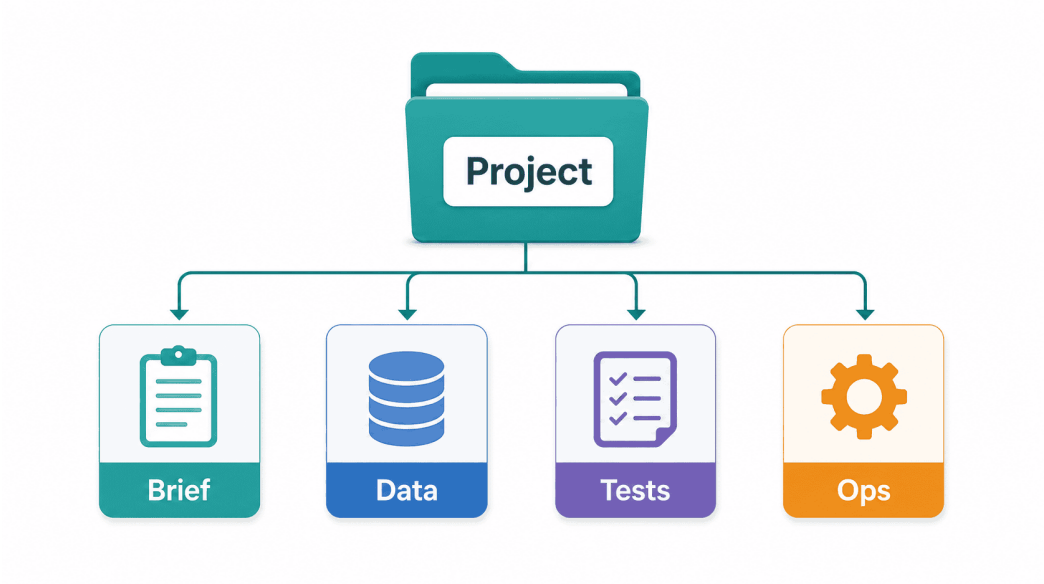
Sản phẩm quan trọng nhất ở giai đoạn khởi động là AI product brief một trang. Brief này không nhằm chứng minh nhóm đã có giải pháp cuối cùng, mà nhằm giúp nhóm đặt bài toán đúng trước khi chọn mô hình hoặc viết code. Một brief tốt phải đủ cụ thể để người khác hiểu sản phẩm làm gì, cho ai, dùng dữ liệu nào, đo thành công bằng gì và rủi ro chính nằm ở đâu.

Bảng 1.9. *Mẫu AI product brief một trang cho bài tập lớn*

Mục cần mô tả	Câu hỏi gợi ý
Bài toán	Sản phẩm giúp người dùng giải quyết việc gì? Vì sao việc đó cần AI hoặc hưởng lợi từ AI?
Người dùng	Ai sử dụng kết quả? Người dùng có đủ chuyên môn để kiểm tra kết quả AI không?
Đầu vào	Dữ liệu đưa vào hệ thống là văn bản, ảnh, âm thanh, bảng dữ liệu hay tài liệu? Có dữ liệu nhạy cảm không?
Đầu ra	Hệ thống trả về nhãn, điểm số, gợi ý, văn bản, cảnh báo hay hành động tự động?
Mô hình dự kiến	Có thể dùng mô hình đã huấn luyện, API AI, RAG, mô hình on-device hay quy tắc thủ công?
Metric	Thành công được đo bằng độ chính xác, thời gian xử lý, mức hài lòng, chi phí, độ an toàn hay khả năng giải thích?
Rủi ro	Nếu kết quả sai, thiên lệch, chậm, bị lộ dữ liệu hoặc bị tấn công thì hậu quả là gì?

Bộ hồ sơ tối thiểu.

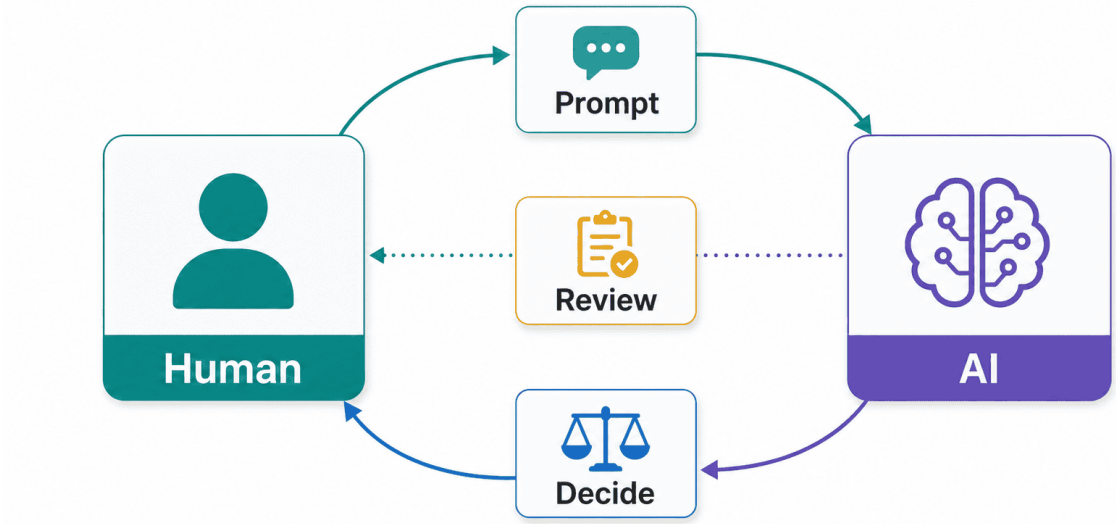
NIST AI RMF gợi ý quản lý rủi ro AI thông qua các hoạt động govern, map, measure và manage [5]. Ở mức bài tập lớn, sinh viên chưa cần xây dựng hệ thống quản trị phức tạp, nhưng cần có hồ sơ tối thiểu để chứng minh sản phẩm được thiết kế có trách nhiệm. Hồ sơ này có thể được bổ sung dần khi nhóm có thêm dữ liệu, API, kiểm thử, log và phản hồi người dùng.



Hình 1.8. *Bộ hồ sơ tối thiểu đi cùng sản phẩm AI trong bài tập lớn*

Dùng AI như cộng sự có kiểm soát.

Sau khi đã hiểu quy trình thủ công, người học có thể dùng AI để tăng tốc phân tích. AI có thể gợi ý câu hỏi làm rõ, so sánh phương án kiến trúc, sinh bản nháp product brief, kiểm tra thiếu sót trong hồ sơ rủi ro hoặc chuyển mô tả tự nhiên thành JSON/API schema ban đầu. Tuy nhiên, con người vẫn quyết định mục tiêu, cung cấp bối cảnh, chọn dữ liệu, kiểm chứng kết quả và chịu trách nhiệm cuối cùng.



Hình 1.9. *Vòng lặp cộng tác người-AI khi lập hồ sơ sản phẩm AI*

Thực hành

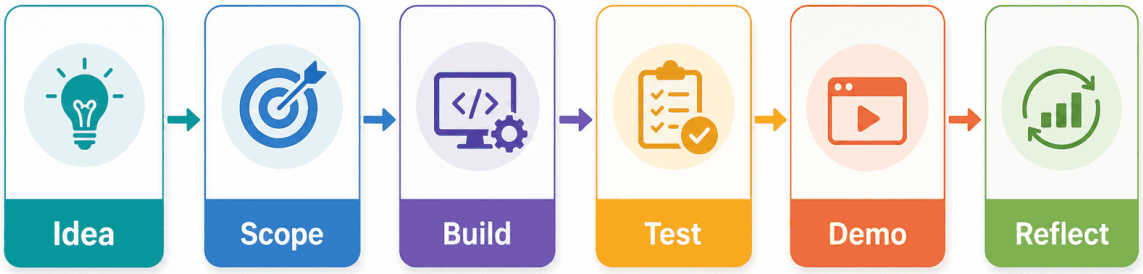
Nhiệm vụ: Lập AI product brief một trang cho bài tập lớn.
Sản phẩm: Một trang gồm bảy mục trong Bảng 1.9. Nhóm được dùng AI để rà soát và cải tiến brief, nhưng phải nộp kèm ghi chú ngắn: AI đã gợi ý gì, nhóm chấp nhận gì, nhóm loại bỏ gì và vì sao.

1.3.3. Thực hành tổng hợp

Mục tiêu của buổi thực hành.

Phần thực hành giúp nhóm chuyển kiến thức tổng quan thành một bộ hồ sơ nhỏ nhưng dùng được. Nhóm chưa cần có mô hình chạy thật, nhưng phải trả lời được: sản phẩm định làm là gì, AI nằm ở đâu, dữ liệu đến từ đâu, kết quả được đánh giá bằng gì và rủi ro chính là gì.

Một sai lầm phổ biến của người mới học AI là bắt đầu bằng tên mô hình: "nhóm em dùng GPT", "nhóm em dùng YOLO", "nhóm em dùng BERT". Cách bắt đầu này dễ làm sản phẩm lệch khỏi nhu cầu thật. Trong thực hành này, nhóm bắt đầu từ người dùng và workflow, sau đó mới chọn kiểu AI phù hợp. Nếu cuối buổi nhóm quyết định không dùng AI cho một phần nào đó vì rule-based đủ tốt, đó vẫn là một quyết định kỹ nghệ hợp lý.



Hình 1.10. *Luồng thực hành tổng hợp lập AI product brief*

Quy trình thực hành gợi ý.

Buổi thực hành có thể tổ chức trong 90-120 phút. Nếu học trên lớp, giảng viên có thể cho nhóm làm nhanh trên giấy hoặc tài liệu chia sẻ. Nếu học trực tuyến, nhóm có thể dùng Google Docs, Markdown hoặc công cụ quản lý dự án. Công cụ không quan trọng bằng chất lượng lập luận.

Bảng 1.10. *Kịch bản thực hành tổng hợp lập AI product brief*

Thời lượng	Hoạt động	Kết quả mong đợi
10 phút	Chọn lĩnh vực và người dùng chính.	Một câu mô tả vấn đề và một nhóm người dùng ưu tiên.
15 phút	Mô tả workflow hiện tại khi chưa có AI.	3-5 bước người dùng đang làm thủ công hoặc bằng phần mềm thường.
15 phút	Xác định điểm AI có thể hỗ trợ.	Một nhiệm vụ AI cụ thể: phân loại, gợi ý, tìm kiếm, tóm tắt, sinh nội dung hoặc phát hiện bất thường.
20 phút	Viết AI product brief.	Bản nháp một trang theo mẫu đã học.
20 phút	Vẽ sơ đồ thành phần và luồng dữ liệu.	Sơ đồ có client, backend, AI component, data store và monitoring/logging.
15 phút	Liệt kê metric, rủi ro và human-in-the-loop.	Ít nhất ba metric, năm rủi ro và một điểm con người cần can thiệp.
10 phút	Rà soát chéo giữa các nhóm.	Nhóm khác đặt câu hỏi để phát hiện giả định mơ hồ.

Trong bước rà soát chéo, nhóm không nên chỉ hỏi "có dùng mô hình nào". Các câu hỏi tốt hơn là: người dùng thật sự cần gì, dữ liệu có sẵn không, nếu AI sai thì ai bị ảnh hưởng, metric nào chứng minh sản phẩm hữu ích, và phần nào của hệ thống có thể demo được trong thời gian môn học.

Mẫu câu hỏi rà soát chéo.

Rà soát chéo giúp nhóm nhìn thấy điểm mù. Một nhóm thường hiểu quá rõ ý tưởng của mình nên dễ bỏ qua giả định chưa viết ra. Nhóm khác, vì chưa quen bối cảnh, sẽ đặt câu hỏi rất gần với người dùng hoặc người đánh giá cuối cùng. Dưới đây là bộ câu hỏi gợi ý. **Rubric đánh giá product brief.**

Rubric dưới đây giúp sinh viên biết product brief được đánh giá theo tiêu chí nào. Mục tiêu không phải làm tài liệu dài, mà là làm tài liệu rõ. Một product brief ngắn nhưng có người dùng cụ thể, dữ liệu rõ, metric đo được và rủi ro thật sẽ tốt hơn một bản trình bày nhiều trang nhưng chỉ nói chung chung về AI.

Bảng 1.11. *Rubric đánh giá AI product brief*

Tiêu chí	Trọng số	Mô tả mức đạt
Bài toán và người dùng	20%	Mô tả được người dùng chính, nhu cầu, workflow hiện tại và lý do AI có thể tạo giá trị.
Nhiệm vụ AI và dữ liệu	20%	Xác định rõ AI làm gì, đầu vào/đầu ra là gì, dữ liệu đến từ đâu và dữ liệu nào không được dùng.
Kiến trúc hệ thống	20%	Sơ đồ có các thành phần phần mềm cơ bản, luồng dữ liệu hợp lý và vị trí của AI trong hệ thống.
Metric và kiểm thử ban đầu	15%	Có metric mô hình, sản phẩm hoặc vận hành phù hợp; biết cách kiểm tra chất lượng ở mức nhập môn.
Rủi ro và trách nhiệm	15%	Nhận diện privacy, security, bias, hallucination, drift hoặc lỗi vận hành; có fallback/human-in-the-loop.
Trình bày và phản biện	10%	Tài liệu ngắn gọn, nhất quán, có ghi giả định; trả lời được câu hỏi rà soát chéo.

Lỗi thường gặp khi lập kế hoạch sản phẩm AI.

Các lỗi dưới đây xuất hiện rất thường xuyên trong những dự án AI nhập môn. Nhận diện sớm các lỗi này giúp nhóm làm rõ sản phẩm ngay từ bản brief đầu tiên.

Bảng 1.12. *Lỗi thường gặp khi lập kế hoạch sản phẩm AI*

Lỗi	Dấu hiệu	Cách sửa
Bắt đầu từ mô hình	Tài liệu mở đầu bằng tên model nhưng không nói rõ người dùng và workflow.	Viết lại problem statement, user story và điểm AI hỗ trợ trước khi chọn model.
Nhiệm vụ AI quá rộng	"Trợ lý học tập thông minh" nhưng không rõ trợ lý làm việc gì.	Thu hẹp thành một tác vụ: hỏi đáp tài liệu, gợi ý tài liệu, phân loại phản hồi hoặc tóm tắt nội dung.
Dữ liệu mơ hồ	Nói "dùng dữ liệu trên mạng" hoặc "dữ liệu sinh viên" mà không nêu nguồn và quyền sử dụng.	Liệt kê nguồn cụ thể, định dạng, quyền dùng, dữ liệu nhạy cảm và dữ liệu loại trừ.
Metric không đo được	Dùng các từ "tốt", "thông minh", "chính xác cao" nhưng không có cách kiểm tra.	Chọn metric quan sát được trong demo: tỷ lệ có nguồn, thời gian phản hồi, số lỗi trên bộ test mẫu.
Không có fallback	Hệ thống luôn trả lời kể cả khi không đủ dữ liệu hoặc confidence thấp.	Thiết kế từ chối có căn cứ, hỏi lại, chuyển người phụ trách hoặc trả lời ở mức tham khảo.
Quên vận hành	Demo chỉ có model notebook, không có API, log, hướng dẫn chạy hoặc kiểm soát phiên bản.	Đưa sản phẩm vào kiến trúc phần mềm tối thiểu, có log, hướng dẫn chạy và mô tả cách kiểm soát phiên bản.

Thực hành

Nhiệm vụ: Hoàn thiện gói nộp AI product brief.
Sản phẩm: Một file PDF hoặc Markdown gồm bốn phần: AI product brief một trang, sơ đồ thành phần, bảng metric/rủi ro, và ghi chú rà soát chéo. Tên file theo mẫu `nhom-XX-chuong-01-product-brief`.

Phiếu tự đánh giá product brief.

Trước khi chốt bản brief, nhóm nên tự đánh giá mức sẵn sàng của ý tưởng. Phiếu này không thay thế rubric chấm điểm, nhưng giúp nhóm phát hiện sớm điểm yếu. Nếu nhiều mục còn ở mức "chưa rõ", nhóm nên quay lại thảo luận bài toán thay vì vội chọn mô hình. Nếu một mục trong phiếu tự đánh giá chưa đạt, nhóm nên viết lại thành một câu hỏi cụ thể. Ví dụ, thay vì ghi "chưa rõ dữ liệu", hãy ghi "dữ liệu phản hồi sinh viên lấy từ nguồn nào, có được phép dùng không, và có cần ẩn thông tin cá nhân không". Câu hỏi cụ thể giúp nhóm tiến lên; nhận xét chung chung chỉ tạo cảm giác mơ hồ.

Gợi ý đề tài bài tập lớn.

Để sinh viên dễ chuyển từ khái niệm sang sản phẩm, Bảng 1.13 liệt kê một số đề tài phù hợp với phạm vi môn học. Các đề tài này được chọn theo tiêu chí: có người dùng rõ, có thể triển khai bằng web/mobile hoặc API, có dữ liệu ở quy mô vừa phải, có thể đánh giá ở mức nhập môn, và đủ chỗ để thảo luận rủi ro. Nhóm có thể chọn trực tiếp một đề tài, điều chỉnh cho bối cảnh của mình, hoặc dùng bảng như nguồn gợi ý để tạo đề tài khác.

Bảng 1.13. *Gợi ý đề tài bài tập lớn*

Đề tài	Nhiệm vụ AI	Gợi ý metric/rủi ro
Trợ lý hỏi đáp tài liệu môn học	RAG, tìm kiếm ngữ nghĩa, sinh câu trả lời có trích dẫn.	Groundedness, tỷ lệ có nguồn, latency; rủi ro hallucination và tài liệu lỗi thời.
Phân loại phản hồi sinh viên	Phân loại văn bản theo chủ đề, mức độ khẩn cấp hoặc cảm xúc.	F1 theo nhãn, tỷ lệ chuyển đúng người phụ trách; rủi ro bias và thông tin cá nhân.
Gợi ý tài liệu học tập	Gợi ý bài đọc hoặc video theo chủ đề và mức độ người học.	Click-through, mức hữu ích do người dùng đánh giá; rủi ro gợi ý hẹp hoặc nguồn kém tin cậy.
Tóm tắt biên bản họp nhóm	Tóm tắt văn bản, trích xuất quyết định và việc cần làm.	Tỷ lệ việc cần làm đúng, số chỉnh sửa của người dùng; rủi ro bỏ sót quyết định quan trọng.
Kiểm tra chất lượng báo cáo	Phát hiện thiếu mục, lỗi trích dẫn, lỗi định dạng hoặc đoạn văn quá mơ hồ.	Tỷ lệ phát hiện lỗi trên bộ mẫu; rủi ro đánh giá sai hoặc làm sinh viên phụ thuộc.
Nhận diện ảnh thiết bị phòng lab	Phân loại ảnh hoặc phát hiện đối tượng trong ảnh.	Accuracy, confusion matrix, latency trên thiết bị; rủi ro ánh sáng, góc chụp và dữ liệu ít.
Dự báo nguy cơ trễ hạn bài tập	Dự đoán từ log tiến độ, số commit, rà soát hoặc lịch làm việc.	Recall với nhóm nguy cơ cao; rủi ro gán nhãn nhạy cảm và tạo áp lực không cần thiết.
Trợ lý tạo câu hỏi ôn tập	Sinh câu hỏi từ tài liệu và gợi ý đáp án tham khảo.	Tỷ lệ câu hỏi đúng trọng tâm, độ đa dạng; rủi ro câu hỏi sai kiến thức hoặc quá dễ.

Khi chọn đề tài, nhóm nên ưu tiên bài toán có thể demo được trong vài tuần. Một đề tài nhỏ nhưng có API, dữ liệu mẫu, kiểm thử và báo cáo rủi ro sẽ có giá trị học tập cao hơn một ý tưởng lớn nhưng chỉ dừng ở mô hình thử nghiệm. Đề tài cũng nên có phiên bản tối thiểu: nếu không hoàn thành toàn bộ tính năng, nhóm vẫn có thể trình bày một

lát cắt chạy được từ input đến output.

Ba mức tham vọng cho cùng một đề tài.

Một đề tài AI thường có nhiều mức tham vọng. Việc chia mức giúp nhóm không bị kẹt giữa kỳ vì mục tiêu quá lớn. Ví dụ với trợ lý hỏi đáp tài liệu môn học, mức cơ bản có thể chỉ cần upload một số tài liệu, tìm đoạn liên quan và trả lời có trích dẫn. Mức khá có thêm giao diện web, lịch sử câu hỏi, đánh giá câu trả lời và fallback khi thiếu nguồn. Mức nâng cao có thêm quản lý phiên bản tài liệu, dashboard monitoring, kiểm thử prompt injection và quy trình giảng viên duyệt FAQ.

Bảng 1.14. *Ba mức tham vọng cho đề tài trợ lý RAG hỏi đáp tài liệu*

Mức	Phạm vi nên làm	Dấu hiệu hoàn thành
Cơ bản	Ingest tài liệu mẫu, tách đoạn, tìm kiếm liên quan, gọi mô hình sinh câu trả lời có nguồn.	Demo được một luồng câu hỏi-đáp; có ít nhất 20 câu hỏi kiểm thử thủ công.
Khá	Có giao diện web, API rõ schema, lưu lịch sử, đánh giá câu trả lời và fallback khi thiếu căn cứ.	Người dùng thử được; report có metric groundedness, latency và lỗi thường gặp.
Nâng cao	Có quản lý phiên bản tài liệu, dashboard monitoring, kiểm thử prompt injection và workflow giảng viên duyệt nội dung nhạy cảm.	Có log vận hành, rà soát rủi ro, kế hoạch cập nhật tài liệu và kịch bản xử lý sự cố.

Với mọi đề tài, nhóm nên viết trước mức cơ bản như một cam kết tối thiểu. Mức khá và nâng cao là phần mở rộng nếu còn thời gian. Cách chia này cũng giúp giảng viên đánh giá công bằng hơn: một nhóm hoàn thành chắc mức cơ bản và giải thích rõ trade-off thường tốt hơn một nhóm trình bày nhiều tính năng nhưng không có luồng nào chạy ổn định.

Mẫu bảng metric và rủi ro cho gói nộp.

Ngoài product brief, nhóm nên tạo một bảng metric và rủi ro ngay từ giai đoạn khởi động. Bảng này chưa cần số liệu thật, nhưng phải cho thấy nhóm biết mình sẽ đo gì và sợ điều gì. Khi sản phẩm được phát triển thêm, bảng này sẽ trở thành nơi ghi lại kết

quả kiểm thử, log, phản hồi người dùng và quyết định triển khai.

Bảng 1.15. *Mẫu bảng metric và rủi ro ban đầu*

Hạng mục	Cách ghi ban đầu	Cách cập nhật khi có thêm kết quả
Metric mô hình	Dự kiến dùng F1, accuracy, groundedness hoặc citation correctness tùy tác vụ.	Ghi bộ dữ liệu thử nghiệm, kết quả đo, lỗi nổi bật và nhận xét về giới hạn.
Metric sản phẩm	Dự kiến đo thời gian hoàn thành tác vụ, mức hài lòng hoặc số lần người dùng phải sửa.	Thêm kết quả user testing, phản hồi định tính và thay đổi thiết kế sau thử nghiệm.
Metric vận hành	Dự kiến theo dõi latency, error rate, chi phí mỗi request hoặc tỷ lệ fallback.	Thêm log thực nghiệm, biểu đồ hoặc bảng số liệu khi chạy demo/backend.
Rủi ro dữ liệu	Nêu dữ liệu nhạy cảm, dữ liệu lỗi thời, dữ liệu không đại diện hoặc dữ liệu thiếu quyền dùng.	Thêm cách làm sạch, ẩn danh, loại trừ, cập nhật hoặc thay thế nguồn dữ liệu.
Rủi ro AI	Nêu hallucination, bias, confidence thấp, drift, prompt injection hoặc output độc hại.	Thêm test case, guardrail, refusal policy, human-in-the-loop hoặc cảnh báo UI.
Rủi ro phần mềm	Nêu API chậm, lỗi schema, thiếu log, lỗi phân quyền hoặc phụ thuộc dịch vụ ngoài.	Thêm test tự động, retry, timeout, monitoring và hướng dẫn triển khai.

Một bảng metric và rủi ro tốt không cần quá dài, nhưng phải cụ thể. Thay vì ghi "rủi ro bảo mật", hãy ghi "người dùng nhập thông tin cá nhân vào câu hỏi và hệ thống gửi nguyên văn sang AI API". Thay vì ghi "metric độ chính xác", hãy ghi "trong 50 câu hỏi kiểm thử, ít nhất 80% câu trả lời phải có nguồn đúng". Sự cụ thể làm cho rủi ro có

thể quản lý được.

1.4. Case study từ internet: Kaggle Titanic như một sản phẩm AI nhỏ

Case study đầu tiên nên đủ nhỏ để sinh viên nhìn thấy toàn bộ vòng đời AI Engineering. Bộ thi *Titanic - Machine Learning from Disaster* trên Kaggle có dữ liệu hành khách, nhãn sống/sống sót và bài toán dự đoán nhị phân.[1] Không nên dạy case này như một bài "train model cho vui", mà nên xem nó như một sản phẩm AI tối thiểu: người dùng cần dự đoán, dữ liệu có cột thiếu, output là xác suất/nhãn, và báo cáo phải nêu giới hạn đạo đức khi dùng dữ liệu nhân khẩu học.

```
mkdir -p case-studies/titanic/{data,reports,scripts}
cd case-studies/titanic
kaggle competitions download -c titanic -p data
unzip data/titanic.zip -d data
ls -lh data
```

Mã nguồn 1.3. *Lệnh lấy dữ liệu Titanic từ Kaggle và tạo thư mục thực nghiệm*

[1] https://www.kaggle.com/c/titanic

Bảng 1.16. *Cách biến Kaggle Titanic thành case study kỹ nghệ AI*

Góc nhìn	Sinh viên làm gì	Bằng chứng đưa vào báo cáo
Product brief	Viết người dùng giả định, input, output và giới hạn sử dụng.	Một trang brief, không dùng mô hình để ra quyết định thật về con người.
Data note	Mở `train.csv`, thống kê cột thiếu, kiểu dữ liệu, nhãn.	Bảng số dòng, số cột, tỷ lệ thiếu của `Age`, `Cabin`, `Embarked`.
Baseline	Train logistic regression hoặc random forest đơn giản.	Accuracy validation, confusion matrix, lỗi điển hình.
Rủi ro	Phân tích việc dùng `Sex`, `Age`, `Pclass`.	Đoạn nhận xét về bias, dữ liệu lịch sử và giới hạn diễn giải.
Vận hành	Đóng gói script chạy lại từ dữ liệu thô.	Lệnh `python scripts/train.py` tạo `reports/metrics.json`.

Kết quả mong đợi của chương này không phải đứng hạng cao trên Kaggle. Kết quả đúng là sinh viên chứng minh được một vòng lặp hoàn chỉnh: lấy dữ liệu từ nguồn thật, mô tả bài toán, tạo baseline, ghi metric, nêu rủi ro và chỉ ra vì sao đây mới là demo học tập chứ chưa phải hệ thống dùng trong đời thực. Hệ thống thông minh cần được xem như một sản phẩm phần mềm có thành phần AI. Người học cần phân biệt mô hình AI với hệ thống AI hoàn chỉnh, vì sản phẩm thực tế còn có giao diện, API, dữ liệu, kiểm thử, triển khai, giám sát và trách nhiệm vận hành.

Các thuật ngữ AI, ML, DL, GenAI, hệ thống dựa trên luật và AI-enabled software cần được hiểu trong bối cảnh sản phẩm. Kiến trúc khái quát, vòng đời AI Engineering và các khác biệt kỹ nghệ như đầu ra xác suất, phụ thuộc dữ liệu, drift và metric đa lớp là nền tảng để nhóm lập kế hoạch sản phẩm có trách nhiệm.

Sản phẩm học tập quan trọng nhất là AI product brief ban đầu cho bài tập lớn. Brief này giúp nhóm thống nhất bài toán, người dùng, dữ liệu, thành phần AI, metric và rủi ro trước khi đi vào triển khai chi tiết.

Câu hỏi ôn tập

1. Vì sao không nên đồng nhất mô hình AI với toàn bộ hệ thống AI?
2. Hãy phân biệt AI, ML, DL, GenAI và hệ thống dựa trên luật bằng ví dụ cụ thể.
3. Khi nào một giải pháp rule-based có thể phù hợp hơn giải pháp ML hoặc GenAI?
4. Hãy nêu bảy thành phần thường có trong một ứng dụng phần mềm có thành phần AI.
5. Phần mềm truyền thống và phần mềm có thành phần AI khác nhau như thế nào về yêu cầu, kiểm thử và vận hành?
6. Vì sao kết quả AI nên được trả về dưới dạng có cấu trúc trong API?
7. Hãy phân biệt metric mô hình, metric sản phẩm và metric vận hành.
8. Vòng đời AI Engineering gồm những bước nào? Vì sao vòng đời này có tính lặp?
9. Trong case study trợ lý RAG hỏi đáp tài liệu môn học, đâu là dữ liệu đầu vào, đâu là kết quả đầu ra và đâu là rủi ro chính?
10. AI product brief tối thiểu cần có những mục nào?
11. Human-in-the-loop có vai trò gì trong các hệ thống AI có rủi ro?
12. AI có thể hỗ trợ lập product brief ở những bước nào, và vì sao nhóm vẫn phải kiểm chứng kết quả?