TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT
KHOA XÂY DỰNG

ỨNG DỤNG TRÍ TUỆ NHÂN TẠO TRONG THIẾT KẾ VÀ PHÁT TRIỂN BÀI GIẢNG ĐIỆN TỬ

Kinh nghiệm thực tiễn trong khối ngành kỹ thuật

TÀI LIỆU HỘI THẢO CHUYÊN MÔN

Báo cáo viên: **ThS. Nguyễn Phú Hoàng**

Thời gian: 15h00, thứ Bảy ngày 10/10/2026

Địa điểm: Phòng Bộ môn Xây dựng

Nội dung tài liệu phục vụ trao đổi chuyên môn nội bộ.
Mọi ví dụ số liệu mang tính minh họa phương pháp, không dùng để thiết kế thực tế.

Mục lục

Bắt đầu tại đây **3**

Lời nói đầu **4**

1 Tổng quan: Vì sao phải có phương pháp **5**
1.1 Bốn loại sai có quy luật . . . 5
1.2 Vì sao "dặn AI cẩn thận hơn" là vô ích . . . 6
1.3 Kiến trúc bốn bước . . . 6

2 Bước 1: Định hình khung và bối cảnh **7**
2.1 Nguyên lý: bối cảnh là hợp đồng, không phải lời dặn . . . 7
2.2 Bốn thành phần của Hồ sơ Bối cảnh . . . 8
2.3 Template Hồ sơ Bối cảnh dùng chung . . . 10
2.4 Ba mức lưu bối cảnh . . . 11

3 Bước 2: Neo dữ liệu chuẩn **12**
3.1 Nguyên lý: phân biệt ba việc khác nhau . . . 12
3.2 Ba hạng chất lượng của kho nguồn . . . 12
3.3 Xử lý tài liệu scan — vấn đề đặc thù . . . 12
3.4 Bốn mệnh lệnh cốt lõi của prompt neo dữ liệu . . . 13
3.5 Vì sao phải tách kho nguồn theo học phần . . . 13

4 Hai công cụ cụ thể: NotebookLM và Gemini Pro **15**
4.1 Bảng đối chiếu nhanh . . . 15
4.2 Ví dụ 1 — Dùng NotebookLM để chống bịa số liệu . . . 15
4.3 Ví dụ 2 — Dùng NotebookLM để kiểm chéo kết quả Gemini . . . 16
4.4 Ví dụ 3 — Dùng Gemini để viết, cách ra lệnh . . . 17
4.5 Ví dụ 4 — Dùng Gemini để phản biện chính mình . . . 17
4.6 Ví dụ 5 — Cảnh báo về việc dùng AI thay người . . . 18
4.7 Cảnh báo về tính ổn định của thông tin công cụ . . . 18

5 Bước 3: Phát triển nội dung **19**
5.1 Nguyên lý: chia nhỏ là bắt buộc . . . 19
5.2 Sinh lý thuyết theo ba lớp . . . 19
5.3 Ví dụ cụ thể: bài toán sức chịu tải cọc . . . 19
5.4 Sinh bài tập có tham số hóa . . . 21
5.5 Cần có gì trước khi bắt đầu . . . 25
5.6 Sinh slide theo nguyên tắc "Slide as Code" . . . 26
5.7 Sinh sơ đồ: chọn công cụ theo mục đích . . . 26

6 Bước 4: Vòng lặp kiểm duyệt và phản biện **27**
6.1 Nguyên lý: người viết không thể là người kiểm . . . 27
6.2 Kiến trúc ba tầng kiểm duyệt . . . 27
6.3 Tầng 1 — Kiểm tự động . . . 27
6.4 Tầng 2 — Prompt phản biện . . . 29
6.5 Tầng 3 — Thẩm định chuyên môn . . . 29
6.6 Danh sách kiểm tra nghiệm thu . . . 30

7 Kỹ thuật nâng cao **31**
7.1 "Rượu mới bình cũ" — cập nhật quy định mới, giữ nguyên văn phong . . . 31
7.2 Neo vào phong cách của chính người dạy . . . 32
7.3 Tạo ngân hàng câu hỏi từ tài liệu nguồn . . . 32
7.4 Tạo kịch bản tương tác trên lớp . . . 33
7.5 Quyền sử dụng tài liệu và dữ liệu . . . 34
7.6 Khi lỗi vẫn lọt ra: quy trình xử lý sau phát hành . . . 34

8 Mười sáu cái bẫy cần tránh **36**

9 Lộ trình áp dụng **37**
9.1 Lộ trình khởi động cho người bắt đầu từ số không . . . 37
9.2 Bốn giai đoạn triển khai . . . 38
9.3 Thứ tự ưu tiên khi nguồn lực có hạn . . . 38
9.4 Dấu hiệu quy trình đang có vấn đề . . . 39

10 Ghi chú về độ tin cậy của tài liệu **40**
10.1 Những khẳng định có căn cứ cơ chế . . . 40
10.2 Những nội dung cần tự kiểm chứng . . . 40
10.3 Ranh giới của tài liệu . . . 41

A Phụ lục A — Template Hồ sơ Bối cảnh (bản đầy đủ) **42**

B Phụ lục B — Prompt phản biện (bản đầy đủ) **43**

C Phụ lục C — Danh sách kiểm tra nghiệm thu **44**

Danh sách bảng

2 Chuyển đổi từ "lời dặn" sang "điều khoản hợp đồng". . . . 7
3 Mức năng lực và cách kiểm chứng tương ứng. . . . 9
4 So sánh ba cách đưa tài liệu vào AI. . . . 12
5 Phân vai giữa NotebookLM và Gemini Pro trong từng bước quy trình. . . . 15
6 Cách hỏi tốt và cách hỏi tồi trong NotebookLM. . . . 16
7 Ba lớp nội dung lý thuyết và cách kiểm chứng. . . . 19
8 Chọn công cụ vẽ sơ đồ theo loại sơ đồ. . . . 26

Danh sách hình vẽ

1 Kiến trúc bốn bước của quy trình soạn giảng có kiểm soát. . . . 7
2 Bốn thành phần cốt lõi của Hồ sơ Bối cảnh. . . . 8
3 Chuỗi xử lý tài liệu scan. Bước đối chiếu là bước không thể bỏ. . . . 13
4 Nguyên lý sinh bài tập có tham số hóa. . . . 22
5 Kiến trúc ba tầng kiểm duyệt. . . . 27
6 Bốn bước của kỹ thuật "Rượu mới bình cũ". . . . 31

Bắt đầu tại đây

Tài liệu này dày, và có nhiều cách đọc khác nhau tùy mục đích. Để không phải tự đoán nên bắt đầu từ đâu, đây là ba con đường — **chọn một**, đừng đọc trộn lẫn:

Nếu bạn…	Đọc theo đường	Vì sao
Chỉ có 15 phút	Mục 1.1 (bốn loại sai) → Chương 6.2 (ba tầng kiểm duyệt)	Hai chỗ này chứa toàn bộ luận điểm cốt lõi; phần còn lại là triển khai.
Muốn bắt tay làm ngay	Lộ trình ở mục 9 (hai buổi, một chương)	Đi thẳng vào việc, không đọc lý thuyết trước. Quay lại đọc lý thuyết khi gặp chỗ không hiểu.
Muốn hiểu đầy đủ	Đọc tuần tự Chương 1 → 7	Đây là đường dành cho người sẽ dạy lại phương pháp này cho người khác, hoặc sẽ tự chỉnh quy trình.

Hai lưu ý chung cho cả ba đường.

- Nếu tài liệu nguồn của bạn *chưa có dạng văn bản sạch* (còn là bản scan chưa OCR, hoặc OCR chưa đối chiếu), hãy làm bước xử lý OCR trước — xem mục 3.3. Lộ trình hai buổi ở trên giả định nguồn đã sạch.
- Nếu bạn chưa quen dùng Python hoặc chưa từng viết script, đọc trước khung điều kiện tiên quyết ở mục 5.5 — trong đó có phương án thay thế không cần biết code.

Lời nói đầu

Tài liệu này ra đời từ một quan sát đơn giản: trong khối ngành kỹ thuật, việc dùng AI để soạn giáo trình và bài giảng điện tử **không** thất bại vì AI viết dở. Nó thất bại vì người dùng tin vào đầu ra của AI ở những chỗ không nên tin.

Trí tuệ nhân tạo hiện nay viết văn trôi chảy, trình bày công thức đúng cú pháp, và dẫn chiếu tài liệu có vẻ rất chuyên nghiệp. Chính sự trôi chảy đó tạo ra một hiệu ứng tâm lý nguy hiểm: người đọc hạ thấp mức độ cảnh giác. Trong khi đó, bốn loại lỗi nghiêm trọng nhất của AI trong tài liệu kỹ thuật — sai loại hình (trộn hai hệ tiêu chuẩn), sai số học trong phép tính nhiều bước, trôi ký hiệu trong văn bản dài, và dùng công thức ngoài miền áp dụng — đều *không* làm cho văn bản trở nên khó đọc. Chúng chỉ lộ ra khi đối chiếu với nguồn gốc.

Vì vậy, tài liệu này không trình bày AI như một công cụ tự động hóa. Nó trình bày AI như một **thợ viết bản nháp rất nhanh nhưng không đáng tin về sự thật**, và tập trung vào câu hỏi: phải thiết kế quy trình như thế nào để lỗi bị chặn ở tầng cơ chế, chứ không dựa vào việc dặn dò AI hãy cẩn thận.

Cách đọc tài liệu. Nếu thầy/cô chỉ có mười lăm phút, xin đọc Mục 1.1 (bốn loại sai) và Chương 6.2 (kiến trúc kiểm duyệt ba tầng). Hai phần này chứa toàn bộ giá trị cốt lõi; các phần còn lại là chi tiết triển khai. Nếu thầy/cô muốn bắt tay làm ngay, đọc thêm Chương 4 (hai công cụ cụ thể: NotebookLM và Gemini Pro) — phần đó trả lời câu hỏi "mở công cụ nào, dán cái gì".

Giá trị đối với người học. Cần nói rõ ngay từ đầu, vì đây là câu hỏi quan trọng nhất và cũng dễ bị bỏ qua nhất. Quy trình trong tài liệu này **không** tự động nâng chất lượng sư phạm của bài giảng. Nó tạo ra ba giá trị, nhưng phải đọc cho đúng bản chất của chúng:

1. **Nhiều bài tập hơn và đúng hơn.** Vì số liệu do máy sinh nên chấm lại được, và mỗi đề đều có kiểm miền giá trị hợp lệ — sinh viên không còn gặp đề vô lý.
2. **Nội dung cập nhật kịp hơn.** Khi quy định thay đổi, chi phí cập nhật bài giảng thấp hơn nhiều so với sửa tay từng trang.
3. **Nhất quán hơn.** Ký hiệu và quy ước thống nhất trong toàn bộ học phần, giảm nhầm lẫn cho người học khi chuyển giữa các chương.

Cần trung thực về bản chất của ba giá trị này: **cả ba đều là hệ quả của khâu soạn, và cả ba đều chưa được đo.** Chúng là *điều kiện cần*, không phải kết quả đã kiểm chứng. Nói thẳng hơn nữa: giá trị trực tiếp và chắc chắn nhất của quy trình là *giảm công soạn và giảm số lỗi lọt ra*; phần "người học được lợi gì" chỉ thành hiện thực nếu có thêm hai điều kiện — (a) số bài tập tăng đi kèm *phản hồi* cho người học, và (b) nội dung cập nhật thật sự chạm tới chỗ người học cần. Nếu thiếu (a) và (b), quy trình chỉ làm giảng viên nhàn hơn, chứ không làm người học khá hơn.

Vì vậy, khi triển khai (Giai đoạn 2 ở mục 9), bắt buộc phải đặt ít nhất một chỉ tiêu *phía người học* để kiểm chứng — nếu không, ta không có cách nào biết mình đang thật sự cải tiến hay chỉ đang tự thấy yên tâm.

Phần còn lại — cách đặt vấn đề, chọn ví dụ để sinh viên bám vào, thiết kế hoạt động trên lớp — vẫn hoàn toàn là việc của người dạy, và không có quy trình nào trong tài liệu này làm thay được.

Phạm vi. Mọi ví dụ số liệu trong tài liệu dùng chung một mạch địa kỹ thuật — sức chịu tải cọc và lún móng nông — để minh họa phương pháp. Các con số được chọn để làm ví dụ, không tra cứu từ tiêu chuẩn hiện hành và không dùng cho thiết kế thực tế.

1. Tổng quan: Vì sao phải có phương pháp

1.1 Bốn loại sai có quy luật

Khi để AI soạn tài liệu kỹ thuật tự do, sai sót không xảy ra ngẫu nhiên. Chúng rơi vào bốn nhóm có quy luật rõ ràng, và mỗi nhóm đòi một cách chặn khác nhau.

1. **Sai loại hình (category error).** AI hoán vị giữa những hệ thống đang tồn tại song song. Ví dụ: lấy hệ số an toàn của một *hệ tiêu chuẩn này* nhưng lại dẫn chiếu sang một *hệ tiêu chuẩn khác*; hoặc trộn quy ước ký hiệu của hai trường phái thiết kế khác nhau trong cùng một công thức. Trường hợp hay gặp nhất trong xây dựng: trộn hệ số của *phương pháp ứng suất cho phép* với hệ số của *phương pháp trạng thái giới hạn* — hai phương pháp tính song song tồn tại, và hệ số của chúng *không* dùng thay cho nhau được.

   Đặc điểm nhận biết: văn bản đọc *rất trôi chảy*, chỉ sai khi đối chiếu với nguồn gốc. Đây là loại nguy hiểm nhất vì **không phép kiểm tự động nào bắt được** — muốn phát hiện phải mở nguồn ra đối chiếu, và phải biết nguồn nào mới là nguồn đúng.

2. **Sai số học (arithmetic drift).** Mô hình ngôn ngữ không cộng trừ trên con số theo cách máy tính làm. Nó dự đoán token tiếp theo trôi chảy nhất. Với phép tính nhiều bước, sai số tích lũy và thường xuất hiện ở *bước trung gian* — nơi khó bị để ý nhất — chứ không ở bước đầu hay bước cuối.

3. **Sai trôi ký hiệu (notation drift).** Trong văn bản dài, AI dần dần đổi cách viết ký hiệu: đổi chỉ số trên thành chỉ số dưới, đổi chữ cái biểu diễn cùng một đại lượng, đổi đơn vị. Mỗi lần thay đổi đều nhỏ và hợp lý tại chỗ, nên rất khó bắt bằng mắt.

4. **Đúng công thức nhưng sai điều kiện áp dụng.** Công thức được chép nguyên vẹn, ký hiệu đúng, đơn vị đúng, số học đúng — nhưng nó được dùng cho một tình huống nằm ngoài miền mà công thức đó được xây dựng. Ví dụ: dùng công thức sức chịu tải lập cho cọc đóng để tính cọc khoan nhồi; hoặc áp dụng công thức lún cho nền đồng nhất khi nền thực tế nhiều lớp rất khác nhau.

   Đặc điểm nhận biết: **mọi phép kiểm tự động đều cho qua**. Kiểm thứ nguyên qua, kiểm ký hiệu qua, máy tính lại số qua. Chỉ có hiểu biết chuyên môn về *điều kiện hình thành* của công thức mới phát hiện được. Đây là loại khó nhất trong bốn loại, vì khác với loại 1, văn bản *không có gì bất thường để mà nghi ngờ*.

Vì sao loại sai thứ tư cần được nêu riêng. Ba loại sai đầu là các lỗi *nội tại*: muốn thấy chúng, chỉ cần đọc kỹ văn bản và đối chiếu với quy tắc đã đặt ra. Loại sai thứ tư không nằm ở văn bản, mà nằm ở *quan hệ giữa văn bản và thực tế sử dụng* — nó chỉ lộ ra khi biết công thức đó được xây dựng cho tình huống nào. Đó là lý do vì sao mỗi công thức, ngoài "công thức" và "đơn vị", bắt buộc phải có thêm hai dòng: **điều kiện áp dụng** và **dấu hiệu dùng sai**.

Cách phân loại dễ gây nhầm — cần nói rõ. Trực giác ban đầu thường xếp cả ba loại sai đầu (loại 1, 2, 3) vào nhóm "kiểm được bằng đối chiếu cơ học" và giao hết cho tầng 1 (kiểm tự động). Cách xếp đó **sai**, và cần tách lại cho đúng:

- **Loại 2 (sai số học)** và **loại 3 (trôi ký hiệu)** mới là loại *kiểm được bằng máy*: máy chạy lại số, máy đối chiếu bảng ký hiệu. Tầng 1 chặn được *chỉ hai* loại này.
- **Loại 1 (sai loại hình)** *không* kiểm được bằng máy — dù nó nằm trong nhóm "ba loại đầu" dễ bị tưởng nhầm là kiểm được. Muốn biết một hệ số có bị lấy sai hệ tiêu chuẩn hay không, phải *đối chiếu với nguồn* — và việc phán đoán nguồn nào đúng là việc của con người. Tầng 2 có thể *nêu nghi vấn*; tầng 3 *quyết định*.
- **Loại 4 (sai miền áp dụng)** cũng không kiểm được bằng máy, và khó hơn loại 1 ở chỗ nó *trông*

hoàn toàn hợp lệ. Tầng 3 là tầng duy nhất bắt được nó một cách chắc chắn.

Nói gọn: **tầng 1 chặn loại 2 và 3; tầng 2 nêu nghi vấn cho cả bốn loại nhưng không phán quyết; tầng 3 quyết định loại 1 và 4.** Không tầng nào một mình chặn được cả bốn loại — đó chính là lý do phải có ba tầng.

1.2 Vì sao "dặn AI cẩn thận hơn" là vô ích

Điều quan trọng cần hiểu: cả bốn loại lỗi trên **không** phải do AI lười biếng hay thiếu hiểu biết. Chúng là *hệ quả tất yếu của kiến trúc*: mô hình sinh văn bản theo xác suất, nên nó tối ưu cho tính trôi chảy, không tối ưu cho tính đúng.

Điều này dẫn tới một kết luận có tính kiến trúc cho toàn bộ tài liệu:

Nguyên lý thiết kế. Không thể nâng chất lượng đầu ra bằng cách yêu cầu AI chính xác hơn. Chỉ có thể nâng chất lượng bằng cách **thiết kế quy trình** sao cho mỗi loại lỗi bị chặn bởi một cơ chế cụ thể — cơ chế đó phải hoạt động độc lập với thiện chí của mô hình.

Toàn bộ phần còn lại của tài liệu là triển khai nguyên lý này thành bốn bước công việc.

1.3 Kiến trúc bốn bước

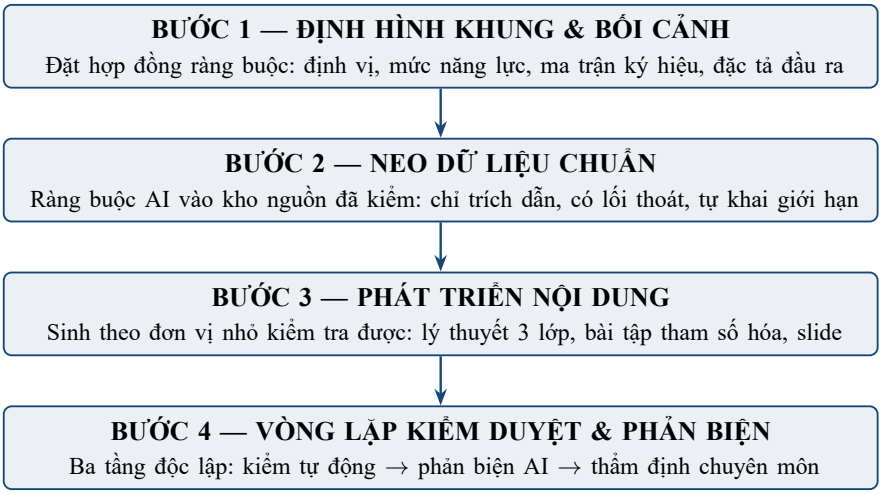
Bảng ánh xạ Bước – Chương – Thuật ngữ (để không bị lẫn). Tài liệu dùng nhiều từ dễ trùng nghĩa; bảng dưới đây cố định nghĩa của từng từ.

Bước	Chương tương ứng	Ghi chú
Bước 1 — Định hình khung & bối cảnh	Chương 2	Làm một lần cho mỗi học phần
Bước 2 — Neo dữ liệu chuẩn	Chương 3	Làm một lần, cập nhật khi quy định đổi
(Chương 4 — Hai công cụ)	Chương 4	Không phải một bước; là *công cụ* dùng cho cả bốn bước
Bước 3 — Phát triển nội dung	Chương 5	Chạy lặp cho từng mục nội dung
Bước 4 — Vòng lặp kiểm duyệt	Chương 6	Chạy lặp, gắn với từng mục
Kỹ thuật nâng cao	Chương 7	Không thuộc bốn bước; dùng khi đã quen

Quy ước dùng từ (tránh lẫn ba thứ vốn đều gọi là "tầng"):

- **Tầng kiểm duyệt** — tầng 1, 2, 3 (kiểm tự động / AI phản biện / người thẩm định).
- **Hạng nguồn** — ba hạng chất lượng của kho nguồn (nguồn gốc / diễn giải / ma trận ký hiệu). Trước đây gọi là "tầng kho nguồn"; nay đổi thành "hạng" để không trùng với tầng kiểm duyệt.
- **Mức lưu bối cảnh** — ba nơi bối cảnh được lưu (phiên / dự án / tệp). Trước đây gọi là "tầng kiểm soát bối cảnh"; nay đổi thành "mức lưu".
- **Lớp** — ba lớp nội dung lý thuyết (cơ chế / mô hình / quy định), phân biệt với "tầng".

Điểm cần lưu ý về thứ tự: bốn bước này **không** phải bốn giai đoạn tuần tự làm một lần rồi xong. Bước 1 và Bước 2 là *hạ tầng* — làm một lần cho mỗi học phần, dùng lại cho mọi lần soạn sau. Bước 3 và Bước 4 là *vòng lặp* — chạy lặp lại cho từng mục nội dung cho đến khi đạt.



Hình 1: Kiến trúc bốn bước của quy trình soạn giảng có kiểm soát.

2. Bước 1: Định hình khung và bối cảnh

2.1 Nguyên lý: bối cảnh là hợp đồng, không phải lời dặn

Sai lầm phổ biến nhất là viết một câu lệnh dài, dặn dò đủ điều, rồi hy vọng AI ghi nhớ. Cách làm đó thất bại vì mọi điều khoản đều ở dạng *mong muốn* — tức là dạng không kiểm tra được.

Cách đúng: coi phần bối cảnh là một **hợp đồng ràng buộc**. Mỗi điều khoản phải ở dạng kiểm tra được — nghĩa là, sau khi AI trả lời, ta có thể xác định bằng máy móc hoặc bằng thao tác rõ ràng rằng điều khoản đó được tuân thủ hay bị vi phạm.

Bảng 2: Chuyển đổi từ "lời dặn" sang "điều khoản hợp đồng".

Dạng mong muốn (kém hiệu quả)	Dạng hợp đồng (kiểm tra được)
"Hãy viết cho chính xác"	"Không đưa ra kết quả số nếu không trình bày đủ các bước tính dẫn tới kết quả đó."
"Dùng ký hiệu thống nhất"	"Ký hiệu D_f chỉ được viết đúng một cách trong toàn tài liệu. Nếu không chắc, hỏi lại thay vì tự chọn."
"Đừng bịa số liệu"	"Nếu thông tin không có trong tài liệu nguồn, trả lời đúng cụm từ `KHÔNG CÓ TRONG NGUỒN`."
"Trình bày dễ hiểu"	"Mỗi chủ đề trình bày theo đúng bốn phần: cơ chế, công thức, điều kiện áp dụng, ví dụ."

Chi tiết then chốt. Điều khoản thứ ba — cho AI một câu trả lời hợp lệ để nói "tôi không biết" — là điều khoản quan trọng nhất và hay bị bỏ sót nhất. Nếu không có lối thoát đó, AI buộc phải bịa, vì bịa là cách duy nhất để hoàn thành yêu cầu. *Về mặt cơ chế*, cho nó một lối thoát sẽ bớt được áp lực phải bịa — nhưng mức giảm cụ thể là bao nhiêu thì tài liệu này không đo, và người dùng nên tự kiểm trên chính công việc của mình.

Một giới hạn của chính phương pháp này — cần nói rõ để không hiểu sai. Ở mục 1.1, tài liệu khẳng định "dặn AI cẩn thận hơn là vô ích". Nhưng rõ ràng cả chương này toàn là các *chỉ thị bằng lời*: lối thoát, tự khai giới hạn, bắt buộc nêu điều kiện áp dụng. Vậy có mâu thuẫn không?
Không — nhưng phải phân biệt hai loại chỉ thị:

- **Chỉ thị làm rõ yêu cầu** (hữu ích): chúng nói cho AI biết *cần làm gì*, và biến một mục tiêu mơ hồ

thành một đầu ra kiểm tra được. Ví dụ: "nếu không có trong nguồn, viết KHÔNG CÓ TRONG NGUỒN".

- **Chỉ thị đòi sự cẩn thận** (vô ích): chúng chỉ yêu cầu AI *cố gắng hơn*. Ví dụ: "hãy viết thật chính xác".

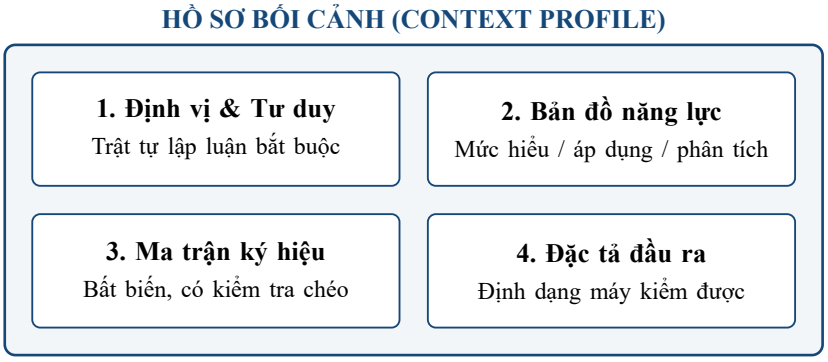
Cái tài liệu phủ nhận là loại thứ hai. Nhưng phải trung thực về một điều: **ngay cả loại thứ nhất cũng không được bảo đảm thi hành** — mô hình vẫn có thể không tuân thủ, và càng ngày càng ít tuân thủ khi cuộc trò chuyện dài ra. Đó chính là lý do phải có tầng 1 và tầng 3: chỉ thị làm rõ *giảm* số lỗi, cơ chế kiểm *chặn* lỗi. Thiếu cơ chế kiểm, mọi chỉ thị — dù viết tốt đến đâu — vẫn chỉ là lời hứa.

2.2 Bốn thành phần của Hồ sơ Bối cảnh

Cách đếm cho thống nhất (tránh nhầm khi đọc). Hồ sơ Bối cảnh có **bốn thành phần cốt lõi**. Trong template ở mục 2.3 và Phụ lục A, bốn thành phần này được viết thành các *mục đánh số* để dễ điền — vì vậy số mục trong template *nhiều hơn* bốn (một số thành phần được tách ra cho rõ). Đừng hiểu "bốn thành phần" và "số mục trong template" là hai thứ phải bằng nhau.
Ánh xạ để tra cứu:

Thành phần	Mục tương ứng trong template
1. Định vị & tư duy	Mục "ĐỊNH VỊ" (và "MỤC TIÊU" — xem ghi chú dưới)
2. Bản đồ năng lực	Mục "MỨC NĂNG LỰC"
3. Ma trận ký hiệu	Mục "MA TRẬN KÝ HIỆU"
4. Đặc tả đầu ra	Mục "QUY TẮC" và "ĐẶC TẢ ĐẦU RA"

Ngoài ra, từ phần đính chính về chuẩn đầu ra (mục 7.3), Hồ sơ Bối cảnh nhận thêm một thứ là *đầu vào bắt buộc* chứ không phải thành phần thứ năm: **chuẩn đầu ra học phần**. Nó không nằm trong hồ sơ, nó là thứ *quyết định* nội dung của hồ sơ.



Hình 2: Bốn thành phần cốt lõi của Hồ sơ Bối cảnh.

Thành phần 1 — Định vị và tư duy

Không gán cho AI một chức danh chung chung kiểu "giáo sư kỹ thuật". Chức danh không tạo ra hành vi. Thay vào đó, gán cho nó **một trật tự lập luận bắt buộc phải đi theo**:

Hiện tượng → Cơ chế → Mô hình → Công thức → Điều kiện áp dụng → Ứng dụng

Mỗi câu trả lời phải đi đủ trật tự này. Giá trị của ràng buộc: nó *buộc AI bộc lộ chỗ nó không chắc*. Nếu AI nhảy thẳng tới "công thức" mà bỏ qua "cơ chế", đó là dấu hiệu nó đang chép mẫu chứ không đang suy luận.

Thành phần 2 — Bản đồ năng lực

Khai báo rõ mỗi mục nội dung ở mức năng lực nào, vì cùng một kiến thức nhưng ở mức khác nhau thì yêu cầu kiểm chứng hoàn toàn khác nhau.

Bảng 3: Mức năng lực và cách kiểm chứng tương ứng.

Mức	Đặc điểm nội dung	Cách kiểm chứng
Hiểu bản chất	Giải thích cơ chế, không có số liệu	Kiểm bằng diễn giải lại. Rủi ro thấp.
Áp dụng	Có công thức, có số liệu, có quy trình tính	**Chạy lại phép tính bằng máy.** Đây là mức AI tự tin nhất và cũng sai nhiều nhất.
Phân tích	So sánh phương án, chọn giải pháp	Đối chiếu nhiều phương án. AI mạnh ở liệt kê, nhưng phải cảnh giác với phương án bịa.

Kỹ thuật thực chiến. Đánh dấu mức năng lực ngay trong bản nháp, ví dụ `[B3]` cho mức áp dụng. Khi chuyển sang bước thẩm định, ta biết chính xác đoạn nào *bắt buộc* phải chạy máy kiểm tra, thay vì chạy tràn lan.

Thành phần 3 — Ma trận ký hiệu bất biến

Đây là công cụ chống trôi ký hiệu hiệu quả nhất. Năm quy tắc:

```
QUY TẮC KÝ HIỆU (bất biến trong toàn bộ tài liệu)
1. Mỗi đại lượng có ĐÚNG MỘT ký hiệu. Không có ngoại lệ "hoặc".
2. Mỗi ký hiệu mang ĐÚNG MỘT nghĩa. Không tái sử dụng cho đại lượng khác.
3. Phân biệt giá trị đặc trưng / tính toán / tiêu chuẩn bằng CHỈ SỐ, không đổi chữ cái.
4. Trước khi viết: in lại ma trận một lần để tự xác nhận.
5. Sau khi viết: liệt kê mọi ký hiệu đã dùng thực tế, đối chiếu với ma trận, báo cáo sai lệch.
```

Hai dòng cuối cùng là phần then chốt. Chúng biến việc kiểm tra ký hiệu từ *đọc dò bằng mắt* — vốn rất dễ bỏ sót — thành *thao tác xuất dữ liệu để so khớp tự động*.

Một xung đột cần giải quyết: ký hiệu của học phần so với ký hiệu của tiêu chuẩn. Quy tắc "mỗi đại lượng một ký hiệu" đúng cho *bài giảng*. Nhưng có hai chỗ nó không áp được, và nếu không xử lý sẽ gây lỗi ngược:

- Khi **trích nguyên văn** một đoạn từ tiêu chuẩn hoặc giáo trình, phải giữ *đúng ký hiệu trong bản gốc* — không được "chuẩn hóa" theo ma trận của học phần. Sửa ký hiệu trong một câu trích là làm sai nguồn.
- Khi sinh viên **ra nghề**, họ sẽ gặp ký hiệu của tiêu chuẩn, vốn có thể khác ký hiệu đã dạy.

Cách xử lý — quy tắc ba dòng:

1. **Nội dung giảng** dùng ký hiệu của học phần (theo ma trận).
2. **Khối trích dẫn** giữ nguyên ký hiệu gốc, và đặt trong dấu trích rõ ràng để máy kiểm không báo lỗi nhầm.

3. **Ma trận ký hiệu** có thêm một cột ghi "ký hiệu tương ứng trong tiêu chuẩn", để học phần chủ động *dạy* sự chuyển đổi thay vì để sinh viên tự phát hiện.

Nếu không có dòng thứ ba, sinh viên gặp ký hiệu lạ trong tài liệu thực tế sẽ không biết nó tương ứng với cái gì đã học — và bộ kiểm tầng 1, nếu không biết quy tắc trích dẫn, sẽ báo lỗi cho những đoạn trích hoàn toàn đúng.

Thành phần 4 — Đặc tả đầu ra

Nguyên tắc: đầu ra phải ở định dạng mà **máy kiểm tra được**.

- Công thức ở dạng văn bản thuần (LaTeX), không phải ảnh. Lý do: còn tìm kiếm được, so khớp được, chạy lại được.
- Bảng thay cho đoạn văn mô tả, ở mọi chỗ có so sánh từ hai đối tượng trở lên.
- Sơ đồ ở dạng mã (Mermaid hoặc TikZ), không phải mô tả bằng lời. Lý do: mã thì sửa được khi số liệu đổi; lời thì phải vẽ lại từ đầu.
- Quy trình ở dạng danh sách có thứ tự, không viết bằng văn xuôi.

2.3 Template Hồ sơ Bối cảnh dùng chung

Template sau dùng được cho mọi học phần kỹ thuật. Phần trong ngoặc vuông là chỗ cần điền.

```
[ĐỊNH VỊ]
Hỗ trợ biên soạn tài liệu giảng dạy cho học phần: [TÊN HỌC PHẦN].
Trật tự lập luận bắt buộc cho mọi chủ đề:
  Hiện tượng - Cơ chế - Mô hình - Công thức - Điều kiện áp dụng - Ứng dụng.
Nếu thiếu căn cứ ở bất kỳ mắt nào, DỪNG LẠI và nêu rõ đang thiếu gì.

[ĐỐI TƯỢNG & MỨC NĂNG LỰC]
Người học: [mô tả nền tảng đã có, KHÔNG mô tả bằng năm học].
Với mỗi mục nội dung, ghi rõ mức: [B2-hiểu] / [B3-áp dụng] / [B4-phân tích].

[CHUẨN ĐẦU RA HỌC PHẦN - ĐẦU VÀO BẮT BUỘC]
Liệt kê các CĐR của học phần, mỗi CĐR một mã: [CĐR1] ... [CĐR2] ...
Mỗi mục nội dung phải gắn với ít nhất một mã CĐR.
Mục nào không gắn được với CĐR nào: nêu rõ để người dùng quyết định.

[QUY TẮC KÝ HIỆU - BẤT BIẾN]
Một ký hiệu = một đại lượng. Một đại lượng = một ký hiệu.
Phân biệt giá trị đặc trưng / tiêu chuẩn / tính toán bằng CHỈ SỐ, không đổi chữ.
Đơn vị: hệ SI. Khi có đơn vị thực hành khác, ghi cả hai.
Định kỳ in lại bảng ký hiệu để tự kiểm tra.

[QUY TẮC TRUNG THỰC - QUAN TRỌNG NHẤT]
- Chỉ dùng thông tin trong tài liệu nguồn được cung cấp.
- Khi dẫn chiếu: nêu rõ vị trí (chương / mục / bảng / trang) trong nguồn.
- Nếu không có trong nguồn: viết đúng cụm KHÔNG CÓ TRONG NGUỒN.
- Chỉ lập BIỂU THỨC, không tự tính ra số. Mọi con số kết quả sẽ do người dùng chạy
script.

[ĐẶC TẢ ĐẦU RA]
- Công thức: LaTeX, mỗi ký hiệu kèm nghĩa và đơn vị.
- Mỗi công thức BẮT BUỘC có hai mục: "Điều kiện áp dụng" và "Dấu hiệu dùng sai".
- Sơ đồ: mã Mermaid hoặc TikZ, không mô tả bằng lời.
- Quy trình: danh sách có thứ tự, không văn xuôi.
```

```
- Cuối tài liệu: bảng ký hiệu thực tế + báo cáo sai lệch so với quy tắc.
```

2.4 Ba mức lưu bối cảnh

Không phải công cụ nào cũng cho phép đặt bối cảnh ở cùng một mức lưu. Phân biệt ba mức để biết mình đang ở đâu:

1. **Mức phiên.** Bối cảnh sống trong một cuộc trò chuyện. Mất khi đóng cửa sổ. Dễ dùng, không bền.
2. **Mức dự án.** Bối cảnh lưu cùng bộ tài liệu, tự nạp lại mỗi lần mở. Bền hơn, phù hợp một học phần.
3. **Mức tệp.** Bối cảnh nằm trong thư mục dự án dưới dạng một tệp văn bản. Bền nhất, và quan trọng hơn cả: *kiểm soát được phiên bản, so sánh được thay đổi, chuyển được giữa các công cụ*.

Khuyến nghị. Luôn giữ nguồn sự thật ở **mức tệp**, rồi dán hoặc đồng bộ lên mức phiên / mức dự án. Lý do thực tế: khi đổi công cụ AI — và chắc chắn sẽ đổi, vì thị trường này thay đổi rất nhanh — tệp vẫn còn nguyên. Giữ nguồn sự thật ở dạng tệp là cách *giảm* phụ thuộc vào một nhà cung cấp; nó không xóa bỏ hoàn toàn sự phụ thuộc đó, vì dù sao bạn vẫn đang dùng công cụ của họ để xử lý.

3. Bước 2: Neo dữ liệu chuẩn

3.1 Nguyên lý: phân biệt ba việc khác nhau

Cần tách bạch ba việc mà người ta hay gọi chung là "cho AI đọc tài liệu". Chúng có cơ chế hoàn toàn khác nhau và hiệu quả chống bịa khác nhau.

Bảng 4: So sánh ba cách đưa tài liệu vào AI.

Cách làm	Cơ chế	Chống bịa được không
Dán văn bản vào ô chat	Văn bản nằm trong bộ nhớ tạm	Chỉ một phần — và mất tác dụng khi bối cảnh tràn.
Nạp tệp vào dự án / notebook	Hệ thống tự truy hồi đoạn liên quan	Tốt — có dẫn chiếu vị trí để kiểm chứng.
Tinh chỉnh mô hình (fine-tune)	Thay đổi trọng số của mô hình	**Không** — vẫn bịa như thường.

Ngộ nhận cần tránh. Nhiều người cho rằng fine-tune là giải pháp chống bịa mạnh nhất, nên đầu tư rất nhiều công sức. Thực tế ngược lại: fine-tune dạy mô hình *văn phong và định dạng*, không dạy *sự thật*. Nó không phải công cụ chống ảo giác. Muốn chống bịa, phải dùng truy hồi có dẫn chiếu.

3.2 Ba hạng chất lượng của kho nguồn

Chất lượng kho nguồn quyết định trần chất lượng đầu ra. Xếp theo mức đóng góp:

Hạng 1 — Văn bản gốc của quy định
(bắt buộc phải có)

Đây là hạng *duy nhất* mà AI được phép trích dẫn. Nếu thiếu hạng này, mọi thứ phía sau trở nên vô nghĩa.

Hạng 2 — Tài liệu diễn giải
(dùng để dạy, cấm trích dẫn)

Giáo trình, bài giảng mẫu, ghi chú. Giá trị: cung cấp *cách diễn đạt* và *trình tự sư phạm*. Nhưng tuyệt đối không cho phép lấy số liệu từ đây, vì tài liệu diễn giải thường đã đơn giản hóa hoặc đã lỗi thời.

Hạng 3 — Ma trận ký hiệu và quy ước
(do người dạy tự viết)

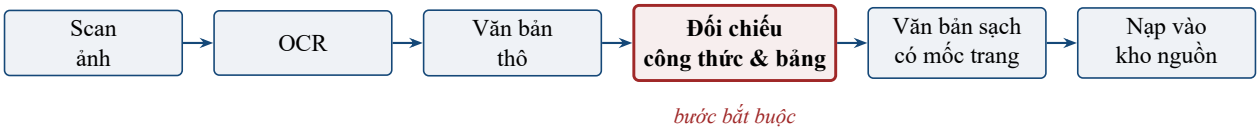
Không phải tài liệu tham khảo, mà là *luật chơi*.

Kỹ thuật. Đặt tên tầng ngay trong câu lệnh: "chỉ trích dẫn từ nhóm NGUỒN GỐC; nhóm DIỄN GIẢI chỉ dùng để tham khảo cách trình bày". Nếu không tách tầng, AI sẽ trộn đều và ta không biết con số nào đến từ đâu.

3.3 Xử lý tài liệu scan — vấn đề đặc thù

Tài liệu quy định ở Việt Nam phần lớn ở dạng scan. Chuỗi xử lý đúng:

Vì sao bước đối chiếu là bắt buộc. OCR xử lý chữ rất tốt nhưng xử lý *công thức và bảng* rất kém. Chỉ số trên/dưới bị nuốt, ký hiệu Hy Lạp bị nhận nhầm thành chữ Latin, dấu căn và phân số bị phá



Hình 3: Chuỗi xử lý tài liệu scan. Bước đối chiếu là bước không thể bỏ.

vỡ cấu trúc.
Nếu nạp thẳng bản OCR vào kho nguồn, AI sẽ đọc được một công thức *trông hợp lệ nhưng đã méo từ gốc* — và dẫn chiếu **đúng số trang**. Người kiểm tra mở đúng trang đó ra, thấy nội dung khớp, bèn tin. Đây là dạng sai nguy hiểm nhất: **sai được bảo chứng bằng dẫn chiếu chính xác.**

Sản phẩm cần tạo sau khi OCR:

1. Tệp văn bản đã đối chiếu, có mốc `[tr.X]` tại mỗi chuyển trang.
2. Tệp đính chính riêng: liệt kê mọi chỗ OCR nghi ngờ kèm giá trị đúng đã kiểm.
3. Với công thức và bảng: chép lại bằng LaTeX / Markdown và đối chiếu thủ công từng dòng.

3.4 Bốn mệnh lệnh cốt lõi của prompt neo dữ liệu

```
1. PHẠM VI
Chỉ trả lời dựa trên tài liệu đã nạp. Không dùng kiến thức nền của mô hình,
kể cả khi bạn chắc chắn về điều đó.

2. DẪN CHIẾU
Mỗi khẳng định kỹ thuật phải kèm vị trí trong nguồn.
Nếu không định vị được vị trí, KHÔNG được khẳng định.

3. LỐI THOÁT
Không có trong nguồn thì trả lời: KHÔNG CÓ TRONG NGUỒN.
Đây là câu trả lời ĐÚNG, không phải thất bại.

4. TỰ KHAI GIỚI HẠN
Sau mỗi câu trả lời, ghi một dòng "Điểm chưa chắc chắn:"
liệt kê những chỗ bạn phải suy đoán.
```

Mệnh lệnh 4 là mệnh lệnh ít người dùng nhất nhưng có giá trị cao nhất: nó buộc AI *tự chỉ ra chỗ yếu*, thay vì để người dùng tự đi tìm. Không phải lúc nào AI cũng khai đủ, nhưng danh sách nó đưa ra luôn là điểm khởi đầu tốt hơn danh sách trống.

3.5 Vì sao phải tách kho nguồn theo học phần

Ba lý do, xếp theo mức quan trọng:

1. **Chống nhiễu bối cảnh.** Trộn nhiều học phần làm tài liệu truy hồi bị pha loãng; tỉ lệ lấy đúng đoạn cần thiết giảm xuống.
2. **Chống lẫn hệ quy ước.** Hai học phần khác nhau có thể dùng ký hiệu trùng tên nhưng khác nghĩa. Trộn vào nhau chính là tự tay tạo ra lỗi loại hình đã nói ở Mục 1.1.
3. **Cô lập lỗi.** Khi phát hiện sai, biết ngay sai xuất phát từ kho nào.

Quy ước đặt tên. Một kho = một học phần + một phiên bản quy định. Ghi rõ ngày tạo trong tên. Khi quy định thay đổi, *tạo kho mới* chứa quy định mới và *giữ nguyên* kho cũ ở trạng thái lưu trữ — không xóa, không trộn vào nhau. Lý do giữ kho cũ: còn cần để dạy lại lớp cũ, đối chiếu khi có khiếu nại, và để hiểu vì sao số liệu năm trước khác năm nay.
Hai kho phải *tách hẳn*. Không nạp quy định mới vào rồi nạp chồng lên kho cũ, vì khi đó mô hình không có cách nào phân biệt điều khoản nào đang còn hiệu lực — và ta lại rơi vào đúng cái bẫy "kết quả lai giữa hai hệ" đã nói ở bẫy 9. Cách làm đúng: notebook cũ đổi tên thành `[TÊN HỌC PHẦN] --- LƯU TRỮ --- [ngày]`, và mọi câu hỏi cho công việc mới chỉ đặt trong notebook mới.

4. Hai công cụ cụ thể: NotebookLM và Gemini Pro

Phần này trả lời câu hỏi thực tế nhất mà giảng viên hay đặt ra sau khi đọc xong bốn bước: *vậy cụ thể thì mở công cụ nào, bấm vào đâu, dán cái gì?* Hai công cụ được chọn vì chúng phổ biến, miễn phí ở mức dùng được, và — quan trọng hơn — chúng đảm nhận hai vai **khác nhau** trong quy trình, không thay thế nhau.

Nguyên tắc phân vai giữa hai công cụ.

- **NotebookLM** = **kho nguồn có dẫn chiếu**. Nó chỉ trả lời dựa trên tài liệu bạn nạp, và mỗi câu trả lời đều kèm dẫn chiếu bấm được tới đúng đoạn gốc. Vai của nó là *chống bịa*.
- **Gemini Pro** = **xưởng viết và xưởng phản biện**. Nó viết được văn dài, dựng được bảng, chạy được biểu thức, và — khi được giao vai — đóng được vai trò người phản biện độc lập. Vai của nó là *sản xuất và kiểm tra*.

Sai lầm phổ biến nhất là dùng Gemini để hỏi sự thật chuyên môn, rồi dùng NotebookLM để viết văn dài. Đó là đảo ngược vai. Kết quả là vừa bịa số, vừa tốn công sửa văn.

4.1 Bảng đối chiếu nhanh

Bảng 5: Phân vai giữa NotebookLM và Gemini Pro trong từng bước quy trình.

Bước	NotebookLM	Gemini Pro
Định hình bối cảnh	Không dùng	Viết và tinh chỉnh Hồ sơ Bối cảnh
Neo dữ liệu	**Vai chính** — nạp nguồn, hỏi có dẫn chiếu	Hỗ trợ làm sạch văn bản OCR
Phát triển nội dung	Tra cứu số liệu gốc để đối chiếu	**Vai chính** — viết lý thuyết, bài tập, slide
Kiểm duyệt tầng 1	Không dùng	Viết script kiểm, chạy biểu thức
Kiểm duyệt tầng 2	Hỏi chéo cùng bộ nguồn, so với kết quả Gemini	**Vai chính** — đóng vai phản biện độc lập
Kiểm duyệt tầng 3	Không dùng	Không dùng — đây là việc của con người

4.2 Ví dụ 1 — Dùng NotebookLM để chống bịa số liệu

Đây là tình huống hay gặp nhất: bạn cần một con số hoặc một quy định cụ thể từ tài liệu gốc, và bạn cần biết chắc nó đến từ đâu.

Các bước thao tác

1. **Tạo một notebook cho đúng một học phần.** Không gộp hai học phần vào một notebook — vì lý do đã nói ở Mục 3.5: ký hiệu trùng tên khác nghĩa sẽ lẫn vào nhau.
2. **Nạp nguồn theo đúng thứ tự ưu tiên.** Nạp tài liệu gốc của quy định trước, tài liệu diễn giải sau. Có thể nạp được tệp PDF, Word, văn bản, Markdown, và cả liên kết web dạng văn bản.
3. **Ngay sau khi nạp, đặt câu hỏi kiểm tra kho.** Đây là bước ít người làm nhưng tiết kiệm rất nhiều thời gian về sau:

```
Liệt kê mọi công thức xuất hiện trong tài liệu đã nạp, kèm số trang.
Với mỗi công thức, nêu đúng nguyên văn ký hiệu và đơn vị như trong bản gốc.
Nếu chỗ nào bạn đọc không chắc (ví dụ ký tự bị lỗi khi nhập),
đánh dấu [KHÔNG CHẮC] thay vì đoán.
```

Câu trả lời sẽ lộ ngay hai loại vấn đề: (a) công thức bị OCR làm méo, và (b) trang nào bị nhập thiếu. Cả hai đều phải sửa *trước khi* bắt đầu soạn.

4. **Đặt câu hỏi hẹp, có phạm vi.** Câu hỏi hẹp cho kết quả tốt hơn câu hỏi rộng:

Bảng 6: Cách hỏi tốt và cách hỏi tồi trong NotebookLM.

Hỏi tồi (nhận câu trả lời tồi)	**Hỏi tốt (nhận câu trả lời kiểm được)**
Tóm tắt tài liệu này.	Công thức tính sức chịu tải cho cọc khoan nhồi nằm ở trang nào? Chép nguyên văn công thức.
Hệ số an toàn lấy bao nhiêu?	Hệ số an toàn xuất hiện ở những trang nào? Liệt kê từng giá trị kèm ngữ cảnh sử dụng.
So sánh hai tiêu chuẩn.	Hai tài liệu đã nạp khác nhau ở điểm nào về cách xác định sức chịu tải? Nêu đúng chỗ khác nhau, kèm số trang cả hai bên.

5. **Đọc dẫn chiếu như một người kiểm tra.** Đây là thao tác quan trọng nhất và cũng là thao tác duy nhất mà NotebookLM có lợi thế lớn: mỗi câu trả lời đều kèm dẫn chiếu bấm được tới đúng đoạn văn gốc. Bấm vào đó, đọc nguyên văn đoạn *xung quanh*, rồi tự hỏi: đoạn văn này có thật sự nói điều AI vừa khẳng định không?

> **Cái bẫy của dẫn chiếu.** Dẫn chiếu đúng *vị trí* không bảo đảm khẳng định đúng *nội dung*. NotebookLM rất ít khi bịa ra vị trí không tồn tại, nhưng nó vẫn có thể *suy diễn quá xa* từ đoạn văn đó — đặc biệt với câu hỏi dạng "vậy suy ra là…". Luôn đọc cả đoạn xung quanh, không chỉ dòng được tô sáng.

6. **Lưu lại câu trả lời đáng tin.** Khi một câu trả lời đã được kiểm và dùng làm cơ sở để soạn bài, lưu nó lại thành ghi chú trong notebook. Lần sau bạn (hoặc đồng nghiệp dạy cùng học phần) sẽ không phải tra lại từ đầu.

4.3 Ví dụ 2 — Dùng NotebookLM để kiểm chéo kết quả Gemini

Đây là một kỹ thuật mà nhiều người dùng một công cụ sẽ không bao giờ nghĩ tới: dùng công cụ thứ hai để *kiểm* công cụ thứ nhất.

1. Gemini viết xong một mục lý thuyết, có dẫn chiếu `[tr.45]`.
2. Bạn mở NotebookLM (đã nạp đúng bộ nguồn đó) và hỏi:

```
Trang 45 của tài liệu đã nạp nói gì?
Chép nguyên văn đoạn liên quan tới [chủ đề vừa viết].
Sau đó cho biết: đoạn văn đó có khẳng định [kết luận của Gemini]
hay không? Nếu chỉ gợi ý mà không khẳng định, nói rõ là ``gợi ý''.
```

3. So hai kết quả.

Vì sao cách này hiệu quả. Hai mô hình "nhớ" khác nhau và "đọc" khác nhau. Nếu cả hai cùng khẳng định một điều dựa trên cùng một đoạn văn, mức tin cậy tăng lên đáng kể. Nếu chúng lệch nhau, đó chính xác là chỗ bạn cần mở bản gốc ra đọc.

4.4 Ví dụ 3 — Dùng Gemini để viết, cách ra lệnh

NotebookLM bị giới hạn: nó trả lời *từ nguồn*, nên nó không viết được văn dài theo ý bạn. Việc đó phải giao cho Gemini.

Quy trình ba bước: khoá nguồn, rồi mới viết

1. **Tạo một Gem (trợ lý riêng) cho việc soạn bài.** Đây là nơi lưu Hồ sơ Bối cảnh một lần rồi dùng lại nhiều lần. Phần chỉ dẫn của Gem chính là nội dung Phụ lục A của tài liệu này.
2. **Nạp Hồ sơ Bối cảnh trước, nạp yêu cầu sau.** Đây là quy tắc thứ tự: khi AI đọc yêu cầu trước khi đọc luật chơi, nó sẽ bắt đầu viết ngay theo thói quen, và các ràng buộc ở sau bị đối xử như lời nhắc phụ.
3. **Dán đoạn văn gốc đã kiểm làm "neo".** Trước khi yêu cầu Gemini viết, dán vào đoạn nguyên văn bạn đã xác minh từ NotebookLM, kèm một câu khoá:

```
Dưới đây là NGUỒN GỐC duy nhất được phép dùng.
Mọi con số, mọi ký hiệu phải khớp với đoạn này.
Không dùng kiến thức nền. Chỉ lập biểu thức --- không tự tính ra số.
--- NGUỒN GỐC BẮT ĐẦU ---
[dán nguyên văn đoạn đã xác minh]
--- NGUỒN GỐC KẾT THÚC ---
Nhiệm vụ: Viết mục lý thuyết cho nội dung trên, trình bày theo ba lớp:
(1) cơ chế không có công thức, (2) mô hình và công thức,
(3) quy định và thực hành. Mỗi công thức bắt buộc có mục ``điều kiện áp dụng''.
```

4. **Cuối mỗi lần gọi, đọc dòng "Điểm chưa chắc chắn".** Nếu Gemini không tự ghi dòng đó, hãy nhắc — vì theo Hồ sơ Bối cảnh, đó là nghĩa vụ bắt buộc.

4.5 Ví dụ 4 — Dùng Gemini để phản biện chính mình

Tầng 2 của kiến trúc kiểm duyệt có một điều kiện khó: người phản biện phải *độc lập*. Cùng một tài khoản Gemini, nếu mở một cuộc trò chuyện mới hoàn toàn và dán prompt phản biện ở Phụ lục B, bạn đã phần nào thoả mãn điều kiện đó — vì mô hình không còn nhớ nó vừa viết gì.

Giới hạn của cách này. Cùng một mô hình, dù ở cuộc trò chuyện mới, vẫn chia sẻ cùng thiên kiến và cùng lỗ hổng đào tạo. Nó bắt được lỗi *trình bày, ký hiệu, logic* — nhưng sẽ *không* bắt được lỗi mà cả mô hình lẫn nó cùng tin là đúng. Vì vậy cách này **không thay thế** tầng 3 (người thẩm định). Nó chỉ làm tầng 3 nhẹ việc hơn.

Mẹo dùng hai công cụ làm phản biện chéo. Nhờ Gemini phản biện bản nháp của chính nó; đồng thời mở NotebookLM hỏi cùng bộ nguồn xem có đoạn nào *mâu thuẫn* với bản nháp không. Hai công cụ này làm việc theo hai *cơ chế* khác nhau — NotebookLM bám vào đoạn văn được truy hồi, Gemini viết và lập luận — nên chúng ít có khả năng cùng bỏ sót một lỗi giống hệt nhau.

Đừng thổi phồng tính độc lập của hai công cụ. Một cách nói hay gặp — nhưng sai — là cho rằng hai công cụ "có nguồn gốc mô hình khác nhau" nên hai người kiểm là độc lập thật sự. Thực tế: cả hai đều là sản phẩm của Google, và NotebookLM được biết là chạy trên họ mô hình Gemini. Vì vậy chúng *có thể* chia sẻ thiên kiến và lỗ hổng đào tạo giống nhau — chúng ta chưa kiểm chứng được điều này và cũng không nên giả định là không.
Hệ quả thực tế: dùng hai công cụ này giúp *giảm tương quan lỗi*, chứ *không* tạo ra hai người kiểm độc lập thật sự. Muốn có tính độc lập thật, phải dùng mô hình của một nhà cung cấp khác cho tầng 2. Đây là điều nên tự kiểm tra lại trước khi dùng, vì họ mô hình của từng sản phẩm thay đổi theo thời gian.

4.6 Ví dụ 5 — Cảnh báo về việc dùng AI thay người

Đây không phải ví dụ về công cụ, mà là ví dụ về *cái bẫy lớn nhất* khi có công cụ trong tay. Tình huống dưới đây là dạng đã được ghi nhận nhiều lần trong thực tế, nên cần ghi lại thành tình huống cụ thể.

Tình huống điển hình. Giảng viên A dùng Gemini sinh một chương bài giảng, đọc thấy văn rất trôi chảy, công thức trình bày đẹp, dẫn chiếu ghi `[TCVN, tr.45]` gọn gàng. A cho rằng "đã có dẫn chiếu là đã kiểm", nên đưa thẳng ra lớp. Ba tuần sau, một sinh viên làm bài tập không ra kết quả — hoá ra công thức trong bài giảng bị sai một chỉ số dưới, và con số ở ví dụ không khớp với chính công thức đó.
Chỗ gãy không nằm ở AI. Nó nằm ở giả định "có dẫn chiếu = đã kiểm". Dẫn chiếu là *điểm bắt đầu* của việc kiểm, không phải chứng nhận đã kiểm xong.
Cách chặn. Tuyệt đối không xuất bản nội dung kỹ thuật mà chưa chạy qua danh sách kiểm tra ở Phụ lục C. Nếu không có thời gian chạy hết danh sách, hãy chạy tối thiểu nhóm A (công thức và số liệu) — đó là nhóm có thể gây hậu quả nặng nhất.

4.7 Cảnh báo về tính ổn định của thông tin công cụ

Phần này chắc chắn sẽ lỗi thời. Tên gọi, giới hạn số nguồn, hạn mức sử dụng, và tên gọi của tính năng bảo vệ tùy chỉnh của cả hai công cụ đều *thay đổi thường xuyên*. Ví dụ: tính năng bảo vệ tùy chỉnh của Gemini đã từng được gọi là "Gems", và tài liệu chính thức đã thông báo chuyển dần sang một hệ tên gọi khác. Tương tự, NotebookLM cũng đã được đổi tên thương hiệu.
Vì vậy, hãy đọc phần này như **mô tả cách làm việc**, không phải **bảng tra cứu tính năng**. Trước khi dùng, luôn tự xác nhận trong trang cấu hình của chính sản phẩm: hiện có bao nhiêu nguồn, giới hạn dung lượng là bao nhiêu, tính năng tùy chỉnh đang tên gì.

5. Bước 3: Phát triển nội dung

5.1 Nguyên lý: chia nhỏ là bắt buộc

Yêu cầu AI viết "một chương" là yêu cầu tồi — không phải vì AI không viết được, mà vì **không thể thẩm định một chương trong một lần đọc**. Khi khối lượng vượt quá khả năng kiểm tra của người đọc, chất lượng kiểm tra tụt xuống mức "đọc thấy trôi chảy thì cho qua". Và đó chính xác là điều kiện để lỗi loại hình và lỗi trôi ký hiệu sống sót qua vòng kiểm duyệt.

Nguyên tắc chia. Mỗi lần sinh ra *một đơn vị mà người dạy kiểm tra được trong một lượt* — thường là một mục nhỏ, một công thức, một ví dụ.

5.2 Sinh lý thuyết theo ba lớp

Với mỗi chủ đề, yêu cầu AI trả lời thành ba khối **tách biệt rõ ràng**.

Bảng 7: Ba lớp nội dung lý thuyết và cách kiểm chứng.

Lớp	Nội dung	Cách kiểm chứng
Cơ chế	Hiện tượng gì xảy ra, vì sao, điều kiện nào thì *không* xảy ra. Không có công thức.	Đối chiếu với **checklist ý cốt lõi** do giảng viên viết trước, và với tài liệu gốc đáng tin. "Diễn giải lại" là chưa đủ — xem cảnh báo bên dưới.
Mô hình	Công thức, kèm giải nghĩa từng ký hiệu, đơn vị, và **điều kiện áp dụng**.	Chạy máy với một bộ số cụ thể. Điều kiện áp dụng là phần hay bị bỏ nhất và cũng gây sai nhiều nhất khi dạy.
Quy định	Lấy từ kho nguồn quy định. Bắt buộc có dẫn chiếu vị trí.	Đối chiếu nguyên văn với nguồn, từng điều khoản một.

Lợi ích của việc tách lớp. Khi AI trả lời sai, ta biết ngay sai ở tầng nào: lớp Mô hình sai số → vấn đề tính toán; lớp Quy định sai dẫn chiếu → vấn đề nguồn hoặc truy hồi; lớp Cơ chế sai → vấn đề mô hình hiểu bài. **Ba loại lỗi, ba cách sửa khác nhau. Gộp lại thì không sửa được cái nào.**

Vì sao không thể kiểm lớp Cơ chế bằng cách "nhờ AI diễn giải lại". Hai lý do, và cả hai đều nghiêm trọng.
Thứ nhất, *vi phạm nguyên lý người viết không tự kiểm*. Nhờ chính mô hình đã viết đoạn đó diễn giải lại là giao việc kiểm cho người viết — đúng điều mục 6.1 cấm.
Thứ hai, và tinh tế hơn: **một lỗi ở lớp Cơ chế thường là lỗi *thiếu ý*, không phải lỗi *nói sai***. Ví dụ ở mục 5.3, AI bỏ sót ý "hai thành phần huy động ở chuyển vị khác nhau" — văn bản vẫn trôi chảy, vẫn đúng, chỉ *mất* một ý then chốt. Nếu đưa đoạn văn đó cho AI diễn giải lại, nó sẽ diễn giải *đúng cái phần còn lại* và không bao giờ tự phát hiện ra ý đã mất. Diễn giải lại chỉ bắt được lỗi khi AI nói *sai*; nó mù với lỗi khi AI nói *thiếu*.
Cách kiểm đúng: **giảng viên viết checklist ý cốt lõi trước**, rồi đối chiếu đoạn AI viết với checklist đó. Checklist này phải viết *trước khi* đọc bản nháp — nếu viết sau, ta lại bị bản nháp dẫn dắt.

5.3 Ví dụ cụ thể: bài toán sức chịu tải cọc

Để thấy rõ cách tách lớp, xét một chủ đề địa kỹ thuật điển hình: *xác định sức chịu tải của cọc đơn theo chỉ số xuyên tiêu chuẩn (SPT)*. Dưới đây là kết quả khi yêu cầu AI trình bày theo đúng ba lớp.

Lớp 1 — Cơ chế (không có công thức)

Cọc truyền tải trọng xuống đất qua hai cơ chế đồng thời. Thứ nhất, *ma sát thành* phát sinh dọc theo mặt tiếp xúc giữa thân cọc và đất, tỉ lệ với diện tích mặt bên và cường độ ứng suất tiếp tại mặt tiếp xúc. Thứ hai, *sức kháng mũi* phát sinh tại đáy cọc, tỉ lệ với diện tích tiết diện ngang và sức kháng của đất dưới mũi.

Điều quan trọng về cơ chế: hai thành phần này **huy động tối đa ở các mức chuyển vị khác nhau**. Ma sát thành thường đạt tối đa ở chuyển vị nhỏ hơn nhiều so với sức kháng mũi. Hệ quả sư phạm: khi tính tổng sức chịu tải bằng cách cộng thẳng hai thành phần, ta đã ngầm giả định chúng cùng huy động — giả định này không đúng với mọi trường hợp.

> Nhận xét về chất lượng đầu ra: đoạn trên đúng về cơ chế, nhưng điểm "hai thành phần huy động ở chuyển vị khác nhau" là điểm mà AI rất hay bỏ sót. Nếu bỏ sót, nội dung vẫn đọc trôi chảy, vẫn đúng công thức, nhưng *mất đi lý do vì sao không được cộng thẳng hai thành phần* — và mất luôn cơ sở để hiểu vì sao trong thực hành người ta phải đưa vào các hệ số điều chỉnh riêng cho từng loại cọc và từng loại đất. Đây chính là chỗ mà người dạy phải bổ sung, và cũng là ví dụ điển hình cho loại lỗi "đúng mà thiếu ý" đã cảnh báo ở mục 5.2.

Lớp 2 — Mô hình và công thức

Một dạng công thức thực hành phổ biến có cấu trúc:

$$Q_u \;=\; Q_s + Q_p \;=\; \sum_{i=1}^{n} f_{s,i}\, A_{s,i} \;+\; q_p\, A_p \tag{1}$$

trong đó:

Ký hiệu	Ý nghĩa	Đơn vị
Q_u	Sức chịu tải cực hạn của cọc đơn	kN
Q_s	Tổng thành phần ma sát thành	kN
Q_p	Thành phần sức kháng mũi	kN
$f_{s,i}$	Ứng suất tiếp đơn vị lớp đất thứ i	kPa
$A_{s,i}$	Diện tích mặt bên lớp đất thứ i	m^2
q_p	Sức kháng đơn vị dưới mũi cọc	kPa
A_p	Diện tích tiết diện ngang mũi cọc	m^2

Điều kiện áp dụng — phần quan trọng nhất và hay bị bỏ nhất:

- Công thức (1) chỉ áp dụng khi cọc chịu nén, tải trọng tác dụng dọc trục.
- Với cọc chịu kéo, số hạng Q_p không được tính.
- Với nhóm cọc, tổng sức chịu tải *không* bằng số cọc nhân sức chịu tải cọc đơn, do hiệu ứng nhóm.
- Các hệ số $f_{s,i}$ và q_p xác định theo tương quan thực nghiệm, nên *mang tính thống kê* — phải dùng đúng tương quan tương ứng với loại đất và loại cọc.

> **Điểm cần lưu ý khi dạy.** Dòng "tổng sức chịu tải nhóm cọc không bằng số cọc nhân cọc đơn" là chỗ sinh viên hiểu sai nhiều nhất. AI thường bỏ dòng này vì nó không nằm trong công thức, mà nằm trong *điều kiện áp dụng*. Đây là lý do vì sao phải yêu cầu AI nêu điều kiện áp dụng một cách tường minh, thay vì chỉ đưa công thức.

Dấu hiệu dùng sai — phần bắt buộc thứ hai, bổ sung cho "điều kiện áp dụng". Đây là các dấu hiệu *nhìn thấy được* cho biết có thể công thức đang bị dùng ngoài miền:

- Đề bài *không* nói loại cọc và cách thi công (đóng / khoan nhồi / ép) — nhưng lại yêu cầu tính Q_u. Vì f_s và q_p phụ thuộc mạnh vào loại cọc, thiếu thông tin này là dấu hiệu đang dùng công thức ngoài miền.
- Đề bài cho nhiều lớp đất khác nhau rõ rệt nhưng lại chỉ đưa *một* giá trị f_s chung — dấu hiệu đã bỏ qua tính phân lớp.
- Kết quả tính ra có Q_s hoặc Q_p chiếm gần như toàn bộ Q_u (trên 90%) — cần nghi ngờ vì trong thực tế hai thành phần thường cùng đóng góp.
- Tính cho cọc khoan nhồi nhưng dùng tương quan xây dựng cho cọc đóng — dấu hiệu của loại sai "đúng công thức, sai miền" đã nói ở mục 1.1.

Lớp 3 — Quy định và thực hành

Ở lớp này mới cho phép lấy từ kho nguồn quy định. Nội dung sẽ gồm: các tương quan thực nghiệm được quy định cho từng loại đất, hệ số điều kiện làm việc, quy định về số điểm thí nghiệm tối thiểu, và hệ số an toàn áp dụng cho cọc đơn và nhóm cọc.

Để thấy lớp này khác hai lớp trên ở chỗ nào, đây là một dòng hoàn chỉnh theo đúng bảng bốn cột đã định nghĩa:

Nội dung	Giá trị / Quy định	Vị trí trong nguồn	Mức tin cậy
Số điểm thí nghiệm tối thiểu cho một cọc	[điền giá trị từ bản gốc]	[điền số điều / bảng / trang]	XÁC NHẬN

Vì sao ví dụ trên để trống giá trị. Cố ý. Tài liệu này không tra cứu tiêu chuẩn hiện hành, nên không được phép điền một con số cụ thể vào đây — làm vậy là tự tạo ra loại sai "dẫn chiếu có vẻ đúng". Điều cần nhìn ở ví dụ này là *cấu trúc*: lớp Quy định khác hai lớp kia ở chỗ **mọi dòng đều phải có vị trí dẫn chiếu và mức tin cậy**. Hai lớp Cơ chế và Mô hình thì không cần cột dẫn chiếu, vì chúng không lấy từ văn bản quy định.

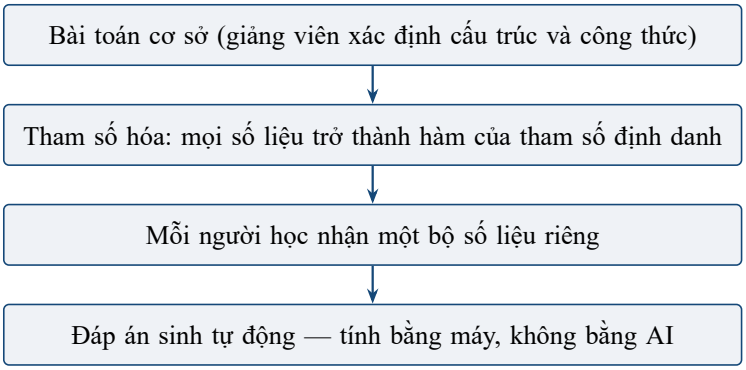
```
Với mỗi mục ở lớp này, yêu cầu AI xuất bảng:
| Nội dung | Giá trị / Quy định | Vị trí trong nguồn | Mức tin cậy |
Mức tin cậy nhận một trong ba giá trị: XÁC NHẬN / SUY LUẬN / KHÔNG CÓ TRONG NGUỒN.
Nếu mức là SUY LUẬN hoặc KHÔNG CÓ TRONG NGUỒN: đánh dấu rõ và không
được dùng làm cơ sở để giảng dạy trước khi giảng viên kiểm lại.
```

5.4 Sinh bài tập có tham số hóa

Giới hạn phạm vi — đọc trước khi dùng. Bài tập tham số hóa là công cụ đo năng lực ở mức **áp dụng**, và chỉ ở mức đó. Nó kiểm tra được: người học có biết chọn đúng công thức, thay đúng số, và ra đúng kết quả hay không. Nó **không** kiểm tra được mức *phân tích*: hiểu vì sao công thức đúng, biết khi nào nó không dùng được, hay đánh giá một kết quả có hợp lý không.
Nếu dùng bài tập tham số hóa cho mọi mục ở mức phân tích, người học sẽ học cách *thay số máy móc* — và đó chính là điều mà chương này muốn tránh. Với mức phân tích, phải dùng câu hỏi mở: cho một kết quả số cụ thể và hỏi "kết quả này có hợp lý không, vì sao?"; hoặc đưa hai phương án và yêu cầu chọn kèm biện luận.

Ý tưởng cốt lõi: **bài tập không cố định, mà sinh ra từ một hàm theo tham số định danh.**
Bốn lợi ích, xếp theo giá trị thực tế:

1. **Đáp án do máy tính, không do AI đoán.** Đây là lợi ích lớn nhất, và cũng là lợi ích hay bị bỏ qua nhất.



Hình 4: Nguyên lý sinh bài tập có tham số hóa.

2. **Tái sử dụng.** Cùng một bài toán, đổi tham số cho học kỳ sau mà không phải soạn lại.
3. **Truy vết.** Biết chính xác công thức nào sinh ra đáp số nào.
4. **Giảm chép trực tiếp.** Mỗi người học một bộ số liệu, một đáp số riêng.

Về lợi ích thứ tư — đừng nói quá. Một cách nói hay gặp — nhưng quá lời — là gọi việc đổi số liệu theo từng sinh viên là "chống chép bài". Thực chất nó chỉ làm được ba điều sau, không hơn:

- **Chỉ giảm được việc chép *nguyên đáp số*** giữa các bài. Sinh viên vẫn chia sẻ được *cách làm*, và có thể dùng AI để giải đề của mình — không có cách nào chặn điều đó bằng việc đổi số.
- **Đáp số phải thực sự không đoán được.** Nếu các tham số sinh theo một quy luật tuyến tính đơn giản (ví dụ tải trọng tăng đều theo số thứ tự), thì bất kỳ ai biết một đáp số là suy ra được các đáp số còn lại. Cách làm đúng: *sinh tham số từ hàm băm của mã định danh* (xem script ở mục 5.4) để các đáp số không còn quan hệ cộng trừ với nhau.
- **Nên thêm phần buộc sinh viên tự lập luận** — một câu hỏi ngắn yêu cầu giải thích, hoặc vấn đáp. Đây mới là thứ thật sự không thể chép, và cũng chính là loại câu hỏi mà AI không làm hộ được.

Nói gọn: đổi số giúp *giảm chép trực tiếp*, chứ không *chống* được chép. Muốn chống, phải hỏi vào phần lập luận.

Nguyên tắc vàng. AI viết **công thức**, máy chạy **số**. Không bao giờ để AI làm cả hai. Biểu thức do AI dựng thì chỉ cần kiểm một lần, dùng cho cả nghìn bài. Con số do máy tính thì đúng *với điều kiện công thức đã được kiểm và đã có test* — xem cảnh báo ngay dưới đây.

Nguyên tắc vàng này có một lỗ hổng, phải bịt trước khi dùng. "Kiểm một lần" có nghĩa là nếu lần kiểm đó *sai*, thì lỗi sẽ nhân lên thành cả nghìn đáp án sai — chứ không phải chỉ một. Máy chỉ đảm bảo *tính nhất quán với công thức*, không đảm bảo *công thức đúng*.
Vì vậy, bắt buộc phải có bước kiểm thứ ba, không được bỏ: **đưa vào script hai đến ba bài giải tay độc lập có đáp án đã biết** (lấy từ giáo trình hoặc bài đã công bố), và để script tự so bằng `assert`. Chỉ khi các test này qua, script mới được phép sinh đề hàng loạt. Đây là hiện thân của nguyên lý ở mục 6.1: người viết công thức không thể tự xác nhận công thức của mình.

Ví dụ: script sinh đề và đáp án

Xét bài toán tính áp lực gây lún tại đáy móng nông — một bước quen thuộc trong tính lún. Bài toán có cấu trúc đơn giản nhưng đủ để minh họa nguyên lý.

Giả thiết của bài toán — phải ghi rõ trong đề, không được để ngầm. Đây cũng là một bài học phương pháp: một đề bài thiếu giả thiết thì sinh viên giỏi cũng không làm đúng được, vì họ không biết đang tính theo quy ước nào.

- N_k là tải trọng tiêu chuẩn **tại đáy móng**, và *đã bao gồm* trọng lượng bản thân móng cùng đất đắp trên móng. Đây là quy ước của bài toán này; nếu dùng quy ước khác (N_k là tải ở mặt đất tự nhiên), phải cộng thêm $\gamma_{tb} D_f$ vào p_{tb}.
- Nền đất đồng nhất trong phạm vi ảnh hưởng; γ là dung trọng tự nhiên.
- Mực nước ngầm nằm **dưới** đáy móng.
- Tải trọng tác dụng đúng tâm; áp lực phân bố đều trên đáy móng.

Dấu hiệu dùng sai công thức: mực nước ngầm nằm *trên* đáy móng (khi đó phải dùng γ' thay cho γ hoặc tách riêng phần dưới mực nước); nền nhiều lớp rất khác nhau; tải lệch tâm gây áp lực phân bố không đều.

```python
# SINH DE VA DAP AN: ap luc gay lun tai day mong nong
# Bien so theo ma so nguoi hoc (stt)
# LUU Y: day la vi du minh hoa PHUONG PHAP, so lieu gia dinh

import os
import random

def tham_so_hoa(stt):
    """
    Sinh bo so lieu TU MOT HAM BAM cua ma so, khong dung bang tra.
    Moi tham so bien thien trong mien thuc te -> kiem mien moi co y nghia.
    """
    rng = random.Random(stt)          # hat giong tat dinh theo ma so
    N_k   = rng.randrange(700, 1301, 10)   # kN  - tai trong tieu chuan
    b     = rng.choice([1.5, 2.0, 2.5])    # m   - canh mong
    l     = rng.choice([2.0, 2.5, 3.0])    # m   - canh mong con lai
    D_f   = rng.uniform(1.0, 2.5)          # m   - chieu sau dat mong
    gamma = rng.uniform(16.0, 19.0)        # kN/m3 - dung trong tu nhien
    return N_k, b, l, D_f, gamma


def tinh_ap_luc_gay_lun(stt, kiem_tra=True):
    """
    Tra ve (de_bai, dap_an) cho mot ma so nguoi hoc.
    de_bai va dap_an la HAI dau ra TACH ROI (ghi ra hai tep khac nhau).
    """
    N_k, b, l, D_f, gamma = tham_so_hoa(stt)

    # --- 1. Kiem tra mien gia tri hop le (co the that su kich hoat) ---
    if kiem_tra:
        if N_k <= 0:
            raise ValueError(f"Tai trong khong hop le: N_k={N_k}")
        if b <= 0 or l <= 0:
            raise ValueError("Kich thuoc mong phai duong")
        if not (0.5 <= D_f <= 6.0):
            raise ValueError("Chieu sau mong ngoai mien thuc te")

```

```
    # --- 2. Tinh cac gia tri trung gian ---
    sigma_zg0 = gamma * D_f              # kPa - ung suat ban dau tai day mong
    A_m       = b * l                    # m2  - dien tich mong
    p_tb      = N_k / A_m                # kPa - ap luc trung binh day mong

    # --- 3. Tinh ket qua ---
    p_gl = p_tb - sigma_zg0              # kPa - ap luc gay lun

    # --- 4. KIEM TRA HOP LY VAT LY (buoc hay bi bo nhat) ---
    if kiem_tra:
        if p_gl <= 0:
            raise ValueError(
                f"p_gl = {p_gl:.2f} <= 0: tai trong nho hon ap luc ban dau, "
                "de bai vo nghia ve mat vat ly"
            )
        if p_tb > 500.0:
            raise ValueError(
                f"p_tb = {p_tb:.2f} kPa qua lon so voi nen thong thuong"
            )

    # --- 5. Xuat HAI dau ra TACH ROI ---
    de_bai = (...)
    dap_an = (...)

    return de_bai, dap_an


# ======================================================================
# TEST DAP AN DA BIET - BAT BUOC CHAY TRUOC KHI SINH DE HANG LOAT
# Moi test la mot bai giai tay doc lap, co dap an tu giao trinh.
# ======================================================================
def test_dap_an_da_biet():
    # Bai 1: mong 2x3 m, N_k = 600 kN, D_f = 1.5 m, gamma = 18 kN/m3.
    #   Giai tay: A_m = 6 m2; sigma_zg0 = 27 kPa;
    #             p_tb = 100 kPa; p_gl = 100 - 27 = 73 kPa.
    N_k, b, l, D_f, gamma = 600.0, 2.0, 3.0, 1.5, 18.0
    p_tb = N_k / (b * l)
    p_gl = p_tb - gamma * D_f
    assert abs(p_tb - 100.0) < 1e-9, "Test 1a that bai"
    assert abs(p_gl - 73.0) < 1e-9,  "Test 1b that bai"

    # Bai 2: mong 1.5x2.0 m, N_k = 480 kN, D_f = 2.0 m, gamma = 17 kN/m3.
    #   Giai tay: A_m = 3 m2; sigma_zg0 = 34 kPa;
    #             p_tb = 160 kPa; p_gl = 160 - 34 = 126 kPa.
    N_k, b, l, D_f, gamma = 480.0, 1.5, 2.0, 2.0, 17.0
    p_tb = N_k / (b * l)
    p_gl = p_tb - gamma * D_f
    assert abs(p_tb - 160.0) < 1e-9, "Test 2a that bai"
    assert abs(p_gl - 126.0) < 1e-9, "Test 2b that bai"

    print("Tat ca test dap an da biet: DAT")


# --- Vi du su dung ---
if __name__ == "__main__":
    test_dap_an_da_biet()             # chay test TRUOC

    # Ghi de va dap an ra HAI thu muc khac nhau
    os.makedirs("de", exist_ok=True)
    os.makedirs("dap_an", exist_ok=True)
    for ma_so in [1, 7, 25]:
        de, dap = tinh_ap_luc_gay_lun(ma_so)
        with open(f"de/de_{ma_so}.txt", "w", encoding="utf-8") as f:
```

```
            f.write(de)
        with open(f"dap_an/dap_an_{ma_so}.txt", "w", encoding="utf-8") as f:
            f.write(dap)
    print("Da ghi 3 de vao thu muc 'de/' va 3 dap an vao thu muc 'dap_an/'")
```

Listing 1: Script sinh đề và đáp án theo tham số định danh người học.

Kết quả khi chạy với ba mã số khác nhau (rút gọn — mỗi mã số sinh một cặp tệp riêng):

```
Tat ca test dap an da biet: DAT

Da ghi 3 de vao thu muc 'de/' va 3 dap an vao thu muc 'dap_an/'

--- Thu muc de/de_1.txt ---
DE BAI (ma so 1)
  Mong don kich thuoc 2.5 x 2.0 m, dat sau 1.38 m.
  Tai trong tieu chuan N_k = 780 kN.
  Dung trong tu nhien cua dat gamma = 17.5 kN/m3.
  YEU CAU: tinh ap luc gay lun tai day mong.

--- Thu muc dap_an/dap_an_1.txt ---
DAP AN (ma so 1)
  B1. sigma_zg0 = 17.5 * 1.38 = 24.18 kPa
  B2. A_m = 2.5 * 2.0 = 5.00 m2
  B3. p_tb = 780 / 5.00 = 156.00 kPa
  B4. p_gl = 156.00 - 24.18 = 131.82 kPa
  B5. Kiem tra: p_gl > 0 va p_tb = 156 < 500 kPa → DAT
  KET QUA: p_gl = 131.82 kPa
```

Sáu đặc điểm của script này đáng chú ý. (i) Có bước *kiểm tra miền giá trị hợp lệ* — và vì mọi tham số nay đều biến thiên, phép kiểm này *thực sự có thể kích hoạt*, không còn là trang trí. (ii) Có bước *kiểm tra hợp lý vật lý* (B5: $p_{gl} > 0$, p_{tb} không vượt ngưỡng vô lý) — đúng như danh sách nghiệm thu yêu cầu. (iii) Có *test đáp án đã biết* chạy trước khi sinh đề hàng loạt. Không có bước này, một công thức sai sẽ nhân thành nghìn đáp án sai. (iv) Đề bài và đáp án ghi ra *hai thư mục riêng* — đúng nghĩa "hai đầu ra vật lý tách rời". (v) Đáp án in ra *từng bước trung gian*, để người học đối chiếu được chỗ mình sai. (vi) Tham số sinh từ *hàm băm của mã số* (không dùng bảng tra tuyến tính), nên các đáp số không suy ra được từ nhau.

5.5 Cần có gì trước khi bắt đầu

Quy trình này có những bước đòi hỏi kỹ năng mà không phải giảng viên nào cũng có. Nói thẳng ra để không ai bị bất ngờ giữa chừng:

Cần	Dùng ở đâu	Nếu chưa có thì làm gì
Tài khoản NotebookLM và Gemini (bản miễn phí đủ dùng)	Chương 6.2, Chương 4	Không có phương án thay thế tương đương; đây là điều kiện tối thiểu.
Biết viết script Python cơ bản	Mục 5.4 (sinh bài tập), kiểm thử nguyên	**Phương án 1:** nhờ AI viết script, rồi tự đọc và chạy thử bằng hai bài giải tay. **Phương án 2:** nếu chưa làm được, dùng bảng tính (Excel) cho bước tính số — vẫn giữ được ý "máy tính, không phải AI".
Biết LaTeX hoặc Markdown cơ bản	Đặc tả đầu ra, Slide as Code	Không bắt buộc. Có thể dùng Word/PowerPoint quen thuộc; phần "Slide as Code" chỉ là tùy chọn nâng cao.
Quen một công cụ vẽ sơ đồ mã (Mermaid hoặc TikZ)	Mục sinh sơ đồ	Mermaid dễ hơn TikZ nhiều; nếu chưa quen, bắt đầu bằng Mermaid.

Nguyên tắc giảm rào cản. Nếu chỉ đọc được một phần của tài liệu này vì thiếu kỹ năng, **hãy bắt đầu bằng phần không cần code**: ma trận ký hiệu, Hồ sơ Bối cảnh, và kho nguồn NotebookLM. Ba việc đó đã chặn được phần lớn lỗi, và không đòi hỏi biết lập trình. Phần script và Slide as Code có thể để sau.

5.6 Sinh slide theo nguyên tắc "Slide as Code"

Slide viết bằng mã có ba ưu điểm mà slide kéo thả không có:

- **Kiểm soát phiên bản.** So sánh được hai phiên bản bài giảng, biết chính xác đã thay đổi gì.
- **Tái sử dụng.** Công thức viết một lần, dùng đồng thời ở slide, giáo trình, và đề thi.
- **Sinh hàng loạt.** Cùng một nguồn, xuất nhiều định dạng: trình chiếu, PDF phát tay, bản in.

Nguyên tắc thiết kế nội dung slide kỹ thuật

1. Một slide = một ý. Không gộp hai công thức độc lập vào cùng một slide.
2. Công thức đặt riêng dòng, không chèn giữa câu văn.
3. Hình vẽ minh họa nên sinh từ mã. Nếu chỉ có ảnh, không sửa được khi số liệu thay đổi.
4. Slide chỉ chứa từ khóa và công thức cốt lõi; phần giải thích nằm ở lời giảng. *Slide không phải giáo trình thu nhỏ.*

5.7 Sinh sơ đồ: chọn công cụ theo mục đích

Bảng 8: Chọn công cụ vẽ sơ đồ theo loại sơ đồ.

Loại sơ đồ	Công cụ	Khi nào dùng
Sơ đồ khối quy trình, sơ đồ tư duy, dòng thời gian	Mermaid	Mọi sơ đồ mô tả *logic*, không cần tọa độ chính xác.
Sơ đồ kết cấu, sơ đồ đặt tải, mặt cắt	TikZ	Khi cần *tọa độ và hình học chính xác*.
Biểu đồ số liệu (quan hệ giữa các đại lượng)	PGFPlots	Khi có dữ liệu, cần trục tọa độ và tỉ lệ đúng.

Quy tắc chọn. Nếu sơ đồ cần *biết chính xác vị trí một điểm*, dùng TikZ. Nếu chỉ cần *thể hiện trình tự hoặc quan hệ*, dùng Mermaid. Đừng dùng TikZ cho sơ đồ khối — sẽ mất nhiều thời gian căn chỉnh mà không tăng giá trị sư phạm.

Lưu ý khi nhờ AI viết mã TikZ. Mã TikZ rất dài, và AI hay sinh ra mã *biên dịch được nhưng hình sai*: đối xứng nhầm, tải đặt sai chiều, tỉ lệ méo. Đây là lỗi **không thể tự phát hiện bằng cách đọc mã** — phải biên dịch và *nhìn hình thật*. Vì vậy, luôn có một lượt render và quan sát hình riêng biệt trong quy trình, không gộp vào lượt kiểm tra mã.

6. Bước 4: Vòng lặp kiểm duyệt và phản biện

6.1 Nguyên lý: người viết không thể là người kiểm

Đây là nguyên lý cốt lõi của toàn bộ Bước 4, và nó *không* liên quan đến AI. Người — hoặc mô hình — vừa viết ra một nội dung thì không thể kiểm tra nội dung đó một cách khách quan, vì nó đọc lại bằng chính những giả định đã dùng lúc viết. Kết quả: nó luôn thấy "đúng như dự định".

Do đó: **phiên kiểm tra phải là phiên khác**, tốt nhất là mô hình khác. Nếu buộc phải dùng cùng mô hình, tối thiểu phải mở cuộc trò chuyện mới hoàn toàn — không có lịch sử, chỉ có bản nháp và bản quy tắc.

6.2 Kiến trúc ba tầng kiểm duyệt



Hình 5: Kiến trúc ba tầng kiểm duyệt.

Điểm mấu chốt về thứ tự. Tầng 1 phải chạy *trước* tầng 2. Nếu đảo thứ tự, AI sẽ tốn công phản biện ngữ nghĩa trên một văn bản còn lỗi ký hiệu và số học, và những lỗi đó làm nhiễu nhận định của nó. **Dọn lỗi cơ học trước, rồi mới bàn chuyện logic.**

6.3 Tầng 1 — Kiểm tự động

Bốn việc, tất cả đều có thể biến thành script chạy tự động:

1. **Đối chiếu ma trận ký hiệu.** Trích mọi ký hiệu xuất hiện trong bản nháp, so với bảng quy tắc. Báo cáo ba loại vi phạm: ký hiệu lạ, ký hiệu dùng hai nghĩa, và đại lượng có hai ký hiệu.

2. **Kiểm tra cú pháp công thức.** Dò các lỗi hay gặp: thiếu dấu đóng ngoặc, chỉ số lồng không cân bằng, ký tự đặc biệt chưa escape, công thức bị ngắt dòng giữa chừng.

3. **Kiểm tra đơn vị.** Đây là kỹ thuật mạnh nhất và ít được dùng nhất — *với các công thức có dạng giải tích*. Với mỗi công thức, kiểm tra thứ nguyên hai vế có khớp không; với mỗi phép cộng hoặc trừ, các số hạng có cùng đơn vị không. **Rất nhiều lỗi công thức bị bắt bằng cách này mà không cần hiểu môn học.**

   Giới hạn của kiểm thứ nguyên — điều quan trọng phải nói rõ. Kiểm thứ nguyên *không* áp dụng được cho các tương quan thực nghiệm — và đây lại chính là loại công thức phổ biến nhất

trong địa kỹ thuật. Ví dụ một tương quan dạng "$q_p = k \cdot N$" với N là chỉ số SPT (không thứ nguyên): để cho ra đơn vị kPa, hệ số k phải mang đơn vị kPa ngầm bên trong. Bộ kiểm thứ nguyên tự động, nếu không biết điều đó, sẽ coi công thức này là *sai* và báo động giả — hoặc bị bỏ qua và không kiểm gì cả.
Cách xử lý: với mọi hệ số của tương quan thực nghiệm, bắt buộc ghi *đơn vị ngầm* ngay trong ma trận ký hiệu (ví dụ "$k = 400$ kPa/vòng SPT"). Với các công thức không thể kiểm thứ nguyên, ghi rõ "*không áp dụng kiểm thứ nguyên — phải kiểm tay theo bảng gốc*" để người dùng biết chỗ này không được bảo vệ tự động.

4. **Chạy lại số học.** Phân biệt hai loại số: **số đầu vào** (tải trọng, kích thước, hệ số — lấy từ nguồn hoặc do giảng viên đặt, có thể nhập tay) và **số kết quả** (mọi giá trị tính ra ở bước trung gian và kết quả cuối). Số kết quả *bắt buộc* phải do script sinh ra. Nếu bản nháp có một số kết quả mà không script nào sinh ra nó → đó là con số không có nguồn gốc, phải loại bỏ và tính lại từ đầu.

   Luồng số *đi vào* bản nháp như thế nào. Ở mục 2.2, quy tắc đặt ra là AI không tự tính số, chỉ lập biểu thức. Vậy con số đi vào bài giảng bằng đường nào? Bằng một chuỗi ba bước, và mỗi bước có người chịu trách nhiệm khác nhau:

	Bước	Ai làm	Sản phẩm
1	Lập biểu thức	AI	$p_{gl} = N_k/A_m - \gamma D_f$ — chỉ công thức, không số
2	Cho số đầu vào	Giảng viên	Giá trị N_k, b, l, D_f, γ và nguồn của chúng
3	Chạy ra số	Máy (script)	Toàn bộ số kết quả, in từng bước trung gian

   Nói cách khác: quy tắc không phải "bài giảng không được có con số", mà là **"mọi con số *tính toán* phải do máy sinh; số đầu vào phải truy được nguồn"**. Ví dụ ở mục 5.4 minh họa đúng chuỗi ba bước này.

Ví dụ: kiểm tra thứ nguyên bắt lỗi công thức

Xét một công thức sai cố ý, để thấy sức mạnh của kỹ thuật kiểm thứ nguyên:

$$p_{gl} = \frac{N_k}{A_m} - \gamma\, D_f^2 \tag{2}$$

Kiểm thứ nguyên từng số hạng, với đơn vị cơ bản kN, m, kPa:

Số hạng	Biểu thức	Thứ nguyên	Kết quả
Số hạng 1	N_k/A_m	kN / m^2	kPa ✓
Số hạng 2	$\gamma\, D_f^2$	(kN/m^3) × m^2 = kN/m	**không phải kPa** ×

Kết luận: phép trừ ở công thức (2) *không hợp lệ về thứ nguyên* — số hạng thứ hai phải là γD_f (đơn vị kPa), không phải γD_f^2.

Một ghi chú trung thực về chính ví dụ này. Bảng trên đây rất dễ bị viết sai ở dòng kết quả trung gian: rất nhiều người (kể cả người viết tài liệu này) ghi "= kN" thay vì "= kN/m". Kết luận ("không phải kPa") khi đó vẫn đúng, nhưng *con số trung gian thì sai* — đúng loại lỗi mà mục 1.1 gọi là sai số học ở bước trung gian. Nó chỉ bị phát hiện khi tài liệu được đưa qua một vòng phản biện độc lập, *không* bởi người viết.
Điều đáng nói: người viết đã đọc lại đoạn này nhiều lần và không thấy. Đây là minh chứng cụ thể

cho nguyên lý ở mục 6.1: người viết không thể tự kiểm khách quan sản phẩm của chính mình — kể cả khi họ là người *dạy* cách kiểm.

Ý nghĩa thực tiễn. Lỗi này bị phát hiện *mà không cần biết gì về địa kỹ thuật*. Chỉ cần biết đơn vị. Đây chính là loại kiểm tra nên được tự động hóa sớm, vì nó rẻ và bắt được một lớp lỗi rất hay gặp trong các công thức giải tích — *với điều kiện* đã xử lý đúng các tương quan thực nghiệm như cảnh báo ở trên.

6.4 Tầng 2 — Prompt phản biện

Prompt phản biện hiệu quả phải có **định dạng đầu ra bắt buộc**. Nếu không, AI sẽ khen rồi thôi.

```
[ĐỊNH VỊ] Người phản biện độc lập. KHÔNG phải người viết.
[Nhiệm vụ] Tìm lỗi. KHÔNG khen. KHÔNG đề xuất cải thiện văn phong.
[Đính kèm bắt buộc] Bản nháp + ma trận ký hiệu + các đoạn nguồn liên quan.

[Định dạng đầu ra BẮT BUỘC] Lập bảng, mỗi dòng một phát hiện:
| # | Vị trí | Loại lỗi | Mô tả | Mức độ (Cao/Vừa/Thấp) |

[Rà bốn nhóm bắt buộc]
  1. Lỗi ký hiệu (trái quy tắc ở mục [X])
  2. Lỗi logic (kết luận không theo tiền đề)
  3. Lỗi điều kiện áp dụng (dùng công thức ngoài miền hợp lệ)
  4. Lỗi dẫn chiếu (khẳng định không có trong nguồn)

[Quy tắc báo cáo]
- Không tìm thấy lỗi ở một nhóm: ghi rõ "đã rà, không phát hiện".
- Không tìm thấy lỗi nào cả: nói rõ đã rà những gì.
```

Dòng cuối cùng là phòng vệ quan trọng: nếu AI nói "không có lỗi" mà không nói nó đã rà những gì, thì kết quả đó **không có giá trị** — nó có thể chỉ đang làm hài lòng người hỏi.

Giới hạn của tầng 2. AI phản biện vẫn có thể bỏ sót, và tệ hơn, có thể *báo lỗi giả* — bắt lỗi những chỗ thực ra đúng. Nếu sửa theo mọi báo cáo mà không kiểm lại, chất lượng sẽ *tệ đi* chứ không tốt lên.
Vì vậy: **kết quả tầng 2 là danh sách giả thuyết, không phải danh sách lỗi.** Mỗi mục phải được người kiểm lại trước khi sửa.

6.5 Tầng 3 — Thẩm định chuyên môn

Phần này không thể ủy thác cho AI. Giảng viên kiểm bốn điều:

1. **Đúng chuyên môn** — điều duy nhất AI không thể làm thay.
2. **Hợp trình độ người học** — nội dung đúng nhưng quá cao hoặc quá thấp vẫn là lỗi sư phạm.
3. **Khả thi khi dạy thật** — thời lượng, thiết bị, sĩ số lớp.
4. **Nhịp độ** — chỗ nào cần chậm lại, chỗ nào cần câu hỏi ngắt nhịp.

Bốn điều trên là *phạm vi* kiểm. Nhưng tầng 1 và tầng 2 có định dạng đầu ra bắt buộc, còn tầng 3 — tầng quan trọng nhất — lại thường được làm theo cảm tính. Đó là một nghịch lý cần sửa. Dưới đây là bộ câu hỏi cố định để tầng 3 không còn là "đọc rồi thấy ổn":

```
RUBRIC TẦNG 3 --- hỏi lần lượt, trả lời bằng chữ, không trả lời trong đầu

Nhóm A --- Sư phạm
A1. Đoạn này giúp người học làm được việc gì mới mà trước đó chưa làm được?
A2. Nếu bỏ hẳn đoạn này, người học mất gì? (Nếu không trả lời được → đoạn này chưa
cần.)
A3. Điểm neo vào thực tế / nghề nghiệp nằm ở đâu?

Nhóm B --- Độ khó và trình độ
B1. Người học cần sai ở đâu mới mất điểm ở đoạn này?
B2. Có chỗ nào đúng về kỹ thuật nhưng vượt quá nền tảng hiện có của lớp?
B3. Có chỗ nào quá dễ, khiến người học bỏ qua vì nghĩ mình đã biết?

Nhóm C --- Chất lượng chuyên môn
C1. Tôi có thể tự giải thích đoạn này mà không cần đọc lại tài liệu không?
C2. Nếu một đồng nghiệp phản biện, chỗ nào họ sẽ tấn công trước?
C3. Kết luận trung tâm của đoạn này có cần thiết cho mục tiêu học phần không?

Quy tắc: câu nào không trả lời được → đó là chỗ phải sửa, không phải chỗ bỏ qua.
```

Vì sao rubric này hoạt động. Nó biến một quyết định cảm tính ("đọc thấy ổn") thành một loạt câu hỏi trả lời được. Câu B1 ("người học cần sai ở đâu mới mất điểm") là câu mạnh nhất: nếu không trả lời được, nghĩa là đoạn đó *không đo được năng lực gì cả*. Câu C1 ("tự giải thích mà không cần đọc lại") bắt được lỗi "đúng mà vô nghĩa": nội dung đọc trôi chảy nhưng người dạy không thật sự nắm được.

6.6 Danh sách kiểm tra nghiệm thu

Danh sách đầy đủ — sáu nhóm A đến F, khoảng ba mươi mục — nằm ở **Phụ lục C**. Phần này chỉ nêu sáu nhóm và mục đích của từng nhóm, để khi đọc tới đó biết mình đang kiểm vì lý do gì.

Nhóm	Tên	Câu hỏi trung tâm của nhóm
A	Công thức & số liệu	Mọi công thức có điều kiện áp dụng và dấu hiệu dùng sai? Mọi số đều truy được nguồn?
B	Câu hỏi & bài tập	Câu hỏi có đo đúng mức năng lực đã gán? Nhiễu có phải nhằm lẫn thật?
C	Nguồn & dẫn chiếu	Dẫn chiếu có trỏ đúng vị trí? Đã đối chiếu bản OCR với bản gốc?
D	Trình bày & sơ đồ	Sơ đồ đã render và *nhìn* xác nhận? Chỗ so sánh đã dùng bảng?
E	Sư phạm	Đã gán mức năng lực? Có neo thực tế? Ví dụ mẫu có bước kiểm tra kết quả?
F	Bảo trì lâu dài	Kho nguồn có tên và ngày? Khi quy định đổi có tạo kho mới?

Cách dùng thực tế — đừng chạy cả ba mươi mục mỗi lần. Nếu không có thời gian chạy hết Phụ lục C, hãy chạy tối thiểu **nhóm A** — đó là nhóm có thể gây hậu quả nặng nhất nếu bỏ sót. Các nhóm còn lại có thể chạy luân phiên giữa các lần soạn.

7. Kỹ thuật nâng cao

7.1 "Rượu mới bình cũ" — cập nhật quy định mới, giữ nguyên văn phong

Bài toán. Tài liệu cũ soạn theo quy định cũ. Quy định mới ra đời. Cần thay số liệu và công thức, nhưng phải *giữ nguyên cấu trúc sư phạm và cách diễn đạt quen thuộc* — vì bản thân văn phong đó đã được tinh chỉnh qua nhiều năm dạy học, và người học đã quen với nó.

Vì sao không thể chỉ bảo AI "cập nhật theo quy định mới". AI sẽ viết lại từ đầu theo văn phong của nó, và ta mất toàn bộ giá trị tích lũy. Tệ hơn: nó giữ lại những gì *trông như* bản cũ nhưng thực chất đã viết mới — dạng lỗi khó phát hiện nhất.



Hình 6: Bốn bước của kỹ thuật "Rượu mới bình cũ".

Bảng khác biệt quy định — bước không được bỏ

Đây là bước rất dễ bị bỏ sót khi soạn lại bài giảng theo quy định mới, và nó tạo ra một lỗ hổng thật: nếu không biết *điều khoản nào đã thay đổi*, ta không có cách nào biết cần sửa chỗ nào trong bài giảng cũ — và sẽ bỏ sót những thay đổi nhỏ nhưng quan trọng (một hệ số điều chỉnh đổi từ 0,8 thành 0,85 chẳng hạn).

Trước khi soạn lại, lập một bảng như sau:

Điều khoản / mục	Quy định cũ	Quy định mới	Ghi chú
[mã điều khoản]	[giá trị / nội dung]	[giá trị / nội dung]	đổi / thêm mới / bỏ / không đổi
…			

Sau đó yêu cầu AI sửa **trực tiếp trên bản giảng cũ**, ở chế độ có theo dõi thay đổi, và **chỉ chạm vào những mục có trong bảng khác biệt**. Cách này khác hẳn về bản chất so với việc yêu cầu AI "soạn lại theo quy định mới": nó buộc phải giữ nguyên phần không đổi, và ta kiểm được *những gì đã đổi* — thay vì phải đọc lại cả bài để đoán.

Nội dung của "Hướng dẫn phong cách & cấu trúc"

Đây là sản phẩm của Bước 1, và là tài sản dùng lại vĩnh viễn:

- Thứ tự trình bày của mỗi mục.

- Cách đặt câu, cách dùng thuật ngữ.
- Mức độ chi tiết ở mỗi đoạn.
- Cách đánh số, cách đặt tiêu đề.
- Cách trình bày ví dụ và lời giải.

Ba điểm dễ sai nhất. (i) Không tách Bước 1 ra riêng mà gộp với Bước 3 — khiến AI vừa phải bắt chước văn phong vừa phải tra cứu, và nó sẽ ưu tiên tra cứu, làm mất văn phong cũ. (ii) Bỏ qua bảng khác biệt quy định ở Bước 2 — dẫn tới việc sửa mà không biết mình đang sửa cái gì, và bỏ sót các thay đổi nhỏ. (iii) Lẫn quy định cũ và mới trong cùng một kho nguồn — dẫn tới kết quả "lai" giữa hai hệ quy định.

7.2 Neo vào phong cách của chính người dạy

Bước 1 ở mục trên có thể nâng lên một mức: thay vì trích xuất phong cách của *một tài liệu*, trích xuất phong cách của *chính giảng viên* qua nhiều tài liệu.

Cách làm. Lấy từ ba đến năm đoạn do chính giảng viên viết, thuộc các loại khác nhau (giải thích khái niệm, lời giải bài tập, nhận xét kết quả). Yêu cầu AI phân tích và trả về:

- Cách mở một khái niệm mới.
- Cách chuyển ý giữa các phần.
- Những câu mẫu hay dùng.
- Mức độ dùng thuật ngữ tiếng nước ngoài.
- Cách nhấn mạnh một điểm quan trọng.

Giá trị. Sản phẩm là một "hồ sơ phong cách" dùng lại vĩnh viễn. Tài liệu do AI soạn sẽ đọc lên *vẫn là giọng của chính người dạy*, nên người học không thấy xa lạ, và người dạy tiết kiệm được công biên tập văn phong.

7.3 Tạo ngân hàng câu hỏi từ tài liệu nguồn

Cách làm đúng. Không yêu cầu AI "viết 50 câu trắc nghiệm" — nó sẽ ra 50 câu giống nhau và nông. Thay vào đó:

1. Lấy **chuẩn đầu ra học phần (CĐR)** làm điểm xuất phát: mỗi mục nội dung và mỗi câu hỏi phải gắn được với một CĐR nào đó. Đây là ràng buộc *duy nhất* chi phối toàn bộ việc sinh câu hỏi.
2. Với mỗi CĐR, yêu cầu sinh câu hỏi ở *nhiều mức*: hiểu, áp dụng, phân tích.
3. Bắt buộc mỗi câu kèm: mã CĐR, đáp án, và **giải thích vì sao các phương án nhiễu sai**.

Đừng lấy "danh sách mục tiêu" từ tài liệu nguồn. Một cách làm hay gặp — nhưng sai về nguyên tắc — là "trích xuất danh sách mục tiêu từ tài liệu nguồn". Tài liệu quy định (tiêu chuẩn, giáo trình) *liệt kê nội dung*, chứ không liệt kê *điều người học cần làm được*. Nếu lấy mục tiêu từ đó, ngân hàng câu hỏi sẽ bám vào nội dung quy định thay vì bám vào năng lực — và câu hỏi sinh ra sẽ kiểm trí nhớ chứ không kiểm năng lực.
Đúng ra: **mục tiêu lấy từ CĐR của học phần; tài liệu nguồn chỉ dùng để tìm nội dung kỹ thuật**. Đây cũng là lý do vì sao ở mục 2.2, CĐR phải là đầu vào của Hồ sơ Bối cảnh, chứ không phải một

thứ suy ra về sau.

Phần giải thích phương án nhiễu là phần có giá trị giảng dạy cao nhất và hay bị bỏ nhất. Nó biến một câu trắc nghiệm từ công cụ chấm điểm thành công cụ dạy học.

Kiểm chứng bắt buộc. Mọi câu hỏi có số liệu phải được tính lại bằng máy. Câu hỏi trắc nghiệm có đáp án số là chỗ AI sai nhiều nhất, vì nó sinh đáp án nhiễu bằng cách cộng trừ tùy ý — và đáp án nhiễu đó thường trùng với đáp án đúng do lỗi tính.

Phương án nhiễu: "kèm giải thích" là chưa đủ. Một cách làm hay gặp là đề xuất "kèm giải thích vì sao nhiễu sai". Giải thích là tốt cho người học, nhưng nó *không làm cho nhiễu bớt tùy ý*. Một phương án nhiễu tốt phải là *kết quả của một sai lầm có thật* mà sinh viên hay mắc — không phải một con số được cộng trừ ngẫu nhiên.
Cách làm: cho AI liệt kê *các mô hình lỗi* thường gặp (ví dụ: quên trừ σ_{zg0}; dùng D_f^2 thay vì D_f; nhầm đơn vị kN với kPa), rồi **sinh nhiễu bằng script** theo đúng các mô hình đó — mỗi phương án nhiễu gắn một nhãn lỗi. Khi đó việc chấm bài còn cho biết sinh viên mắc *loại* lỗi nào, chứ không chỉ biết đúng hay sai.

Thang mức năng lực phải nhất quán — và câu hỏi mở phải có rubric. Tài liệu dùng thang ba mức **B2-hiểu / B3-áp dụng / B4-phân tích** (theo thứ bậc nhận thức quen thuộc). Một lỗi rất dễ mắc là ở mục tạo ngân hàng câu hỏi lại ghi bốn mức (thêm "nhận biết"), gây mâu thuẫn với thang ba mức dùng trong Hồ sơ Bối cảnh — và mâu thuẫn này làm cho việc kiểm "câu hỏi đo đúng mức đã gán" không thực hiện được. Vì vậy tài liệu thống nhất về **ba mức** cho mọi chỗ.
Về câu hỏi mở: yêu cầu "có ít nhất một câu hỏi mở" mà không kèm cách chấm thì câu hỏi mở sẽ khó chấm nhất quán. Mẫu rubric ngắn sau dùng được cho hầu hết câu hỏi mở kỹ thuật:

Tiêu chí	Chưa đạt	Đạt	Tốt
Chọn đúng phương án / hướng giải	Chọn sai	Chọn đúng	Chọn đúng và nêu được vì sao
Lập luận	Không nêu lý do	Nêu lý do nhưng thiếu căn cứ	Lý do gắn với công thức / điều kiện áp dụng
Nhận diện giới hạn	Không thấy	Nêu có nhưng chung	Chỉ đúng điều kiện khiến cách giải không dùng được

7.4 Tạo kịch bản tương tác trên lớp

Ngoài nội dung, AI hỗ trợ tốt việc dựng kịch bản dạy học:

Câu hỏi gợi mở theo từng mục
Không đưa đáp án, chỉ đưa câu hỏi dẫn dắt từng bước dẫn tới kết luận.

Dự đoán hiểu nhầm phổ biến
Yêu cầu liệt kê những chỗ người học hay hiểu sai, kèm cách phát hiện và cách chữa.

Tình huống đảo lớp
Giao trước phần đọc, thiết kế hoạt động trên lớp để kiểm tra và đào sâu.

Lưu ý — nhưng đừng vì thế mà bỏ kiểm. Đây là khu vực AI *trông có vẻ* an toàn nhất, vì nó không cần số liệu chính xác. Nhưng "không cần số" không có nghĩa là "không có lỗi" — nó chỉ có nghĩa là **loại lỗi ở đây khác**: một danh sách "hiểu nhầm phổ biến" do AI viết có thể là những hiểu nhầm *không hề tồn tại* ở lớp của bạn, và một "cách chữa" có thể chứa lỗi kỹ thuật. Cả hai đều lọt qua tầng 1 vì không có số để kiểm.
Vì vậy đừng gọi đây là "rủi ro thấp", hãy gọi đúng là **rủi ro khác loại**. Cách kiểm: đối chiếu danh sách hiểu nhầm với *bài làm thật của sinh viên khóa trước* — đó là nguồn dữ liệu đáng tin duy nhất; và kiểm kỹ phần "cách chữa" như kiểm mọi nội dung kỹ thuật khác. Nếu muốn bắt đầu dùng AI ở mức an toàn, đây vẫn là chỗ bắt đầu tốt — nhưng phải kiểm theo cách khác, không phải bỏ kiểm.

7.5 Quyền sử dụng tài liệu và dữ liệu

Phần này ít được nói tới nhưng lại là câu hỏi đầu tiên mà đồng nghiệp sẽ đặt ra khi nghe báo cáo. Quy trình trong tài liệu có những thao tác *nạp tài liệu lên dịch vụ đám mây* (mục 3.3, chương 4), nên có ba việc phải cân nhắc trước khi áp dụng:

1. **Quyền sử dụng tiêu chuẩn.** Nhiều tiêu chuẩn kỹ thuật có bản quyền và giấy phép hạn chế việc sao chép, phân phối, hoặc đưa vào hệ thống của bên thứ ba. **Kiểm giấy phép của bộ tiêu chuẩn mình dùng** trước khi nạp toàn văn lên công cụ đám mây. Khi cần hỏi đáp trong phạm vi nội bộ, có thể chỉ nạp các đoạn trích ngắn mà mình có quyền sử dụng.

2. **Dữ liệu nội bộ và dữ liệu người học. Không nạp** tài liệu mật của đơn vị, hồ sơ nhân sự, hay *dữ liệu cá nhân và kết quả học tập của sinh viên* lên công cụ công cộng. Bài tập tham số hóa *không* cần dữ liệu thật của sinh viên — chỉ cần một mã định danh do giảng viên tự đặt là đủ.

3. **Quy định của đơn vị về dùng AI.** Nhiều trường và khoa đã có hoặc đang xây quy định riêng về việc dùng AI trong giảng dạy và trong công bố tài liệu. **Hỏi đơn vị trước khi áp dụng**, thay vì tự suy đoán. Việc nạp tài liệu giảng dạy lên dịch vụ bên ngoài cũng nên báo và xin ý kiến nếu đơn vị chưa có quy định rõ.

Nguyên tắc chung. Nếu không chắc một loại tài liệu hoặc dữ liệu có được nạp lên công cụ đám mây hay không, **cách an toàn nhất là không nạp**. Quy trình này không đòi hỏi phải nạp mọi thứ — có thể chạy với các đoạn trích do mình tự gõ lại, dù mất công hơn một chút.

7.6 Khi lỗi vẫn lọt ra: quy trình xử lý sau phát hành

Không quy trình nào đạt tỉ lệ không sót. Ví dụ 5 ở chương 4 kể đúng một tình huống: lỗi được một sinh viên phát hiện sau ba tuần. Vậy khi lỗi đã ra tới người học, phải làm gì?

1. **Gắn phiên bản lên tài liệu phát hành.** Mỗi bộ slide và mỗi tài liệu bài tập ghi một mã phiên bản (ví dụ `2026.10-v1`) và ngày phát hành. Không có mã phiên bản thì không thể thông báo đính chính chính xác cho ai.

2. **Có kênh để người học báo lỗi.** Sinh viên là người phát hiện lỗi sớm nhất nhưng thường không có cách nào báo. Chỉ cần một dòng ở cuối tài liệu: "Nếu thấy nội dung chưa đúng, gửi cho thầy/cô qua [kênh]". Điều này biến người học thành một tầng kiểm miễn phí.

3. **Có trang đính chính.** Khi xác nhận lỗi, phát hành một bản đính chính ngắn: lỗi ở đâu, sai thế nào, đúng là gì, ảnh hưởng tới ai. Không lặng lẽ sửa — vì bản cũ vẫn đang lưu hành ở nhiều máy.

4. **Ghi mỗi lỗi vào nhật ký.** Đây là điểm quan trọng nhất và hay bị bỏ nhất: mỗi lỗi lọt ra đều là dữ liệu cho biết *mắt xích nào trong quy trình đang yếu*. Lỗi thứ nguyên lọt ra nghĩa là kiểm thứ nguyên chưa

chạy; lỗi dẫn chiếu lọt ra nghĩa là kho nguồn chưa được đối chiếu. Nhật ký này nối trực tiếp vào bước "ghi lại chỗ gãy" ở mục 9.

Đừng để việc xử lý lỗi biến thành việc chữa cháy. Phản ứng tự nhiên khi bị phát hiện lỗi là sửa ngay chỗ đó rồi thôi. Nhưng nếu không hỏi "vì sao lỗi này lọt qua được quy trình", thì loại lỗi đó sẽ lọt lại lần sau ở chỗ khác. Mỗi lỗi lọt ra phải trả lời được hai câu: (a) nó đã lọt qua mắt xích nào, và (b) cần thêm hay sửa gì ở mắt xích đó.

8. Mười sáu cái bẫy cần tránh

#	Cái bẫy	Dấu hiệu nhận biết	Cách chặn
1	Tin vào con số trôi chảy	Kết quả có đơn vị đẹp, tròn trịa	Mọi số phải sinh từ script
2	Tin vào dẫn chiếu có vẻ đúng	Có số trang, có số điều khoản	Đối chiếu nguyên văn — dẫn chiếu cũng có thể bịa
3	Trôi ký hiệu trong văn bản dài	Ký hiệu "tự nhiên" đổi sau vài trang	Ma trận ký hiệu + kiểm tự động
4	Sửa theo mọi báo cáo phản biện	Sửa rất nhiều mà chất lượng không lên	Coi báo cáo là giả thuyết, kiểm lại từng cái
5	Tin rằng ngữ cảnh càng dài càng tốt	Nạp thật nhiều tài liệu cho "yên tâm"	Chất lượng truy hồi giảm khi kho pha loãng
6	Bỏ qua điều kiện áp dụng	Công thức đúng nhưng dùng sai miền	Bắt buộc nêu điều kiện ở mỗi công thức
7	Đọc mã sơ đồ thay vì nhìn hình	Mã "có vẻ đúng" nên cho qua	Luôn render và quan sát hình thật
8	Gộp đề và đáp án vào một tệp	Tiện lúc soạn	Tách thành hai đầu ra vật lý
9	Trộn quy định cũ và mới	Kết quả "lai" giữa hai hệ	Loại bỏ nguồn cũ trước khi nạp nguồn mới
10	Kho nguồn là bản OCR chưa đối chiếu	Dẫn chiếu đầy đủ nhưng số liệu sai	Đối chiếu công thức và bảng thủ công sau OCR
11	Fine-tune để dạy sự thật	Tốn nhiều công mà vẫn bịa	Fine-tune chỉ dạy văn phong; sự thật dùng truy hồi
12	Phó mặc khâu cuối cho AI	Không có tầng thẩm định của người	Tầng 3 không thể ủy thác
13	Đúng công thức, sai miền áp dụng	Mọi phép kiểm tự động đều qua	Bắt buộc ghi "điều kiện áp dụng" & "dấu hiệu dùng sai"
14	Thẩm định tầng 3 theo cảm tính	Đọc thấy "ổn" là cho qua	Dùng bộ rubric tầng 3, trả lời bằng chữ từng câu
15	Nhiễu trắc nghiệm do AI cộng trừ tùy ý	Một phương án trùng đáp án đúng	Sinh nhiễu bằng script theo mô hình lỗi có nhãn
16	Script không có test đáp án đã biết	"Kiểm một lần dùng cả nghìn bài"	Bắt buộc 2–3 test giải tay chạy trước khi sinh đề

9. Lộ trình áp dụng

9.1 Lộ trình khởi động cho người bắt đầu từ số không

Giai đoạn 1 ở dưới mô tả đầy đủ hạ tầng cần dựng. Nhưng với một giảng viên chưa có gì, yêu cầu dựng cả hạ tầng trước khi thấy được giá trị là một rào cản thật — và rào cản đó thường dẫn tới việc bỏ cuộc ngay từ tuần đầu.

Đọc giả định trước khi bắt đầu — nếu không, lộ trình sẽ vượt giờ ngay. Lộ trình dưới đây giả định **nguồn đã có dạng văn bản sạch, không phải bản scan**. Nếu tài liệu của thầy/cô là bản scan, phải cộng thêm thời gian OCR và đối chiếu công thức thủ công (mục 3.3) — việc đó với một chương có thể mất thêm vài giờ, và *không thể* ép vào 90 phút.
Lộ trình cố tình **tách thành hai buổi riêng** vì một buổi 90 phút không đủ cho cả việc dựng và việc chạy. Người mới thường đánh giá thấp ba việc tốn thời gian nhất: viết Hồ sơ Bối cảnh lần đầu, viết script cho người chưa quen code, và tầng 3.

Buổi 1 — dựng hạ tầng tối thiểu cho một chương (khoảng 75 phút, chưa đụng vào tầng 3):

Thời lượng	Việc	Cách làm và sản phẩm
15 phút	Viết ma trận ký hiệu	Liệt kê đúng những ký hiệu của *một chương* đã chọn. Ba cột: ký hiệu, ý nghĩa, đơn vị. Không cần đủ cả học phần.
20 phút	Viết Hồ sơ Bối cảnh	Dùng Phụ lục A làm mẫu, điền cho đúng chương đó. **Việc này tốn thời gian hơn dự kiến** — đây là lần đầu làm, các lần sau chỉ sửa vài dòng.
15 phút	Tạo notebook và kiểm kho nguồn	Nạp tài liệu gốc của chương. Dùng câu hỏi ở mục 4.2 để phát hiện chỗ công thức bị méo.
25 phút	Viết script chạy số	Chỉ cho *một* công thức của chương. **Nếu chưa biết code**, nhờ AI viết script rồi tự đọc kỹ và chạy bằng hai bài giải tay. Thời gian này có thể lâu hơn với người mới.

Buổi 2 — chạy hết một vòng trên một mục nhỏ (khoảng 90 phút, có tầng 3):

Thời lượng	Việc	Cách làm và sản phẩm
25 phút	Sinh một mục lý thuyết	Dán Hồ sơ Bối cảnh, dán đoạn nguồn đã xác minh, yêu cầu viết theo ba lớp *và* kèm "điều kiện áp dụng" + "dấu hiệu dùng sai".
20 phút	Chạy kiểm tra tầng 1	Kiểm ký hiệu, thứ nguyên, và tính lại số bằng script. Sửa hết lỗi cơ học trước khi sang bước sau.
15 phút	Chạy phản biện tầng 2	Cuộc trò chuyện mới, dán Phụ lục B, **đính kèm cả đoạn nguồn**. Coi kết quả là giả thuyết.
25 phút	**Thẩm định tầng 3**	Trả lời bộ rubric ở mục 6.5. **Bước này không được bỏ** — đây là tầng duy nhất bắt được lỗi "đúng công thức, sai miền".
5 phút	Ghi lại chỗ gãy	Danh sách những chỗ phải sửa tay → đây là dữ liệu để cải tiến quy trình.

Vì sao chỉ một chương. Mục đích của hai buổi này **không phải** để có một sản phẩm hoàn chỉnh, mà để bạn *chạm tay vào từng mắt xích một lần*. Sau đó, bạn sẽ biết chính xác mắt xích nào tốn thời gian nhất với môn mình dạy, và đó là thứ tự ưu tiên thật của bạn — không phải thứ tự lý thuyết trong tài liệu.

9.2 Bốn giai đoạn triển khai

Giai đoạn 1

Dựng hạ tầng — làm một lần cho mỗi học phần.

- Viết ma trận ký hiệu cho từng học phần.
- Viết template Hồ sơ Bối cảnh cho từng học phần.
- Xây kho nguồn đã xử lý (OCR + đối chiếu) cho học phần trọng tâm.
- Tạo thư mục dự án theo quy ước thống nhất.

Giai đoạn 2

Thử nghiệm trên phạm vi hẹp.

- Chọn *một mục nhỏ* đã có bài giảng cũ làm đối chứng.
- Chạy đủ bốn bước trên mục đó.
- So sánh với bài giảng cũ: chỗ nào tốt hơn, chỗ nào tệ hơn, mất bao nhiêu công.
- **Đo ba chỉ tiêu** trước và sau, ghi lại con số cụ thể: (a) giờ công soạn mục đó; (b) số lỗi phát hiện được ở tầng 2 và tầng 3; (c) một chỉ tiêu phía người học (tỷ lệ làm đúng bài tập mẫu, hoặc một khảo sát ngắn). Không có (c) thì không biết được quy trình có giúp ích gì cho người học hay không — xem mục 9.2.
- **Mục đích của giai đoạn này không phải tạo sản phẩm, mà là hiệu chỉnh quy trình.**

Giai đoạn 3

Mở rộng có kiểm soát.

- Áp dụng cho một chương trọn vẹn.
- Trong lúc làm, ghi lại mọi chỗ phải sửa thủ công → đó chính là danh sách cải tiến quy trình.

Giai đoạn 4

Chuẩn hóa.

- Chốt template và quy ước.
- Viết hướng dẫn để người khác dùng lại (đồng nghiệp, trợ giảng).

9.3 Thứ tự ưu tiên khi nguồn lực có hạn

Nếu chỉ làm được vài việc, hãy làm theo đúng thứ tự sau — xếp theo tỉ lệ giữa giá trị thu được và công sức bỏ ra:

1. **Ma trận ký hiệu.** Rẻ nhất, và chặn được loại lỗi phổ biến nhất (trôi ký hiệu) cùng một phần lỗi loại hình.
2. **Kiểm thứ nguyên tự động.** Rẻ, áp được cho mọi công thức, và bắt được lớp lỗi công thức mà không cần hiểu môn học. Nên làm ngay sau ma trận ký hiệu.

3. **Script chạy lại số liệu — kèm test đáp án đã biết.** Chặn được loại lỗi nguy hiểm nhất, *với điều kiện* đã có test; không có test thì script chỉ nhân lỗi lên nhanh hơn.
4. **Kho nguồn đã đối chiếu.** Không có nó thì truy hồi chỉ tạo ảo giác an toàn.
5. **Template Hồ sơ Bối cảnh.** Tiết kiệm công về lâu dài.
6. **Prompt phản biện có định dạng.** Cải thiện chất lượng kiểm tra.
7. **Hồ sơ phong cách cá nhân.** Cho sản phẩm đọc lên giống chính mình.

9.4 Dấu hiệu quy trình đang có vấn đề

- Đọc bản nháp và thấy *không có gì để sửa* → quy trình kiểm tra đang bị bỏ qua, hoặc đã "quen mắt" với sản phẩm của AI.
- Sửa nhiều vòng mà bản nháp *không tốt lên* → đang sửa triệu chứng chứ không sửa nguyên nhân ở tầng thấp.
- Mỗi lần soạn lại phải *làm lại từ đầu* → chưa có hạ tầng (ma trận, template, kho nguồn).
- Không nói được chính xác *con số nào do máy sinh* → kỷ luật chưa được thiết lập.

10. Ghi chú về độ tin cậy của tài liệu

Phần này liệt kê trung thực những gì tài liệu khẳng định chắc chắn và những gì người đọc cần tự kiểm chứng. Mục đích: để người đọc biết chỗ nào có thể tin ngay, chỗ nào phải kiểm lại.

10.1 Những khẳng định có căn cứ cơ chế

Các khẳng định dưới đây có căn cứ từ cơ chế hoạt động của mô hình ngôn ngữ, không phụ thuộc vào phiên bản công cụ cụ thể:

1. Mô hình ngôn ngữ sinh văn bản theo xác suất, do đó tối ưu cho tính trôi chảy, không tối ưu cho tính đúng.
2. Phép tính số học nhiều bước trong mô hình ngôn ngữ không đáng tin; cần chuyển sang script để tính.
3. Mô hình không có ký ức bền vững về quy ước đã đặt ra trong một cuộc trò chuyện dài, nên quy ước có thể trôi dần — đây là *cơ chế*, và cũng giải thích hiện tượng trôi ký hiệu.
4. Người viết không thể tự kiểm tra khách quan nội dung do chính mình viết.
5. Tinh chỉnh mô hình dạy văn phong và định dạng, không dạy sự thật.
6. Chất lượng truy hồi giảm khi kho nguồn bị pha loãng bởi tài liệu không liên quan.
7. OCR xử lý công thức và bảng kém hơn xử lý văn bản chữ.

Cần phân biệt rõ hai loại, đừng gộp. Bảy điểm trên là *cơ chế* — hệ quả của cách mô hình hoạt động, nên còn đúng khi công cụ đổi phiên bản, đổi tên, hoặc biến mất. Nhưng có một nhóm khẳng định khác trong tài liệu *không* phải cơ chế, mà là **quan sát thực nghiệm** — tức là phụ thuộc vào từng mô hình và từng phiên bản, và có thể sai:

- "AI hay bỏ sót ý X" (ví dụ bỏ sót điểm hai thành phần huy động ở chuyển vị khác nhau).
- "AI sinh phương án nhiễu bằng cách cộng trừ tùy ý".
- "AI đúng ở bước đầu và bước cuối, sai ở bước trung gian".
- "Lỗi ở bước trung gian khó bị để ý nhất".

Các quan sát này nhất quán với cơ chế, và có căn cứ từ kinh nghiệm dùng thực tế — nhưng chúng *là quan sát, không phải định luật*. Một phiên bản mô hình mới có thể giảm hẳn loại lỗi này (đặc biệt khi mô hình được phép gọi công cụ tính toán bên ngoài). Vì vậy: đừng thiết kế quy trình dựa trên *giả định* rằng một loại lỗi cụ thể sẽ luôn tồn tại. Hãy thiết kế quy trình sao cho *nếu* lỗi đó xảy ra thì cơ chế kiểm bắt được — đó là điều duy nhất không phụ thuộc phiên bản.

Về nguồn dẫn. Tài liệu này không kèm danh mục tài liệu tham khảo vì là bản báo cáo hội thảo nội bộ, không phải bài báo khoa học. Các khẳng định cơ chế ở nhóm trên xuất phát từ cách mô hình ngôn ngữ được xây dựng và huấn luyện (điều có thể tự kiểm bằng tài liệu kỹ thuật của các nhà cung cấp mô hình); các quan sát ở nhóm dưới xuất phát từ kinh nghiệm soạn giảng thực tế của tác giả. Người đọc nên coi nhóm dưới là *giả thuyết cần tự kiểm* trên công việc của chính mình, không phải kết luận đã được đo.

10.2 Những nội dung cần tự kiểm chứng

Các nội dung dưới đây **thay đổi theo thời gian** và không thể xác nhận cho ngày hôm nay. Người đọc cần tự kiểm tra trước khi dùng:

- **Tên gọi và tính năng cụ thể của từng công cụ AI.** Thị trường này đổi tên và gộp tính năng liên tục; một công cụ có thể đã được đổi tên, sáp nhập, hoặc bỏ tính năng đang cần dùng.

- **Giới hạn kỹ thuật:** số tệp tối đa, dung lượng mỗi tệp, số ký tự, kích thước cửa sổ bối cảnh. Luôn kiểm trong trang cấu hình của chính sản phẩm đó.
- **Điều kiện sử dụng của bản miễn phí và bản trả phí.** Thay đổi thường xuyên và không có thông báo trước.
- **Cú pháp cụ thể của từng công cụ trình diễn.** Cú pháp có thể thay đổi giữa các phiên bản chính.
- **Khả năng render công thức của từng bộ công cụ.** Mức hỗ trợ ký hiệu khác nhau giữa các bộ render.

Cách tự kiểm nhanh. Dựng một trang thử tối thiểu (một công thức, một sơ đồ, một bảng) → chạy toàn bộ chuỗi xuất bản trên đó → xác nhận đầu ra đúng. Làm một lần cho mỗi công cụ mới, mất vài phút, nhưng tránh được việc phát hiện lỗi giữa lúc đang soạn hàng trăm trang.

10.3 Ranh giới của tài liệu

Tài liệu này mô tả **phương pháp**. Nó không thay thế:

- Kiến thức chuyên môn của người dạy — không có quy trình nào bù được phần này.
- Việc thẩm định nội dung chuyên môn — tầng 3 luôn là con người.
- Việc tra cứu quy định hiện hành — mọi số liệu phải lấy từ bản gốc còn hiệu lực.

Nói cách khác: tài liệu này giúp dùng AI tốt hơn, *không* giúp hiểu môn học thay.

Hai phạm vi tài liệu này *không* bao gồm — nói rõ để không kỳ vọng sai.

- **Khâu phân phối và chấm bài.** Tài liệu dừng ở việc *soạn ra* đề bài và đáp án. Việc đưa đề tới từng sinh viên, thu bài, và trả phản hồi là việc của hạ tầng lớp học (hệ thống quản lý học tập, hoặc quy trình thủ công) — không nằm trong phạm vi ở đây. Cần lưu ý: nếu mỗi sinh viên một đề mà vẫn chấm tay, khối lượng chấm sẽ *tăng lên*. Giá trị thật của việc cá nhân hóa đề chỉ đến khi có cơ chế chấm tự động hoặc phản hồi tự động.
- **Đo lường chất lượng câu hỏi theo dữ liệu.** Tài liệu hướng dẫn cách *sinh* câu hỏi, không hướng dẫn cách *kiểm định* chúng bằng số liệu thi cử (độ khó, độ phân biệt, tỷ lệ chọn từng phương án nhiễu). Muốn cải tiến ngân hàng câu hỏi qua các học kỳ, phải thu và phân tích dữ liệu bài làm — đó là bước riêng, không có trong tài liệu này.

Nêu hai điểm này không phải để giảm giá trị tài liệu, mà để người áp dụng biết mình cần thêm gì — và biết rằng *thiếu hai khâu đó thì giá trị tài liệu bị chặn trần*.

A. Phụ lục A — Template Hồ sơ Bối cảnh (bản đầy đủ)

Bản này in ra được để dán trực tiếp vào công cụ AI.

```
[MỤC 0 - ĐỊNH VỊ]
Hỗ trợ biên soạn tài liệu giảng dạy cho học phần: [TÊN HỌC PHẦN].
Bạn là thợ viết bản nháp, KHÔNG phải người thẩm định cuối cùng.
Trật tự lập luận bắt buộc:
  Hiện tượng - Cơ chế - Mô hình - Công thức - Điều kiện áp dụng - Ứng dụng
Nếu thiếu căn cứ ở bất kỳ mắt nào: DỪNG LẠI, nêu rõ đang thiếu gì.

[MỤC 1 - ĐỐI TƯỢNG]
Người học đã có nền tảng: [liệt kê kiến thức tiên quyết].
KHÔNG mô tả người học bằng năm học, mô tả bằng nền tảng.

[MỤC 2 - CHUẨN ĐẦU RA]
Các CĐR của học phần: [CĐR1] [CĐR2] …
Mỗi mục nội dung phải gắn với ít nhất một mã CĐR.

[MỤC 3 - MỨC NĂNG LỰC]
Mỗi mục nội dung phải được gán mức:
  [B2-hiểu]   [B3-áp dụng]   [B4-phân tích]
Mọi nội dung ở mức [B3] phải có ít nhất một ví dụ số cụ thể.

[MỤC 4 - MA TRẬN KÝ HIỆU]
[Mỗi dòng: ký hiệu | ý nghĩa | đơn vị]
Quy tắc: một ký hiệu = một đại lượng; một đại lượng = một ký hiệu.
Phân biệt giá trị đặc trưng / tính toán / tiêu chuẩn bằng CHỈ SỐ.
Đầu tài liệu: in lại ma trận. Cuối tài liệu: liệt kê ký hiệu thực dùng + báo cáo sai
lệch.

[MỤC 5 - QUY TẮC TRUNG THỰC]
1. Chỉ dùng thông tin trong tài liệu nguồn. Không dùng kiến thức nền.
2. Mọi khẳng định kỹ thuật phải kèm vị trí trong nguồn.
3. Không có trong nguồn: viết "KHÔNG CÓ TRONG NGUỒN".
4. Không tự tính số. Lập biểu thức, để người dùng chạy script.
5. Cuối mỗi câu trả lời: ghi dòng "Điểm chưa chắc chắn: ..."

[MỤC 6 - ĐẶC TẢ ĐẦU RA]
- Công thức: LaTeX. Mỗi ký hiệu kèm nghĩa + đơn vị.
- Luôn có HAI mục cho mỗi công thức: "Điều kiện áp dụng" và "Dấu hiệu dùng sai".
- Sơ đồ: mã Mermaid hoặc TikZ. Không mô tả bằng lời.
- Quy trình: danh sách có thứ tự. Không viết văn xuôi.
- Bảng cho mọi chỗ có so sánh từ hai đối tượng trở lên.

[MỤC 7 - ĐỊNH DẠNG XUẤT]
Markdown chuẩn. Công thức LaTeX đặt trong dấu đô-la: một cặp đô-la cho công thức
nằm trong dòng, hai cặp đô-la cho công thức đứng riêng thành dòng.
Đánh dấu mức năng lực ngay trong văn bản: [B2], [B3], [B4].
```

B. Phụ lục B — Prompt phản biện (bản đầy đủ)

```
[ĐỊNH VỊ]
Bạn là thẩm định viên độc lập. Bạn KHÔNG viết tài liệu này.
Bạn KHÔNG có lợi ích nào trong việc bảo vệ nó.

[NHIỆM VỤ]
Tìm lỗi trong bản nháp đính kèm.
KHÔNG khen. KHÔNG nhận xét văn phong. KHÔNG đề xuất cải thiện.
Chỉ tìm lỗi và báo cáo.

[TÀI LIỆU PHẢI ĐÍNH KÈM --- BẮT BUỘC]
Bản nháp cần thẩm định.
Ma trận ký hiệu của học phần (để kiểm nhóm 1).
Các đoạn nguồn gốc liên quan (để kiểm nhóm 4).
Nếu thiếu nguồn gốc: bỏ qua nhóm 4 và ghi rõ
"KHÔNG ĐỦ NGUỒN ĐỂ KIỂM DẪN CHIẾU" --- không được đoán.

[ĐỊNH DẠNG ĐẦU RA - BẮT BUỘC]
Lập bảng, mỗi dòng một phát hiện:
| # | Vị trí | Loại lỗi | Mô tả cụ thể | Mức độ |
Mức độ: Cao (làm sai kết quả) / Vừa (gây hiểu nhầm) / Thấp (khó đọc).

[BỐN NHÓM BẮT BUỘC RÀ]
1. LỖI KÝ HIỆU: ký hiệu trái quy tắc ở mục [X].
2. LỖI LOGIC: kết luận không suy ra được từ tiền đề.
3. LỖI ĐIỀU KIỆN ÁP DỤNG: công thức dùng ngoài miền hợp lệ.
4. LỖI DẪN CHIẾU: khẳng định không có trong tài liệu nguồn.

[QUY TẮC BÁO CÁO]
- Nhóm nào không có lỗi: ghi "Đã rà, không phát hiện".
- Không có lỗi nào cả: nói rõ đã rà những gì, theo trình tự nào.
- Không được bịa lỗi để tỏ ra hữu ích.
- Nếu không chắc một chỗ có phải lỗi không: ghi mức độ "Thấp"
  và nêu rõ lý do chưa chắc.
```

C. Phụ lục C — Danh sách kiểm tra nghiệm thu

☐	**Nội dung kiểm tra**

A. Về công thức và số liệu

- ☐ Mọi công thức đều có mục "điều kiện áp dụng" tường minh?
- ☐ Mọi công thức đều có mục "dấu hiệu dùng sai" (dùng ngoài miền)?
- ☐ Mọi ký hiệu khớp ma trận quy tắc, không có ngoại lệ ngầm?
- ☐ Mọi con số *kết quả* đều được sinh từ script, không có số tính tay?
- ☐ Mọi số *đầu vào* đều truy được nguồn?
- ☐ Đã kiểm thứ nguyên hai vế ở mọi công thức?
- ☐ Với các tương quan thực nghiệm (không kiểm được thứ nguyên): đã ghi đơn vị ngầm của hệ số và đánh dấu "kiểm tay"?
- ☐ Đã kiểm cùng đơn vị ở mọi phép cộng và phép trừ?
- ☐ Script đã chạy qua ít nhất 2–3 test đáp án đã biết trước khi sinh đề hàng loạt?
- ☐ Mọi bảng có số liệu đều có dòng nhắc "minh họa, không dùng thiết kế thực tế"?

B. Về câu hỏi và bài tập

- ☐ Mỗi mục nội dung và mỗi câu hỏi gắn được với ít nhất một CĐR?
- ☐ Mỗi câu hỏi chỉ có một đáp án đúng?
- ☐ Không có câu hỏi nào gợi ý đáp án ngay trong đề?
- ☐ Câu hỏi đo đúng mức năng lực đã gán (áp dụng ≠ phân tích)?
- ☐ Phương án nhiễu là những nhầm lẫn *thật* mà sinh viên hay mắc?
- ☐ Mức phân tích đã dùng câu hỏi mở, không dùng bài thay số?
- ☐ Câu hỏi mở có rubric chấm kèm theo?

C. Về nguồn và dẫn chiếu

- ☐ Mọi dẫn chiếu đều trỏ đúng vị trí trong nguồn gốc?
- ☐ Đã đối chiếu công thức và bảng trong bản OCR với bản gốc?
- ☐ Kho nguồn đã tách hẳn khỏi kho cũ (không trộn)?
- ☐ Không có khẳng định nào lấy từ hạng tài liệu diễn giải?

D. Về trình bày và sơ đồ

- ☐ Sơ đồ đã được render và *nhìn hình* xác nhận?
- ☐ Quy trình trình bày dạng danh sách có thứ tự, không văn xuôi?
- ☐ Mọi chỗ so sánh từ hai đối tượng trở lên đã dùng bảng?

E. Về sư phạm

- ☐ Đã gán mức năng lực (hiểu / áp dụng / phân tích) cho từng mục?
- ☐ Mọi mục ở mức áp dụng đều có ví dụ số cụ thể?
- ☐ Mỗi khái niệm mới có ít nhất một điểm neo thực tế hoặc một câu hỏi gợi vấn đề?
- ☐ Ví dụ mẫu giải theo từng bước, có nêu bước kiểm tra kết quả?
- ☐ Đã có ít nhất một câu hỏi mở cho người học tự lập luận?
- ☐ Nội dung phù hợp trình độ thực tế của lớp?

F. Về bảo trì lâu dài

- ☐ Kho nguồn được đặt tên có học phần và ngày tạo?
- ☐ Khi quy định thay đổi, đã tạo kho mới thay vì trộn vào kho cũ?
- ☐ Khi quy định thay đổi, đã lập bảng khác biệt quy định cũ–mới?
- ☐ Tài liệu phát hành có mã phiên bản và ngày?
- ☐ Có kênh để người học báo lỗi và có trang đính chính khi cần?
- ☐ Có ghi lại lý do khi giữ nguyên một ý kiến phản biện?

☐	**Nội dung kiểm tra**

G. Về thẩm định của người dạy (tầng 3)

☐ Đã trả lời bộ rubric tầng 3 (mục 6.5), không bỏ qua câu nào?
☐ Có chỗ nào đúng kỹ thuật nhưng không phục vụ CĐR nào không?
☐ Có chỗ nào vượt quá nền tảng thực tế của lớp?
☐ Có nội dung nào "đúng mà vô nghĩa" — đọc trôi chảy nhưng không dạy được gì?