TỦ SÁCH ROBBIE

Làm việc cùng tác nhân AI

Từ giao việc đến một hệ thống chạy được

- Nguồn và hướng dẫn
- Bản đầu có căn cứ
- Kiểm, sửa và làm tiếp

Thực hành với Codex, Claude Code, Antigravity

Bản biên tập 0.3 • Tháng 11/2026

Mục lục

Lời mở đầu 3

1. Giao một công việc có thể làm tiếp 5

2. Một thư mục đủ để bắt đầu 8

3. Giao việc, đặt quy tắc và chia vai 12

4. Tạo bản đầu có căn cứ và một phép kiểm 16

5. Cho tác nhân làm, xem lỗi rồi sửa 20

6. Tiếp tục qua phiên mới và phối hợp nhiều tác nhân 24

7. Từ bài viết sang video, báo cáo và công cụ nhỏ 29

8. Giữ cách làm có ích qua nhiều lần sử dụng 34

Phụ lục A. Tự tạo bộ thực hành 38

Phụ lục B. Những tên có thể gặp khi dùng công cụ 44

Lời mở đầu

Chiều thứ Sáu, bạn nhờ AI viết báo cáo tuần. Bản đầu khá ổn. Bạn sửa vài con số, thêm một việc còn dở rồi gửi đi. Đến thứ Sáu tuần sau, bạn lại tìm bảng số, mở cuộc trò chuyện cũ và giải thích những điều vừa giải thích tuần trước.

Đó là một việc nhỏ, nhưng làm mãi cũng mệt. Nhiều người đã biết giao việc cho AI vẫn đang tự nối các bước như thế. AI làm giúp từng đoạn; người dùng nhớ chỗ lưu, chuyển tệp và nhắc việc tiếp theo.

Cuốn sách này dành cho lúc bạn muốn giao thêm phần nối ấy. Bạn sẽ dùng một tác nhân AI có khả năng đọc và sửa tệp, viết chương trình, chạy lệnh rồi xem kết quả để làm tiếp. Cách làm này thường được gọi là coding agent (tác nhân có khả năng lập trình). Codex, Claude Code và Antigravity là những môi trường mà bạn có thể gặp khi bắt đầu.

Bạn không cần tự viết được một chương trình hoàn chỉnh trước khi đọc sách. Tuy vậy, bạn sẽ cần hiểu một vài thứ rất cụ thể: dữ liệu nào đang được dùng, chương trình thay đổi tệp nào, phép kiểm đang xác nhận điều gì và bản nào là bản sẽ đưa vào công việc. Chúng ta sẽ tìm hiểu những điều đó qua một dự án có thể mở ra xem, sửa và chạy lại.

Dự án ấy là một xưởng nội dung nhỏ. Đầu tiên chỉ có ba ghi chép và một yêu cầu viết bài. Sau đó, bạn sẽ thêm quy tắc, tạo bản giao việc có căn cứ, cho chạy phép kiểm, xử lý một câu sai nghĩa và thử tiếp tục công việc trong phiên mới. Đến những chương sau, cùng cách tổ chức này sẽ được dùng cho báo cáo số liệu và một công cụ nội bộ.

Một số kinh nghiệm trong sách đến từ cách Robbie đã làm việc với AI: đổi cách phân vai khi hiểu đề bài rõ hơn, giữ lại quyết định để lần sau tiếp tục và thử chuyển ghi chép thành video. Những câu chuyện ấy được kể đầy đủ ngay tại chỗ cần dùng. Các nhóm người, số liệu và tệp trong bài thực hành là tình huống được tạo để học; chúng không mô tả kết quả kinh doanh hay dữ liệu riêng của Robbie.

Trong phần giải thích, sách ưu tiên từ tiếng Việt. Khi một tên tiếng Anh có ích cho việc tra cứu, nghĩa tiếng Việt nằm ngay cạnh. Tên sản phẩm được giữ nguyên. Tên tệp và phần mã cũng được giữ đúng cách viết để bạn có thể dùng lại; nội dung và cách đọc của chúng được giải thích bên cạnh.

Bạn có thể đọc cuốn sách mà chưa mở máy. Tuy nhiên, từ chương 2, việc có một thư mục để làm theo sẽ giúp nhiều chi tiết dễ hiểu hơn. Toàn bộ dữ liệu mẫu và chương trình kiểm cần thiết nằm trong phụ lục. Gói tệp đi kèm giúp bớt việc sao chép, nhưng không có phần kiến thức bắt buộc nào chỉ nằm trong gói đó.

Bạn sẽ làm gì trong lần đọc đầu?

Đọc chương 1 với một công việc đang lặp lại của bạn trong đầu. Ghi ra phần bạn vẫn phải tự chuyển tay giữa các công cụ. Bạn chưa cần chọn ngay phần sẽ tự động hóa; mục đích là nhận ra thao tác nào đang lấy công của mình.

Sang chương 2, mở một thư mục mới và làm theo ba ghi chép mẫu. Giữ dữ liệu ngắn để bạn tự đọc hết. Ở chương 3, chọn đúng một góc viết và lưu bản giao việc. Đến chương 4, bạn có thể xem chương trình kiểm nhận vào gì và trả về điều gì; nếu chưa muốn chép mã, đọc phần giải thích rồi quay lại phụ lục khi mở máy.

Chương 5 sẽ cố tình đưa lỗi vào để bạn xem cách sửa. Khi đã tạo được một bản giao việc đúng dữ kiện, dùng chương 6 để thử đóng phiên và tiếp tục vào lần sau. Đừng thay dữ liệu thật quá sớm trong lần đọc đầu; với mẫu nhỏ, bạn còn nhìn rõ được lỗi đến từ dữ liệu, lời giao hay bộ kiểm.

Nếu đã sử dụng tác nhân thực thi thường xuyên, bạn có thể bắt đầu ở chương 5. Trước khi thử, đọc ba nguồn ở chương 2 và cấu trúc bản giao việc ở chương 4 để biết chính xác phép kiểm đang xét điều gì. Những phần về video và bảng số ở chương 7 giúp bạn chọn hướng mở rộng sau khi đã chạy được lượt đầu.

1. Giao một công việc có thể làm tiếp

Một kho ghi chép thường lớn lên khá tự nhiên. Có ý được viết ngay sau cuộc họp, có đoạn chép từ sổ tay, có bài đã sửa nhiều lần mà tên tệp vẫn là "bản mới". Khi chỉ có vài tệp, người viết còn nhớ. Đến lúc muốn lấy chất liệu cho ba bài trong tuần, thời gian tìm lại có thể dài hơn thời gian viết.

Giả sử bạn đang có ba mươi ghi chép như vậy. Bạn đã từng đưa từng đoạn cho AI và nhận về một bài đọc được. Bây giờ bạn muốn nó tự tìm trong thư mục, đề xuất bài nên viết, giữ lại lý do chọn và cho biết mỗi nhận định dựa vào đâu. Cuối tuần sau, khi thêm hai ghi chép mới, bạn muốn tiếp tục từ kết quả đang có.

Yêu cầu này đưa ta đến một câu hỏi khá thực tế: trong phần việc ấy, bạn đang làm gì chỉ vì công cụ chưa được giao đủ bối cảnh và cách thao tác?

Có thể bạn đang tự mở tệp, chép đoạn cần dùng và đặt lại tên cho bản nháp. Có thể bạn còn phải đọc lại toàn bộ cuộc trò chuyện để nhớ lý do bỏ một ý. Những thao tác này có cơ hội chuyển cho tác nhân thực thi. Nhưng trước khi giao, ta cần biến điều mình vẫn ngầm nhớ thành dữ liệu mà nó đọc được.

Một lần giao việc khác với trước

Bạn tạo thư mục chứa các ghi chép, rồi mở nó trong công cụ đang dùng. Lời giao ban đầu có thể rất ngắn:

> Đọc các ghi chép trong thư mục nguồn. Đề xuất ba bài cho người đi làm đã sử dụng AI. Với mỗi bài, nêu câu hỏi trung tâm, đoạn nguồn sẽ dùng và điều chưa đủ dữ kiện. Lưu kết quả thành một tệp riêng. Giữ nguyên các ghi chép ban đầu.

Lúc này, tác nhân có một việc để thực hiện và một nơi để đặt kết quả. Bạn có thể xem danh sách tệp nó đã đọc, mở bản đề xuất, chỉ vào một dòng và yêu cầu sửa. Sau khi đóng cuộc trò chuyện, tệp vẫn ở đó.

Ở lượt sau, bạn bổ sung rằng hai đề xuất quá giống nhau. Một cách sửa là viết lại cả ba. Cách khác là giữ đề xuất tốt, thay một đề xuất trùng ý và ghi lý do thay. Bạn có thể giao cách thứ hai ngay khi cần. Chưa phải dựng một hệ thống lớn để có lợi ích này.

Robbie từng thay đổi cách phối hợp giữa các AI theo quá trình hiểu công việc. Có lúc một AI giúp diễn đạt yêu cầu để bên khác thực hiện. Khi đã nắm nghiệp vụ đủ rõ, Robbie làm việc trực tiếp với bên triển khai. Kinh nghiệm ấy gợi ra một lựa chọn hữu ích cho dự án mới: bắt đầu bằng số vai vừa đủ, rồi bỏ hoặc thêm vai khi công việc cho thấy cần.

Cần biết bao nhiêu về lập trình?

Với lần thử đầu, bạn nên biết cách mở đúng thư mục, nhận ra tệp mới và xem thay đổi trước khi dùng. Khi tác nhân đề nghị chạy một lệnh, bạn cần hiểu mục đích của lệnh và phạm vi tác động. Nó có thể giải thích trước bằng lời thông thường: lệnh này đọc các tệp nguồn, tính một mã nhận diện cho mỗi tệp và ghi báo cáo vào thư mục kết quả.

Nếu chưa hiểu, bạn có thể yêu cầu nó thử trên một bản sao nhỏ. Đây cũng là cách học kỹ thuật khá gần công việc: gặp đến đâu, làm rõ đến đó. Bạn học ý nghĩa của đường dẫn vì đang cần biết kết quả được lưu ở đâu. Bạn học mã thoát vì muốn phân biệt một lệnh chạy xong với một phép kiểm đạt.

Bạn không phải đọc hết mọi dòng mã ngay lập tức. Tuy vậy, những phần quyết định cách dùng kết quả cần được làm rõ. Nếu báo cáo loại đơn hàng bị hủy khỏi doanh thu, bạn cần biết chương trình nhận diện đơn bị hủy như thế nào. Một phép thử hai dòng dữ liệu có thể giải thích điều ấy dễ hơn cả trang mô tả.

Có lúc bạn sẽ gặp một đoạn chương trình quá khó để tự đánh giá. Khi đó, thu nhỏ chức năng, nhờ giải thích hoặc đưa cho người phù hợp xem là những lựa chọn cụ thể. Chưa hiểu mà vẫn thấy đầu ra đẹp là một trạng thái rất dễ gặp, nhất là khi AI viết nhanh hơn tốc độ bạn đọc.

Có nên bỏ qua các công cụ làm việc khác?

Một tác nhân có thể viết mã để xử lý tài liệu, tạo bảng, dựng bản trình bày hoặc ghép video. Đó là lý do người dùng ngoài ngành phần mềm nên thử môi trường này sớm hơn. Cùng một yêu cầu có thể để lại cả kết quả lẫn công cụ nhỏ dùng cho lần sau.

Nhưng tên "lập trình" chưa quyết định môi trường nào làm việc tốt hơn. Một ứng dụng chuyên xử lý tài liệu có thể đã có sẵn kết nối đến nơi bạn lưu tệp, thư viện tạo bản trình bày và giao diện để sửa trực tiếp. Tác nhân trong một thư mục khác có thể cần bạn thiết lập thêm những phần đó. Ngược lại, khi việc có nhiều bước lặp và cần kiểm bằng chương trình, khả năng viết công cụ ngay trong lúc làm có thể rất đáng giá.

Bạn sẽ thấy khác biệt rõ hơn nếu đem một đầu việc của mình ra so. Giữ nguyên dữ liệu và yêu cầu cuối cùng. Ghi thời gian chuẩn bị, số lần sửa, lỗi còn lại và khả năng dùng lại khi nguồn thay đổi. Nếu chỉ nhìn phút đầu tiên tạo bản nháp, bạn chưa nhìn tới phần việc xảy ra vào tuần sau.

Trong cuốn sách này, ta chọn một việc đủ nhỏ để đọc hết kết quả, nhưng đủ dài để phải giữ trạng thái. Đó là bản giao việc cho một bài viết, được tạo từ ba ghi chép. Đến cuối dự án, bạn sẽ có thể chỉ ra tệp nào chứa nguồn, tệp nào ghi điều đã chọn và phép kiểm nào đã chạy. Từ đó, việc giao thêm một phần cho AI sẽ dễ cân nhắc hơn.

2. Một thư mục đủ để bắt đầu

Trước khi viết thêm quy tắc, hãy chuẩn bị một chỗ làm việc mà bạn tự tìm lại được. Thư mục của dự án có thể nằm ở nơi bạn vẫn lưu công việc. Trong bài thực hành, ta đặt tên nó là xuong_noi_dung.

Workspace (không gian làm việc) là tên thường dùng cho phạm vi mà công cụ mở để xử lý dự án. Trong sách, phạm vi ấy chứa dữ liệu nguồn, hướng dẫn, quy định, chương trình và kết quả. Khi bạn đổi sang dự án khác, cần kiểm xem công cụ đã chuyển sang đúng thư mục chưa. Việc đang mở một cửa sổ mới chưa chắc đã đổi nơi đọc và ghi tệp.

Ba ghi chép đầu tiên

Tạo thư mục nguon rồi lưu ba tệp văn bản dưới đây. Bạn có thể tự làm hoặc nhờ tác nhân tạo đúng nội dung. Các đoạn này là dữ liệu luyện tập; cứ giữ ngắn như vậy để mình còn tự kiểm hết.

Tên tệp thứ nhất là N01.txt:

```
Nhóm đã chuyển năm ghi chép thành hai bản nháp video.
Hai bản nháp chưa được công bố.
Chưa có số liệu người xem.
```

Tên tệp thứ hai là N02.txt:

```
Người phụ trách đã đọc lời thoại và xem ảnh các cảnh.
Người phụ trách chưa nghe toàn bộ âm thanh.
Nhóm muốn ghi riêng từng phần đã kiểm trong báo cáo.
```

Tên tệp thứ ba là N03.txt:

```
Nhóm đang thử phân vai: chọn ý, làm bản đầu và xem lại.
Chưa so thời gian với cách làm cũ.
Cách phân vai còn có thể thay đổi sau lần thử.
```

Ba ghi chép này cho biết khá ít. Ta biết đã có bản nháp, có một lượt xem chưa bao phủ âm thanh và có cách phân vai đang thử. Ta chưa biết video hay đến

đâu, cách mới nhanh hơn bao nhiêu hoặc người xem có thích hay không. Khoảng trống ấy sẽ giúp ta nhận ra một số lỗi ở chương sau.



Hình 1. Dữ liệu nguồn, hướng dẫn và kết quả có chỗ riêng để tìm lại.

Đặt hướng dẫn cạnh dữ liệu

Trong thư mục dự án, thêm ba tệp: DU_AN.md ghi mục tiêu và sản phẩm cần có; QUY_TAC.md ghi các quy ước áp dụng cho nhiều lượt; TRANG_THAI.json ghi công việc đang dở. Đuôi .md là Markdown (văn bản có ký hiệu định dạng đơn giản). Đuôi .json là JSON (dữ liệu có cấu trúc theo trường), thuận tiện cho chương trình đọc.

Bạn có thể bắt đầu bằng nội dung sau trong DU_AN.md:

```
Mục tiêu: chuẩn bị một bài về cách báo rõ phần đã kiểm.
Người đọc: người đi làm đã dùng AI vào công việc.
Nguồn: ba tệp trong thư mục nguon.
Sản phẩm: bản giao việc, bản nháp và ghi chú kiểm tra.
Nơi lưu: mỗi lượt dùng một thư mục trong luot_chay.
```

Tệp này giống một lời nhắc ngắn về dự án. Nó chưa cần chứa mọi quy trình. Nếu bạn đổi người đọc từ người đi làm sang quản lý nhóm, thay đổi ở đây rồi yêu cầu tác nhân xem những phần nào của bản giao việc bị ảnh hưởng.

Đối với QUY_TAC.md, trước mắt chỉ cần hai điều đã sát bài toán: giữ nguyên dữ liệu nguồn trong lượt viết bài; mỗi nhận định về việc đã xảy ra phải chỉ ra đoạn ghi chép hỗ trợ. Những quy ước khác có thể bổ sung khi gặp vấn đề thật.

Tệp trạng thái ban đầu có thể rất ngắn:

```
{
  "cong_viec": "bai-ve-pham-vi-kiem-tra",
  "trang_thai": "chua_lam",
  "buoc_tiep": "doc-nguon-va-de-xuat-goc-viet"
}
```

Các tên trường nghĩa là công việc đang làm, tình trạng hiện tại và bước tiếp theo. Chưa có đường dẫn tới sản phẩm vì sản phẩm chưa được tạo. Đừng ghi trước rằng bản nháp đã hoàn thành chỉ để tệp trạng thái trông đầy đủ.

Nhờ công cụ đọc đúng điểm bắt đầu

Mỗi môi trường có cách nạp hướng dẫn riêng. Trong lần đầu, bạn có thể bỏ qua việc tự động và nói trực tiếp:

> Đọc DU_AN.md, QUY_TAC.md và TRANG_THAI.json trong thư mục đang mở. Cho biết mục tiêu hiện tại và những tệp sẽ dùng. Sau đó đề xuất một góc viết từ ba nguồn. Lưu kết quả của lượt này vào luot_chay/01.

Khi đã thấy cách này hoạt động, bạn có thể đặt một tệp chỉ dẫn đầu vào theo công cụ. Codex thường dùng AGENTS.md (tệp hướng dẫn tác nhân); Claude Code có CLAUDE.md (tệp hướng dẫn dự án cho Claude Code). Antigravity có cơ chế đọc quy tắc dự án. Cách tự động đọc có thể khác giữa phiên bản và bề mặt sử dụng, nên hãy kiểm trong phiên thực tế. Nội dung cốt lõi vẫn nằm ở các tệp tiếng Việt ta vừa tạo.

Một tệp chỉ dẫn đầu vào đủ ngắn có thể ghi: “Trước khi làm, đọc DU_AN.md, QUY_TAC.md và TRANG_THAI.json. Chỉ mở thêm các nguồn liên quan tới nhiệm vụ hiện tại.” Nếu đổi công cụ, bạn chỉ phải đổi cách dẫn vào, còn mục tiêu và trạng thái vẫn dùng tiếp được.

Dữ liệu nguồn và bản đang sửa

Trong thư mục luot_chay/01, tác nhân sẽ tạo những tệp có thể sửa: bản giao việc, báo cáo kiểm và bản nháp. Nguồn ở nguon là đầu vào của lượt này. Cách chia ấy giúp bạn biết chỗ nào được thay khi câu viết sai. Nếu bản nháp nói video đã công bố, ta sửa bản nháp; không sửa ghi chép ban đầu để câu viết trở thành đúng.

Tệp quy tắc mới chỉ hướng dẫn hành vi. Nếu bạn muốn công cụ thực sự không có quyền sửa thư mục nguồn, cần cấu hình quyền ở môi trường chạy. Bài thực hành chưa tự khóa quyền ghi. Vì thế, sau một lượt, việc xem danh sách tệp đã thay đổi vẫn hữu ích.

Khi chuyển sang dữ liệu thật, có thể giữ nguyên cấu trúc này và thay ba ghi chép bằng tài liệu bạn được phép sử dụng. Thư mục dữ liệu thử nên tách khỏi dữ liệu đang dùng hằng ngày. Việc tác nhân có thể đọc một tệp không có nghĩa tệp ấy cần được đưa vào mọi dự án.

Thử lần quay lại đầu tiên

Sau khi nhận đề xuất, bạn chọn một góc viết và nhờ ghi lựa chọn ấy vào trạng thái. Đóng phiên, mở một phiên khác rồi chỉ dẫn về ba tệp đầu vào. Nếu tác nhân tìm được góc đã chọn và tiếp tục đúng bản, thư mục bắt đầu làm được việc mà trước đây bạn phải nhớ.

Nếu nó hỏi lại một điều đã ghi, hãy xem tệp có quá dài, thông tin có nằm ở hai nơi hoặc phiên mới có thực sự đọc được không. Ta sửa đúng chỗ gây vướng. Chưa cần thêm một kho trí nhớ lớn chỉ vì lần đầu quay lại chưa suôn sẻ.

3. Giao việc, đặt quy tắc và chia vai

Với ba ghi chép đã có, tác nhân có thể viết ngay một bài về làm video cùng AI. Bản đầu có lẽ sẽ khá trôi chảy. Nhưng người đọc cần nhận được điều gì từ bài ấy? Họ nên thấy một câu chuyện làm nội dung, một cách báo cáo kết quả hay một hướng dẫn kiểm video?

Bạn cần chọn một hướng trước khi giao phần triển khai. Trong bài thực hành, ta chọn vấn đề rất cụ thể: người nhận báo cáo không biết câu "đã kiểm video" bao gồm những việc gì. Bài sẽ giúp họ ghi rõ đã đọc lời thoại, xem hình hay nghe âm thanh tới đâu.

Một góc viết như vậy có thể dùng cả ba nguồn, nhưng N02 là căn cứ chính. N01 chỉ giúp hiểu nhóm đang có bản nháp. N03 cho biết cách phân vai còn đang thử. Nếu mọi đoạn nguồn được đưa vào bài với trọng lượng như nhau, bài sẽ nhanh chóng tản sang chuyện làm nội dung nói chung.

Một bản giao việc đọc xong biết làm gì

Brief (bản giao việc tóm tắt) là tên bạn thường gặp khi làm nội dung. Trong cuốn này, ta dùng "bản giao việc" và giữ nó đủ ngắn để cả người viết lẫn người xem lại có thể đọc trước khi bắt đầu.

Một bản cho tình huống đang làm có thể như sau:

> Viết cho người đi làm đang dùng AI tạo nội dung. Bài giải thích vì sao cần ghi riêng phần đã đọc, đã xem và đã nghe trong một lượt kiểm video. Dùng tình huống nhóm đã đọc lời thoại, xem ảnh cảnh nhưng chưa nghe hết âm thanh. Người đọc xong có thể viết một báo cáo ngắn nói rõ phần đã kiểm và phần còn thiếu. Chưa có dữ liệu để nói cách làm này tiết kiệm bao nhiêu thời gian.

Đoạn ấy đã xác định người đọc, vấn đề, tình huống và kết quả mong muốn. Tác nhân còn được tự chọn cách mở bài và ví dụ phụ, trong giới hạn dữ liệu.

Nếu bạn muốn một bài dài thay vì đoạn ngắn, thêm độ dài dự kiến và nơi người đọc sẽ dùng bài.

Ở đây, tự do viết và căn cứ dữ kiện được đặt cạnh nhau. Một câu diễn đạt mới có thể không xuất hiện nguyên văn trong ghi chép. Tuy nhiên, một sự việc mới như "video đã đạt hàng nghìn lượt xem" cần căn cứ riêng. Yêu cầu giữ dữ kiện không buộc người viết chép lại cả nguồn.

Phân biệt điều đổi mỗi lượt với điều dùng lâu

Chủ đề tuần này nằm trong nhiệm vụ hiện tại. Quy ước giữ đúng trạng thái chưa công bố có thể dùng cho nhiều tuần, nên đặt vào tệp quy tắc. Cách tìm căn cứ và xuất báo cáo là quy trình có thể dùng lại. Đường dẫn của bản nháp hiện hành nằm trong trạng thái.

Trong các công cụ, bạn sẽ gặp rule (quy tắc), workflow (quy trình thực hiện) và skill (gói hướng dẫn thao tác dùng lại). Chúng thường được trình bày ở những chỗ khác nhau. Ta có thể chọn chỗ đặt bằng câu hỏi khá đơn giản: điều này áp dụng cho riêng bài hôm nay hay cho nhiều nhiệm vụ cùng loại?

Ví dụ, câu "giữ bài dưới 1.000 từ" có thể chỉ đúng với lần đăng này. Câu "không biến đề nghị thành kết quả đã đạt" đúng với nhiều bài dựa trên ghi chép. Quy trình "đọc nguồn, tách nhận định, viết, kiểm" lại mô tả một chuỗi thao tác. Gom cả ba vào một tệp dài khiến việc sửa một nhiệm vụ dễ ảnh hưởng sang các nhiệm vụ khác.

Bạn chưa cần đóng gói thành một kỹ năng có thể cài đặt ngay. Cứ giữ quy trình ở dạng văn bản, dùng vài lần rồi xem đoạn nào thực sự được lặp lại. Một bước chỉ xuất hiện một lần vì lỗi đặc biệt chưa chắc nên thành quy tắc chung.

Quy tắc tốt có một ví dụ để hiểu

Giả sử bản đầu viết: "Nhóm đã hoàn tất kiểm video trước khi công bố." N02 chỉ nói đã đọc lời thoại và xem ảnh, chưa nghe hết âm thanh. Bạn có thể sửa bằng một quy tắc kèm ví dụ:

> Khi ghi chép mới xác nhận một phần công việc, giữ nguyên phạm vi đó. “Đã đọc lời thoại và xem ảnh cảnh” chưa đủ để viết “đã kiểm xong video”. Nếu chưa biết một phần đã làm hay chưa, ghi rõ phần chưa xác định hoặc bỏ kết luận rộng.

Tác nhân nhận được cả điều cần giữ và chỗ hai cách viết khác nhau. Lần sau, khi gặp báo cáo “đã xem tài liệu”, nó có cơ hội hỏi tài liệu nào và phạm vi nào, thay vì chỉ nhớ một câu cấm chung.

Cũng cần xem lại các quy tắc sau một thời gian. Có thể bạn từng yêu cầu mọi bài đều phải có bảng, nhưng giờ đang viết lời kể để nghe. Quy tắc cũ không còn hợp. Nếu tác nhân cứ làm sai định dạng, nguyên nhân có thể nằm ở những chỉ dẫn trái nhau mà bạn đã để tích lũy.

Chia vai để rõ trách nhiệm trong luồng

Một vai nên có đầu vào, đầu ra và quyền thay đổi cụ thể. Với bài hiện tại, có thể chia thành người lập căn cứ, người viết và người xem lại. Một tác nhân vẫn có thể lần lượt làm cả ba, miễn báo cáo nói đúng rằng đây là tự kiểm.

Người lập căn cứ đọc ba ghi chép, viết ra các nhận định có thể dùng cùng đoạn hỗ trợ. Người viết dựa vào bản giao việc ấy để tạo bài. Người xem lại đọc bản nháp và nguồn, tìm câu lệch nghĩa hoặc câu hứa nhiều hơn bài làm được. Họ không cần sửa mọi câu theo sở thích của mình.

Một lời giao cho vai xem lại có thể là:

> Đọc bản nháp cùng ba nguồn. Chỉ ra câu nào diễn giải quá dữ kiện hoặc làm người đọc hiểu sai trạng thái công việc. Với mỗi lỗi, ghi câu đang có, điều nguồn thực sự nói và đề nghị sửa. Tách góp ý về giọng văn khỏi lỗi dữ kiện. Lượt này chưa sửa bản gốc.

Cách giao ấy giúp người viết nhận được việc có thể xử lý. “Bài chưa sâu” hoặc “cần tự nhiên hơn” có thể đúng, nhưng vẫn cần chỉ ra đoạn đang vướng và vì sao. Nếu người xem lại chỉ đổi cả bài thành giọng của họ, bạn sẽ khó biết phần nào là lỗi bắt buộc sửa.

Robbie từng muốn một cộng sự AI có thể bất đồng, với điều kiện phản biện nói rõ căn cứ. Khi đem vào công việc, mong muốn ấy trở thành một yêu cầu rất thực dụng: hãy chỉ ra dữ kiện bị bỏ sót, giả định chưa kiểm hoặc kết quả thử đang khác điều ta mong đợi. Người chủ trì có thể giữ lựa chọn ban đầu sau khi xem phản biện; ít nhất lý do đã rõ hơn.

Quyền thực hiện nằm ở đâu?

"Bạn là người xem lại" chưa tự làm tác nhân mất quyền sửa tệp. "Chỉ đọc nguồn" cũng chưa tự khóa thư mục. Các giới hạn thực sự phụ thuộc vào cấu hình công cụ và môi trường chạy. Khi một bước chỉ cần đọc, có thể cấp quyền đọc nếu công cụ hỗ trợ, hoặc tách bản sao dữ liệu thử để giảm hậu quả của thao tác nhầm.

Tương tự, việc một tác nhân có quyền tạo bản nháp không tự cho nó quyền đăng bài. Phần xuất bản có tài khoản, đối tượng và lịch riêng. Khi chưa cần bước đó, ta giữ gói nháp ở trong thư mục để xem. Điều này cũng làm bài thực hành dễ lặp hơn: bạn có thể sửa nhiều lần mà chưa tạo thêm bài trên kênh thật.

Sau chương này, dự án đã có một góc viết rõ và các quy tắc sát với lỗi có thể gặp. Ở bước tiếp theo, ta sẽ cho bản giao việc một cấu trúc mà chương trình đọc được, rồi dùng nó để tìm một số lỗi trước khi viết dài.

4. Tạo bản đầu có căn cứ và một phép kiểm

Một bản giao việc bằng văn bản đủ cho người viết đọc. Khi muốn kiểm hàng loạt, bạn sẽ có lợi hơn nếu những thông tin cần kiểm nằm ở các trường rõ ràng. Chương trình có thể tìm trường nhận định và căn cứ, thay vì đoán câu nào trong cả trang là câu cần đối chiếu.

Ta vẫn dùng bài về phạm vi kiểm video. Trong bản giao việc, có một nhận định: người phụ trách chưa nghe toàn bộ âm thanh. Căn cứ là câu tương ứng trong N02. Ta sẽ lưu mối liên hệ ấy trong một tệp dữ liệu.

Claim (mệnh đề cần căn cứ) là thuật ngữ thường dùng cho câu nhận định. Evidence (căn cứ hỗ trợ) là thông tin được dùng để xem xét câu ấy. Trong các ví dụ sau, tên trường đã đổi sang tiếng Việt để bạn dễ đọc. Bạn sẽ không cần nhớ hai từ tiếng Anh này mới theo được bài.

Viết ra điều cần được đối chiếu

Tạo ban_giao.json trong thư mục luot_chay/01. Tệp có tiêu đề, góc viết và một danh sách mệnh đề. Mỗi mệnh đề chứa nội dung cùng thông tin về nguồn. Cấu trúc rút gọn trông như sau:

```
{
  "tieu_de": "Ghi rõ phần đã kiểm trong một video",
  "goc_viet": "Phân biệt đọc lời, xem hình và nghe âm thanh",
  "menh_de": [
    {
      "noi_dung": "Người phụ trách chưa nghe hết âm thanh.",
      "can_cu": {
        "ma_nguon": "N02",
        "dau_van": "GIÁ_TRỊ_DO_CHƯƠNG_TRÌNH_TÍNH",
        "trich_doan": "Người phụ trách chưa nghe toàn bộ âm thanh."
      }
    }
  ]
}
```

Dòng dấu vân trong mẫu đang là chỗ giữ chỗ để giải thích cấu trúc. Bản đưa vào phép kiểm phải thay nó bằng giá trị thật do chương trình tính từ tệp N02. Phần phụ lục có một lệnh tạo bản mẫu đúng, nên bạn không phải tự chép chuỗi dài ấy.

Các tên trường đọc gần như câu thường: mệnh đề có nội dung, căn cứ có mã nguồn, dấu vân và trích đoạn. Một chương trình kiểm có thể đi đúng theo cấu trúc đó. Nếu thiếu căn cứ hoặc trích đoạn rỗng, nó trả lỗi trước khi người viết phát triển thành cả bài.

Dấu vân giúp biết đang dùng bản nguồn nào

Hash (mã băm, dùng như dấu vân dữ liệu) là một giá trị tính từ nội dung tệp. Khi tệp thay đổi, giá trị tính lại gần như chắc chắn sẽ khác. Bài thực hành dùng thuật toán SHA-256, một cách tính mã băm phổ biến.

Giả sử hôm nay N02 nói chưa nghe âm thanh. Ngày mai bạn bổ sung rằng nhóm đã nghe lại toàn bộ. Đoạn cũ vẫn còn trong tệp nên phép tìm câu có thể vẫn thấy. Nếu bản giao việc giữ dấu vân cũ, chương trình sẽ biết nguồn đang khác với bản lúc lập căn cứ.

Dấu vân không cho biết tệp nào đúng hơn, cũng không tự quyết định câu trong bài phải sửa ra sao. Nó báo rằng có một thay đổi cần xem. Bạn đọc phần mới rồi chọn giữ, sửa hoặc bỏ nhận định. Với bài chưa công bố, việc này thường dễ hơn nhiều so với tìm lại các kết luận đã đi ra nhiều nơi.

Ở dự án lớn, bạn có thể muốn biết chính xác đoạn nào đổi thay vì chỉ biết toàn tệp đổi. Nhưng phép kiểm theo tệp đã đủ hữu ích cho lần đầu và ít công thiết lập. Khi nó báo quá nhiều thay đổi không liên quan, ta mới có dữ kiện để thu hẹp phạm vi.

Chương trình đang kiểm những gì?

Bộ kiểm trong sách làm bốn việc: đọc cấu trúc dữ liệu, tìm nguồn tương ứng, so dấu vân và tìm trích đoạn trong nguồn. Nó lưu một báo cáo cạnh bản giao việc. Báo cáo có mã nhận diện của bản giao việc đã kiểm, để khi bản giao việc đổi bạn biết báo cáo cũ không còn bao phủ bản mới.

Bạn có thể nhờ tác nhân tạo chương trình theo mô tả ấy. Toàn bộ phiên bản dùng trong sách cũng nằm ở phụ lục. Chương trình chỉ dùng các thư viện có sẵn trong Python, ngôn ngữ lập trình thường dùng để xử lý tệp và dữ liệu. Nó không gọi một mô hình AI và không gửi dữ liệu qua mạng.

Lệnh chạy trong thư mục dự án là:

```
python3 chuong_trinh/kiem.py luot_chay/01/ban_giao.json
```

Ở đây, python3 là chương trình thực thi mã Python; chuong_trinh/kiem.py là tệp chứa bộ kiểm; phần cuối là bản giao việc cần kiểm. Dấu gạch chéo phân chia các thư mục trong đường dẫn. Nếu máy bạn không nhận lệnh python3, cần dùng lệnh Python tương ứng của môi trường đang có.

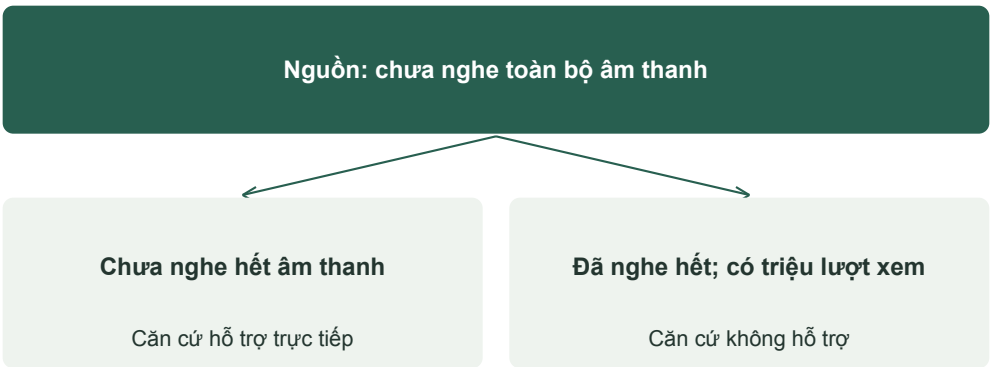
Khi kiểm đạt, báo cáo sẽ ghi kết quả kiểm cơ học là đạt và danh sách lỗi rỗng. Kiểm cơ học ở đây là kiểm cấu trúc và đối chiếu dữ liệu theo quy tắc xác định. Tên gọi được giữ cụ thể như vậy vì chương trình chưa đánh giá được mọi điều người đọc quan tâm.

Theo dõi tác nhân khi chạy lần đầu

Bạn giao nó lập bản giao việc, chạy bộ kiểm và đưa đường dẫn kết quả. Sau đó, mở cả bản giao việc lẫn báo cáo. Nhìn xem câu trong báo cáo có khớp tệp đang đọc, nguồn có đúng N02 và tác nhân có giữ nguyên dữ liệu ban đầu hay không.

Nếu chỉ nhận một tin nhắn “đã xong”, bạn còn thiếu chỗ để xem kết quả. Có thể tác nhân đã tạo tệp thật và chỉ báo quá ngắn. Cũng có thể nó mới viết mô tả dự định. Một yêu cầu tiếp theo rất đơn giản là: “Cho tôi đường dẫn bản giao việc và báo cáo vừa tạo; nêu lệnh đã chạy cùng kết quả trả về.”

Việc nhìn đường đi này đặc biệt hữu ích trong những lượt đầu. Sau khi cách làm ổn định, bạn có thể xem ít chi tiết hơn và tập trung vào những lỗi hoặc thay đổi bất thường. Không nhất thiết bắt mọi lượt đều in toàn bộ nội dung tệp vào cuộc trò chuyện.



Hình 2. Hai câu có thể trỏ cùng một đoạn nguồn. Phải đọc nghĩa mới phân biệt được.

Đọc kết quả đạt cho đúng phạm vi

Giờ hãy đổi nội dung mệnh đề thành: “Nhóm đã nghe toàn bộ âm thanh và video đạt một triệu lượt xem.” Giữ nguyên căn cứ, dấu vân và trích đoạn. Chạy lại phép kiểm.

Bộ kiểm vẫn có thể báo đạt. Mã nguồn tồn tại, dấu vân đúng và câu trích vẫn ở trong N02. Điều sai nằm ở quan hệ giữa câu nhận định và căn cứ: câu mới nói ngược tình trạng âm thanh rồi thêm một kết quả người xem chưa từng có.

Đây là một giới hạn cụ thể mà người dùng nên nhìn thấy sớm. Muốn kiểm nghĩa, phải đọc mệnh đề cùng đoạn nguồn và xét xem điều được nói có thực sự suy ra từ đó hay không. Một tác nhân khác có thể hỗ trợ lượt đọc này, nhưng vẫn cần mô tả tiêu chí và xem chất lượng các phát hiện của nó.

Với bản giao việc ngắn, bạn tự làm được ngay. Đọc câu nhận định, che phần giải thích của tác nhân và tự trả lời: nguồn có nói điều này không, có điều kiện nào bị bỏ, trạng thái có đổi từ đang thử sang đã đạt không? Nếu câu trả lời chưa rõ, thu hẹp câu hoặc quay lại nguồn. Bài viết dài nên được phát triển từ những câu đã đủ căn cứ ấy.

5. Cho tác nhân làm, xem lỗi rồi sửa

Sau lần kiểm đầu tiên, bạn đã có những thứ rời rạc nhưng cần thiết: nguồn, bản giao việc, chương trình và báo cáo. Bây giờ ta nối chúng thành một lượt làm việc mà tác nhân có thể tự tiếp tục khi gặp lỗi thông thường.

Harness (bộ khung điều phối thực thi) là tên gọi cho phần tổ chức việc gọi mô hình, sử dụng công cụ, đưa kết quả trở lại và quyết định bước tiếp theo. Công cụ bạn đang dùng đã cung cấp phần lớn cơ chế này. Trong bài thực hành, ta bổ sung hướng dẫn dự án, bộ kiểm và cách ghi trạng thái. Chưa cần tự viết một chương trình gọi mô hình nhiều lần.

Một loop (vòng thực hiện và nhận phản hồi) hữu ích sẽ có điều mới để dựa vào sau mỗi lượt. Nếu tác nhân sửa bản giao việc vì thiếu trích đoạn, lần kiểm tiếp phải xác nhận lỗi đó đã hết. Nếu nó chỉ viết lại lời giải thích mà tệp vẫn thiếu, công việc chưa tiến ở điểm cần tiến.

Theo một lỗi từ lúc xuất hiện

Giả sử tác nhân tạo căn cứ với mã nguồn N20, trong khi thư mục chỉ có N01, N02 và N03. Bạn cho chạy bộ kiểm. Báo cáo nói không tìm thấy mã nguồn. Đây là lỗi liên kết, chưa phải kết luận rằng nội dung mệnh đề sai.

Tác nhân đọc báo cáo, mở thư mục nguồn, nhận ra tên đúng là N02 rồi sửa mã. Nó cũng cần tính lại dấu vân theo đúng tệp ấy. Sau khi chạy lại, lỗi nguồn không tồn tại biến mất. Lượt sửa có thông tin mới: tên tệp trong thư mục và thông báo từ bộ kiểm.

Nếu nó cứ đổi ngẫu nhiên N20 thành N01, rồi N03, bạn nên dừng kiểu sửa đó. Yêu cầu nó tìm câu trích trong các nguồn, xác định tệp chứa câu và báo cách đã xác định. Thông tin cần bổ sung là nơi câu thực sự xuất hiện, chứ không phải một lời nhắc mạnh hơn rằng hãy cẩn thận.

Đến lỗi thứ hai, câu trích không khớp. Tác nhân đã chép “Người phụ trách chưa nghe hết âm thanh” trong khi N02 ghi “chưa nghe toàn bộ âm thanh”. Ý nghĩa gần nhau nhưng phép kiểm đang tìm đoạn nguyên văn. Tác nhân cần giữ câu diễn giải ở phần nội dung, còn phần trích đoạn lấy đúng chữ trong nguồn.

Hai việc ấy dễ bị lẫn. Nếu bạn muốn câu trích được phép chuẩn hóa khoảng trắng hoặc dấu xuống dòng, đó là một thay đổi trong tiêu chí kiểm. Cần mô tả rõ rồi thử lại các trường hợp liên quan. Không nên âm thầm nới bộ kiểm mỗi khi một bản giao việc bị lỗi.



Hình 3. Kết quả kiểm tạo thông tin cho lượt sửa; bước đọc nghĩa vẫn cần làm riêng.

Giữ bộ kiểm ổn định trong lượt sửa

Tác nhân có thể viết được chương trình kiểm, nên cũng có thể sửa nó. Khả năng này hữu ích khi bạn đang phát triển công cụ. Nhưng trong một lượt chỉ có nhiệm vụ lập bản giao việc, hãy giữ bộ kiểm làm điều kiện đầu vào.

Giả sử báo cáo bắt lỗi dấu vân. Một đường sửa đúng là đọc lại nguồn, xem thay đổi và cập nhật bản giao việc theo quyết định mới. Nếu tác nhân bỏ hẳn

phép so dấu vân để báo cáo chuyển sang đạt, bạn đã mất khả năng phát hiện loại lỗi ban đầu.

Cũng có lúc chương trình thực sự sai. Ví dụ, nó đọc tệp tiếng Việt bằng cách mã hóa không phù hợp và làm hỏng dấu. Khi đó, tách ra một việc sửa bộ kiểm: giữ tệp đầu vào gây lỗi, ghi kết quả mong muốn và thử bản sửa. Sau khi chấp nhận tiêu chí mới, quay lại bản giao việc. Cách tách này giữ được lý do của từng thay đổi.

Cho tác nhân biết khi nào nên dừng

Lời giao cho lượt thực hành có thể như sau:

> Tạo bản giao việc từ ba nguồn rồi chạy phép kiểm. Nếu lỗi nằm ở cấu trúc hoặc thông tin căn cứ, sửa bản giao việc và kiểm lại. Giữ nguồn cùng bộ kiểm không đổi. Sau tối đa ba lượt sửa, nếu còn lỗi thì ghi điều đang thiếu và dừng ở đó. Khi kiểm cơ học đạt, đọc lại nghĩa của từng mệnh đề trước khi báo hoàn tất.

Giới hạn ba lượt giúp mô tả cách làm mong muốn. Nó chưa tự tạo ra bộ đếm bắt buộc ở môi trường chạy. Nếu muốn cho tác nhân làm khi bạn không theo dõi, cần kiểm công cụ có cơ chế giới hạn số lượt, thời gian hoặc chi phí nào và cơ chế ấy đã được áp dụng chưa.

Điều kiện dừng cũng có thể đến từ nghiệp vụ. Nếu bài đòi số liệu người xem mà nguồn ghi chưa có, tác nhân nên trả về phần thiếu. Tiếp tục tìm cách viết cho đủ số đoạn sẽ chỉ làm bài xa nguồn hơn. Một kết quả hữu ích lúc này là bản giao việc đã thu hẹp hoặc một câu hỏi cụ thể cho người giữ dữ liệu.

Đọc lỗi nghĩa cùng tác nhân

Sau khi đạt kiểm cơ học, bản giao việc có thể còn câu: “Cách phân vai mới đã giúp nhóm làm nhanh hơn.” N03 nói chưa so thời gian với cách cũ. Bạn giao tác nhân xem lại riêng câu này cùng nguồn.

Một phản hồi dùng được sẽ giải thích rằng câu đã chuyển một cách thử thành kết luận hiệu quả. Đề nghị sửa có thể là: “Nhóm đang thử tách phần chọn ý,

làm bản đầu và xem lại; hiệu quả thời gian chưa được đo.” Câu sửa giữ thông tin đang có và dành phần kết quả cho một lần đo sau.

Bạn cũng có thể hỏi thêm vì sao nó đã viết câu đầu. Đôi khi tác nhân sẽ nói đã suy từ lợi ích thông thường của phân vai. Lý do đó giúp xác định lỗi: nó mang một kỳ vọng chung áp lên sự việc cụ thể. Quy tắc cần bổ sung là tách kỳ vọng khỏi kết quả đã quan sát, kèm chính cặp câu vừa sửa làm ví dụ.

Không phải phản hồi nào của tác nhân cũng đáng giữ. Nếu nó tự nhận lỗi rồi sửa sang một câu vẫn quá rộng, hãy quay lại đoạn nguồn. Một lời giải thích dài về sự thận trọng chưa thay được việc sửa đúng mệnh đề đang có vấn đề.

Thử làm hỏng để hiểu bộ kiểm

Bạn có thể nhờ tạo bản sao tạm của dữ liệu rồi thử các thay đổi có chủ đích. Đổi mã nguồn thành tệp không có, thay đoạn trích bằng câu tự viết, sửa nội dung nguồn nhưng giữ dấu vân cũ. Mỗi trường hợp cần cho kết quả lỗi tương ứng.

Sau đó giữ nguyên căn cứ và đổi mệnh đề sang ý trái nguồn. Bộ kiểm cơ học sẽ vẫn chấp nhận trường hợp này. Đây là kết quả mong đợi của phép thử giới hạn: ta biết chính xác phần nào cần bước đọc nghĩa.

Nếu một bộ kiểm được quảng bá là bắt lỗi diễn giải, tình huống vừa rồi lại phải bị phát hiện. Muốn đánh giá nó, hãy chuẩn bị một tập câu có đáp án và lý do. Có câu được hỗ trợ trực tiếp, câu thiếu điều kiện, câu nói ngược nguồn và câu chưa đủ dữ liệu để kết luận. Những câu quá dễ chỉ cho biết hệ thống làm được phần dễ.

Ghi lại điều vừa học ở đâu?

Kết quả của lượt này nằm cạnh bản giao việc: báo cáo kiểm, ghi chú tự soát và bản sửa. Một lỗi có thể lặp lại ở nhiều bài thì thêm vào quy trình hoặc quy tắc cùng ví dụ. Một chi tiết riêng của N02 thì để trong ghi chú của lượt này.

6. Tiếp tục qua phiên mới và phối hợp nhiều tác nhân

Bạn đã chọn góc viết, sửa căn cứ và có bản giao việc đạt kiểm cơ học. Đúng lúc ấy cần nghỉ. Nếu ngày mai phải kể lại cả cuộc trao đổi, phần việc giữ bối cảnh vẫn đang nằm ở bạn.

Lúc này, tệp trạng thái bắt đầu có ích. Nó không cần kể lại toàn bộ diễn biến. Nó cần cho người tiếp theo biết mục tiêu hiện tại, bản nào đang dùng, phần nào đã kiểm và việc còn lại. Mỗi thông tin quan trọng nên trỏ được tới một tệp có thể mở.

Bàn giao một lượt chưa xong

Giả sử đã có luot_chay/01/ban_giao.json, nhưng chưa đọc lại nghĩa. Trạng thái có thể ghi:

```
{
  "cong_viec": "bai-ve-pham-vi-kiem-tra",
  "trang_thai": "cho-doc-lai-nghia",
  "san_pham": "luot_chay/01/ban_giao.json",
  "buoc_tiep": "doi-chieu-tung-menh-de-voi-nguon"
}
```

Một ghi chú bằng lời có thể bổ sung: đã kiểm mã nguồn, dấu vân và trích đoạn; cần xem câu nói về hiệu quả phân vai; chưa phát triển thành bài dài vì góc viết còn chờ chốt. Phần “vì sao chưa viết dài” giúp phiên sau hiểu lựa chọn, thay vì coi bản giao việc là việc dang dở cần làm nhanh cho xong.

Robbie từng muốn giữ lại các quyết định cùng nguyên nhân để việc đổi phiên hoặc đổi mô hình không làm mất phần đã hiểu. Ghi một lựa chọn mà thiếu lý do sẽ khó dùng khi hoàn cảnh đổi. Lưu cả cuộc trò chuyện lại có thể khiến người đọc mất lâu để tìm điều cần thiết. Bản bàn giao ngắn dẫn tới chi tiết là một cách cân bằng hai nhu cầu ấy.

Đối chiếu trạng thái với tệp thực tế

Trạng thái có thể cũ. Nếu nó ghi đã kiểm mà báo cáo không tồn tại, phiên sau cần kiểm lại. Nếu báo cáo có nhưng bản giao việc đã được sửa, dấu vân của hai bản sẽ khác. Kết quả kiểm lần trước chỉ áp dụng cho bản cũ.

Một lời giao mở phiên mới nên nhắc thẳng việc đối chiếu:

> Đọc mục tiêu và trạng thái dự án. Mở các tệp được dẫn để xác nhận chúng đang tồn tại và thuộc đúng phiên bản. Tiếp tục từ bước chưa có bằng chứng hoàn thành. Nếu trạng thái mâu thuẫn với tệp, ghi lại chỗ mâu thuẫn rồi xử lý bước còn thiếu.

Bài thử này giúp phát hiện nhiều vấn đề mà việc ngồi nhìn cấu trúc thư mục không cho thấy. Có thể đường dẫn viết sai, tên bản mới nhất không rõ hoặc thông tin quyết định đang nằm trong một đoạn dài. Sửa theo lần quay lại thực tế sẽ tiết kiệm công hơn cố đoán mọi thứ ngay từ đầu.

Dùng bối cảnh vừa đủ cho việc trước mắt

Một tác nhân viết bài về báo cáo kiểm video chưa cần đọc toàn bộ lịch sử cá nhân, dự án giao dịch và cấu hình website của bạn. Context (ngữ cảnh làm việc) có giới hạn; dữ liệu không liên quan còn làm phần cần dùng khó được chú ý hơn.

Hãy để điểm vào dẫn tới mục tiêu và trạng thái. Khi nhiệm vụ cần viết, mở thêm quy trình biên tập. Khi cần kiểm dữ kiện, mở nguồn. Một tóm tắt có thể giúp chọn đúng tài liệu, nhưng nhận định phụ thuộc ngoại lệ cụ thể vẫn cần quay lại đoạn gốc.

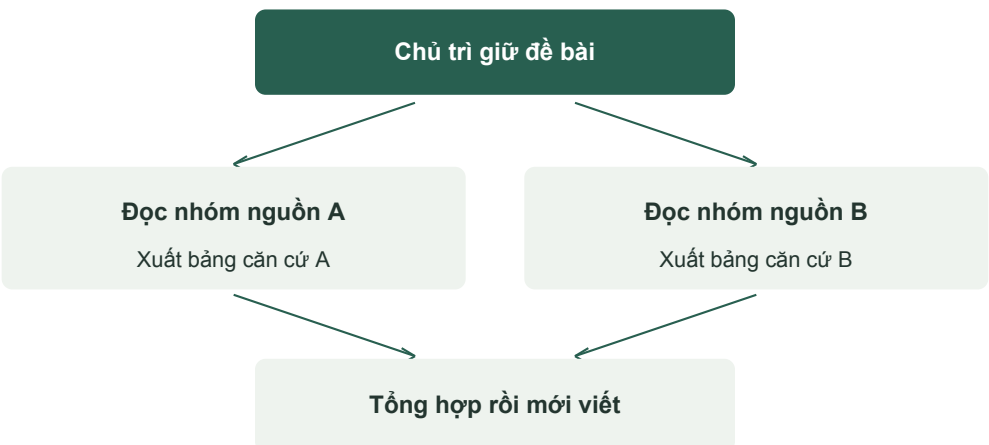
Cách tổ chức này cũng giúp khi đổi công cụ. Thông tin quyết định nằm trong tệp dự án, đọc được ngoài một cuộc trò chuyện. Bộ nhớ của ứng dụng có thể giảm công nhắc lại, nhưng bạn vẫn có đường riêng để xem bản đang dùng và lý do đã chọn.

Khi nào nên mở thêm một tác nhân?

Bạn có thể đang muốn một bên nghiên cứu, một bên viết và một bên phản biện cùng lúc. Trước khi mở, nhìn vào quan hệ phụ thuộc. Người viết cần bản giao việc đã chốt. Người xem lại cần bản nháp và nguồn. Nếu cả ba bắt đầu từ một yêu cầu rộng, họ có thể hiểu ra ba bài khác nhau.

Multi-agent (phối hợp nhiều tác nhân) có ích khi có công việc đủ tách biệt. Chẳng hạn, một bên đọc nhóm ghi chép về dữ liệu, bên khác đọc nhóm ghi chép về làm video. Cả hai xuất cùng mẫu: câu hỏi đáng viết, nhận định có căn cứ và phần chưa rõ. Người chủ trì so hai kết quả rồi chọn bài.

Nếu chỉ có ba tệp ngắn, một tác nhân làm lần lượt thường dễ theo dõi hơn. Lợi ích của thêm người xem cần được so với công đọc kết quả, xử lý khác biệt và chờ nhánh chậm. Số phiên mở ra chưa nói được gói cuối đã tốt hơn bao nhiêu.



Hình 4. Tách phần độc lập; đầu ra của các nhánh cùng một mẫu.

Đừng để hai bên cùng sửa một bản

Giả sử người viết đang sửa mở bài, người xem lại cũng viết đè vào chính tệp ấy. Bạn có thể mất thay đổi hoặc không biết câu mới đến từ ai. Trong lượt đầu, cách đơn giản là giữ bản nháp cho người viết và yêu cầu người xem lại tạo tệp nhận xét riêng.

Mỗi nhận xét nên có vị trí, vấn đề và cách sửa đề nghị. Người viết xử lý từng mục rồi ghi điều đã đổi. Nếu không đồng ý, giữ lý do để chủ trì quyết định. Khi công việc đòi sửa đồng thời, có thể dùng nhánh riêng trong Git (công cụ lưu lịch sử thay đổi) rồi hợp nhất, nhưng chi phí ấy chỉ đáng trả khi có nhu cầu thật.

Người chủ trì cũng cần giữ một nơi ghi trạng thái chung. Các nhánh không tự sửa mục tiêu của nhau. Khi một nguồn thay đổi, chủ trì xác định nhánh nào bị ảnh hưởng và yêu cầu xem lại phần đó. Nếu mỗi bên có một bản trạng thái tự gọi là hiện hành, việc tổng hợp sẽ nhanh chóng trở thành một nhiệm vụ riêng.

Phiên xem lại có thể bỏ sót điều gì?

Một phiên tách khỏi người viết tạo khoảng cách hữu ích, nhưng vẫn có thể cùng hiểu sai một nguồn. Vì vậy, hãy đánh giá lượt xem bằng phát hiện cụ thể. Nó có tìm ra câu trái N02 không? Có chỉ đúng điều kiện bị bỏ trong N03 không? Có đưa cả những góp ý chỉ là sở thích thành lỗi bắt buộc sửa không?

Bạn có thể dùng câu sai nghĩa ở chương trước để thử người xem lại. Nếu nó chấp nhận câu “đã nghe toàn bộ âm thanh” khi nguồn nói chưa nghe, cần sửa cách giao hoặc cách đọc nguồn trước khi tin nó trên hàng chục bài. Một lời nhận xét dài và lịch sự vẫn có thể không làm được việc bạn cần.

Chạy lại khi có hành động bên ngoài

Trong bài thực hành, các lượt chủ yếu đọc và ghi tệp nên dễ làm lại. Khi nối tới gửi thư, đăng bài hoặc cập nhật một hệ thống dùng chung, một lần chạy lại có thể tạo hành động thứ hai.

Ví dụ, tác nhân đã gửi bài nhưng phiên dừng trước khi ghi trạng thái. Phiên mới thấy chưa có dấu “đã gửi” và gửi thêm lần nữa. Cách xử lý cần dựa vào việc kiểm kết quả ở nơi nhận, dùng mã định danh của lần thực hiện và cơ chế chống trùng nếu dịch vụ hỗ trợ. Chỉ thêm một câu nhắc trong tệp trạng thái không giải quyết được khoảng dừng giữa hai thao tác.

Ta chưa thêm bước gửi vào dự án này. Gói kết quả sẽ dừng ở bản nháp và báo cáo để xem. Khi mở rộng sau này, bạn đã biết hành động bên ngoài cần được thiết kế như một bước riêng, có cách xác nhận thành công và cách tiếp tục sau lỗi.

7. Từ bài viết sang video, báo cáo và công cụ nhỏ

Đến đây, dự án đã có thể tạo một bản giao việc kèm căn cứ và giữ trạng thái qua phiên. Bạn có thể dùng nó để viết bài. Nhưng nếu mục tiêu chỉ là tự động tạo nhiều bài hơn, phần chọn ý và xem lại rất dễ bị ngập.

Ta thử phát triển một góc đã chốt thành gói nhỏ: một bài đọc và một đoạn video ngắn. Cùng lúc, hãy giữ được mối liên hệ với nguồn, để nếu N02 thay đổi thì biết cần xem lại phần nào.

Viết bài từ một tình huống đủ rõ

Lời mở có thể bắt đầu bằng chính báo cáo gây hiểu nhầm:

> Bạn nhận được một tin nhắn: “Đã kiểm video, có thể gửi.” Bạn tưởng người làm đã xem và nghe toàn bộ. Khi mở lên, phần lời đọc bị hụt một câu. Hỏi lại mới biết người ấy đã đọc kịch bản và xem ảnh các cảnh, chưa nghe bản âm thanh cuối.

Đây là một tình huống minh họa để người đọc nhận ra vấn đề. Nó không phải diễn biến mới được xác nhận từ N02. Trong bản giao việc, hãy cho phép tạo tình huống giả định và yêu cầu giữ rõ cách kể ấy, tránh dùng ngôi tác giả như thể chính mình đã gặp.

Từ mở bài đó, phần giải thích có thể đi vào ba công việc khác nhau: đọc lời giúp tìm lỗi nội dung; xem hình giúp kiểm bố cục; nghe toàn bộ giúp phát hiện lời bị hụt hoặc nhịp chưa phù hợp. Cuối bài, đưa một mẫu báo cáo mà người đọc có thể sửa cho việc của họ.

Mẫu ấy có thể là: “Đã đọc kịch bản và xem ảnh từng cảnh. Chưa nghe toàn bộ bản âm thanh cuối. Cần nghe lại trước khi gửi.” Nó giúp người nhận biết bước tiếp theo. Nếu sau đó đã nghe đủ, cập nhật phạm vi và kết quả, giữ đúng bản video được nghe.

Tác nhân có thể viết nhiều cách mở bài khác nhau. Bạn nên chọn theo sự rõ ràng và độ khớp với người đọc, thay vì chỉ chọn câu nghe gây chú ý. Khi mở bài kể một sự cố mà phần sau không giải quyết, người đọc dễ thấy bị dẫn sang chuyện khác.

Chuyển thành lời kể và hình minh họa

Một bài viết có thể dừng lâu ở một câu. Video cần người xem theo kịp trong thời gian thực. Bạn có thể nhờ tác nhân viết lời kể từ bài đã chốt, rồi đọc thành tiếng để xem câu nào quá dài hoặc thiếu chủ ngữ.

Đối với tình huống này, ba cảnh đã đủ. Cảnh đầu là màn hình báo cáo "đã kiểm video". Cảnh tiếp theo mở ra các việc đã làm: đọc lời và xem ảnh. Cảnh cuối là báo cáo được viết lại, nêu rõ còn phần âm thanh. Người xem nhìn được sự thay đổi ngay trên nội dung, thay vì chỉ nghe một lời khuyên về sự rõ ràng.

Storyboard (bảng phân cảnh) là tên thường dùng cho bản mô tả cảnh trước khi dựng. Trong bảng này, nên có điều người xem cần hiểu, lời sẽ nghe và hình sẽ thấy. Nếu ba cột ấy chỉ diễn đạt lại cùng một câu, hình có thể chưa giúp thêm điều gì.

Bạn có thể giao như sau:

> Tạo bảng ba cảnh từ bài đã chốt. Mỗi cảnh ghi ý cần hiểu, lời kể và hình minh họa. Hình ưu tiên báo cáo trước và sau khi sửa. Đừng dùng ảnh nhân vật như thể họ là người thật trong tình huống. Giữ lời đủ ngắn để đọc thử tự nhiên.

Chưa cần thêm số giây chính xác trước khi có bản đọc. Thời lượng dự kiến là ước lượng; khi có giọng thực tế mới tính lại độ dài cảnh và phụ đề. Một câu dài hơn dự kiến có thể buộc bạn đổi cả nhịp dựng.

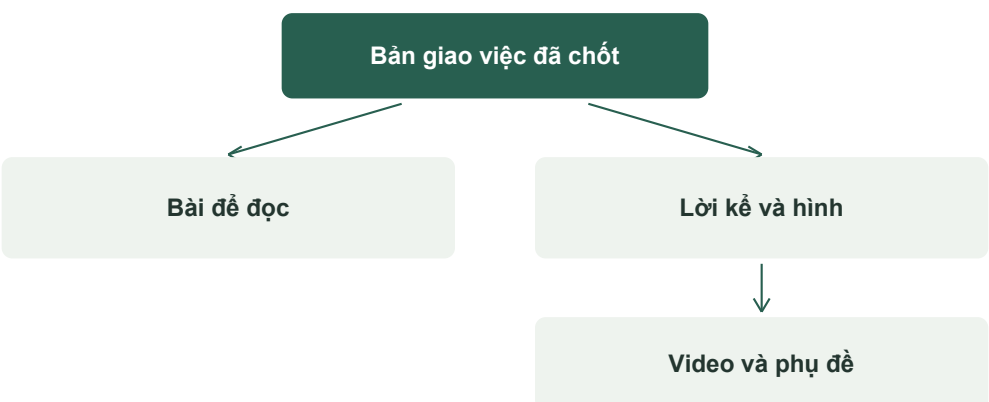
Tác nhân giúp ở khâu sản xuất như thế nào?

Nếu môi trường có công cụ phù hợp, tác nhân có thể tạo hình, gọi dịch vụ giọng đọc, ghép cảnh và xuất phụ đề. Một số thao tác được viết thành chương trình để lần sau dùng lại. Các công cụ tạo hình và âm thanh cần được kết nối riêng; việc tác nhân viết được lệnh gọi chưa chứng minh tài khoản đã dùng được dịch vụ.

Khi chương trình dựng báo thành công, mở sản phẩm để xem. Mã thoát bằng không chỉ cho biết lệnh đã kết thúc theo cách chương trình coi là thành công. Nó chưa nói chữ có bị che, lời có bị hụt hoặc hình có đúng ý bài.

Contact sheet (bảng ghép các khung hình đại diện) giúp xem nhanh bố cục nhiều cảnh. Nó không cho biết toàn bộ nhịp chuyển và âm thanh. Bạn có thể nhờ tác nhân xuất bảng hình để kiểm chữ, rồi xem và nghe bản video hoàn chỉnh cho các phần còn lại.

Robbie từng thử chuyển ghi chép sang video và nhận ra có một bản trước mặt giúp cuộc trao đổi cụ thể hơn: đoạn nào kể chưa rõ, hình nào chưa thêm ý, nhịp nào cần sửa. Các bản đầu còn thô. Kinh nghiệm có thể dùng lại là dựng một đoạn mẫu để xem sớm, trước khi tạo hàng loạt cảnh theo một hướng chưa hợp.



Hình 5. Các định dạng dùng chung góc viết, nhưng cần cách kiểm phù hợp từng loại.

Thay nội dung bằng số liệu

Bây giờ thử chuyển cách tổ chức sang báo cáo tuần. Dữ liệu mới là một bảng đơn hàng. Quy tắc của bài tập: chỉ tính doanh thu của đơn hoàn tất; bản ghi trùng hoàn toàn chỉ tính một lần; cùng mã nhưng khác thông tin phải báo để xem, không tự chọn một bản.

CSV (tệp bảng phân tách cột bằng dấu phẩy) dưới đây có thể lưu thành don_hang.csv. Đơn vị tiền trong toàn bảng là đồng. Giá trị số để nguyên, không thêm dấu ngăn hàng nghìn.

```
ma,thang,trang_thai,so_tien
A01,1,hoan_tat,1000000
A02,1,hoan_tat,600000
A03,1,da_huy,300000
A04,2,hoan_tat,1500000
A05,2,hoan_tat,900000
A02,1,hoan_tat,600000
A06,2,dang_xu_ly,400000
```

Bạn có thể tự tính trước: tháng 1 có 1.600.000 đồng, tháng 2 có 2.400.000 đồng. Tổng hai tháng là 4.000.000 đồng. Bản A02 lặp lại không được cộng thêm, đơn hủy A03 và đơn đang xử lý A06 chưa được tính. Theo quy tắc này, doanh thu tháng 2 tăng 50% so với tháng 1.

Giao tác nhân viết chương trình đọc bảng, áp dụng quy tắc rồi xuất số tổng hợp cùng danh sách dòng bị loại. Điều đáng xem không chỉ là tổng cuối. Bạn muốn biết nó đã bỏ dòng nào và theo điều kiện nào, để một kết quả tình cờ đúng không che cách làm sai.

Thử sửa dòng A02 thứ hai thành 650.000. Hai dòng cùng mã đã mâu thuẫn. Theo quy tắc, chương trình phải báo phần cần xử lý. Nếu nó tự lấy dòng cuối hoặc lấy số lớn hơn, bạn đã tìm được một chỗ hiểu sai trước khi dùng bảng dài.

Cùng cấu trúc dự án vẫn dùng được: tài liệu mục tiêu ghi báo cáo cần có, tệp quy tắc ghi cách tính, thư mục nguồn chứa bảng và mỗi lượt có báo cáo riêng.

Bộ kiểm đổi theo nghiệp vụ; phép tìm trích đoạn của bài viết không phù hợp để xác nhận tổng doanh thu.

Phát triển thành một công cụ nhỏ

Khi cách tính đã ổn trên vài trường hợp, bạn có thể muốn một biểu mẫu để tải bảng và nhận kết quả. Tác nhân có thể dựng giao diện ấy, nhưng phần giao diện chưa nên thay đổi cách tính. Giữ các ví dụ đã biết đáp án để chạy lại sau mỗi lần sửa.

Một lỗi thường xuất hiện khi công cụ được dùng bởi người khác: họ đưa bảng có tên cột khác, có ô trống hoặc dùng dấu chấm trong số tiền. Những trường hợp này cần được hỏi và xử lý theo cách đã thống nhất. Việc tự đoán mọi định dạng có thể tạo lỗi khó thấy hơn một thông báo rõ rằng đầu vào chưa phù hợp.

Bạn không cần biến cả xưởng nội dung thành phần mềm quản lý doanh nghiệp. Khi một đoạn chương trình đã giải quyết được phần lặp, hãy dùng nó vài lần. Nhu cầu tiếp theo sẽ cụ thể hơn: cần tải nhiều tệp cùng lúc, cần giữ danh sách dòng lỗi hay cần người khác tiếp tục mà không nhờ bạn mở máy. Mỗi nhu cầu ấy dẫn tới một thay đổi có thể kiểm được.

8. Giữ cách làm có ích qua nhiều lần sử dụng

Một tác nhân có thể làm ra bản đầu rất nhanh. Sau vài tuần, câu hỏi sẽ chuyển sang việc khác: bạn có bớt công chưa, công cụ có chạy lại được không và những lỗi cũ có quay lại không? Đó là lúc nên nhìn cả quá trình, từ chuẩn bị dữ liệu đến bản thật sự được dùng.

Có những việc đáng làm dù chưa tiết kiệm thời gian. Bạn có thể dùng AI để thử một hình thức nội dung trước đây chưa làm được, hoặc hiểu một phần kỹ thuật đang cần. Chỉ cần gọi đúng mục đích ấy khi đánh giá. Chi phí học không nhất thiết là thất bại; nó cũng chưa tự trở thành số phút đã tiết kiệm.

Ghi tổng công của một lượt

Trong một lượt làm bài, bạn có thể mất mười phút chọn nguồn, năm phút giao việc, hai mươi phút đọc và sửa. Tác nhân chạy thêm mười phút mà bạn không phải theo dõi toàn bộ. Ghi riêng thời gian người dùng bỏ công và thời gian hệ thống chạy sẽ dễ hiểu hơn cộng tất cả thành một con số duy nhất.

Đừng quên phần dựng ban đầu: tạo thư mục, viết quy tắc, sửa bộ kiểm và thử lại. Những phần ấy có thể dùng cho nhiều lượt, nên cần xem sau một khoảng đủ dài. Nếu đầu việc chỉ làm một lần, công thiết lập có thể không được hoàn lại.

Giả sử cách cũ cần 60 phút cho một gói nội dung. Cách mới cần 120 phút chuẩn bị lần đầu, rồi 30 phút công của bạn cho mỗi gói, gồm kiểm và sửa. Nếu chất lượng, độ khó tương đương và chưa tính bảo trì, bốn gói là điểm hai cách dùng cùng lượng thời gian: 240 phút. Từ gói sau, chênh lệch bắt đầu có lợi cho cách mới.

Đây là phép tính minh họa. Nếu mỗi gói mới lại phải sửa bộ kiểm, hoặc nguồn khó hơn, kết quả sẽ khác. Chi phí dịch vụ và thời gian học thêm cũng có thể ảnh hưởng quyết định. Ghi điều kiện bên cạnh số đo để sau này không đem một kết quả của bài dễ áp cho mọi việc.

Khi bản nháp trở thành hàng đợi

Bạn có thể tạo mười bài trong khi chỉ đủ sức xem hai. Thêm lượt viết sẽ không làm số bài dùng được tăng tương ứng. Chỗ cần sửa có thể là bước chọn ý hoặc tiêu chuẩn xem lại.

Với kho nội dung, một cách thử là lập đề xuất ngắn trước. Bạn chọn ba ý đáng phát triển, tác nhân mới viết dài. Một cách khác là kiểm căn cứ ngay sau bản giao việc để tránh sửa sai nghĩa ở cả bài, lời kể và video. Hai lựa chọn này giảm công làm lại theo những cách khác nhau; nên thử trên chính luồng đang vướng.

Bạn cũng có thể giảm số định dạng của một gói. Một ý đủ cho bài đọc chưa chắc đã cần video. Chọn hình thức theo điều người đọc cần sẽ giúp gói nhỏ hơn và dễ hoàn thành hơn. Phần chưa triển khai được giữ lại với lý do, thay vì tạo thêm tệp nháp khó tìm.

Đóng gói phần đã dùng được

Sau vài lượt, bạn sẽ thấy những đoạn hướng dẫn lặp lại: cách đọc nguồn, cách ghi căn cứ, lệnh kiểm và cách ghi nhận xét. Đó là lúc phù hợp để đóng gói thành một kỹ năng dùng lại. Giữ cả ví dụ khó như mệnh đề sai nghĩa nhưng trích đoạn đúng, vì nó cho lần sau biết giới hạn cần chú ý.

Một gói thao tác nên nói khi nào dùng, cần đầu vào gì, xuất kết quả ở đâu và kiểm bằng cách nào. Chủ đề hôm nay và mã lượt chạy tiếp tục nằm trong nhiệm vụ cụ thể. Khi thử gói mới, dùng lại một bài cũ để xem tác nhân có tạo đúng sản phẩm mà không phải nhắc nhiều lần không.

Nếu lần nào cũng phải sửa cả gói, hãy xem phạm vi có quá rộng. Quy trình nghiên cứu nguồn và quy trình dựng video có vài điểm nối, nhưng không nhất thiết phải được đọc trọn vẹn trong mọi nhiệm vụ. Một chương trình tính dấu vân có thể đứng riêng, không cần gói hướng dẫn dài để chạy.

Đổi công cụ mà không đánh mất dự án

Có thể bạn muốn thử cùng việc trong Codex, Claude Code rồi Antigravity. Trước khi chuyển, giữ bản mục tiêu, quy tắc, dữ liệu, chương trình và trạng thái ở những tệp có thể đọc bình thường. Ghi thêm môi trường đã dùng để biết một lệnh phụ thuộc vào thư viện nào.

MCP (giao thức kết nối mô hình với công cụ và dữ liệu) là một cách các môi trường có thể cung cấp tích hợp. Có tích hợp trong công cụ này chưa làm công cụ khác tự có quyền đọc cùng tài khoản. Khi chuyển, kiểm lại kết nối nào dùng được và quyền nào đã được cấp.

Tương tự, sandbox (môi trường thực thi được giới hạn quyền) thuộc cách chạy của từng công cụ. Một tệp quy tắc viết chỉ đọc không thay được giới hạn quyền ghi. Những bước tạo tệp trong dự án có thể dễ mang sang, còn bước truy cập dịch vụ bên ngoài thường cần thiết lập và xác nhận lại.

Tên mô hình cũng chưa đủ để biết môi trường có chạy được dự án. Nếu muốn dùng DeepSeek hay nhà cung cấp khác, kiểm khả năng đọc và ghi tệp, chạy chương trình, trả kết quả quan sát và giữ trạng thái. Đó là những điều bài thực hành cần. Một giao diện chỉ trả lời văn bản có thể chưa cung cấp đủ, dù mô hình vẫn hiểu bài toán.

Chọn phần kỹ thuật đáng học tiếp

Khi làm vài lượt, bạn sẽ biết phần nào thường xuyên khiến mình phải chờ. Nếu hay vướng đường dẫn và bản mới nhất, học kỹ hơn cách tổ chức tệp và lịch sử thay đổi. Nếu báo cáo sai do dữ liệu đầu vào, dành thời gian hiểu cấu trúc bảng và cách kiểm từng trường. Nếu video khó xem, học cách đọc lời, bố cục và thời lượng sẽ gần nhu cầu hơn học thêm cấu hình mô hình.

Robbie từng đặt câu hỏi liệu sự hiểu biết của mình có đi cùng tốc độ tạo sản phẩm hay không. Một cách tiếp cận là chọn đúng phần ảnh hưởng tới việc chấp nhận kết quả: vì sao tin con số này, vì sao dùng câu kết luận đó, vì sao bước gửi chỉ được thực hiện một lần. Bạn có thể làm rõ từng phần qua một ví dụ nhỏ và một lần thử có thể quan sát.

Thành thạo một công cụ không đòi bạn nhớ mọi tùy chọn. Bạn cần tìm lại đúng thông tin khi cần và đủ hiểu để đặt câu hỏi tiếp. Sau một thời gian, chính các lỗi đã xử lý sẽ thành những phần hướng dẫn sát công việc nhất trong kho của bạn.

Khép lại lần thử đầu

Trở lại thư mục xưởng nội dung. Một gói đầu tiên hoàn chỉnh có bản giao việc, bài viết, báo cáo kiểm cơ học, ghi chú đọc lại nghĩa và trạng thái chỉ rõ bản đang dùng. Nếu đã làm video, thêm bản dựng cùng phạm vi đã xem và nghe.

Mở một phiên mới rồi nhờ tác nhân chuẩn bị gói tiếp theo từ dữ liệu mới, dùng lại cách làm đã có. Xem nó tiếp tục được phần nào, chỗ nào vẫn cần bạn giải thích và lỗi nào xuất hiện lần đầu. Ghi một thay đổi cụ thể cho lượt sau.

Bạn có thể dừng ở một xưởng nhỏ đủ dùng. Khi việc quay lại hằng tuần bớt phải bắt đầu từ đầu, những tệp và chương trình đã tạo đang làm đúng công việc của chúng. Nếu sau vài lượt công kiểm vẫn cao hơn cách cũ, bản ghi sẽ giúp bạn biết nên thu hẹp, đổi công cụ hay dừng phần tự động hóa nào.

Phụ lục A. Tự tạo bộ thực hành

Phần này chứa toàn bộ chương trình để tạo dữ liệu, kiểm bản giao việc và thử lỗi. Bạn có thể đọc mã để hiểu hoặc giao tác nhân tạo các tệp theo nội dung trong sách. Máy cần Python từ bản 3.9 trở lên; các chương trình dùng thư viện chuẩn có sẵn.

Nếu đã tạo ba nguồn ở chương 2, bạn có thể giữ nguyên chúng và chỉ thêm bộ kiểm. Chương trình khởi tạo bên dưới dành cho một thư mục chưa có nguồn hoặc kết quả; nó sẽ dừng khi thấy dữ liệu đã tồn tại để tránh ghi đè.

Chuẩn bị một thư mục mới

Tạo thư mục xuong_noi_dung, bên trong có thư mục chuong_trinh. Lưu ba đoạn mã ở các mục tiếp theo thành đúng tên tệp được ghi. Các đường dẫn tính từ thư mục xuong_noi_dung, nên mở cửa sổ lệnh tại đó trước khi chạy.

Nếu dùng gói tệp đi kèm, ba chương trình cùng các tệp hướng dẫn đã có sẵn. Dữ liệu mẫu cũng đã được tạo. Bạn chạy lệnh kiểm và thử lỗi; không cần chạy lại khởi tạo trong thư mục đã có dữ liệu.

Python có các từ khóa bắt buộc như import (nạp thư viện), def (khai báo hàm) và return (trả kết quả). Phần mã được giữ nguyên để có thể chạy; không thay từ khóa bằng bản dịch. Tên biến của bộ mẫu dùng tiếng Việt không dấu. Các mục phía dưới giải thích chương trình làm gì để bạn đọc theo cả khi chưa quen cú pháp.

Tệp tạo dữ liệu ban đầu

Lưu đoạn sau thành chuong_trinh/khoi_tao.py. Nó tạo ba nguồn giống chương 2, tính dấu vân thật của N02 và điền vào bản giao việc mẫu. Chương trình không tự viết bài hay đánh giá nội dung.

```
"""Tạo dữ liệu mẫu trong một thư mục chưa khởi tạo."""
import hashlib
import json
from pathlib import Path
```

```
GOC = Path(__file__).resolve().parents[1]
# Dừng nếu có dữ liệu để tránh ghi đè công việc đang có.
if (GOC / 'nguon').exists() or (GOC / 'luot_chay').exists():
    raise SystemExit('Đã có dữ liệu; giữ nguyên và dùng thư mục mới.')

(GOC / 'nguon').mkdir()
dich = GOC / 'luot_chay' / '01'
dich.mkdir(parents=True)
cac_nguon = {
    'N01': ('Nhóm đã chuyển năm ghi chép thành hai bản nháp video.\n'
            'Hai bản nháp chưa được công bố.\n'
            'Chưa có số liệu người xem.\n'),
    'N02': ('Người phụ trách đã đọc lời thoại và xem ảnh các cảnh.\n'
            'Người phụ trách chưa nghe toàn bộ âm thanh.\n'
            'Nhóm muốn ghi riêng từng phần đã kiểm trong báo cáo.\n'),
    'N03': ('Nhóm đang thử phân vai: chọn ý, làm bản đầu và xem lại.\n'
            'Chưa so thời gian với cách làm cũ.\n'
            'Cách phân vai còn có thể thay đổi sau lần thử.\n'),
}
for ma, noi_dung in cac_nguon.items():
    (GOC / 'nguon' / (ma + '.txt')).write_text(
        noi_dung, encoding='utf-8')
tep = GOC / 'nguon' / 'N02.txt'
ban_giao = {
    'tieu_de': 'Ghi rõ phần đã kiểm trong một video',
    'goc_viet': 'Phân biệt đọc lời, xem hình và nghe âm thanh',
    'menh_de': [{
        'noi_dung': 'Người phụ trách chưa nghe hết âm thanh.',
        'can_cu': {
            'ma_nguon': 'N02',
            'dau_van': hashlib.sha256(tep.read_bytes()).hexdigest(),
            'trich_doan': 'Người phụ trách chưa nghe toàn bộ âm thanh.',
        },
    }],
}
(dich / 'ban_giao.json').write_text(
    json.dumps(ban_giao, ensure_ascii=False, indent=2),
    encoding='utf-8')
print('Đã tạo ba nguồn và bản giao việc mẫu trong luot_chay/01.')
```

Chạy lệnh python3 chuong_trinh/khoi_tao.py một lần trong thư mục mới. Nếu thành công, mở nguon để xem ba tệp và luot_chay/01/ban_giao.json để xem bản giao việc. Chuỗi ở trường dau_van đã được tính từ nội dung N02, thay cho phần giữ chỗ ở chương 4.

Tệp kiểm bản giao việc

Lưu đoạn tiếp theo thành chuong_trinh/kiem.py. Hàm dau_van tính mã nhận diện của một tệp. Hàm kiem trả danh sách lỗi. Phần main nhận đường dẫn từ lệnh, chạy kiểm và lưu báo cáo.

Báo cáo có bốn trường. kiem_co_hoc_dat cho biết có lỗi cơ học hay không; dau_van_ban_giao nhận diện đúng bản vừa kiểm; loi liệt kê vấn đề; chua_kiem nêu những phần vẫn cần bước đánh giá khác. Giá trị true trong dữ liệu JSON nghĩa là đúng, false nghĩa là sai, null nghĩa là không có giá trị ở trường đó.

```
"""Kiểm cấu trúc, phiên bản nguồn và trích đoạn."""
import hashlib
import json
import sys
from pathlib import Path

GOC = Path(__file__).resolve().parents[1]


def dau_van(tep):
    return hashlib.sha256(tep.read_bytes()).hexdigest()


def kiem(tep, thu_muc_nguon=None):
    thu_muc_nguon = thu_muc_nguon or GOC / 'nguon'
    nguon = {p.stem: p for p in thu_muc_nguon.glob('*.txt')}
    loi = []
    try:
        du_lieu = json.loads(tep.read_text(encoding='utf-8'))
    except (OSError, ValueError) as exc:
        return ['Không đọc được bản giao việc: ' + str(exc)]
    if not isinstance(du_lieu, dict):
        return ['Bản giao việc phải là một đối tượng dữ liệu.']
    for truong in ['tieu_de', 'goc_viet']:
        gia_tri = du_lieu.get(truong)
        if not isinstance(gia_tri, str) or not gia_tri.strip():
            loi.append('Thiếu nội dung ở trường ' + truong)
    danh_sach = du_lieu.get('menh_de')
    if not isinstance(danh_sach, list) or not danh_sach:
        return loi + ['Cần ít nhất một mệnh đề.']
    for so, muc in enumerate(danh_sach, 1):
        nhan = 'Mệnh đề ' + str(so) + ': '
        if not isinstance(muc, dict):
            loi.append(nhan + 'sai cấu trúc.'); continue
```

```
        noi_dung = muc.get('noi_dung')
        if not isinstance(noi_dung, str) or not noi_dung.strip():
            loi.append(nhan + 'chưa có nội dung.')
        can_cu = muc.get('can_cu')
        if not isinstance(can_cu, dict):
            loi.append(nhan + 'thiếu căn cứ.'); continue
        ma = can_cu.get('ma_nguon')
        tep_nguon = nguon.get(ma) if isinstance(ma, str) else None
        if tep_nguon is None:
            loi.append(nhan + 'không tìm thấy mã nguồn.'); continue
        if can_cu.get('dau_van') != dau_van(tep_nguon):
            loi.append(nhan + 'dấu vân nguồn đã khác.')
        trich = can_cu.get('trich_doan')
        if not isinstance(trich, str) or not trich.strip():
            loi.append(nhan + 'trích đoạn đang rỗng.')
        elif trich not in tep_nguon.read_text(encoding='utf-8'):
            loi.append(nhan + 'trích đoạn không có trong nguồn.')
    return loi


def main():
    if len(sys.argv) != 2:
        print('Cần truyền đường dẫn bản giao việc.'); return 2
    tep = Path(sys.argv[1]).resolve()
    if not tep.is_relative_to(GOC / 'luot_chay'):
        print('Bản giao việc phải nằm trong luot_chay.'); return 2
    loi = kiem(tep)
    bao_cao = {
        'kiem_co_hoc_dat': not loi,
        'dau_van_ban_giao': dau_van(tep) if tep.is_file() else None,
        'loi': loi,
        'chua_kiem': ['nghĩa mệnh đề', 'giọng văn', 'quyền đăng'],
    }
    if tep.parent.is_dir():
        dich = tep.parent / 'bao_cao_kiem.json'
        dich.write_text(json.dumps(bao_cao, ensure_ascii=False,
                                   indent=2), encoding='utf-8')
    print(json.dumps(bao_cao, ensure_ascii=False, indent=2))
    return 1 if loi else 0


if __name__ == '__main__':
    sys.exit(main())
```

Chạy python3 chuong_trinh/kiem.py luot_chay/01/ban_giao.json. Với bản mẫu vừa khởi tạo, kết quả kiểm cơ học là đạt. Báo cáo được lưu tại luot_chay/01/bao_cao_kiem.json. Chương trình cũng trả mã thoát: 0 khi

kiểm đạt, 1 khi có lỗi trong bản giao việc, 2 khi cách gọi hoặc vị trí tệp không phù hợp.

Nếu chỉ đổi câu ở trường noi_dung thành một nhận định sai nhưng giữ đúng căn cứ, báo cáo có thể vẫn đạt. Hàm kiểm chưa đọc nghĩa câu. Đây là giới hạn cần giữ rõ khi dùng kết quả.

Tệp thử các lỗi có chủ đích

Lưu đoạn sau thành chuong_trinh/thu_loi.py. Nó sao chép dữ liệu vào thư mục tạm rồi thử năm trường hợp. Sau khi kết thúc, nguồn gốc vẫn giữ nguyên. Kết quả được ghi vào luot_chay/01/thu_loi.json.

```
"""Thử bộ kiểm trên bản sao tạm, không sửa nguồn gốc."""
import copy
import json
import tempfile
from pathlib import Path
from kiem import GOC, kiem

mau = json.loads((GOC / 'luot_chay/01/ban_giao.json').read_text(
    encoding='utf-8'))
ket_qua = []
with tempfile.TemporaryDirectory() as tam:
    goc_tam = Path(tam)
    nguon = goc_tam / 'nguon'
    nguon.mkdir()
    for tep in (GOC / 'nguon').glob('*.txt'):
        (nguon / tep.name).write_bytes(tep.read_bytes())

    def thu(ten, du_lieu, mong_dat):
        tep = goc_tam / 'ban_giao.json'
        tep.write_text(json.dumps(du_lieu, ensure_ascii=False),
                       encoding='utf-8')
        loi = kiem(tep, nguon)
        dat = not loi
        if dat != mong_dat:
            raise RuntimeError('Khác dự kiến: ' + ten + str(loi))
        ket_qua.append({'tinh_huong': ten,
                        'kiem_co_hoc_dat': dat, 'loi': loi})

    thu('Mẫu đúng', mau, True)
    ban = copy.deepcopy(mau)
    ban['menh_de'][0]['can_cu']['ma_nguon'] = 'KHONG_CO'
    thu('Nguồn không có', ban, False)
    ban = copy.deepcopy(mau)
```

```
    ban['menh_de'][0]['can_cu']['trich_doan'] = 'Câu tự tạo.'
    thu('Trích đoạn tự tạo', ban, False)
    tep = nguon / 'N02.txt'
    cu = tep.read_text(encoding='utf-8')
    tep.write_text(cu + 'Đã thêm ghi chú.\n', encoding='utf-8')
    thu('Nguồn đổi, dấu vân cũ', mau, False)
    tep.write_text(cu, encoding='utf-8')
    ban = copy.deepcopy(mau)
    ban['menh_de'][0]['noi_dung'] = (
        'Đã nghe hết âm thanh; video đạt một triệu lượt xem.')
    thu('Sai nghĩa, căn cứ còn đúng hình thức', ban, True)

(GOC / 'luot_chay/01/thu_loi.json').write_text(
    json.dumps(ket_qua, ensure_ascii=False, indent=2),
    encoding='utf-8')
print('Năm tình huống cho kết quả đúng dự kiến.')
```

Chạy python3 chuong_trinh/thu_loi.py. Khi mọi trường hợp cho kết quả đúng dự kiến, chương trình in thông báo “Năm tình huống cho kết quả đúng dự kiến.” Bạn vẫn nên mở tệp kết quả để xem trường hợp sai nghĩa được chấp nhận bởi phép kiểm cơ học.

Các chương trình này đã được chạy với dữ liệu mẫu khi biên soạn bản sách. Lượt thử xác nhận cấu trúc kiểm và điểm mù được mô tả; nó không so sánh năng lực giữa các sản phẩm AI. Nếu bạn thay chương trình hoặc nguồn, chạy lại những trường hợp liên quan rồi ghi kết quả mới.

Giao một lượt mới cho tác nhân

Khi hiểu bản mẫu, tạo lượt luot_chay/02 và dùng lời giao sau. Bạn có thể dán nguyên đoạn vào công cụ đang mở đúng thư mục:

> Đọc ba nguồn và các hướng dẫn dự án. Tạo một bản giao việc mới cho bài giải thích phạm vi kiểm video, lưu trong luot_chay/02/ban_giao.json. Dùng cùng cấu trúc với mẫu ở lượt 01. Tính dấu vân từ tệp thật, chạy chương trình kiểm rồi xử lý lỗi của bản giao việc. Giữ nguồn và chương trình kiểm không đổi. Sau đó đọc lại nghĩa từng nhận định, ghi phần đã đối chiếu và điểm còn thiếu vào một tệp riêng. Dừng ở gói nháp để xem.

Bạn có thể ngắt sau bước tạo bản giao việc, ghi trạng thái rồi mở phiên mới để thử tiếp tục từ phần còn thiếu.

Phụ lục B. Những tên có thể gặp khi dùng công cụ

Các thuật ngữ dưới đây giúp bạn nhận ra phần tương ứng trong giao diện hoặc tài liệu. Trong mạch chính, sách ưu tiên nghĩa tiếng Việt để việc đọc không phụ thuộc vào khả năng nhớ tên gọi.

Agent (tác nhân thực thi) là hệ thống dùng mô hình để chọn bước tiếp theo và gọi công cụ trong một nhiệm vụ. Coding agent (tác nhân có khả năng lập trình) có thể viết, sửa và chạy mã trong môi trường được cung cấp. Khả năng cụ thể phụ thuộc công cụ cùng quyền đang có.

Model (mô hình) là thành phần tạo nội dung và quyết định từ ngữ cảnh được đưa vào. Tool (công cụ) thực hiện thao tác như đọc tệp, chạy chương trình hoặc truy vấn dịch vụ. Runtime (môi trường thực thi) là nơi các thao tác được chạy, với thư viện, cấu hình và quyền tương ứng.

Workspace (không gian làm việc) chứa dữ liệu, hướng dẫn, quy định, chương trình và kết quả của dự án. Context (ngữ cảnh) là thông tin đang được đưa vào để mô hình xử lý ở một thời điểm; không phải mọi tệp trong không gian làm việc đều đã nằm trong ngữ cảnh.

Prompt (lời giao hoặc yêu cầu cho AI) mô tả việc cần làm trong lượt hiện tại. Rule (quy tắc) hướng dẫn hành vi dùng lâu hơn một lượt. Workflow (quy trình thực hiện) mô tả chuỗi bước. Skill (gói hướng dẫn thao tác dùng lại) đóng gói một loại công việc, có thể kèm tài liệu và chương trình.

Harness (bộ khung điều phối thực thi) tổ chức việc gọi mô hình và công cụ, nhận kết quả rồi điều khiển lượt tiếp. Loop (vòng thực hiện và phản hồi) lặp việc làm, quan sát và điều chỉnh theo một điều kiện. Hook (điểm gắn xử lý tự động) cho phép một số môi trường chạy kiểm tra hoặc hành động tại những thời điểm nhất định của luồng.

Brief (bản giao việc tóm tắt) giữ mục tiêu, người đọc và kết quả cần có. Claim (mệnh đề cần căn cứ) là câu nhận định đang được xem xét. Evidence (căn cứ hỗ trợ) là dữ kiện dùng để xét mệnh đề ấy. Review (xem lại và đánh giá) cần nói rõ đã đọc hoặc thử phần nào, theo tiêu chí gì.

Schema (cấu trúc dữ liệu quy định) mô tả các trường, kiểu và điều kiện của dữ liệu. JSON (định dạng dữ liệu có cấu trúc) được dùng trong các tệp trạng thái và bản giao việc của sách. CSV (tệp bảng phân tách cột bằng dấu phẩy) được dùng cho ví dụ đơn hàng.

Hash (mã băm) là giá trị tính từ dữ liệu. Trong bài thực hành, nó được dùng làm dấu vân để nhận ra phiên bản thay đổi. Nó không tự đánh giá mức đáng tin của nguồn. Log (nhật ký thực thi) ghi lại sự kiện hoặc kết quả để xem lại quá trình.

Multi-agent (phối hợp nhiều tác nhân) là cách chia nhiệm vụ cho nhiều tác nhân rồi kết hợp kết quả. Handoff (bàn giao) giữ thông tin cần để bên tiếp theo tiếp tục. Checkpoint (mốc lưu trạng thái) là điểm bạn có thể dùng để xác định công việc đã đi tới đâu và phần nào cần làm tiếp.

API (giao diện lập trình để phần mềm gọi dịch vụ) là một cách chương trình tương tác với hệ thống khác. MCP (giao thức kết nối mô hình với công cụ và dữ liệu) là một cách cung cấp tích hợp cho tác nhân. Sandbox (môi trường thực thi được giới hạn quyền) giúp giới hạn phạm vi thao tác ở môi trường hỗ trợ; hiệu lực phụ thuộc cấu hình thực tế.

Tên sản phẩm, tệp lệnh và tùy chọn có thể thay đổi theo phiên bản. Khi tìm một chức năng trong công cụ, hãy xác định mình đang cần giải quyết phần nào của công việc: nạp hướng dẫn, đọc dữ liệu, chạy lệnh, giới hạn quyền hay xem kết quả. Sau đó đối chiếu với chức năng thực sự có trong môi trường đang dùng.