

Ex01:

Product Owner (PO):

"Chào mọi người, mục tiêu của Sprint này là hoàn thiện tính năng *Quản lý khóa học*, cho phép giảng viên tạo khóa học và sinh viên có thể xem danh sách khóa học khả dụng."

Developer (Dev):

"Vâng, em xin demo tính năng *Tạo khóa học*. Giảng viên có thể nhập tên khóa học, mô tả và thời lượng. Sau khi bấm 'Create', hệ thống sẽ lưu lại và hiển thị trong danh sách. Đây là ví dụ một khóa học mới đã được tạo."

Stakeholder:

"Tính năng hoạt động ổn. Tuy nhiên phần giao diện nhập thông tin hơi đơn giản. Tôi đề xuất bổ sung kiểm tra dữ liệu (validation) và thêm phần chọn ảnh đại diện khóa học để tăng tính trực quan."

PO:

"Ghi nhận ý kiến. Nhóm phát triển sẽ đưa yêu cầu này vào backlog để xem xét ưu tiên cho Sprint tiếp theo."

Ex02:

a. Product Backlog Item (PBI)

User Story – Role / Goal / Benefit

Title: Thêm bộ lọc thống kê theo ngày cho báo cáo hoạt động

User Story:

Là một trưởng phòng vận hành, tôi muốn xem được thống kê theo ngày hoặc theo 24 giờ gần nhất thay vì chỉ theo tuần, để tôi có thể kiểm tra chi tiết hoạt động của người dùng và theo dõi các vấn đề phát sinh một cách chính xác hơn.

b. Acceptance Criteria (AC)

(Dạng Given – When – Then)

AC1 – Lọc thống kê theo ngày

Given người dùng đang ở màn hình thống kê hoạt động
When chọn ngày cụ thể trên bộ lọc (ví dụ: 03/11)
Then hệ thống hiển thị dữ liệu chi tiết đúng với ngày đã chọn.

AC2 – Lọc thống kê trong 24 giờ gần nhất

Given người dùng mở trang thống kê
When chọn chế độ “Trong 24 giờ gần nhất”
Then hệ thống hiển thị dữ liệu hoạt động từ thời điểm hiện tại lùi về 24 giờ trước.

AC3 – Không có dữ liệu

Given người dùng chọn một khoảng ngày không có dữ liệu
When nhấn “Áp dụng”
Then hệ thống hiển thị thông báo “Không có dữ liệu cho khoảng thời gian này”.

Ex03:

a. Timeline Sprint (14 ngày)

Sprint Duration: 2 tuần (14 ngày)

Ngày 1 → Kickoff Sprint, phân chia công việc, review backlog.
Ngày 2 → Bắt đầu phát triển tính năng “Tạo công việc”.
Ngày 3 → Hoàn thiện UI cơ bản, bắt đầu logic backend.
Ngày 4 → Test nội bộ tính năng “Tạo công việc”.
Ngày 5 → Bắt đầu phát triển tính năng “Danh sách công việc”.
Ngày 6 → Tích hợp API hiển thị danh sách.
Ngày 7 → Gặp bug đồng bộ dữ liệu giữa client & server.
Ngày 8 → Fix bug, ổn định API.
Ngày 9 → Phát triển tính năng “Bộ lọc công việc theo trạng thái”.
Ngày 10 → Demo nội bộ lần 1.
Ngày 11 → Điều chỉnh UI theo feedback.
Ngày 12 → Viết test & chuẩn bị tài liệu.
Ngày 13 → Hoàn thiện & chạy test toàn hệ thống.
Ngày 14 → Sprint Review + Sprint Retrospective.

b. 5 Sự kiện quan trọng trong Sprint

Sự kiện 1 — Ngày 4: Hoàn tất test nội bộ tính năng “Tạo công việc”

Mô tả:

Team đã hoàn thành UI + logic thêm công việc và thử nghiệm nội bộ không phát sinh lỗi lớn.

Ảnh hưởng:

Tích cực — giúp cả nhóm yên tâm chuyển sang tính năng tiếp theo đúng timeline.

Bài học:

Test nhỏ và liên tục giúp giảm rủi ro dồn lỗi về cuối Sprint.

Sự kiện 2 — Ngày 7: Xuất hiện bug đồng bộ dữ liệu giữa client và server

Mô tả:

API trả về dữ liệu chậm khiến UI hiển thị danh sách công việc sai thời điểm.

Ảnh hưởng:

Tiêu cực — làm chậm việc phát triển trong 1 ngày, ảnh hưởng kế hoạch.

Bài học:

Cần thống nhất giao thức API rõ ràng từ đầu Sprint để tránh hiểu sai.

Sự kiện 3 — Ngày 10: Demo nội bộ lần 1

Mô tả:

Team demo các tính năng đã hoàn thiện: Tạo công việc, Danh sách, Bộ lọc.

Ảnh hưởng:

Tích cực — nhận được feedback sớm về UI và trải nghiệm người dùng.

Bài học:

Demo nội bộ giữa Sprint giúp phát hiện vấn đề sớm hơn Sprint Review.

Sự kiện 4 — Ngày 11: Điều chỉnh UI theo phản hồi

Mô tả:

Team cập nhật lại màu sắc, spacing và icon để giao diện rõ ràng hơn.

Ảnh hưởng:

Tích cực — cải thiện trải nghiệm người dùng mà không ảnh hưởng nhiều đến tiến độ.

Bài học:

Feedback nhỏ nhưng đúng thời điểm giúp giao diện tốt hơn mà không tốn quá nhiều công sức.

Sự kiện 5 — Ngày 14: Sprint Review & Retrospective

Mô tả:

Trình bày sản phẩm với PO + Stakeholder, ghi nhận 2 yêu cầu mới về lọc nâng cao và thêm trang phân tích.

Ảnh hưởng:

Tích cực — giúp định hướng backlog Sprint tiếp theo.

Bài học:

Phản hồi thực tế từ Stakeholder quyết định rõ giá trị thực sự của sản phẩm.

Ex04:

Kỹ thuật chọn: *Mad – Sad – Glad*

1. Vì sao kỹ thuật này phù hợp với trạng thái hiện tại của team?

- Team đang căng thẳng, tinh thần tụt dốc → *Mad – Sad – Glad* giúp mọi người bộc lộ cảm xúc thật một cách an toàn.
- Kỹ thuật này tập trung vào việc “**xả nén**” cảm xúc, rất phù hợp khi team đang stress và ít giao tiếp.

2. Nó giúp team giải quyết vấn đề gì?

- Giúp phát hiện những điều khiến team **bực bội (Mad)**, **buồn/áp lực (Sad)** — đây chính là các root-cause gây bug dồn và stress.
- Đồng thời tìm lại năng lượng tích cực từ phần **Glad**, giúp team phục hồi tinh thần và tăng kết nối.

3. Vì sao các kỹ thuật khác ít phù hợp hơn?

- **Start–Stop–Continue**: tập trung vào hành động, không phù hợp khi cảm xúc team đang bị “tắc”.
- **4Ls**: khá phân tán, không đi thẳng vào vấn đề cảm xúc.
- **Sailboat**: tốt cho định hướng dài hạn, nhưng team đang thiếu kết nối, không phải thiếu mục tiêu.
- **Timeline**: phù hợp phân tích sự kiện, nhưng không giải tỏa được trạng thái stress của team.

Ex05:

(A) Mục tiêu của buổi Sprint Review

1. Xác nhận tính năng đăng nhập hoạt động đúng như kỳ vọng trên cả web và mobile.
2. Trình bày luồng đăng nhập hoàn chỉnh cho stakeholders để kiểm tra trải nghiệm người dùng.
3. Thu thập phản hồi về UI, tốc độ phản hồi, thông báo lỗi, và tính năng bảo mật.
4. Kiểm tra sự phù hợp của tính năng với Sprint Goal và định hướng sản phẩm.
5. Xác định các cải tiến hoặc thay đổi cần đưa vào Product Backlog cho Sprint sau.

(B) Nội dung 1 phút mở đầu của cuộc họp

Product Owner nói gì?

“Xin chào mọi người, cảm ơn đã tham dự buổi Sprint Review hôm nay. Trong Sprint này, mục tiêu chính của chúng tôi là hoàn thiện tính năng đăng nhập hệ thống để đảm bảo người dùng có thể truy cập nhanh, an toàn và ổn định. Đây là tính năng nền tảng cho các chức năng tiếp theo, nên rất mong nhận được phản hồi chi tiết từ mọi người.”

Scrum Master nói gì?

“Cảm ơn PO. Buổi Sprint Review hôm nay sẽ kéo dài khoảng 30 phút. Chúng ta sẽ xem demo tính năng đăng nhập, sau đó dành thời gian cho câu hỏi và phản hồi. Mọi người có thể góp ý trực tiếp hoặc ghi lại để bàn trong phần cuối buổi.”

Dev Lead nói gì?

“Trong phần demo, tôi sẽ trình bày luồng đăng nhập gồm: nhập tài khoản, xử lý lỗi, đăng nhập thành công và chuyển hướng sang Dashboard. Chúng tôi cũng chuẩn bị vài tình huống lỗi để stakeholders có thể đánh giá trải nghiệm người dùng.”

(C) Dev sẽ demo tính năng như thế nào (6–8 câu chi tiết)

1. Dev mở giao diện đăng nhập và giới thiệu bố cục gồm ô nhập email, mật khẩu và nút “Đăng nhập”.
2. Sau đó, dev nhập một email không hợp lệ để mô tả cách hệ thống hiển thị lỗi ngay lập tức.

3. Dev tiếp tục nhập đúng định dạng email nhưng sai mật khẩu để minh họa lỗi xác thực từ backend.
4. Dev dùng một tài khoản hợp lệ để thực hiện đăng nhập thành công và trình bày tốc độ phản hồi của API.
5. Sau khi đăng nhập thành công, hệ thống tự động điều hướng sang trang Dashboard — dev giải thích dữ liệu hiển thị ban đầu.
6. Dev mô tả cách hệ thống lưu token bảo mật và hướng dẫn stakeholders kiểm tra trạng thái đăng nhập trong Local Storage.
7. Nếu có thời gian, dev bật chế độ mạng chậm để minh họa việc loading và cách UI xử lý trạng thái chậm.
8. Cuối cùng, dev tổng hợp lại toàn bộ luồng và hỏi feedback trực tiếp từ stakeholders về UX, tốc độ và thông báo lỗi.

Ex06:

(A) Tên Retrospective + kỹ thuật sử dụng

Tên buổi Retro: “*Improve Flow – Improve Discipline*” *Sprint Retrospective*

Kỹ thuật sử dụng (2 kỹ thuật):

1. **Timeline** – để cả team nhìn lại toàn bộ Sprint và xác định điểm phát sinh vấn đề (late bug, spillover, daily trễ).
2. **Start – Stop – Continue** – để đưa ra hành động cải tiến rõ ràng, dễ ưu tiên và dễ thực thi.

(B) Agenda chi tiết cho buổi Retrospective 90 phút

Tổng thời lượng: 90 phút

1. **Ice-breaking – Warm-up (10 phút)**
Mini-game “One Word Check-in”: mỗi người nói 1 từ mô tả cảm xúc của mình về Sprint vừa rồi.
2. **Stage 1 – Gather Data: Timeline Review (20 phút)**
 - o Team cùng dựng lại timeline 2 tuần.
 - o Ghi các sự kiện chính: Daily trễ, bug phát hiện muộn, 3 task spillover, thiếu ý kiến cải tiến.
 - o Mỗi thành viên dán note diễn tả “điểm tắc” họ nhìn thấy.
3. **Stage 2 – Generate Insights (15 phút)**
 - o Nhóm note tương đồng lại với nhau.
 - o Phân tích nguyên nhân gốc rễ (why-why analysis).

- Chỉ ra 3–4 điểm gây ảnh hưởng mạnh nhất.
- 4. **Stage 3 – Start – Stop – Continue (25 phút)**
 - Mỗi thành viên viết 3 nhóm: Start / Stop / Continue.
 - Scrum Master gom nhóm, ưu tiên theo voting.
 - Chọn ra các hành động khả thi cho Sprint sau.
- 5. **Stage 4 – Define Action Items (15 phút)**
 - Chốt 5 hành động cụ thể kèm người phụ trách & deadline.
 - Đảm bảo từng action measurable.
- 6. **Close-out (5 phút)**
 - Mỗi thành viên chia sẻ 1 điều tâm đắc của buổi Retro.
 - Scrum Master nhắc lại các cam kết Sprint sau.

(C) 5 Action Items (đầy đủ: Ai – Làm gì – Khi nào – Đo lường thế nào)

Action Item 1

- **Ai:** Scrum Master
- **Làm gì:** Thiết lập quy tắc “Daily đúng giờ – bắt đầu đúng phút”.
- **Khi nào:** Áp dụng ngay từ Sprint kế tiếp.
- **Đo lường:** Daily bắt đầu đúng giờ $\geq 90\%$ số buổi.

Action Item 2

- **Ai:** QA Lead
- **Làm gì:** Thiết lập quy trình test sớm (Shift-left), test ngay khi dev xong từng module nhỏ.
- **Khi nào:** Trong 3 ngày đầu Sprint.
- **Đo lường:** Số bug phát hiện từ ngày cuối Sprint giảm $\geq 50\%$.

Action Item 3

- **Ai:** Dev Team
- **Làm gì:** Chia nhỏ task để tránh spillover, mỗi task ≤ 1.5 ngày effort.
- **Khi nào:** Từ buổi Sprint Planning kế tiếp.
- **Đo lường:** Giảm spillover từ 3 task xuống ≤ 1 task mỗi Sprint.

Action Item 4

- **Ai:** Tất cả thành viên
- **Làm gì:** Mỗi người mang ít nhất 1 ý kiến cải tiến trong Retro.
- **Khi nào:** Mỗi buổi Retrospective của Sprint sau.
- **Đo lường:** Số lượng ý kiến tăng từ mức hiện tại (rất ít) lên ≥ 8 ý kiến/buổi.

Action Item 5

- **Ai:** Dev + QA
- **Làm gì:** Thực hiện review chung vào giữa Sprint để rà soát tiến độ & chất lượng.
- **Khi nào:** Ngày thứ 7 của Sprint.
- **Đo lường:** Bug nghiêm trọng giảm từ cuối Sprint, tiến độ bám sát kế hoạch hơn.

Ex07:

Format	Thời gian đề xuất	Phù hợp khi team...	Ưu điểm	Nhược điểm	Điểm dễ dẫn dắt (1–5)
Start – Stop – Continue	30–45 phút	Cần hành động cụ thể, rõ ràng ngay; team quen Retro nhưng thiếu định hướng cải tiến	Dễ hiểu, nhanh, tập trung vào hành động; phù hợp mọi team	Ít đào sâu nguyên nhân; dễ thành “checklist”	5
Sailboat	45–60 phút	Team cần nhìn toàn cảnh sprint, xác định risk, blocker, cơ hội	Hình ảnh trực quan; kích thích thảo luận; giúp team thấy tổng thể	Tốn thời gian nếu team đông; dễ lan man nếu không kiểm soát tốt	4
Mad – Sad – Glad	40–50 phút	Team căng thẳng, ít nói thật; cần giải tỏa cảm xúc và khơi mở chia sẻ	Tập trung vào cảm xúc; giúp team mở lòng; tạo sự thấu hiểu	Không tạo hành động cụ thể nếu không kết hợp thêm kỹ thuật khác	3
4Ls (Liked, Learned, Lacked, Longed for)	45–60 phút	Team cần phân tích học được gì, thiếu gì; phù hợp bước chuyển giai đoạn hoặc kết thúc release	Cân bằng giữa cảm xúc – kiến thức – cải tiến; đào sâu lý do	Gồm nhiều mục nên tốn thời gian; dễ chòng chéo ý	4
KALM (Keep, Add, More, Less)	30–45 phút	Team vận hành ổn, muốn tối ưu quy trình và tăng hiệu suất	Tập trung tối ưu hóa; dễ thấy cơ hội cải tiến ngay; đơn giản	Không phù hợp với team đang xung đột hoặc stress	5

Ex08:

(A) Bạn sẽ nói gì đầu tiên để lấy lại không khí?

Scrum Master đứng lên, giọng bình tĩnh và trung lập:

“Cả team mình hãy tạm dừng một chút. Mục tiêu của buổi Retrospective không phải là tìm lỗi cá nhân, mà là tìm giải pháp để Sprint sau tốt hơn. Chúng ta cùng hướng vào quy trình, không hướng vào con người. Mình đề nghị mọi người hít thở nhẹ, lấy lại tinh thần và chúng ta sẽ tiếp tục theo đúng format.”

(B) Kỹ thuật để mọi người không đổ lỗi cá nhân

Tôi sử dụng kỹ thuật “Silent Writing + Timeline Mapping”:

1. Silent Writing – Viết im lặng (5 phút)

- Mỗi người tự viết ra vấn đề mà họ thấy trong Sprint
- Không nói chuyện
- Tập trung vào *sự kiện, quy trình, công cụ*, không nhắc tên cá nhân

→ Kỹ thuật này giúp loại bỏ cảm xúc, giảm tranh luận, giảm đổ lỗi.

2. Timeline Mapping – Vẽ dòng thời gian Sprint (10 phút)

- Gắn các ghi chú lên timeline từ ngày 1 → ngày cuối Sprint
- Nhìn lại:
 - Khi nào Daily Scrum bị muộn?
 - Khi nào bug xuất hiện?
 - Task nào bị spillover?
- Nhìn vào **dữ liệu khách quan**, không còn chỗ cho “tôi nghĩ”, “tôi cảm thấy”, hay “tại bạn”.

→ Chuyển focus từ “Con người sai” → “Quy trình đang gặp lỗi”.

(C) 3 câu nói “thần chú” để xử lý khi có người đổ lỗi

1. “Hãy nói về *quy trình*, không phải *con người*.”

Giúp nhắc team quay lại mục tiêu cốt lõi.

2. “Điều này xảy ra vào thời điểm nào trong Sprint? Chúng ta cùng nhìn vào dữ liệu nhé.”

Kéo tranh luận từ cảm xúc → dữ liệu timeline.

3. “Giải pháp chúng ta có thể làm là gì cho Sprint sau?”

Chuyển focus sang tương lai, giảm căng thẳng.

(D) Action Item giải quyết tận gốc vấn đề đổ lỗi

Action Item 1 – Chuẩn hóa quy trình báo bug

- **Ai:** Tester Lead
- **Làm gì:** Thêm bước QA review sớm trước ngày cuối Sprint
- **Khi nào:** Áp dụng ngay Sprint sau
- **Đo lường:** 80% bug được báo trước ngày thứ 3 của Sprint

Action Item 2 – Thiết lập nguyên tắc “No Blame, Only Data” trong Retro

- **Ai:** Scrum Master
- **Làm gì:** Mở đầu mỗi Retro bằng quy tắc “Không đổ lỗi cá nhân”
- **Khi nào:** Từ Sprint tiếp theo
- **Đo lường:** Không còn trường hợp nhắc tên cá nhân trong Retro

Action Item 3 – Quy định Daily Scrum đúng giờ

- **Ai:** Cả team
- **Làm gì:** Đứng dậy và bắt đầu Daily đúng 9:00, không chờ người đến trễ
- **Khi nào:** Sprint tới
- **Đo lường:** Tỷ lệ họp muộn giảm còn <10%

Action Item 4 – Checklist coding standard cho Dev

- **Ai:** Dev Lead
- **Làm gì:** Tạo checklist coding standard (clean code, format, naming...)
- **Khi nào:** Hoàn thành trong 2 ngày
- **Đo lường:** 100% Pull Request pass checklist trước khi merge

Action Item 5 – Cấm dùng điện thoại trong Retro

- **Ai:** Scrum Master
- **Làm gì:** Đặt khay thu điện thoại khi vào phòng họp
- **Khi nào:** Từ Retro kế tiếp
- **Đo lường:** 0 trường hợp sử dụng điện thoại trong buổi Retro

