Bài 1: Mô phỏng một buổi sprint review

- Scrum master tổ chức cuộc họp
- Các thanh viên tham gia: Scrum master, PO, Developer, Stakeholder
- Quy trình
    - PO liệt kê các task (tính năng trong Sprint)
        - Thêm sản phẩm vào giỏ hàng
        - Tăng giảm , xóa sản phẩm trong giỏ hàng
        - Xem chi tiết sản phẩm trong giỏ hảng
    - Các thanh viên trong team demo về tính năng
    - Khách hàng nhận xét
        - Tôi hài lòng với các chức năng
        - Muốn bổ sung thêm khi thêm sản phẩm vào giỏ hàng thì có hiệu ứng sản phẩm bay vào giỏ hàng
    - Team hội ý và sau đó PO đưa vào Product backlog
- Kết thúc cuộc họp

Bài 2:

a. viết lại phản hồi thành 1 product backlog item

Role	Goal	bennrfit
Khách hàng	Xem báo cáo ngày dễ dàng	Dễ dàng lọc các dữ liệu

b. Viết 1 bộ Acceptance Criteria cho PBI này

Given	When	Then
Trưởng phòng vận hành	Muốn xem dữ liệu theo ngày	Xem chi tiết từng ngày mà không bị rối
Trưởng phòng vận hành	Click chọn ngày	Nếu ngày không có dữ liệu thì hiển thị thông báo không có kết quả phù hợp

Trưởng phòng vận hành	Click xem dữ liệu theo khung giờ cụ thể	Kết quả trả về theo khung giờ trong ngày phải chọn ngày trước khi chọn giờ

Bài 3:

A) Timeline Sprint (Ngày 1 → Ngày 14)

Ngày	Timeline (diễn biến chính)
1	Sprint Planning, chốt Sprint Goal + chọn PBI (Login, CRUD Task, Board)
2	Setup repo, CI cơ bản, tạo UI layout + routing
3	Làm API Login + token + màn hình đăng nhập
4	CRUD Task (Create/List), validate form
5	Tích hợp FE–BE, fix lỗi mapping dữ liệu
6	Làm Kanban Board kéo-thả (UI)
7	Review nội bộ giữa Sprint, cắt bớt scope "tag nâng cao"
8	Hoàn thiện kéo-thả + update trạng thái task
9	Phát sinh bug: task bị trùng/nhảy cột sai khi kéo nhanh
10	Tối ưu API + thêm test cơ bản cho service task
11	QA test, log nhiều bug UI/edge case
12	Fix bug theo mức độ ưu tiên, hardening
13	Freeze tính năng, chuẩn bị demo + slide Sprint Review
14	Sprint Review demo + Sprint Retrospective, chốt action cải tiến

---

B) 5 sự kiện quan trọng trong Sprint

#	Ngày xảy ra	Mô tả sự kiện	Ảnh hưởng	Bài học (1 câu)

1	Ngày 1	Sprint Planning chốt Sprint Goal "User đăng nhập + quản lý task + kéo thả board" nhưng estimate hơi lạc quan	Tiêu cực nhẹ: dễ over-commit, cuối Sprint phải cắt scope	Đặt Sprint Goal rõ và **để buffer** cho tích hợp + QA.
2	Ngày 5	Tích hợp FE–BE gặp lỗi format ngày/ID (task hiển thị sai, update thất bại)	Tiêu cực: mất ~0.5–1 ngày để debug dữ liệu	Chuẩn hóa **contract API** (schema) ngay từ đầu, có ví dụ payload.
3	Ngày 7	Mid-sprint review: nhận ra "tag nâng cao" không cần cho demo → quyết định drop	Tích cực: giảm áp lực, tập trung vào giá trị chính	Dám **cắt scope** để bảo vệ Sprint Goal quan trọng hơn "làm cho nhiều".
4	Ngày 9	Bug kéo-thả: kéo nhanh gây duplicate task / sai trạng thái do cập nhật không đồng bộ	Tiêu cực: QA phát hiện trễ, tốn thời gian fix	Tính năng UI phức tạp cần **test sớm** và log rõ hành vi edge case.
5	Ngày 14	Sprint Review demo ổn nhưng QA note "cần checklist DoD rõ hơn" (test, review, tiêu chí done)	Tích cực: team thống nhất cải tiến quy trình	DoD phải cụ thể (test, code review, acceptance) để tránh bug dồn cuối Sprint.

Bài 4:

Nội dung	Trả lời (dạng 5–7 dòng, thực tế)
Kỹ thuật chọn	Mad – Sad – Glad
Vì sao phù hợp với trạng thái hiện tại của team?	Team đang stress, thiếu năng lượng, ít giao tiếp và "không ai nói thật" trong Daily, nên cần một format đi từ cảm xúc trước để mở khóa chia sẻ. Mad–Sad–Glad cho phép nói điều bực, điều buồn, điều tích cực theo cách an toàn, dễ mở lời hơn so với hỏi "vấn đề là gì" ngay. Khi mọi người được "xả" cảm xúc đúng cách, không khí bớt nặng và kết nối lại.
Nó giúp team giải quyết vấn đề gì?	(1) Giải tỏa cảm xúc và giảm căng thẳng đang tích tụ. (2) Khơi gợi sự chia sẻ thật: từ "Mad/Sad" thường lòi ra nguyên nhân như over-commit, review kém, thiếu test, scope đổi… (3) Từ các ý "Mad/Sad", Scrum Master gom nhóm và dùng 5 Whys nhanh để ra root-cause của bug dồn & QA quá tải, rồi chốt 1–2 action rõ ràng cho Sprint 6.

Vì sao kỹ thuật khác ít phù hợp hơn?	Start–Stop–Continue và 4Ls thiên về "hành động/bài học", nhưng team đang khó nói thẳng, dễ trả lời cho có. Sailboat và Timeline tốt để phân tích tiến trình, nhưng khi cảm xúc đang nặng + thiếu an toàn tâm lý, team có thể né tránh và không nói "cái đau" (stress, áp lực, bug). Vì vậy Mad–Sad–Glad hợp nhất để mở khóa cảm xúc trước, rồi mới đi tới nguyên nhân và cải tiến.

Bài 5:

(A) Mục tiêu của buổi Sprint Review (tính năng Login)

- Xác nhận tính năng **đăng nhập hoạt động đúng** theo yêu cầu (đúng account, đúng phân quyền cơ bản).
- Nhận phản hồi từ stakeholders về **trải nghiệm đăng nhập** (form, thông báo lỗi, "ghi nhớ đăng nhập", luồng quên mật khẩu nếu cần).
- Kiểm tra mức độ sẵn sàng để tích hợp với các chức năng sau đăng nhập (dashboard, danh sách công việc…).
- Ghi nhận đề xuất cải tiến và **điều chỉnh ưu tiên** cho Sprint tiếp theo (ví dụ thêm MFA/OTP, giới hạn đăng nhập sai, UX).

(B) 1 phút mở đầu cuộc họp – Ai sẽ nói gì? (mô phỏng nguyên văn)

Vai trò	**Lời nói trong 1 phút đầu (nguyên văn mẫu)**
Product Owner (PO)	"Chào mọi người, Sprint này mục tiêu của chúng ta là đưa vào luồng **Login** để người dùng truy cập hệ thống an toàn trước khi dùng các chức năng chính. Hôm nay team sẽ demo đăng nhập với tài khoản mẫu và các tình huống sai mật khẩu để mọi người xác nhận đúng nhu cầu. Sau demo, em mong nhận góp ý về trải nghiệm và các yêu cầu bổ sung như 'ghi nhớ đăng nhập', giới hạn đăng nhập sai, hoặc OTP."
Scrum Master (SM)	"Cảm ơn mọi người đã tham gia Sprint Review. Thời lượng buổi hôm nay khoảng **20–30 phút**: 5 phút recap, 10 phút demo, còn lại Q&A và góp ý. Mình tập trung vào **sản phẩm chạy được**, góp ý dựa trên trải nghiệm thực tế; các ý mới sẽ được PO ghi lại để đưa vào Product Backlog."

Dev Lead	"Team sẽ demo luồng Login end-to-end: mở màn hình đăng nhập, nhập tài khoản đúng để vào trang chính, sau đó thử vài trường hợp lỗi như sai mật khẩu và bỏ trống dữ liệu. Cuối demo tụi em sẽ show log/hiển thị để mọi người thấy hệ thống phản hồi đúng như mong đợi."

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

1. Dev mở trình duyệt và truy cập URL hệ thống, màn hình đầu tiên hiển thị là **Trang Đăng nhập** gồm Email/Username, Password và nút **Login**.
2. Dev giới thiệu **dữ liệu mẫu**: tài khoản staff01, mật khẩu 123456 (hoặc mẫu do PO cung cấp) và nhắc rằng tài khoản này có quyền "User/Staff".
3. Dev nhập đúng thông tin và bấm Login, hệ thống gửi request xác thực và trả về token/session; màn hình chuyển sang **Dashboard/Home** và hiển thị tên người dùng ở góc phải.
4. Dev nhấn F5 để refresh trang nhằm chứng minh người dùng **vẫn đăng nhập** (nếu có cơ chế lưu phiên) và không bị bật về trang login.
5. Tiếp theo, Dev đăng xuất (Logout), quay lại màn hình login và thử nhập **sai mật khẩu**, hệ thống phải hiển thị thông báo rõ ràng như "Sai mật khẩu" và không cho vào hệ thống.
6. Dev thử trường hợp **để trống** username hoặc password, hệ thống phải hiện validate ngay dưới ô nhập (ví dụ: "Vui lòng nhập mật khẩu") trước khi gửi request.
7. Dev thử nhập liên tiếp sai 3 lần (nếu có), để stakeholders xác nhận có cần **giới hạn đăng nhập sai** hoặc captcha/lock account cho Sprint sau.
8. Cuối cùng, Dev tóm tắt kết quả mong đợi: đăng nhập đúng → vào hệ thống; đăng nhập sai/thiếu dữ liệu → báo lỗi đúng và an toàn; PO ghi nhận góp ý về UX và các yêu cầu mở rộng.

Bài 6:

(A) Tên Retrospective + kỹ thuật sử dụng (không cần bảng)

Tên: "Retro Sprint: Đúng giờ – Hạn chế spillover – Bug báo sớm"
Kỹ thuật lấy ý kiến (≥2): Timeline (nhìn lại theo tiến trình) + **Start–Stop–Continue** (tạo ý kiến cải tiến) + **Dot Voting** (chọn ưu tiên)

(B) Agenda Retrospective 90 phút (bảng)

Giai đoạn	Thời gian	Kỹ thuật	Làm gì ra kết quả gì
Set the stage	0–10’	Check-in 1 từ	Ổn định không khí, nhắc Prime Directive, mục tiêu: đúng giờ + giảm spillover + bug sớm
Gather data	10–30’	**Timeline**	Dán note theo ngày: Daily trễ, 3 task spillover, bug báo ngày cuối, sự kiện liên quan
Generate insights	30–45’	Gom nhóm + “Vì sao?” nhanh	Nhóm vấn đề: (1) Daily trễ (2) Spillover (3) Bug muộn → chốt 1–2 nguyên nhân chính/nhóm
Generate ideas	45–60’	**Start–Stop–Continue**	Mỗi người viết 2–3 ý cho Start/Stop/Continue; đọc nhanh, gộp ý trùng
Decide what to do	60–75’	**Dot Voting**	Mỗi người 3 phiếu → chọn Top 3 cải tiến ưu tiên Sprint tới
Close + Commit	75–90’	Chốt plan + cam kết	Chuyển Top 3 thành Action Items (Ai/Khi nào/Metric), mỗi người chốt 1 cam kết

---

(C) Action Items sau Retro (bảng) — (Ai – Làm gì – Khi nào – Đo lường)

#	Ai chịu trách nhiệm	Làm gì	Khi nào (deadline)	Đo lường thế nào
1	Scrum Master	Đổi Daily sang giờ cố định + nhắc lịch tự động; áp dụng rule “bắt đầu đúng giờ (trễ tối đa 2 phút)”	Từ **ngày 1 Sprint tới**	≥ **90%** Daily bắt đầu trong ±2 phút
2	Dev Lead	Rà lại capacity khi Sprint Planning: trừ **15% buffer** + không “nhét thêm” ngoài thỏa thuận	**Trước Sprint Planning** Sprint tới	Spillover giảm còn ≤ **1 task/Sprint**
3	Dev Team	Quy tắc tách task: task > 1 ngày phải split (UI/API/Test) để không kẹt cuối Sprint	Từ **ngày 2 Sprint tới**	100% task >1 ngày được split; giảm ticket “kẹt” cuối Sprint
4	Tester/QA	Thiết lập “Test sớm”: ticket chuyển “Ready for Test” thì **test trong 24h**; không dồn ngày cuối	Từ **ngày 3 Sprint tới**	≥ **80% bug** được log **trước 2 ngày cuối Sprint**

5	Dev + QA (pair)	Thêm checkpoint giữa Sprint (ngày 5/10): rà ticket đã "Ready for Test", ưu tiên test/fix sớm	**Giữa Sprint tới**	Bug phát hiện ngày cuối giảm **≥ 50%** so Sprint trước
6	Scrum Master	Tăng ý kiến cải tiến: cuối Retro bắt buộc mỗi người đóng góp **ít nhất 1 note** (Start/Stop/Continue)	Từ **Retro Sprint tới**	Mỗi Retro có ≥ **N note** (N = số thành viên); có ít nhất **3 action** được tạo

---

Bài 7:

Format	**Thời gian đề xuất**	**Phù hợp khi team…**	**Ưu điểm**	**Nhược điểm**	**Điểm số dễ dẫn dắt (1–5)**
Start – Stop – Continue	45–60'	Muốn **ra action cải tiến nhanh**, team đã tương đối ổn, có thể nói thẳng	Dễ hiểu, đi thẳng vào cải tiến, tạo action items nhanh	Dễ nông, ít đào root-cause; team ngại nói thật sẽ "viết cho có"	**5**
Sailboat	60–90'	Team cần **nhìn mục tiêu + rào cản + rủi ro**, muốn tạo động lực/định hướng	Trực quan, tạo "bức tranh chung", dễ ra ưu tiên (anchor/wind/rocks)	Nếu facil không chặt dễ lan man; ít hợp khi cần xả cảm xúc mạnh	**4**
Mad – Sad – Glad	45–75'	Team **căng thẳng, "âm", ít chia sẻ**, cần	Kéo được "sự thật" ra, tăng an toàn tâm lý, giảm căng thẳng	Có thể sa vào than phiền nếu không chốt action; nhạy cảm với team toxic	**4**

		mở khoá cảm xúc trước			
4L (Liked, Learned, Lacked, Longed for)	60–90’	Team muốn **tổng kết toàn diện** sau Sprint (được–mất–học–mong muốn)	Toàn diện, cân bằng tích cực/tiêu cực, dễ rút bài học	Nhiều mục nên tốn thời gian; dễ thành “kể chuyện” nếu không vote	**3**
KALM (Keep, Add, More, Less)	45–60’	Team cần **tinh chỉnh quy trình** (giữ gì, thêm gì, tăng/giảm gì)	Rõ ràng, hướng hành động, hợp cải tiến process	Khá giống Start/Stop/Continue nên thiếu chiều sâu cảm xúc; dễ chung chung	**5**

Bài 8 :

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

“Mình tạm dừng 30 giây nhé. Mục tiêu Retro là **cải thiện cách làm việc**, không phải tìm người chịu tội. Mình nhắc lại *Prime Directive*: mọi người đều đã làm tốt nhất có thể trong điều kiện lúc đó. Từ giờ mình đề nghị nói theo **sự kiện – tác động – nhu cầu**, không gọi tên công kích cá nhân. OK mình bắt đầu lại.”

Kịch bản xử lý từng phút (đề xuất khung 20 phút “dập lửa” + đưa về đúng đường ray)

Thời điểm	**Bạn làm gì / bạn nói gì (cụ thể)**	**Mục tiêu**
00:00–00:01	“Stop 30 giây. Mình thấy không khí đang nóng. Cho mình reset lại.”	Cắt vòng xoáy công kích
00:01–00:03	Nhắc Prime Directive + rule: “Nói về **process/system**, không nói ‘anh/chị’. Dùng ‘Tôi thấy… tôi bị ảnh hưởng… tôi cần…’”	Tạo an toàn tâm lý

00:03–00:04	Với người chơi điện thoại: "Mình xin mọi người **cất điện thoại 10 phút** để tập trung. Nếu có việc gấp, báo mình rồi ra ngoài 2 phút."	Lấy lại sự tôn trọng & tập trung
00:04–00:06	Can PO chen ngang: "PO giúp mình 1 việc: mình sẽ có **Parking Lot**. Ý nào 'không quan trọng' hoặc ngoài phạm vi, mình ghi lại, cuối buổi mình quay lại quyết định ưu tiên."	Giữ tiếng nói PO nhưng không phá nhịp
00:06–00:08	Chuyển format sang **Silent Writing**: "Giờ 2 phút, mọi người viết im lặng: *Sự kiện cụ thể xảy ra? (fact) – Ảnh hưởng? – Mình mong muốn gì?* Không ghi tên người."	Giảm đổ lỗi, tăng dữ liệu
00:08–00:10	Bạn thu note, dán lên bảng theo 3 nhóm: **Late Daily / Spillover / Bug late / Code quality / Handoff**	Chuyển từ cá nhân sang chủ đề
00:10–00:12	Dùng **Timeline mini**: "Bug được báo ngày nào? Ticket chuyển QA ngày nào? Code review ngày nào?" (chỉ hỏi dữ kiện)	Trở về "sự thật theo thời gian"
00:12–00:14	Khi Dev A đổ lỗi: "Mình nghe được *'tester làm muộn'*. Mình đổi câu hỏi: **Điều gì trong quy trình** khiến bug bị phát hiện muộn?"	Reframe khỏi công kích
00:14–00:16	Khi Tester B nói "code bẩn": "Mình đổi sang mô tả được kiểm chứng: **điều gì khó test/khó đọc?** Ví dụ: thiếu log, thiếu unit test, naming, không có checklist review?"	Từ phán xét → tiêu chí kỹ thuật
00:16–00:18	Chọn 1 vấn đề nóng nhất bằng **Dot Voting** (mỗi người 2 phiếu)	Chốt ưu tiên, tránh tranh cãi lan man
00:18–00:20	Dẫn **5 Whys** ngắn cho vấn đề top 1 (ví dụ "bug báo muộn") và chốt 2–3 giải pháp khả thi	Đi đến root-cause + hành động

Sau 20 phút, Retro đã về "đúng đường": tiếp tục 30–40 phút để đào sâu + 20 phút cuối để chốt action/owner/deadline/metrics.

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

Bạn dùng combo này (rất hiệu quả lúc căng):

1. **Silent Writing** (viết im lặng, không tranh cãi, giảm công kích)
2. **Timeline** (bám dữ kiện theo thời gian, tránh "cảm giác")
3. (Tuỳ chọn) **5 Whys** theo **process** để ra root-cause, không hỏi "ai sai".

(C) 3 câu "thần chú" khi có người đổ lỗi

1. "Mình chuyển từ *ai sai* sang **cái gì trong quy trình** đang khiến chuyện này lặp lại?"
2. "Bạn cho mình **1 sự kiện cụ thể** (ngày/giờ/ticket) và **tác động** của nó, không dùng nhãn 'lười/bẩn/kém' nhé."
3. "Mình ghi nhận cảm xúc đó. Giờ mình cần **đề xuất thay đổi 1 bước** để Sprint sau không gặp lại."

(D) Action Items để giải quyết tận gốc vấn đề "đổ lỗi" (có gốc kỹ thuật + gốc hành vi)

Bạn chốt ít nhất các action sau (đúng kiểu Ai–Làm gì–Khi nào–Đo lường):

1. **Dev Lead** – tạo **Code Review Checklist + Definition of Done** (unit test tối thiểu, naming, log, error handling) – **trước Sprint kế tiếp** – **Đo:** 100% PR có checklist tick, bug do "code quality" giảm.
2. **QA Lead / Tester B** – đặt rule "Ticket vào Ready for Test → test trong 24h" + báo bug ngay khi thấy – **từ ngày 2 Sprint tới** – **Đo:** ≥80% bug được log trước 2 ngày cuối Sprint.
3. **Dev + QA** – thêm "mid-sprint test checkpoint" (ngày 5/10) để handoff sớm – **Sprint tới** – **Đo:** không còn dồn bug ngày cuối.
4. **Scrum Master** – thiết lập **Working Agreement** Retro/Daily: không công kích cá nhân, nói theo "fact-impact-need", điện thoại để chế độ im lặng – **ngay từ buổi tiếp theo** – **Đo:** Retro không bị ngắt vì tranh cãi cá nhân; mọi người tham gia đầy đủ.
5. **PO** – dùng **Parking Lot** + chốt "ưu tiên sau demo" thay vì cắt ngang – **Sprint tới** – **Đo:** số lần PO chen ngang giảm rõ; vẫn ghi nhận đủ yêu cầu.