Hands-On Learining:

Agile QA Exercise: Interpreting QA Metrics & Improving Quality

Case Study

Overview

(this is the same organization and project that we’ve used in the previous exercises)

Streamline Retail is a mid-sized company that provides Point-of-Sale (POS) and inventory management solutions for small to medium retail stores. Their product suite includes a desktop management platform and a companion mobile app used by store managers to track inventory, process sales, and receive real-time analytics.

Over the past five years, Streamline Retail has grown rapidly, but its product quality has not kept pace. The development team recently adopted Scrum to improve delivery cadence and quality. QA processes are still evolving.

Project Scope: Streamline Mobile App 2.0

The goal of this project is to **redesign and rebuild the mobile app (iOS and Android)** used by retail store managers. The new version must:

- Provide **real-time inventory tracking**
- Allow **barcode scanning** for product lookup
- Display **sales dashboards**
- Support **offline mode**
- Integrate securely with the main backend

The project is expected to be delivered in **5 increments over 4 months**, with an MVP ready by the end of sprint 3.

Your role

As a QA engineer, you are responsible for improving quality. You’ve been collecting key QA metrics for three months since the launch of the first product increment.

Metric	Month 1	Month 2	Month 3	Observation
Test Coverage	55%	65%	72%	Increasing, but still below ideal (80%+)
Test Reliability	90%	85%	80%	Dropping, suggesting flaky tests or environment issues
Mean Time to Detect (MTTD)	5 days	3 days	2 days	Improving, but defects still escaping to production
Mean Time to Repair (MTTR)	7 days	6 days	5 days	Slow but improving bug fixes
Customer Satisfaction (CSAT)	60%	64%	68%	Increasing, indicating users enjoy the product.
Test Cycle Time	4 days	3 days	3 days	Stabilized, but is it fast enough?
Defect Leakage	25	22	18	High, meaning many issues reach production
User Adoption Rate	40%	42%	38%	Dropping in month 3, indicating user frustration

Hands-on Exercise

The task

1. Identify trends in the metrics: What problems can you infer?
2. Propose 3-5 actionable improvements to address the most critical quality issues.
3. Explain how your proposed changes would impact specific metrics.

Instructions

Create trial account

1. Look at the metrics and try to interpret them in terms of what they mean.
2. Categorise trends into positive and negative ones.
3. For ones you consider negative, try to think of potential root causes. Feel free to make any assumptions needed that are not covered by the case study.
4. For the root causes identified this way, propose actions.
5. Explain how these actions are going to address the root cause, and which visible improvement you are likely to see in the metrics in the future.
6. For the trends you consider positive, think if expediting the trend is needed (to speed it up or scale it even further). If the answer is yes, propose actions to do so.

PAUSE HERE AND DON’T SCROLL UNTIL YOU’VE ATTEMPTED THE EXERCISE.

On the next page, there is the example solution. If you want to attempt this on your own, pause here and don’t scroll next until you are ready.

There are many different ways to accomplish this task, it is ok if your solution is different. Compare your solution with the one provided, if there are discrepancies think if you can explain them.

Example Solution

Observed Issues & Trends:

- **Test Coverage is improving**, which could allow critical bugs to slip through. This is a positive trend. However, **the overall coverage was below the target** level for the whole time, which is a negative trend.
- **Test Reliability is declining,** suggesting tests are failing inconsistently. This is a negative trend.
- **Defect Leakage remains high,** meaning many defects escape into production. This is a negative trend.
- **Customer Satisfaction is rising**, indicating users enjoy the app overall despite leaked defects. This is a positive trend.

User Adoption Rate declined, signaling that users who are frustrated are abandoning the app.This is a negative trend.

MTTD and MTTR are improving. This is a positive trend.

Overall, based on the trends that we see we can assume that our test suites are not providing enough support for the project. They are not very reliable and allow for leaking defects into production. CSAT being on the rise is a rather positive sign, but it happens together with dropping adoption rate which indicates that unhappy users prefer to just drop the product (and as a result potentially not participate in CSAT survey) making it somewhat skewed. We don’t know the reasons people abandon the solution, but can assume the number of production defects may be a contributing factor. MTTD and MTTR are improving, but potentially not improving fast enough for users not to drop the solution when faced with a defect.

Proposed Solutions & Expected Impact

Solution strategy	Potential steps	Impacted Metrics	Expected Outcome
Increase automated test coverage	Make a sprint dedicated to increasing test coverage. Aim at 80% or higher. Revisit the definition of done to ensure automated tests are created.	Test Coverage Defect Leakage Test Cycle Time	Low coverage is a potential root cause of some of the issues. Addressing it we expect to reduce escaped defects and speed up testing.
Improve code review quality	Create checklists for code reviews to cover things like existence of unit tests and coverage.	Test Coverage Defect Leakage	One reason for low test coverage may be that engineers find it acceptable to ship code that is not covered by tests. By introducing clear code review standards you may address this issue.

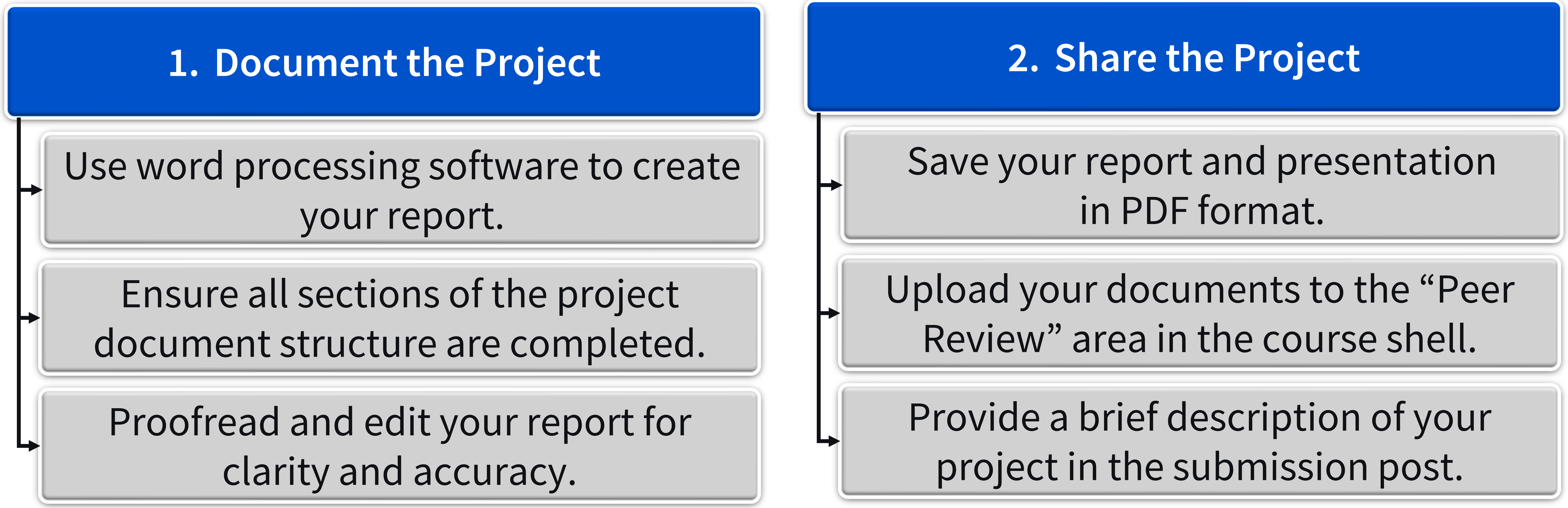
Proposed Solutions & Expected Impact

Solution strategy	Potential steps	Impacted Metrics	Expected Outcome
Stabilize unstable tests & improve CI/CD reliability	Perform analysis of tests that produce false positives most often; identify patterns and potential causes. Suggest a sprint focusing on addressing technical debt and optimising failing tests.	Test Reliability Test Cycle Time	Potential root causes addressed by this action is technical debt and low quality of engineering in automated tests. Solving it ensures consistent test results and reliable deployments
Introduce beta testing with real users	Identify target audiences; define which audiences are more likely to stop using the product by analysing the characteristics of users who dropped the product. Recruit people from those categories to participate in surveys, interviews, focus groups, or beta testing.	Customer Satisfaction User Adoption Rate	Dropping adoption rates may indicate that users are not getting the value they expect from the solution. The best way to validate this assumption is to ask the users. This identifies usability and performance issues, as well missing features before the release.

Proposed Solutions & Expected Impact

Solution strategy	Potential steps	Impacted Metrics	Expected Outcome
Improve developer-QA collaboration via shift-left testing	Introduce three amigos meetings. Introduce defect triaging huddles when testers and developers work together on defects without handover times.	Test Coverage Defect Leakage MTTR	Close collaboration catches bugs earlier, reducing production defects. It also directly addresses MTTR by reducing the time it takes to fix the issue and verify defect resolution.

Instructions for Documenting and Sharing The Project for Peer Review



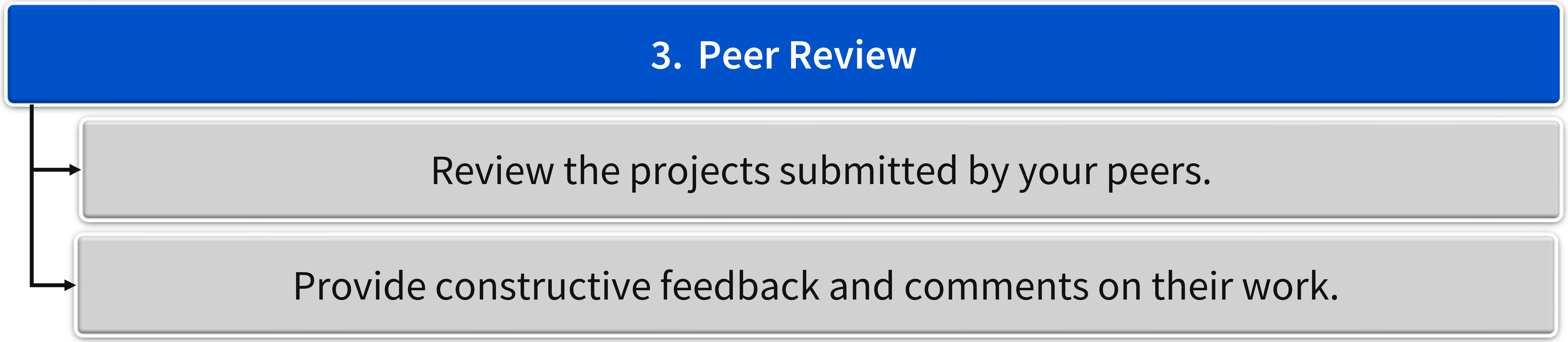
1. Document the Project

- Use word processing software to create your report.
- Ensure all sections of the project document structure are completed.
- Proofread and edit your report for clarity and accuracy.

2. Share the Project

- Save your report and presentation in PDF format.
- Upload your documents to the “Peer Review” area in the course shell.
- Provide a brief description of your project in the submission post.

Instructions for Documenting and Sharing The Project for Peer Review



3. Peer Review

Review the projects submitted by your peers.

Provide constructive feedback and comments on their work.