Quality Assurance makes the systems robust and error prone by detecting the issue before the deployment. However it is very challenging to run QA tests after creating the complete product.

Below project plan ensures that we can detest the issues early in the development before it is too big to manage.

**Integrating QA Early**: A Turnaround Plan for Improving Code Quality

As the project manager, the report from the QA team indicating that only 10% of the submitted code passes foundational QA checks raises a critical red flag. This indicates not only a lack of testing but also a systemic issue in how quality assurance is positioned within our development lifecycle. To address this, I would initiate a quality turnaround plan based on DevOps principles and Shift-Left testing bringing QA into the process from the earliest stages of development.

* **Transparent Communication and Root Cause Analysis**

First, we schedule a cross-functional meeting involving development, QA, and architecture teams to openly communicate the QA findings. The tone will focus on improvement, not blame. Using data from failed QA checks, we’ll identify patterns of failures due to logic errors, missing test cases, or poor requirements? This collaborative analysis will drive a shared understanding of quality expectations and gaps.

* **Integrating QA into the Dev Lifecycle**

Based on DevOps best practices, QA should not be a involved only at the end of development but a continuous part of the build process. We'll implement "Shift-Left" QA strategies by following the below steps

1. Involving QA in requirements and design reviews.
2. Writing unit and integration tests during development, not after.
3. Enforcing pull requests to include automated test coverage.
4. Using CI/CD pipelines to automatically run tests on every build.

* **Training and Ownership**

Many QA failures stem from skill gaps or unclear ownership. I would arrange workshops on test-driven development (TDD) and behavior-driven development (BDD) for developers and testers. Additionally, every sprint will include dedicated QA stories and peer reviews, reinforcing that quality is a shared responsibility.

* **Continuous Feedback and Adjustments**

Finally, I would establish a weekly QA retrospective to review progress, address issues, and recalibrate the plan. QA will become a continuous loop not a late-stage bottleneck.

By embedding QA into the DevOps pipeline and fostering a quality-first culture, we can transform our process from reactive to proactive ultimately delivering more reliable software, faster.

# **References**

Kim, Humble & Willis The DevOps Handbook

<https://learning-oreilly-com.csuglobal.idm.oclc.org/library/view/the-devops-handbook/9781098182281/00-Cover.xhtml>

**Applying Unit Testing, TDD, BDD and Acceptance Testing** by [David Sale](https://learning.oreilly.com/search/?query=author%3A%22David%20Sale%22&sort=relevance&highlight=true)

<https://learning-oreilly-com.csuglobal.idm.oclc.org/library/view/testing-python-applying/9781118901243/?ar=&email=MT3OraT%2B%2FQPH11KS%2F7Dup5a1kwBcMookmqoXI%2BcUMLbAvNW2BuWroTeSYIPg6DX6&tstamp=1747358881&id=F648E41B8D9F813CC7C94FFA96CD31FDE3D31858>