### **Feedback on Improving Quality in the Team**

#### **1. Requirement Clarity & Refinement**

* Encourage **early collaboration** between developers, testers, and product managers to refine requirements before development begins.
* Implement **formal requirement reviews** where testers can provide feedback on testability and edge cases.
* Maintain a **well-documented PRD (Product Requirements Document)** with clear acceptance criteria and expected outcomes.

#### **2. Test-Driven Development (TDD) & Behavior-Driven Development (BDD)**

* Promote **TDD practices** to improve code quality by writing test cases before development.
* Adopt **BDD (Behavior-Driven Development)** using Cucumber to bridge the gap between business and technical teams.

#### **3. Shift-Left Testing Approach**

* Introduce **unit tests** as part of development to catch defects early.
* Automate API and integration tests to validate backend functionality before UI testing.
* **Enforce static code analysis tools** (SonarQube, Checkstyle) to ensure code quality.

#### **4. CI/CD Pipeline Enhancements**

* Ensure every pull request triggers **automated regression tests** to avoid regressions.
* Include **performance and security testing** in the CI/CD pipeline.
* Establish clear **test execution reports** with logs and failure analysis to debug issues faster.

#### **5. Improved Test Automation Strategy**

* Enhance test automation coverage, focusing on **critical business flows first**.
* Improve the **stability of test scripts** by handling flaky tests proactively.
* Encourage **parallel execution** to reduce test execution time.

#### **6. Code Review & Pair Testing**

* Implement **stricter code review processes** where automation engineers also review automation scripts for reusability and efficiency.
* Encourage **peer testing**, where developers test each other’s features before handover to QA.

#### **7. Continuous Learning & Upskilling**

* Organize **knowledge-sharing sessions** on automation, performance testing, and security testing.
* Encourage certifications and workshops on **new testing frameworks and tools**.

#### **8. Post-Release Monitoring & Bug Analysis**

* Set up **production monitoring alerts** to catch issues early.
* Perform **root cause analysis (RCA)** for high-priority defects and document preventive actions.