Mindset for long run

1. What is Quality

In this mini-doc, we use the term **quality management** rather than quality assurance. Information from testing can help improve the quality but can't "assure" the quality is there. The most popular definition seems to be one from Jerry Weinberg - "**Quality is value to some person**". W Edward Deming defined good quality as: "**predictable degree of uniformity and dependability with a quality standard suited to the customer**". People talk about quality as if there is only one kind – all encompassing, but it's not. For example, product quality is about quality of finished product – what our customers experience. When teams try to measure quality, it's usually about how well they adhere to their processes which is really something different. There are other stakeholders who also have an interest in a product's quality. There are many lenses in how people view quality.

Process quality

In the development cycle, the emphasis is on the practices followed to build the product. The focus is on defect prevention and limiting rework, and this is often where many teams spend their time building it right and measuring the quality of the process. Many teams target testing in this perspective.

Some processes that support quality are: TDD (test-driven development), coding for maintainability, peer reviews, continuous integration, exploratory testing or even conforming to the "Definition of Done". Teams measure process quality, the technology-facing quality that the team owns, the practices that help them build quality in. They aim to answer the question: "Are we building it right?"

Product-based quality

People often assume that using good ingredients should make a good product. If quality is viewed as an inherent characteristic of goods, higher quality means better performance, and enhanced features – things that may increase the cost of the product. This thinking may be slightly flawed. Using the metaphor of cooking, an ordinary cook with good ingredients may not have a good outcome, but a great chef with ordinary ingredients may produce something magical. We need to understand what our product is and how it is put together.

Some processes that teams can do to help support product quality are: ATDD (acceptance test-driven development) or BDD (behavior-driven development). Testing quality attributes such as security, performance or reliability is another example. The question to be answered is: "Does it work as expected (or desired)?"

User-based quality

The user-based perspective is what is most often used to talk about quality, and it is highly subjective and highly personal.

It assumes that consumers possess sufficient information to evaluate product quality. If they do not, they will rely on other cues when making that assessment. Let's consider a cup of coffee. Some folks prefer a simple black medium roast coffee to a well-made cappuccino, but others take the cappuccino every time. Who is your consumer? Who is evaluating?

2. Team Player

Value:

Detailed-oriented + Independent for end-to-end test + Great verbal_written skills + Technology ambition + Punctual

Character: Whoever you want to be in the team

Responsible: Product/Process/User-based Quality

3. Career Path

[3-6 months]	Formal Employee +		Project Members	
Interview 1:	6 weeks	+	8 weeks	[Just an observation]
Interview 2:	15 weeks	+	18 weeks	[Just an observation]

[1 –2 years] Technical Specialist

Annual Review 1: Discussion (Nov)

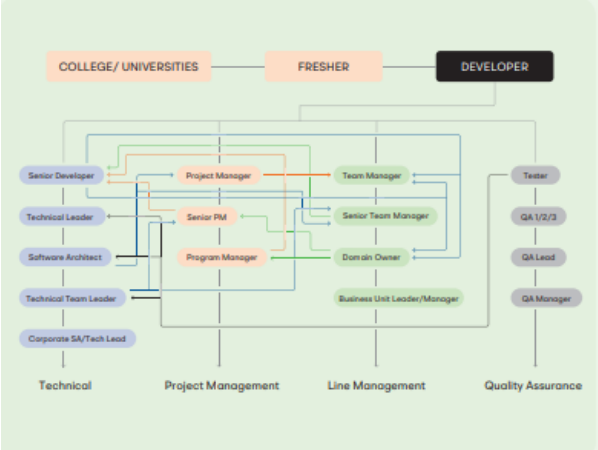


Figure: Career path provide in FPT Software – Start With You Program

?? IVS – Structure of Test Team: Who and What

4. Reward

- Salary: ???
- Credit: ???
- Award: ???
- Opportunity: ???

Capstone 1: Fresher (3-6 mths) [craft skills]

1. Overall

To become an Advanced QA Engineer, it's crucial to explore advanced testing topics and stay updated with emerging technologies:

1. 🌐 Web Services/API Testing:

🌐 Testing APIs using Postman and REST Assured: Gain hands-on experience in testing APIs using popular tools like Postman and REST Assured.

🚀 API Automation and Validation Techniques: Learn how to automate API tests and validate API responses.

2. ♿ Web Accessibility Testing:

🌐 Understanding WCAG Guidelines: Learn about the Web Content Accessibility Guidelines (WCAG) and the importance of web accessibility testing.

🧭 Using Tools like Axe and Lighthouse for Accessibility Testing: Familiarize yourself with accessibility testing tools like Axe and Lighthouse.

3. 🔄 Test Automation Maintenance:

🚧 Strategies for Maintaining Automated Test Scripts: Discover best practices for maintaining automated test scripts to reduce flakiness and improve reliability.

🎯 Reducing Flakiness and Improving Reliability: Learn techniques to minimize flakiness and improve the stability of your automated tests.

4. 🗃️ Test Data Management:

📝 Creating and Managing Test Data: Understand the importance of test data and how to manage it effectively for different test scenarios.

🎲 Data-Driven Testing Approaches: Explore data-driven testing approaches to optimize your test coverage.

5. 🌌 Emerging Technologies:

🤖 AI/ML Testing Applications: Discover the role of Artificial Intelligence and Machine Learning in software testing.

🌐 IoT Testing Challenges and Solutions: Learn about the unique challenges of testing Internet of Things (IoT) devices and systems.

2. Skills Training Program

- Environment: Shadow Position + Mentor + Team Support
- Common Weekly Tasks:

Basic Agile SDLC + Test Suite Writing + Regression/Functional/Smoke Test Executing + Test Script Writing/Integration + Bug Investigate/Raise/Closed + Requirement Analyze + Self-study Automation Skills

- Focal of phase 1 [3 months 4/2024 - 7/2024] is crafted auto skills by tool listed on the table below

Automation Tools	**ntermediate Auto**	**Status**
Git	Continuous Integration	Done
Azure CI/CD, Repo	Continuous Integration	80%
Visual Studio Code	Continuous Integration	Done
Jerkins	Continuous Integration	20%
DBeaver (SQL)	Database Testing	Done
Katalon Studio	Develop TS	Done
Cucumber Framework	Develop TS	Done
Selenium	**Develop TS**	**30%**
X-path	Develop TS	Done
Fundamental API	Develop TS	0%
Web Service Testing-Frontend	Experience	70%
Web Service Testing-Backend	**Experience**	**0%**
Performance Testing	**Experience**	**0%**
Python (Advanced)	Programming Language	60%
Java OOP, Array/List, Casting	**Programming Language**	**70%**
Azure Test Plans	Test Maintenance	Done
Jira Test Plans	Test Maintenance	0%

- Focal of phase 1 [3 months 4/2024 - 7/2024] is on project experience with these responsibilities:

Well-prepare for Clients interview, Hone Test Management Skills, Take full-responsibility for Manual Tasks. Illustrate by the Holistic Testing Model Graphic:

Reviewed: Hieu Ho Duc Anh (6pm - 29/7)

3. Better Reference Better Performance

Overall

- Testing Bible: Smoke Testing - javatpoint

- Testing Interview Question Bank: Software Testing Core.xlsx (sharepoint.com)

Communication

- SDLC: UAT/Deployment,
- Interview: PET TRAINING Summary.docx (sharepoint.com)
- Good friend: I used to watch TV series “SUIT”

Manual

- Folder: Readme_Manual (sharepoint.com)
- Label status while execute
- Investigate bug by all 3 layers
- Keep in mind while write TCs: readable, execute regression and smoke
- Leverage test design skills by execute regression test case (GUI, Functional, API, Data, Performance...)
- After Production Smoke Test --> Hotfix --> Bug Deployment

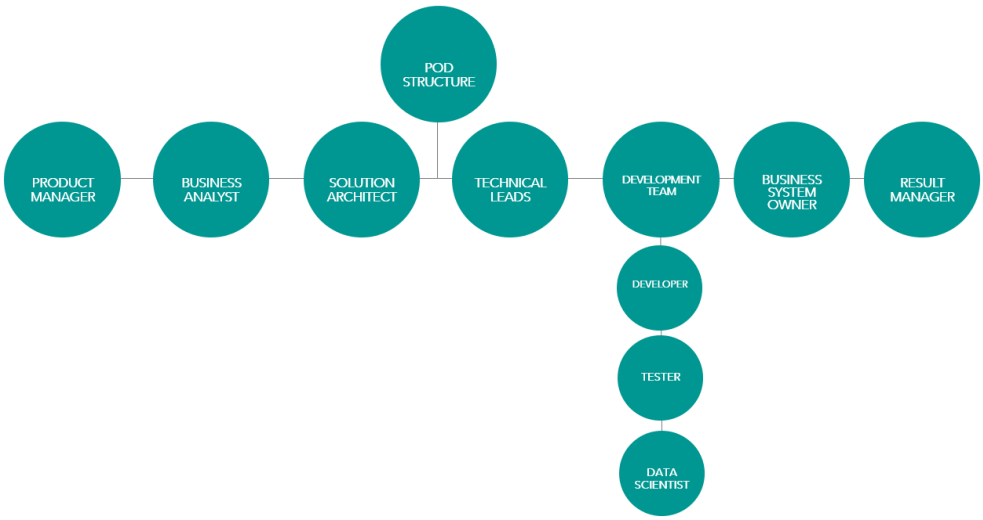
Automation

- Collaborate with Git: Gitflow Workflow | Atlassian Git Tutorial
- Leverage Back-end Testing: Readme Backend Testing (sharepoint.com)
- Check the automation set-up plan after UAT or deployment

Performance

P/s: Detail components to master or discuss and methodology to craft skills are all here

Capstone 2: QA2 (1 - 2 yrs) [continuous proj]



1. Real-world Project: Advanced QA Engineer Skills

Quality is woven into the fabric, Figure 5 is a great visual that goes along with phrase of “Building quality in”. In this capstone, Quality Strategy is the ultimate responsibility, we have testing supports the quality discussion for both products and process quality.

The left side of the loop shows testing done early in the cycle. It creates shared understanding that guides development and prevents defects. The right side of the loop is about what has been built, and finding defects that escaped the build process. It’s also for learning from customer usage and gathering data to help plan the next changes.

Recommend an approach for improve your QA skills as Bottom-up and Big Bag Approach restrict by SDLC project experience. Make sure you have continuous project experience for at least a year to have an evaluation for below task:

+ TCs Writing: Faster, Better Bug Coverage, Re-usable
+ Collaborate: More Convincible, Reasonable, Reliable
+ Test Plan/Management: Right Time, Right People, Right Method, Right Words
+ Trouble Shooting and Problem Solving: Need less support

2. Mentor Role

- People reach out to you for help with your technical and management problem