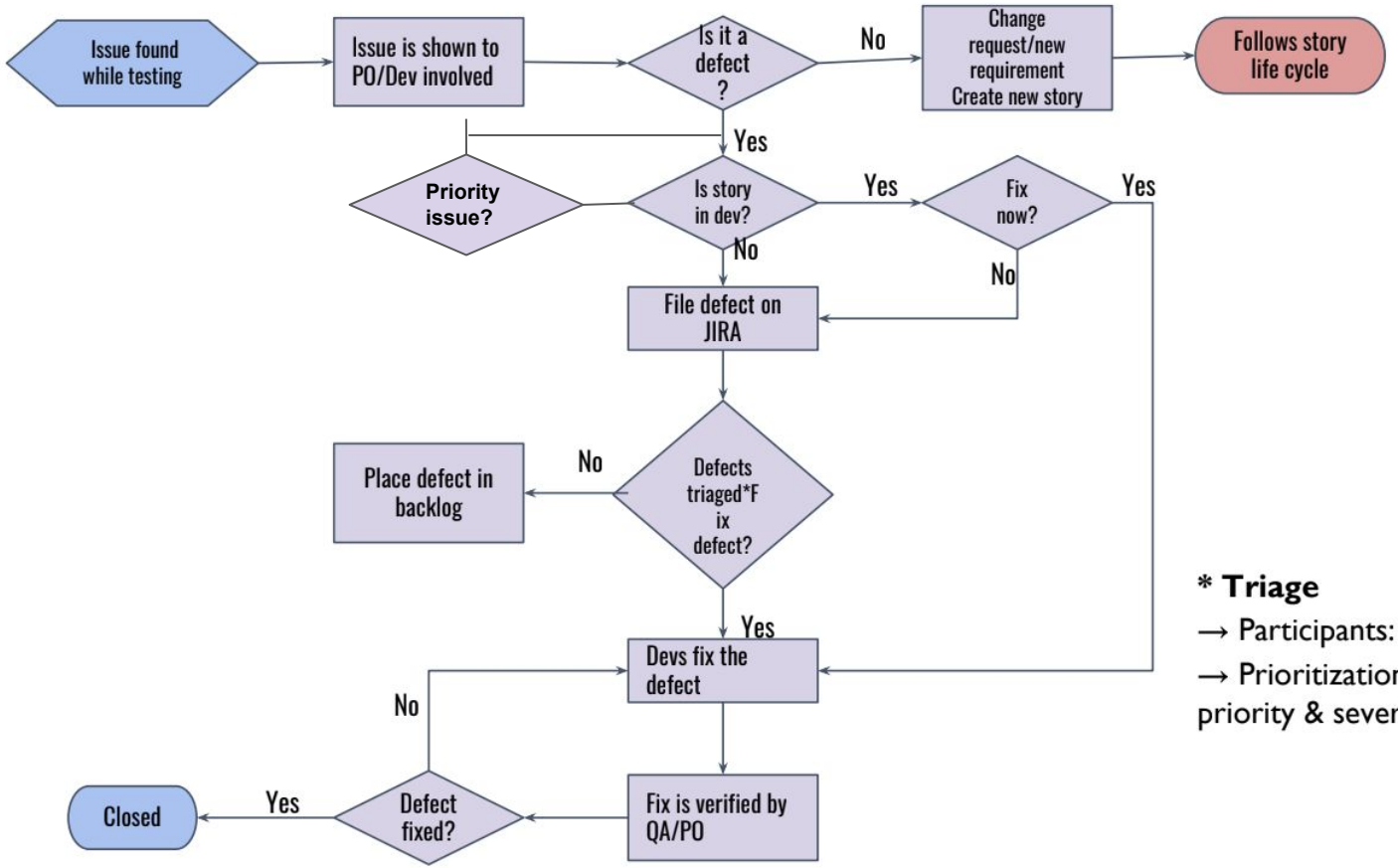
QA Practices

Falabella

Defect life cycle

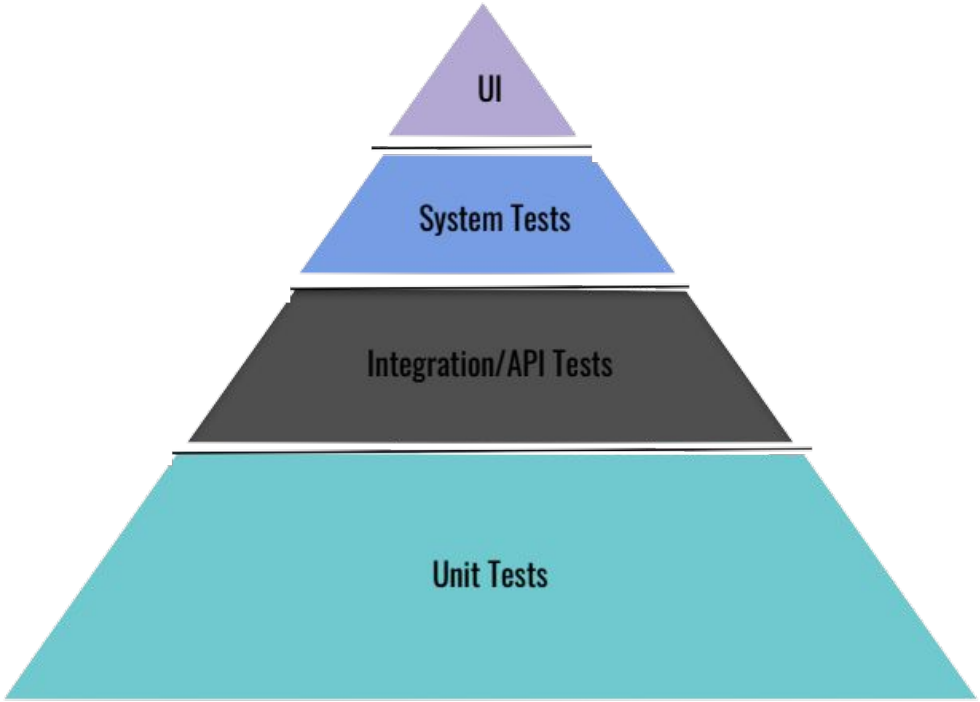


*** Triage**

→ Participants: QAs, TL/Any devs, PMs.

→ Prioritization based on defect priority & severity

Test Pyramid



Tools

1. Unit Testing: jest
2. Contract Testing: Pact with JS
3. API Testing: RestAssured with Java (Framework Exists, can be reused)
4. Performance/Load Testing: Locust with Python (Framework Exists, can be reused)
5. Security Testing: OWASP / Zap
6. Monitoring Tool: Graffana
7. Logging Tool: Fluentd

Other Practices

- Defect Triaging
- Bug Analysis
- Automation Report
- Security Analysis
- Bug Bash
- Define Regression Test Scenario
- Document to create visibility

Practices

- IPM
- Story review
- Kick off
- Dev Huddles / Team Huddles
- Dev Box
- Testing
- Showcase/acceptance
- Bug bash
- Regression/Release testing

> “Why” is more important than “What”

Iteration Planning Meeting

Why ?

- To define iteration deliverables

When ?

- Every iteration

What happens ?

- Review and estimate stories/tech-tasks
- Finalise iteration deliverables
- Callout Risks, Assumptions, Issues, Dependencies

Who takes part ?

- Team along with Product owner

Story Review

Why ?

- To review the analysis for clarity and completeness

What happens ?

- QA reviews the analysis and finds out missed/incorrect understanding of purpose, scope, assumptions, dependencies and acceptance criteria
- Share the findings to the BA for further analysis, if required

When ?

- Every story/task

Who takes part ?

- QA and BA for functional stories
- QA and Dev for tech tasks

Kick Off

Why ?

- To establish a common understanding of what is expected from the story/task
- Minimise rework

What happens ?

- BA explains the purpose, scope, assumptions, dependencies and acceptance criteria
- QA captures testing scenarios and calls out edge cases
- Devs capture dev-tasks

When ?

- Every story/task

Who takes part ?

- BA, UX, Devs and QA

Dev Huddles / Team Huddles

Why ?

- To brainstorm a problem and come-up with a solution as a group

What happens ?

- The pair initiating the huddle talks about the problem and options available
- Brainstorming as a group to add/scrutinise options and finalise an implementation approach
- QA internalises the impact of the decided approach and applies it for testing

When ?

- Whenever a dev-pair or a team member has a need

Who takes part ?

- Devs and QA in case of tech/dev huddles
- Everyone in case of team huddles

Dev Box

Why ?

- To provide immediate feedback on stories to devs
- Minimise rework and reduce cycle time

What happens ?

- Devs callout when story nears dev-completion
- BA and QA will validate the story implementation in developer machine by using the acceptance criteria
- Any observation/feedback will be captured and addressed immediately

When ?

- Every story/task

Who takes part ?

- BA, QA and Devs for functional stories
- QA and Devs for tech tasks

Testing

Why ?

- To ensure a bug-free implementation of the story

What happens ?

- QA tests the story using the test scenarios and/or acceptance criteria
- Captures defects/observations in the story
- Calls for Dev signup (or BA signup)
- Writes automation at required levels
- QA creates separate cards to track bugs which need not block the story

When ?

- Every story

Who takes part ?

- QA drives it and includes Dev/BA on a need basis
- Automation is a combined responsibility between QA and Dev

Standup

Why ?

- To share progress and callout blockers which can feed into signups

What happens ?

- Every team member shares the progress of the task he/she is working on
- Blockers are called-out explicitly so that appropriate actions are taken to unblock
- Usually signups happen following a stand-up

When ?

- Everyday

Who takes part ?

- Everyone in the team

Showcase / Acceptance

Why ?

- To get feedback on the progress of functionality or application

What happens ?

- QAs showcase new features to clients
- PO accepts or gives feedback
- Get quick feedback on evolving features
- Display iteration metrics (feature completion, burnup, velocity)

When ?

- Every iteration

Who takes part ?

- Team along with Product owner

Bug Bash

Why ?

- To quickly uncover issues in exploratory testing mode
- To evaluate app stability

What happens ?

- Entire team sits together and tests the app for a timeboxed duration
- Split features to users (in case of huge app)
- Capture any issue/concerns in a centralized place
- Recreate, triage and follow-up

When ?

- Before every release
- After a feature-set completion

Who takes part ?

- Everyone in the team takes part
- Triage and prioritization by QA, BA and PO

Regression/Release testing

Why ?

- To perform effective regression leveraging automation coverage
- To give a GO or NO-GO for a release

What happens ?

- QA tests regression scenarios which are not covered by automation
- Followed by defect listing and triaging
- QA performs sanity testing when release branch gets promoted to UAT/Preprod environment

When ?

- Before every release
- After any major refactoring

Who takes part ?

- QA drives it and includes Dev/BA on a need basis

Responsibilities of QA

(Apart from Story testing)

- ➔ Defining overall test strategy
- ➔ NFR testing
- ➔ Drive test automation & CI
- ➔ Release management
- ➔ Defect management

Defining overall test strategy

- **Key is to understand the solution architecture**
- Knowing different components and subsystems involved
- Knowing the integration points in the ecosystem
- Understanding the scope of testing
- Defining the different types of testing

NFR testing

- **“How a system is supposed to be”**
- **“Architecturally Significant Requirements”**
- Some non functional requirements:
- Performance
- Security
- Auditability
- Scalability
- Portability

Drive test automation & CI

- Automation is our never-tiring assistant
- Optimal automation coverage
- Ensuring required CI builds/pipelines exist and are visible
- Inferences from CI and actions to follow-up

Release management

- Release planning
- Managing risks and mitigation planning
- Organizing bug-bashes
- Keeping release testing optimal
- Managing release tasks
- Release communication

Defect Management

Defect classification

- Story level defects
- Regression defects
- Cosmetic defects
- Non-functional defects

Defect Management

- Lean & effective tracking
- Bug-bash and defect triaging

Work environment & team dynamics

- Dining culture: Co-located and collaborative
- Flat, open and sign-up culture
- Self disciplined teams
- Cross pollinate by pairing
- Feedback is fundamental