ISTQB foundation level

SDLC -> Software Development Life Cycle Models like Waterfall and Agile

SRS -> Software Requirements Specification

HLD -> High Level Design

Planning for testing is the most important part of testing.

Difference of Vocabularies:

Quality Assurance: Adherence to process, it is `proactive and prevents`, are we doing the right thing

Quality Control: Testing, test design, test execution. `Reactive and detects`. So Testing lives IN the quality Control.

Main umbrella is Quality Management, under which is QA, under which is QC, under which is Testing.

Error, Defect, and Failure are not the same thing.

Error is a mistake. lets say customer and software team are not on the same page on understanding stuff.

Defect is Bug. It is a mistake `in the software.`

A bug(defect) may or may not lead to the failure of a software product.

Failure is simply an observable defect.

These should be caught in testing.

7 Testing Principles

- Testing shows the presence of defects, not their absence.
- Exhaustive testing is impossible, nor practical. An important sampling and testing on it is smart choice.
- Early testing saves money and time.
- Defects cluster together(1 cause another)
- Pesticide Paradox: To find new bugs, you need to make reviews and changes to the test.
- Testing is content dependent. Which app requires more security testing.
- Absence of error is fallacy. Meaning even if you catch tons of defects, you cannot guarantee there is not anymore.

Test Process Fundamentals.

- Test Planning

What test techniques do you use and how? Schedule and deadlines?

- Test Monitoring and Control

What was planned vs what is actually happening? Are we on track on plan and deadlines?

- Test Analysis

Business requirement, functional and non-funcional requrements, designs, documenation,code etc.

All of these are test basis upon which the tests will be written.

- Test Design and Implementation

All so far was to analyze what to test. Now it is time to how to test.

This process entails designing and prioritizing the tests cases, identifying test data.

Design is how to test, implementation is `Do we have eveything we need to test?`

Implemenattion entails creating procedures and automated test scripts, setting up environment,preparing test data.

- Test Execution and Completion

SUT -> System Under Test.

In this step you execute the designed and implemented tests and compare the results with the expected results.

Test completion means creating a test report, also ensureing that all defetc report are documented.

SDLC Models

Regardless which model you choose, there is a common ground of stages:

- Requirement analysis
- Design
- Development
- Testing
- Deployment
- Maintenance

There are 2 Main Models:

1. **Sequential**

- V-Model:

Unlike the waterfall model, the V-Model integrates the stages with testing. Main idea in V-Model is `Early testing`

Each development phase is linked with a testing phase like

low-level design ==> Unit testing.

High-level design ==> Integration testing.

System-design-and-development ==> System testing.

Analysis ==> acceptance testing.

Unlike Waterfall, which is a linear process, the V-Model is more synchronized approach. Testing starts earlier than waterfall.

- Waterfall

simple and in-bulk deployment but not flexible and could be slow. one blockage could cause next steps to be blocked.

Also any phase can become a bottleneck and very late feedback threat.

It adds additional stress to testers.

Each of these steps given above are done sequentially for each big feature requested.

First business analyst makes analysis, then it is designed, then devs develop it and it is handed to tester as version 1.0,

then testers test, if bugs, send back to devs, they fix them, deliver the next version (1.1) and so on then delivered to customer.

then as the situatuion requires, minor updates and maintenance is undertaken.

2. **Iterative(incremental)**

instead of working in bulk, you work in small increments and in each step, you test and fix the bugs. so instead of 10 features

at a time, you work on one or two features at a time.

In this model, work is divided into features or by `fixed time cycles`

Instead of making one major activity at a time, you do everything in small increments in fixed time cycles.

- RUP:
-

Rational Unified Process. bigger deliveries

longer iterations.

- Scrum:

iteartions are shorter, but more frequent. smaller deliveries Part of AGILE

- Kanban:
- Part of AGILE fluid roles, tasks are shared by everyone, no scrum-master, timelines evolve.
- Spiral:

experimental increments, most flexible model.

Test Levels