**Day 6: Test Case Design and Environment setup**

Writing test cases:-

* A test case is a series of steps to verify a particular functionality or flow. A test case does nothing except verifying a particular functionality or flow.
* Well a test case should have fields like test case ID, Test Case Name, Objectives, Pre-condition, Test Data, Steps, expected result, Actual Result, status, defect(if any), Comments(if any)
* Test cases should be designed on functional requirements
* Test cases should be peer reviewed and updated frequently

(Note: A test case is different from test scenario which covers number of flows.)

**How to write a good test case:**

* Keep it simple, clear and provide coverage
* Get expected result from requirement, not current application behavior
* Do not repeat test case
* Follow navigational flow
* Keep the end user in mind
* Keep a balance between positive and negative scenarios
* Review the test case, and keep them updated

Use test case design techniques for sufficient coverage.

Test Case Design Techniques:-

**There are four common test design techniques:**

1. Equivalence Partitioning – based on dividing the test conditions into partitions from which test cases are designed
2. Boundary value analysis – based on testing the boundaries of partition, instead of center
3. Decision table – based on tabular representation of various inputs and their output
4. Use case testing – based on use cases (user interaction from start to end point).

Reviewing Test Cases:-

* Test case review is an important activity that should always be done.
* Even if it is not explicitly mentioned, test case design should include test case review also.
* Test cases should be reviewed by a peer or by your supervisor; even we can review our test cases ourselves.
* Test case is reviewed:

- to ensure coverage

- to find errors

- to ensure compliance to test case design practices

* Test case review should be documented for future reference.

Maintaining Test Cases:-

* As applications change over time, test case maintenance is needed.
* If test cases are not maintained, it can cause incorrect testing, missed scope or increased effort (especially for new resources)
* In an automated project, test case maintenance also requires maintenance of automation scripts
* Maintenance can be simplified by
  + Keeping a track of new requirements and finding the impacted test cases.
  + Keeping test cases in a modular structure.
  + Updating test cases rather than creating new ones.

Test Management Tools:-

* The test management tools were used mostly for creation and execution. But now-a-days they support a lot of activities like reporting, defect tracking, RTM, requirement mapping, etc.
* The common features of STLC are supported by all test management tools
* The difference is in cost, integration with third party API, scalability
* Most common test management tools in market are (2020):
  + ALM
  + QTest
  + PractiTest
  + TFS
  + TestRail

Test Environment Setup:-

* A test environment is setup of hardware and software for testing team.
* Setup is done by system admins, developers, testers, Run smoke test to certify environment health.
* The test environment should be isolated from production environment.
* Common test environments are:

- DEV – where development happens (and unit test)

- QA – where testing happens (regression, re-testing etc.)

- UAT/Pre-Prod – the data more closely mimics production (Acceptance test)

- Production – Live environment (user testing)

- Additionally – Performance – for load Testing.

* Testers spend around ~30-40% of their time setting up test data
* Inadequate or incorrect test data can cause invalid or missed testing.
* Test data should be
  + realistic,
  + valid
  + Versatile.
* Test data can be created or copied from production.
* Boundary value analysis, equivalence, partitioning, acceptor techniques etc should be used for test data creation
* Automated test generation tools are also used nowadays for testing to generation.