Software Testing #2

7 Principles of Testing

7 Principles of Testing

- Testing shows presence of defects
- Exhaustive testing is impossible
- Early Testing
- Defect Clustering
- Pesticide Paradox
- Testing is context dependent
- Absence of Error – Fallacy

1. Testing shows presence of defects

- Testing shows the presence of defects in the software.
- The goal of testing is to make the software fail.
- Sufficient testing reduces the presence of defects
- In case when QAs are unable to find defects after repeated regression testing doesn't mean that the software is bug-free
- **Testing should talk about presence of defects not about absence of defects**

2. Exhaustive testing is impossible

- Yes! Exhaustive testing is not possible
- Exhaustive testing
  - also known as complete testing
  - Testing all the functionalities using all valid and invalid inputs and preconditions is known as Exhaustive testing
  - It is a quality assurance testing technique in which all scenarios or data is tested for testing.
- It is not possible to test for all probable outputs or sequence of operations with exhaustive testing.

3. Early Testing

- Testing should start as early as possible in the SDLC
- Defects in the requirements or design phase are captured in early stages are less expensive to fix
- Conducting early testing reduces the cost of fixing defects
- Bugs in testing phase vs bugs in requirement/design phase

4. Defect Clustering

- Pareto principle-
  - known as the **80/20 rule** or **law of the vital few**
- Defect Clustering means that a small module or functionality contains most of the bugs or it has the most operational failures.
- 80% of issues comes from 20% of modules and remaining 20% of issues from remaining 80% of modules
- Emphasize testing on the 20% of modules where we face 80% of bugs.

5. Pesticide Paradox

- Repetitive use of the same pesticide mix to eradicate insects during farming will over time lead to the insects developing resistance to the pesticide
- The same applies to software testing as well
- If the same set of repetitive tests are conducted, the method will be useless for discovering new defects
- the test cases need to be regularly reviewed & revised, adding new & different test cases to help find more defects

6. Testing is context dependent

- All the developed software's are not identical
- Use a different approach, methodologies, techniques, and types of testing depending upon the application type
- QA for banking app will be different from QA for ecommerce application
- QA for flight software system will be different from QA for payment wallet

7. Absence of Error – Fallacy

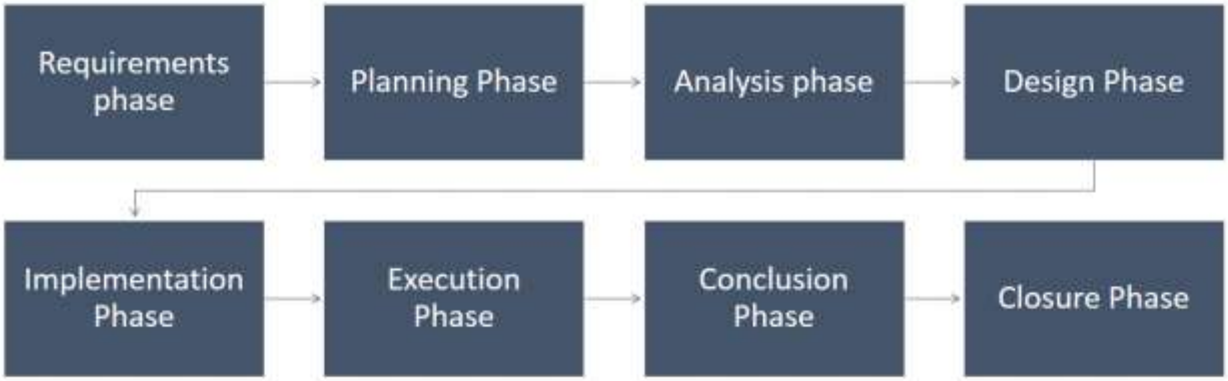
- Is a 99% of bug-free software usable? No!
- Software testing is not mere finding defects, but also to check that software addresses the business needs
- The absence of Error is a Fallacy i.e. Finding and fixing defects does not help if the system build is unusable and does not fulfill the user's needs & requirements.

Software Testing Life Cycle [STLC]s

Software Testing Life Cycle - STLC

- A sequence of activities conducted to perform Software Testing
- In STLC process, each activity is carried out in a planned and systematic way
- Software Testing is not a just a single activity
- A series of activities carried out methodologically to help certify your software product
- Each of these stages has a definite Entry and Exit criteria, Activities & Deliverables associated with it.

Phases of STLC



Requirements phase → Planning Phase → Analysis phase → Design Phase → Implementation Phase → Execution Phase → Conclusion Phase → Closure Phase

What is Entry and Exit Criteria?

- **Entry Criteria**
  - The prerequisite items that must be completed before testing can begin.
- **Exit Criteria**
  - Defines the items that must be completed before testing can be concluded
- We will have Entry and Exit Criteria for all levels in the Software Testing Life Cycle (STLC)
- In ideal world, tester will not enter next phase until exit criteria is full filled for the previous phase

1. Requirements Phase

- Analyze and study the requirements
- Brain storming sessions [and discussions] with other teams and try to find out whether the requirements are testable or not
- The QA team may interact with various stakeholders to understand the requirements in detail
  - Client
  - Business Analyst
  - Technical Leads
  - System Architects
  - Domain experts
- This phase helps to identify the scope of the testing

2. Planning Phase

- In this phase we –
  - Identify the activities and resources which would help to meet the testing objectives
  - On what basis the planning is done –
    - Requirements
    - Test strategy of the organization
    - Risk analysis / Risk Management and mitigation

3. Analysis phases

- This STLC phase defines "WHAT" to be tested
- Identify the test conditions through the requirements document
- Every test condition should be traceable back to the requirement
- Test condition-
  - Is the constraint that you should follow to test an application
  - Test condition can be a piece of functionality. In simple terms the goal of a test cases
  - **Ex-** When User Name and Password are valid then an application will move forward

3. Analysis phase(cont.)

- Factors which effect the identification of test conditions-
  - Levels and depth of testing
  - Complexity of the product
  - Product and project risks
  - Software development life cycle involved
  - Test management
  - Skills and knowledge of the team
  - Availability of the stakeholders.

4. Design Phase

- This STLC phase defines "HOW" to be tested
  - Detail the test condition. Break down the test conditions into multiple sub conditions to increase coverage.
  - Identify and get the test data
  - Identify and set up the test environment.
  - Create the requirement traceability metrics

5. Implementation Phase

- Major task for this phase is to create detailed test case
- Prioritize the test cases also identify which test case will become part of the regression suite or smoke suite
- Carry out the review to ensure the correctness of the test cases
- If project has automation testing-
  - identify the test cases for automation
  - Writing scripts for the selected test cases

6. Execution Phase

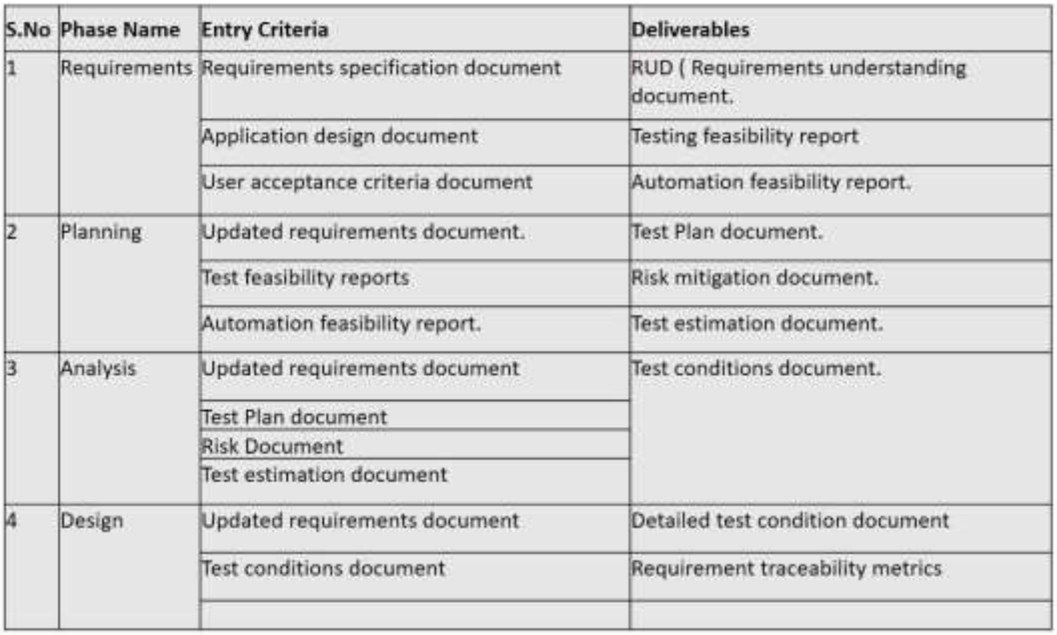
- In this phase, actual execution for test case occurs
  - Execute the test cases
  - log defects/issues/gaps in case of any discrepancy
  - Fill the traceability matrix
- If project has automation testing-
  - Run automation testing scripts
  - log defects/issues/gaps in case of any discrepancy
  - Fill the traceability matrix

7. Conclusion Phase

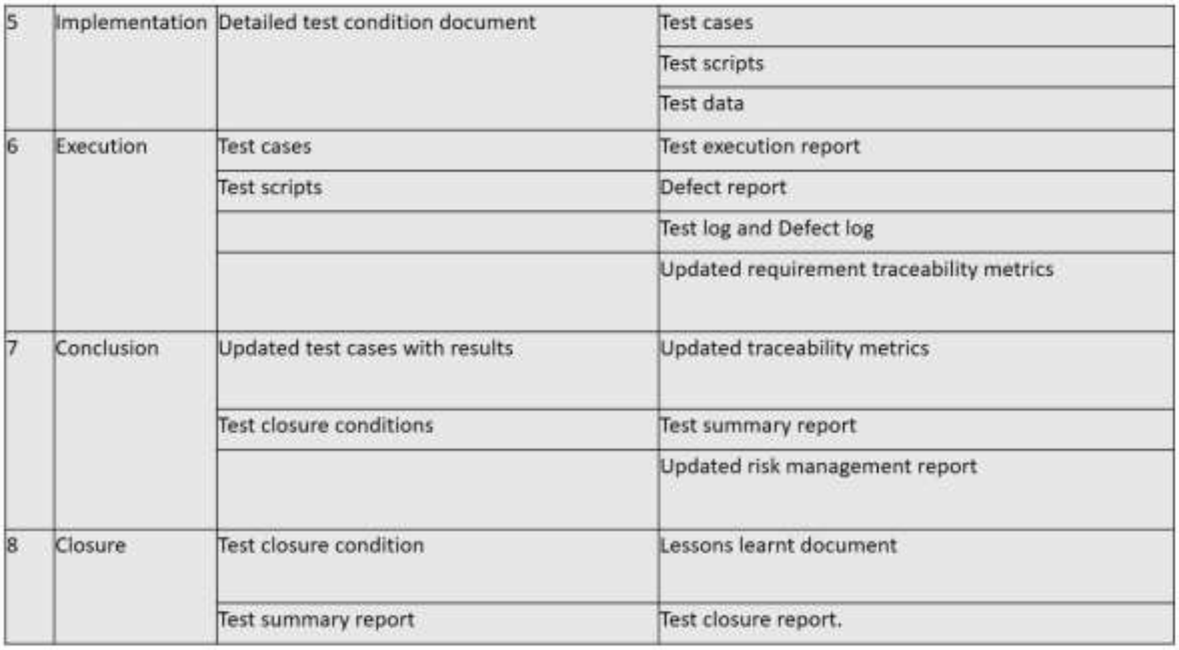
- This phase focuses on the exit criteria and reporting
  - Depending upon project and stakeholder's choice
    - DSR – Daily status report
    - WSR – Weekly status reports
  - If reports are for upper management;
    - No technical details
    - Just summary and how did you mitigate risks
  - But if reports are for Project managers or some technical guys;
    - Need to include all technical details
      - number of test cases passed/failed
      - defects raised
      - severity

8. Closure Phase

- Tasks for this phase are-
  - Check for the completion of the test
  - Confirm of all test cases are executed
  - Ensure that high severity defects are not open
  - Lessons learned document
  - Scope of improvements



S.No	Phase Name	Entry Criteria	Deliverables
1	Requirements	Requirements specification document	RUD (Requirements understanding document.
		Application design document	Testing feasibility report
		User acceptance criteria document	Automation feasibility report.
2	Planning	Updated requirements document.	Test Plan document.
		Test feasibility reports	Risk mitigation document.
		Automation feasibility report.	Test estimation document.
3	Analysis	Updated requirements document	Test conditions document.
		Test Plan document	
		Risk Document	
		Test estimation document	
4	Design	Updated requirements document	Detailed test condition document
		Test conditions document	Requirement traceability metrics



5	Implementation	Detailed test condition document	Test cases
			Test scripts
			Test data
6	Execution	Test cases	Test execution report
		Test scripts	Defect report
			Test log and Defect log
			Updated requirement traceability metrics
7	Conclusion	Updated test cases with results	Updated traceability metrics
		Test closure conditions	Test summary report
			Updated risk management report
8	Closure	Test closure condition	Lessons learnt document
		Test summary report	Test closure report.

Thank You