**Test Plan**

Test Plan is a document which derives all future testing activities. Test manager will review and approve the Test plan.

**Test plan consists of 15 attributes:**

1) **Objective**: It speaks about the aim of writing the test plan. We can mention which model is used and what process is used.

2) **Scope**: Here we mention features which need to be tested and features which need not be tested.

3) **Test methodology**: Based on type of application, we decide what type of testing needs to be done.

a) Web based application

b) Client Server application

c) Stand alone application

4) **Test approach**: It explains how we test the application. Several approaches are:

a) Writing Test cases

b) Writing Test scenarios

c) Writing Test cases and Test scenarios

d) Writing the flowchart

5) **Assumption**: While writing Test plan, team will assume few aspects: a) Assumption from resource point of view

b) Assumption from technology point of view

c) Assumption from development team point of view

d) Assumption from Knowledge Transfer point of view

e) Assumption from supporting documents point of view

6) **Risk**: When assumption fails, we will be in Risk, to overcome it, we have a Backup plan.

7) **Backup plan/Mitigation Plan**: If assumption fails, it will lead to 100% risk, by having a Backup plan we can reduce risk to 20%. Example: Lets say 3 TE’s are working on a project, due to some reasons, 1 TE quits job suddenly. Now the team is at risk as we are supposed

to finish the project in a given deadline. To overcome this risk, we will initially assign a secondary TE, so that they can act as a Backup and ensure software is tested in a given timeline before it is released to the customer.

8) **Roles & Responsibilities of TE**:

a) Understand customer requirement

b) Write Test scenarios & Test cases

c) Conduct Brainstorming meeting for better test case coverage & get test cases reviewed

d) Always do optimized testing

e) Conduct Smoke, Functional, Integration, System, adhoc testing etc f) Perform Test case execution

g) Log the defect to the defect tracking tool

h) Give Knowledge Transfer to other team members

9) **Scheduling**: In this section, we mention start and end dates for each and every testing activity.

10) **Defect Tracking**: As soon as TE finds any defect, he will login to the Defect tracking tool and log the defect, generate a defect report, and a unique id will be created. TE will communicate this defect to the developer and the developer will fix the defect and give back to TE. TE will retest to check if the defect is really fixed or not. If a defect is fixed, TE will close the defect. This process is called Defect Tracking.

11) **Test environment**: It is an environment configured for testing, where TE will test the application by executing the test cases. Test environment consists of hardware, software, database, OS, CPU etc details.

12) **Entry & Exit criteria**: These are the set of conditions that should be met in order to

start and end the project. It is also the key attribute of Test plan.

**Functional Testing:**

Entry criteria:

1) Coding should be completed

2) WBT should be done

3) Software should be installed

4) Smoke Testing should be done

5) Functional Test scenarios & Test cases should be ready

6) Resource should be present

Exit Criteria:

1) Based on number of Test cases executed, test case pass% should be 85%

2) Based on number of defects found:

a) There should be no blocker defects

b) Critical defects should not be more than 20

c) Major defects should not be more than 50

d) Minor defects should not be more than 80

**Integration Testing:**

Entry criteria:

1) It should match exit criteria of Functional Testing

2) Functional Test scenarios & Test cases should be ready

3) Resource should be present

Exit Criteria:

1) Based on number of Test cases executed, test case pass% should be

95%

2) Based on number of defects found:

a) There should be no blocker defects

b) There should be no critical defects

c) Major defects should not be more than 20

d) Minor defects should not be more than 40

**System Testing:**

Entry criteria:

1) It should match exit criteria of Integration Testing

2) Integration Test scenarios & Test cases should be ready

3) Resource should be present

Exit Criteria:

1) Based on number of Test cases executed, test case pass%

should be 99%

2) Based on number of defects found:

a) There should be no blocker defects

b) There should be no critical defects

c) Major defects should not be more than 10

d) Minor defects should not be more than 20

13) **Test Automation**: Here we mention which automation tool we use on a project, which features to be tested and which need not be tested, which automation framework we use.

14) **Deliverables**: These are the outcomes of the Testing team. This section contains what we are going to give to customer by the end of the project. The document

includes

– Test plan document, Test scenario & Test case document, Traceability matrix document, Test execution report document, Defect report document, Release notes, Graphs & metrics

**Q) What do you mean by Release notes?**

A) These are the set of documents given to the customer along with software which is signed off by the Test manager.

Release notes consist of – List of Pending and open defects, List of defects fixed in current release found in previous release, platform on which software is tested, platform on which software is not tested, list of features added, deleted, modified, procedure to install the software, version of software.

15) **Templates**: Here, we mention all the empty templates which will be used in future by TE’s.

Templates include – Test case template, Test case review template, Traceability matrix template, Test case execution template, Defect report template, Test plan template.

**Test Strategy**

Test strategy is a high level plan consisting of principles that guide the overall software testing process.

It provides a structured approach to the entire QA team, guiding them towards achieving testing objectives in the most efficient way.

Q**) Why Test Strategy**?

1. To make sure that all purposes are covered entirely and understood by all stakeholders

2. To support quality assurance with respect to planning of resources, language, test, roles and responsibilities etc.

u **Test Strategy document** - It is a well described document in software testing which clearly defines the exact software testing approach and testing objectives of the software

application.

u Components of Test Strategy

u Scope & Overview

u Testing methodology

u Testing environment

u Testing tools

u Release

u Risk analysis

u Review and approvals

**MIND MAPPING**

****

Mind mapping is a valuable technique in software testing for several reasons: **Visualization of Information**: Mind maps provide a visual representation of complex

information, making it easier to understand and remember. In software testing, where there are various test cases, requirements, and interconnections, a visual overview can be immensely helpful.

**Organization of Test Scenarios**: Mind maps help organize and structure test scenarios hierarchically. Testers can break down testing into different levels (e.g., unit testing, integration testing, system testing) and map related test cases under each category.

**Creative Problem Solving:** Mind maps encourage creative thinking and brainstorming. Testers can use mind maps to explore different test scenarios, edge cases, and potential issues that might not be immediately apparent in a linear document.

**Efficient Test Planning**: Test planning involves considering various factors such as test objectives, scope, resources, and schedules. A mind map provides a clear, concise overview of the test plan, making it easier for team members to understand and contribute to the planning process.

**Traceability**: Mind maps can be used to establish traceability between test cases and requirements. This visual linkage helps ensure that every requirement has associated test cases and provides a clear view of test coverage.

**Communication and Collaboration**: Mind maps are effective tools for communicating testing strategies and progress to team members, stakeholders, and even non-technical audiences. They promote collaboration and provide a shared understanding of the testing process.

**Test Case Design**: When designing test cases, mind maps help testers systematically consider various input combinations, scenarios, and potential outcomes. This aids in creating comprehensive test coverage and reduces the likelihood of overlooking critical test scenarios.

**Flexibility and Iteration**: Software testing often involves iterative processes as the software evolves. Mind maps are flexible and can be easily updated or modified as testing progresses or as new information becomes available.

**Test Execution Planning**: Mind maps can assist in planning the execution of test cases. Testers can use the map to determine the optimal sequence of test execution, helping to identify any dependencies or prerequisites.

**Documentation**: While mind maps are not a replacement for detailed documentation, they provide a concise and structured overview that can complement more extensive documentation. This is particularly useful for conveying key testing concepts and strategies.

**In summary, mind mapping is a versatile and powerful tool in software testing. It enhances understanding, aids in planning and execution, supports creative problem-solving, and facilitates communication and collaboration among team members. By leveraging mind maps, testing teams can streamline their processes and improve the effectiveness and efficiency of their testing efforts.**