**SOFTWARE TESTING ASSIGNMENT**

**MODULE 2**

* **What is exploratory testing?**

A testing method in which test designs are not created in advance but testers checks the system on the fly. Here test design and test execution are performed at same time.

* **What is traceability matrix?**

Traceability matrix is a table type document that is used in the development of software application to trace requirements.

* **What is boundary value testing?**

Boundary value analysis is a software testing technique in which tests are designed to include representatives of boundary values in a range.

* **What is Equivalence partitioning testing?**

A software testing technique that divides the input data of a software unit into partitions of equivalent data from which test cases can be derived.

* **What is Integration testing?**

Testing performed to expose defects in the interfaces and in interactions between integrated component and systems.

* **What determines the level of Risk?**

The level of risk is determined by probability of an adverse event and the impact of the event.

* **What is Alpha testing?**

Alpha testing is a type of acceptance testing performed to identify all possible issues and bugs before releasing the final product to end users.

* **What is Beta testing?**

Testing performed by real users of software application in the real environment is beta testing.

* **What is component testing?**

The testing of individual software component is component testing. It is also known as unit testing.

* **What is functional system testing?**

A requirement that specifies a function that a system or system component must performed.

* **What is non functional testing?**

Testing the attributes of a component or system that do not relate to functionality.

* **What is GUI testing?**

GUI is all about what a user see.

Graphical User Interface (GUI) testing is the process of testing the system’s GUI of the System under Test. GUI testing involves checking the screens with the controls like menus, buttons, icons, and all types of bars – tool bar, menu bar, dialog boxes and windows etc.

* **What is Ad hoc testing?**

It is an informal testing type with an aim to break the system. Main aim of this testing is to find defects by random checking.

* **What is load testing?**

Load testing is a kind of performance testing which determines a system’s performance under real-life load conditions. This testing helps determine how the application behaves when multiple users access it simultaneously.

* **What is Stress testing?**

System is stressed beyond its specifications to check how and when it fails. Performed under heavy load like putting large number beyond storage capacity, complex database queries, continuous input to system or database load.

* **What is white box testing and list the types of white box testing?**

Testing based on an analysis of the internal structure of the component or system is known as white box testing.

**White box Testing Techniques:**

* Statement coverage
* Branch Coverage
* Decision Coverage
* **What is black box testing? What are the different black box testing techniques?**

Testing, either functional or non-functional, without reference to the internal structure of the component or system.

**Black box Testing Techniques**:

* Equivalence Partitioning
* Boundary Value Analysis
* Decision Tables
* State Transition Testing
* **Mention what are the categories of defects?**
* **Data Quality/Database Defects:** Deals with improper handling of data in the database.
* **Critical Functionality Defects:** The occurrence of these bugs hampers the crucial functionality of the application.
* **Functionality Defects:** These defects affect the functionality of the application.
* **Security Defects:** Application security defects generally involve improper handling of data sent from the user to the application. These defects are the most severe and given highest priority for a fix.
* **User Interface Defects:** As the name suggests, the bugs deal with problems related to UI are usually considered less severe.
* **Mention what big bang testing is?**
* In Big Bang integration testing all components or modules is integrated simultaneously, after which everything is tested as a whole.
* Big Bang testing has the advantage that everything is finished before integration testing starts.
* **What is the purpose of exit criteria?**
* All the high priority bugs are fixed.
* The rate at which bugs are found is too small.
* The testing budget is exhausted.
* The project duration is completed.
* The risk in the project is under acceptable limit.
* **What is 7 key principles? Explain in detail?**
* **Testing shows the presence of defects.**

Testing can show that defects are present, but cannot prove that there are no defects.

Testing reduces the probability of undiscovered defects remaining in the software but, even if no defects are found, it is not a proof of correctness.

* **Exhaustive testing is impossible.**
* Testing everything including all combinations of inputs and preconditions is not possible. So, instead of doing the exhaustive testing we can use risks and priorities to focus testing efforts. we cannot test everything (i.e. all combinations of inputs and pre-conditions)
* **Early Testing**
* Testing activities should start as early as possible in the software or system development life cycle, and should be focused on defined objectives. Testing activities should start as early as possible in the development life cycle.
* **Defect Clustering**
* A small number of modules contain most of the defects discovered during pre-release testing, or are responsible for the most operational failures. Defects are not evenly spread in a system. They are ‘clustered’
* **Pesticide Paradox**
* If the same tests are repeated over and over again, eventually the same set of test cases will no longer find any new defects.
* To overcome this “pesticide paradox”, the test cases need to be regularly reviewed and revised, and new and different tests need to be written to exercise different parts of the software or system to potentially find more defects.
* **Testing is Context Dependent**
* Testing is a context-dependent principle states that we have multiple fields such as e-commerce websites, commercial websites, and so on are available in the market. There is a definite way to test the commercial site as well as the e-commerce websites because every application has its own needs, features, and functionality.
* **Absence of errors fallacy**
* Once the application is completely tested and there are no bugs identified before the release, so we can say that the application is 99 percent bug-free. The absence of error fallacy means identifying and fixing the bugs would not help if the application is impractical and not able to accomplish the client's requirements and needs.
* **Difference between QA v/s QC v/s Tester.**

|  |  |  |
| --- | --- | --- |
| **QA (Quality Assurance)** | **QC (Quality Control)** | **Tester** |
| QA ensures the implementation of processes, procedures | QC ensures the verification of developed software with respect to documented requirements. | Tester ensures the identification of bugs/error/defects in the Software. |
| Focuses on processes and procedures rather than conducting actual testing on the system. | Focuses on actual testing by executing Software with intend to identify bug/defect through implementation of procedures and process. | Focuses on actual testing. |
| Process oriented activities. | Product oriented activities. | Product oriented activities. |
| Preventive activities. | Corrective process. | Preventive process. |
| QA can be considered a subset of Software Test Life Cycle (STLC). | QC can be considered as the subset of Quality Assurance. | Testing is the subset of Quality Control. |

* **Difference between verification and Validation.**

|  |  |
| --- | --- |
| **Verification** | **Validation** |
| Verification is a process which is performed at development level. | Validation is a process which is performed at testing level. |
| Verification phases are :  Business Requirement Analysis  System Design/ System Requirement  Architectural Design (Technical Specification)  Module Design (Program Specification) | Validation Phases are :  Unit Testing  Integration Testing  System Testing  Acceptance Testing |
| It is the process of evaluating product of development to check whether the specified requirements meet or not. | It is the process of evaluating the product of development to check whether it satisfied business requirements or not. |
| Verification can be achieved by asking Are you building a product right. | Validation can be achieved by asking Are you building a right product. |
| The evaluation of verification can be achieved by planning, Requirement specification, Design Specification, Code specification, and test cases. | The evaluation of validation can be achieved as an actual product. |
| Verification activities are Reviews and Inspections. | Validation activity is Testing. |

* **Explain types of Performance testing.**
* **Load Testing –** Load Testing checks the application’s ability to perform under expected user loads. The main objective of this testing is to identify performance bottlenecks before the software application goes live.
* **Stress Testing –** This testing involves testing an application under extreme workloads to see how it handles high traffic or data processing. Here, it identifies the breaking point of an application.
* **Endurance Testing –** This testing is done to make sure the software can handle the expected load over a long period of time.
* **Spike Testing –** It is used to test the software’s reaction to sudden large spikes in the load generated by users.
* **Volume Testing** – Here, a large amount of data is populated in a database and the overall software system’s behavior is monitored. The main objective of this testing is to check the software application’s performance under varying database volumes.
* **Scalability Testing**– The main objective of scalability testing is to determine the software application’s effectiveness in “scaling up” to support an increase in user load.
* **What is Error, Defect, Bug and failure?**

A mistake in coding is called an error. An error found by the tester is called Defect. Defect accepted by the Development team is known as Bug. Build does not meet requirements is called a bug.

* **Difference between verification and Validation?**

|  |  |
| --- | --- |
| Verification | Validation |
| * It includes checking documents, design, codes and programs. | * It includes testing and validating the actual product. |
| * Verification is the static testing. | * Validation is the dynamic testing. |
| * It does not include the execution of the code. | * It includes the execution of the code. |
| * Methods used in verification are reviews, walkthroughs, inspections and desk-checking. | * Methods used in validation are Black Box Testing, White Box Testing and non-functional testing. |
| * It checks whether the software conforms to specifications or not. | * It checks whether the software meets the requirements and expectations of a customer or not. |
| * It can find the bugs in the early stage of the development. | * It can only find the bugs that could not be found by the verification process. |
| * The goal of verification is application and software architecture and specification. | * The goal of validation is an actual product. |
| * Quality assurance team does verification. | * Validation is executed on software code with the help of testing team. |
| * It comes before validation. | * It comes after verification. |
| * It consists of checking of documents/files and is performed by human. | * It consists of execution of program and is performed by computer. |

* **What is Bug Life Cycle?**

**Defect Life Cycle** or Bug Life Cycle in software testing is the specific set of states that defect or bug goes through in its entire life. The purpose of Defect life cycle is to easily coordinate and communicate current status of defect which changes to various assignees and make the defect fixing process systematic and efficient.

* **Defect States Workflow**
* The number of states that a defect goes through varies from project to project. Below lifecycle diagram, covers all possible states
* **New:** When a new defect is logged and posted for the first time. It is assigned a status as NEW.
* **Assigned:** Once the bug is posted by the tester, the lead of the tester approves the bug and assigns the bug to the developer team
* **Open:** The developer starts analyzing and works on the defect fix
* **Fixed:** When a developer makes a necessary code change and verifies the change, he or she can make bug status as “Fixed.”
* **Pending retest:** Once the defect is fixed the developer gives a particular code for retesting the code to the tester. Since the software testing remains pending from the testers end, the status assigned is “pending retest.”
* **Retest:** Tester does the retesting of the code at this stage to check whether the defect is fixed by the developer or not and changes the status to “Re-test
* **Verified:** The tester re-tests the bug after it got fixed by the developer. If there is no bug detected in the software, then the bug is fixed and the status assigned is “verified.”
* **Reopen:** If the bug persists even after the developer has fixed the bug, the tester changes the status to “reopened”. Once again the bug goes through the life cycle.
* **Closed:** If the bug is no longer exists then tester assigns the status “Closed.”
* **Duplicate:** If the defect is repeated twice or the defect corresponds to the same concept of the bug, the status is changed to “duplicate.”
* **Rejected:** If the developer feels the defect is not a genuine defect then it changes the defect to “rejected.”
* **Deferred:** If the present bug is not of a prime priority and if it is expected to get fixed in the next release, then status “Deferred” is assigned to such bugs
* **Not a bug:** If it does not affect the functionality of the application then the status assigned to a bug is “Not a bug”.
* **Difference between Priority and Severity**

|  |  |  |
| --- | --- | --- |
| Parameters | Severity in Testing | Priority in Testing |
| Definition | Severity is a term that denotes how severely a defect can affect the functionality of the software. | Priority is a term that defines how fast we need to fix a defect. |
| Parameter | Severity is basically a parameter that denotes the total impact of a given defect on any software. | Priority is basically a parameter that decides the order in which we should fix the defects. |
| Relation | Severity relates to the standards of quality. | Priority relates to the scheduling of defects to resolve them in software. |
| Value | The value of severity is objective. | The value of priority is subjective. |
| Change of Value | The value of Severity changes continually from time to time. | The value of Priority changes from time to time. |
| Who Decides the Defect | The testing engineer basically decides a defect’s severity level. | The product manager basically decides a defect’s priority level. |
| Types | There are 5 types of Severities: Cosmetic, Minor, Moderate, Major, and Critical. | There are 3 types of Priorities: High, Medium, and Low. |
|  |  |  |

* **Explain the difference between Functional testing and Non-Functional testing.**

|  |  |
| --- | --- |
| Functional Testing | Non-functional Testing |
| * It verifies the operations and actions of an application. | * It verifies the behaviour of an application. |
| * It is based on requirements of customer. | * It is based on expectations of customer. |
| * It helps to enhance the behaviour of the application. | * It helps to improve the performance of the application. |
| * Functional testing is easy to execute manually. | * It is hard to execute non-functional testing manually. |
| * It tests what the product does. | * It describes how the product does. |
| * Functional testing is based on the business requirement. | * Non-functional testing is based on the performance requirement. |
| * Examples:  1. Unit Testing 2. Smoke Testing 3. Integration Testing 4. Regression Testing | * Examples:  1. Performance Testing 2. Load Testing 3. Stress Testing 4. Scalability Testing |

* **What is the difference between the STLC (Software Testing Life Cycle) and SDLC (Software Development Life Cycle)?**

|  |  |  |
| --- | --- | --- |
| Parameter | SDLC | STLC |
| Origin | Development Life Cycle | Testing Life Cycle |
| Objective | The main object of SDLC life cycle is to complete successful development of the software including testing and other phases. | The only objective of the STLC phase is testing. |
| Requirement Gathering | In SDLC the business analyst gathers the requirements and create Development Plan | In STLC, the QA team analyze requirement documents like functional and non-functional documents and create System Test Plan |
| High & Low- Level Design | In SDLC, the development team creates the high and low-level design plans | In STLC, the test analyst creates the Integration Test Plan |
| Coding | The real code is developed, and actual work takes place as per the design documents. | The testing team prepares the test environment and executes them |
| Maintenance | SDLC phase also includes post- deployment supports and updates. | Testers, execute regression suits, usually automation scripts to check maintenance code deployed. |

* **What is the difference between test scenarios, test cases, and test script?**

|  |  |  |
| --- | --- | --- |
| Test Scenario | Test Case | Test Script |
| Is any functionality that can be tested. | Is a set of actions executed to verify particular features or functionality. | Is a set of instructions to test an app automatically. |
| Is derived from test artefacts like business requirement specification (BRS) and software requirement specification (SRS). | Is mostly derived from test scenario. | Is mostly derived from test cases. |
| Helps test the end-to-end functionally in an agile way. | Helps in exhaustive testing of an app. | Helps in test specific things repeatedly. |
| Is more focused on what to test | Is focused on what to test and how to test. | Is focused on the expected result. |
| Takes less time and fewer resources to create. | Requires more resources and time. | Requires less time for testing but more resources for scripts creating and updating. |
| Includes an end-to-end functionality to be tested. | Include test steps, date, expected results for testing. | Includes different commands to develop a script. |
| The main task is to check the full functionality of a software application. | The main task is to verify compliance with the applicable standards, guidelines, and customer, requirements. | The main task is to verify that nothing is skipped, and the results are true as the desired testing plan. |

|  |  |  |
| --- | --- | --- |
| Allows quickly assessing the testing scope. | Allows detecting errors and defects. | Allows carrying out an automatic execution of test cases. |

* **Explain what Test Plan is? What is the information that should be covered.**
* Test Plan is a document describing the scope, approach, resources, and schedule of intended test activities.
* **What are the different Methodologies in Agile Development Model?**
* Scrum
* Extreme Programming
* Dynamic system Development method(DSDM)
* Test Driven Development(TDD)
* Feature Driven Development(FDD)
* X Breed
* Crystal
* Kanab
* **Explain the difference between Authorization and Authentication in Web testing.**

| **Authentication** | **Authorization** |
| --- | --- |
| * In the authentication process, the identity of users are checked for providing the access to the system. | * While in authorization process, a the person’s or user’s authorities are checked for accessing the resources. |
| * In the authentication process, users or persons are verified. | * While in this process, users or persons are validated. |
| * It is done before the authorization process. | * While this process is done after the authentication process. |
| * It needs usually the user’s login details. | * While it needs the user’s privilege or security levels. |
| * Authentication determines whether the person is user or not. | * While it determines What permission does the user have? |
| * Generally, transmit information through an ID Token. | * Generally, transmit information through an Access Token. |
| * The Open ID Connect (OIDC) protocol is an authentication protocol that is generally in charge of user authentication process. | * The Oath 2.0 protocol governs the overall system of user authorization process. |
| * Popular Authentication Techniques- * Password-Based Authentication * Password less Authentication * 2FA/MFA (Two-Factor Authentication / Multi-Factor Authentication) * Single sign-on (SSO) * Social authentication | * Popular  Authorization Techniques- * Role-Based Access Controls (RBAC) * SON web token (JWT) Authorization * SAML Authorization * Open ID Authorization * Oath 2.0 Authorization |
| * The authentication credentials can be changed in part as and when required by the user. | * The authorization permissions cannot be changed by user as these are granted by the owner of the system and only he/she has the access to change it. |
| * The user authentication is visible at user end. | * The user authorization is not visible at the user end. |
| * The user authentication is identified with username, password, face recognition, retina scan, fingerprints, etc. | * The user authorization is carried out through the access rights to resources by using roles that have been pre-defined. |

* **What are the common problems faced in Web testing?**

Integration testing exposes problems with interfaces among different program components before deployment

* Interoperability. ...
* Security. ...
* Performance. ...
* Usability. ...
* Quality Testing, Exceptional Services.