**Software testing assignment module 2**

1. **What is Exploratory Testing?**

**Exploratory Testing** is a type of software testing where Test cases are not created in advance but testers check the system on the fly. They may note down ideas about what to test before test execution. The focus of exploratory testing is more on testing as a “thinking” activity.

Exploratory Testing is widely used in Agile models and is all about discovery, investigation, and learning. It emphasizes the personal freedom and responsibility of the individual tester.

1. **What is traceability matrix?**

A Traceability Matrix is a document that co-relates any two-baseline documents that require a many-to-many relationship to check the completeness of the relationship.

It is used to track the requirements and to check the current project requirements are met.

1. **What is Boundary Value testing?**

Boundary value analysis is a methodology for designing test cases that concentrates software testing effort on cases near the limits of valid ranges Boundary value analysis is a method which refines equivalence partitioning. Boundary value analysis generates test cases that highlight errors better than equivalence partitioning. The trick is to concentrate software testing efforts at the extreme ends of the equivalence classes. At those points when input values change from valid to invalid errors are most likely to occur. Boundary Value Analysis (BVA) uses the same analysis of partitions as EP and is usually used in conjunction with EP in test case design.

1. **What is Equivalence Partitioning Testing?**

Equivalence Partitioning or Equivalence Class Partitioning is a type of black box testing technique that can be applied to all levels of [software testing](https://www.guru99.com/software-testing.html) like unit, integration, system, etc. In this technique, input data units are divided into equivalent partitions that can be used to derive test cases which reduces time required for testing because of small number of test cases.

* It divides the input data of software into different equivalence data classes.
* You can apply this technique, where there is a range in the input field.

1. **What Is Integration Testing?**

Integration testing -- also known as integration and testing (I&T) -- is a type of [software testing](https://www.techtarget.com/whatis/definition/software-testing) in which the different units, modules or components of a software application are tested as a combined entity. However, these modules may be coded by different programmers.

The aim of integration testing is to test the interfaces between the modules and expose any defects that may arise when these components are integrated and need to interact with each other.

1. **What determines the level of risk?**

Risk should be evaluated at the Business Level, Technological Level, Project Level and Testing Level.

Risk also used to decide where to start and where more testing is needed.

1. **What is Alpha Testing?**

Alpha testing is definitely performed and carried out at the developing organizations location with the involvement of developers.

• It is always performed by the developers at the software development site.

• Sometimes it is also performed by independent testing team.

• It is always performed in virtual Environment.

• It comes under the category of both White box testing and Black box testing

1. **What is Beta Testing?**

Beta Testing is performed by real users and it is unstructured. It can be considered as a form of external [User Acceptance Testing](https://www.practitest.com/qa-learningcenter/best-practices/what-is-uat-testing/).

Users can freely use the application and then they are encouraged to give feedback about their experience. This test is more focussed on performance and scalability. Beta Testing helps reduce product failures and provides higher product quality through customer validation that resulted from their experience with the application.

1. **What is component testing?**

A minimal software are item that can be tested in isolation it means “A unit is the smallest testable part of software.” Component testing is defined as a software testing type, in which the testing is performed on each individual component separately without integrating with other components. It’s also referred to as Module Testing when it is viewed from an architectural perspective. Component Testing is also referred to as Unit Testing, Program Testing, or Module Testing.

Generally, any software as a whole is made of several components. Component Level Testing deals with testing these components individually.

1. **What is functional system testing?**

* Functional system testing – A requirement that specifies a function that a system or system component must perform.
* A requirement may exist as a text document or a model.

1. **What is Non-Functional Testing?**

Non-Functional testing: Testing the attributes of a component or a system that do not relate to functionality, e.g. reliability, efficiency, usability, interoperability, maintainability and portability.

• May be performed at all test levels

• Measuring the characteristics of the system/software that can be quantified on varying scale-e.g. performance test scaling

• Non-functional testing includes, but is not limited to, performance testing, load testing, stress testing, usability testing, maintainability testing, reliability testing and portability testing. It is the testing of “how” the system works. Non – Functional testing may be performed at all test levels.

1. **What Is GUI Testing?**

Graphical User Interface (GUI) testing is the process of testing the system’s GUI of the system under test.GUI testing involves checking the screen with the controls like menus, buttons, icons and all type of bars - tool bar , menu bar , dialog boxes and windows etc.

* There are some Approach of GUI
* Manual based testing
* Record and Replay
* Model based testing

1. **What is Adhoc Testing?**

The error guessing is a technique where the experienced and good testers are encouraged to think of situation in which the software may not be able to cope. Some people seem to be naturally good at testing and others are good tester because they have a lot of experience either as a tester or working with a particular system and so are able to find out its weaknesses.

It also saves a lot of time because of the assumption and guessing made by the experienced tester to find out the defects which otherwise won’t be able to find.

Using experience to postulate errors. Use error guessing to complement test design techniques.

Types of adhoc testing :

* 1. Buddy testing
  2. Pair testing
  3. Monkey testing

1. **What is load Testing?**

It’s performance testing to check systems behaviour under load testing an application under heavy loads. Such as testing of a website under a range of loads to determine at what point the system’s response time degrades or fails.

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

**Load Testing** is a type of [Performance Testing](https://www.geeksforgeeks.org/performance-testing-software-testing/) that determines the performance of a system, software product, or software application under real-life-based load conditions. Basically, load testing determines the behavior of the application when multiple users use it at the same time. It is the response of the system measured under varying load conditions. The load testing is carried out for normal and extreme load conditions.

1. **What is Stress Testing?**

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.

• Stress testing is done to make sure that the system would not crash under crunch situation.

• Stress testing is also known as endurance testing.

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

White Box Testing: Testing based on an analysis of the internal structure of the component or system. The structure-based testing technique is also known as the ‘white-box’ or ‘glass-box’ testing technique because here the testers require knowledge of how the software is implemented, and how it works.

In white-box testing, the tester is concentrating on how the software does it. For example, a structural technique may be concerned with exercising loops in the software. Different test cases may be derived to exercise the loop once, twice, and many times. This may be done regardless of the functionality of the software. Structure-based techniques are also used in system and acceptance testing, but the structures are different. For example, the coverage of menu options or major business transactions could be the structural element in the system or acceptance testing.

Types of testing:

* Statement coverage ,Decision coverage, Condition coverage

1. **What is black box Testing? What are the different black box testing techniques?**

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

Specification-based testing technique is also known as ‘black-box’ or input/output-driven testing techniques because they view the software as a black box with inputs and outputs. The testers have no knowledge of how the system or component is structured inside the box. In black-box testing, the tester is concentrating on what the software does, not how it does it. Specification-based techniques are appropriate at all levels of testing.

* Techniques of Black box testing
* Equivalence partitioning
* Boundary value analysis
* Decision Tables
* State transition testing
* Use case testing Other black box testing
* Syntax or pattern testing

1. **Mention what are the categories of defects?**

* There are some categories which are as below
* Data quality/Database Defects
* Critical Functionality Defects
* Functionality Defects
* Security Defects
* User Interface Defects

1. **Mention what big bang testing is?**

In big bang integration testing all components and modules is integrated simultaneously, after which everything is tested as a whole.

• Big bang testing has the advantages that everything is finished before integration testing starts.

• The major disadvantage is that in general it is time consuming and difficult to trace the cause of failures because of this late integration.

1. **What is the purpose of exist criteria?**

Exit criterion is used to determine whether a given test activity has been completed or NOT. Exit criteria can be defined for all of the test activities right from planning, specification, and execution. Exit criterion should be part of the test plan and decided in the planning stage.

Successful Testing of Integrated Application. Executed Test Cases are documented All High prioritized bugs fixed and closed Technical documents to be submitted followed by release Notes.

1. **When should “Regression Testing “be performed?**

Change in requirements and code Is modified according to the requirements

• New feature is added to the software

• Defect fixing

• Performance issue fix

1. **What is 7 key principles? Explain in detail?**

* Testing shows presence of Defects
* Exhaustive Testing is Impossible
* Early testing
* Defect Clustering
* The pesticide Paradox
* Testing is context dependent
* Absence of Errors Fallacy

1. **Testing shows presence of Defects:**

* Testing can show that defects are present, but cannot prove that there are no defects.
* Ven if no defects are found, it is not a proof of correctness.
* As we find more defects. The probability of undiscovered defects remaining in a system reduces

1. **Exhaustive Testing is Impossible:**

* Testing everything including all combinations of inputs and preconditions is not possible
* For example in an application in one screen there are 15 input fields. Each having 5 possible values then to test all the valid combination you would need
* That is we must Priorities our testing efforts using a Risk Based Approach.

1. **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.

1. **. 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
* In other words, most defects found during testing are usually confined to small number of modules

1. **The pesticide Paradox:**

• If the same tests are repeated over and over again, eventually the same set of test case will no longer find any new defects

• To overcome this “pesticides 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.

1. **Testing is context dependent:**

• Testing is basically context dependent.

• Testing is done differently context

• Different kinds of sites are tested differently

1. **Absence of Errors Fallacy:**

• If the system built is unusable and does not fulfill the user’s needs and expectations then finding and fixing defects does not help.

• Even after defects have been resolved it may still be unusable and/or does not fulfill the users need and expectation

1. **Difference between QA v/s QC v/s Tester**

|  |  |  |  |
| --- | --- | --- | --- |
| 1. Sr no. | Quality assurance | Quality control | Testing |
| 1. | Activities which ensure the implementation of processes, procedures and standards in context to verification of developed software and intended requirements. | Activities which ensure the verification of developed software with respect to documented (or not in some cases) requirements. | Activities that ensure the identification of bugs/errors/defects in the Software. |
| 2. | Focuses on processes and procedures rather than conducting actual testing on the system. | Focuses on actual testing by executing Software with the intent to identify bugs/defects through the implementation of procedures and processes. | Focuses on actual testing. |
| 3. | Process-oriented activities. | Product-oriented activities. | Product-oriented activities. |
| 4. | Preventive activities. | It is a corrective process. | It is a preventive process. |
| 5. | It is a subset of the Software Test Life Cycle (STLC). | QC can be considered as the subset of Quality Assurance. | Testing is the subset of Quality Control. |

**24.Difference between smoke and sanity?**

|  |  |
| --- | --- |
| Smoke testing | Sanity testing |
| Smoke Testing is performed to ascertain that the critical functionalities of the program is working fine | Sanity Testing is done to check the new functionality/bugs have been fixed |
| The objective of this testing is to verify the “stability” of the system in order to proceed with more rigorous testing | The objective of the testing is to verify the “rationality” of the system in order to proceed with more rigorous testing |
| This testing is performed by the developers or testers | Sanity testing in software testing is usually performed by testers |
| Smoke testing is usually documented or scripted | Sanity testing is usually not documented and is unscripted |
| Smoke testing is a subset of Acceptance testing | Sanity testing is a subset of [Regression Testing](https://www.guru99.com/regression-testing.html) |
| Smoke testing exercises the entire system from end to end | Sanity testing exercises only the particular component of the entire system |
| Smoke testing is like General Health Check Up | Sanity Testing is like specialized health checkup |

**25.Difference between verification and validation**

|  |  |  |
| --- | --- | --- |
|  | **Verification** | **Validation** |
| **Definition** | It is a process of checking if a product is developed as per the specifications. | It is a process of ensuring that the product meets the needs and expectations of stakeholders. |
| **What it tests or checks for** | It tests the requirements, architecture, design, and code of the software product. | It tests the usability, functionalities, and reliability of the end product. |
| **Coding requirement** | It does not require executing the code. | It emphasizes [executing the code](https://www.bplogix.com/process-director/low-code-development) to test the usability and functionality of the end product. |
| **Activities include** | A few activities involved in verification testing are requirements verification, design verification, and code verification. | The commonly-used validation activities in software testing are usability testing, performance testing, system testing, security testing, and functionality testing. |
| **Types of testing methods** | A few verification methods are inspection, code review, desk-checking, and walkthroughs. | A few widely-used validation methods are black box testing, white box testing, integration testing, and acceptance testing. |
| **Teams or persons involved** | The quality assurance (QA) team would be engaged in the verification process. | The software testing team along with the QA team would be engaged in the validation process. |
| **Target of test** | It targets internal aspects such as requirements, design, software architecture, database, and code. | It targets the end product that is ready to be deployed. |

**26 Explain types of performance testing.**

* Load testing
* Stress testing
* Endurance testing
* Spike testing
* Volume testing
* Scalability testing

1. **Load testing:**

The purpose of [load testing](https://qualitestgroup.com/initiatives/load-and-performance-testing-services/) is to evaluate the application’s performance under increasingly high numbers of users. Load, or increasing numbers of users are applied to the application under https://qualitestgroup.com/initiatives/load-and-performance-testing-services/test and the results are measured to validate the requirements are met. This load can be the expected concurrent number of users on the application performing a specific number of transactions within the set duration. This test will give out the response times of all the important business-critical transactions. If the database, application server, etc. are also monitored, then this simple test can itself point towards  bottlenecks in the application software.

1. **Stress testing:**

This test pushes an application beyond normal load conditions to determine which components fail first. Stress testing attempts to find the breaking point of the application and is used to evaluate the robustness of the application’s data processing capabilities and response to high volumes of traffic.

1. **Endurance testing:**

Endurance testing evaluates the performance of the system under load over time. It is executed by applying varying loads to the application under test for an extended period of time to validate that the performance requirements related to production loads and durations of those loads are met. Endurance testing can be considered a component of load testing and is also known as soak testing.

1. **Spike testing:**

This testing evaluates the ability of the application to handle sudden volume increases. It is done by suddenly increasing the load generated by a very large number of users. The goal is to determine whether performance will suffer, the system will fail, or it will be able to handle dramatic changes in load. This testing is critical for applications that experience large increases in the number of users; for example, utility customers reporting power outages during storms. This can be considered a component of stress testing.

1. **Volume testing:**

Also known as flood testing, this testing is used to evaluate the application’s ability to handle large volumes of data. The impact on response time and the behaviour of the application are analysed. This testing can be used to identify bottlenecks and to determine the capacity of the system. This type of performance testing is important for applications that deal with big data.

1. **Scalability testing:**

This testing is used to determine your application’s ability to handle increasing amounts of load and processing. It involves measuring attributes including response time, throughput, hits and requests per second, transaction processing speed, CPU usage, Network usage, and more. The results of this testing can be used in the planning and design phases of development which reduces costs and mitigates the potential for performance issues.

**27**. **What are Error, Defects, Bug and Failure?**

“A mistake in coding is called error, error found by tester is called defect, defect accepted by development team then it is called bug, build does not meet the requirements then it is failure”

**28**. **Difference between priority and severity**

**Priority :**

➢ Priority is relative and business focused. Priority defines the order in which we should resolve a defect, should we fix it now or can it wait? This priority status is set by the tester to the developer mentioning the time frame to fix the defect. If high priority is mention then the developer has to fix it at the earliest. The priority status is set based on the customer requirements.

**Severity :**

➢ Severity is absolute and customer focused. It is the extent to which the defect can affect the software In other words it defines the impact that given defect has on the system.

**29. What is Bug Life Cycle?**

• “A computer bug is an error, flow, mistake, failure or fault in a computer program that prevents it from working correctly or produces an incorrect result. Bug arise from mistake and errors made by people, in either a program’s source code or its design.”

• When bug is discovered, it goes through several states and eventually reaches one of the terminal states where it becomes inactive and closed.

**30.** **explain the difference between Functional testing and Non-functional testing.**

|  |  |
| --- | --- |
| **Functional Testing** | **Non-Functional Testing** |
| Functional testing is performed using the functional specification provided by the client and verify the system against the functional requirements. | non functional testing checks the performance reliability, scalability, and other non-functional aspects of the software systems. |
| Functional testing is executed first | Non-functional testing should be performed after functional testing |
| Manual testing or automation tools can be used for functional testing | Using tools will be effective for this testing |
| Business requirements are the inputs to functional testing | Performance parameters like speed, scalability are inputs to non-functional testing |
| Functional testing describes what the product does | Non-functional testing describes how well the product works. |
| Easy to do manual testing | Tough to do manual testing |
| Types of Functional testing are  ∙ Unit Testing  ∙ Smoke Testing  ∙ Sanity Testing  ∙ Integration Testing  ∙ White box testing  ∙ Black Box testing  ∙ User Acceptance testing | Types of Non-functional testing are  ∙ Performance Testing  ∙ Load Testing  ∙ Volume Testing  ∙ Stress Testing  ∙ Security Testing  ∙ Installation Testing  ∙ Penetration Testing  ∙ Compatibility Testing  ∙ Migration Testing |

31. What is the difference between the STLC (Software Testing Life Cycle) and SDLC (Software Development Life Cycle)

|  |  |
| --- | --- |
| **SDLC** | **STLC** |
| SDLC is mainly related to software development. | STLC is mainly related to software testing. |
| Besides development other phases like testing is also included. | It focuses only on testing the software. |
| SDLC involves total of six phases or steps. | STLC involves only five phases or steps. |
| In SDLC, more number of members (developers) are required for the whole process. | In STLC, less number of members (testers) are needed. |
| In SDLC, the development team makes the plans and designs based on the requirements. | In STLC, the testing team(Test Lead or Test Architect) makes the plans and designs. |
| The goal of SDLC is to complete the successful development of software. | The goal of STLC is to complete the successful testing of software. |
| It helps in developing good quality software. | It helps in making the software defects-free. |
| SDLC phases are completed before the STLC phases. | STLC phases are performed after the SDLC phases. |
| Post-deployment support, enhancement, and update are to be included if necessary. | Regression tests are run by the QA team to check deployed maintenance code and maintain test cases and automated scripts. |
| The creation of reusable software systems is the end result of SDLC. | A tested software system is the end result of STLC. |

**32.** **What is the difference between test scenario, test cases and test script?**

❖**Test Scenario**:

* Is any functionality that can be tested
* Is derived from test artifacts like Business Requirements Specification (BRS) and Software Requirements Specification (SRS)
* Helps test the end to end functionality in Agile way
* Is more focused on what to test
* Test less time and fewer resources to create

**❖ Test Cases:**

* Is a set of actions executed to verify particular features or functionality
* Is mostly derived from test scenario
* Helps in exhaustive testing of an app
* Is focused on what to test and how to test
* Requires more resources and time

❖ **Test Script :**

* Is a set of instructions to test an app automatically
* Is mostly derived from test cases
* Helps to test specific things repeatedly
* Is focused on the expected result
* Requires less time for testing but more resources for scripts creating and updating

**33. What is priority?**

Priority is Relative and Business-Focused. Priority defines the order in which we should resolve a defect. Should we fix it now, or can it wait? This priority status is set by the tester to the developer mentioning the time frame to fix the defect. If high priority is mentioned then the developer has to fix it as the earliest. The priority status is set based on the customer’s requirements.

* For example: If the company name is misspelled on the home page of the website, then the priority is high and severity is low to fix it. Priority can be of the following types:
* Low: The defect is an irritant that should be repaired, but a repair can be deferred until after a more serious defect has been fixed.
* Medium: The defect should be resolved in the normal course of development activities. It can wait until a new build or version is created.
* High: The defect must be resolved as soon as possible because the defect is affecting the application or the product severely. The system cannot be used until the repair has been done.
* Critical: Extremely urgent, resolve immediately

**34. What is severity?**

• Severity is absolute and customer focused. It is the extent to which the defect can affect the software. In other word it defines the impact that a given defect has on the system.

• For example : if an application of the web page crashes when a remote link is clicked in this case clicking the remote link by an user is rare but the impact of application crashing severe. So the severity is high but priority is low.

**36 Bugs categories are…**

• There are some categories

➢ Security

➢ Database

➢ Functionality (Critical/General)

➢ UI

**37. Advantages of Bugzilla**

* Open source, free bug tracking tool.
* Automatic [Duplicate Bug Detection](https://cloudinfrastructureservices.co.uk/how-to-setup-bugzilla-issue-tracker-on-azure-aws-gcp/).
* Search option with advanced features.
* File/Modify Bugs By Email.
* Move Bugs Between Installs.
* Multiple [Authentication](https://cloudinfrastructureservices.co.uk/adfs-vs-azure-ad-how-authentication-has-evolved/) Methods ([LDAP](https://cloudinfrastructureservices.co.uk/radius-vs-ldap-vs-kerberos/),[Apache server](https://cloudinfrastructureservices.co.uk/how-to-setup-apache-web-server-mysql-server-on-linux-in-azure-aws-gcp/)).
* Time Tracking.
* Automated bug reporting; has an API to interact with the system.
* Integrated email capabilities.

**38. What are the different Methodology in Agile Development Model ?**

1. Scrum methodology:

Scrum is a lightweight framework of Agile Project Management, it can be adopted to conduct iterative and all types of incremental projects. Due to specific characteristics like simplicity, sustained productivity and strength for blending several underlying approaches adapted by other agile methods, Scrum has obtained popularity over the years.

2. Kanban:

Kanban is an eminently visual workflow management approach, that can be employed for visualizing and thoroughly maintaining the making of products, it focuses on continual delivery of the product , but is not making stress to the entire software development life cycle. Similar to scrum, kanban is the process developed for supporting collaborative teamwork more effectively.

**39. Explain the difference between Authorization and Authentication in web testing. What are the common problems faced in web testing?**

**Authentication:**

* Authentication verifies who the user is.
* Authentication works through passwords, one-time pins, biometric information, and other information provided or entered by the user.
* Authentication is the first step of a good identity and access management process.
* Authentication is visible to and partially changeable by the user.
* Example: By verifying their identity, employees can gain access to an HR application that includes their personal pay information, vacation time, and 401K data.

**Authorization:**

* Authorization determines what resources a user can access.
* Authorization works through settings that are implemented and maintained by the organization.
* Authorization always takes place after authentication.
* Authorization isn’t visible to or changeable by the user.
* Example: Once their level of access is authorized, employees and HR managers can access different levels of data based on the permissions set by the organization.

**40 When to use Usability Testing?**

• Usability testing is a method of testing the functionality of a website, app or other digital product by observing real users as they attempt to complete tasks on it. The users are usually observed by researchers working for a business.

• Usability testing can and should be conducted on the current iteration of a product before beginning any new design work, after you have begun the strategy work around a brand new site or app.

**41. What is the procedure for GUI Testing?**

Check all the GUI elements for size, position, width, length and acceptance of Characters or numbers. For instance, you must be able to provide inputs to the input fields.

• Check you can execute the intended functionality of the application using the GUI.

• Check Error Messages are displayed correctly

• Check for Clear demarcation of different sections on screen

• Check Font used in application is readable

• Check the alignment of the text is proper

• Check the Color of the font and warning messages is aesthetically pleasing

• Check that the images have good clarity

• Check that the images are properly aligned

• Check the positioning of GUI elements for different screen resolution.

**42. Write agile manifesto principles?**

**• Individuals and interactions, Over processes and tools –**

Suppose the team finds any issue in software then they search for another process or tool to resolve the issue. But, in Agile, it is preferable to interact with client, manager or team regarding issue and make sure that the issue gets resolved.

**• Working software, Over comprehensive documentation -** Documentation is needed, but working software is much needed. Agile is not saying that documentation is not needed, but working software is much needed. For example, you have 20-page documents, but you do not have a single prototype of the software. In such a case, the client will not be happy because, in the end, the client needs a document.

**• Customer collaboration, Over contract negotiation –**

Contract negotiation is important as they make the budget of software, but customer collaboration is more important than over contract negotiation. For example, if you stuck with the requirements or process, then do not go for a contract which we have negotiated. You need to interact with the customer, gather their requirements.

**• Responding to change, over following a plan –**

In the waterfall model, everything is planned, i.e., at what time, each phase will be completed. Sometimes you need to implement the new requirements in the middle of the software, so you need to be versatile to make changes in the software