***Module 2 – Manual Testing***

1. **What is Exploratory Testing?**

* **Exploratory Testing** is a software testing technique where testers actively explore the application without predefined test scripts, relying on their understanding of the application, its features, and their creativity to identify defects and issues.
* The goal of exploratory testing is to discover defects that might not be covered by traditional scripted testing, and to gain insights into the software’s behavior in an ad-hoc, flexible manner.
* Example:- flexibility, creativity, real time documentation, experience driven.

1. **What is traceability matrix?**

* A **Traceability Matrix** is a document used in software testing and requirements management to ensure that all requirements are covered by test cases.
* It is a tool that helps track the relationships between requirements, test cases, and other project artifacts.
* The main purpose of the traceability matrix is to verify that the software meets the specified requirements and to ensure full test coverage, leaving no requirement untested.

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

* **Boundary Value Testing (BVT)** is a software testing technique used to identify defects at the boundaries rather than within the middle of input ranges.
* It is based on the principle that errors often occur at the edges or boundaries of input values, rather than in the center.
* This technique is particularly useful for testing input fields, ranges, and conditions that involve numeric or enumerated data.

1. **What is Equivalence partitioning testing?**

* **Equivalence Partitioning** is a software testing technique used to reduce the number of test cases while maintaining good test coverage.
* It is a type of black-box testing method that divides the input data of a program into partitions (or groups), where the program's behavior is expected to be the same for any input within a given partition.
* The idea is that if one test case from a partition passes, all other values in that partition should pass as well. Similarly, if one test case from a partition fails, the rest will also fail.

1. **What is Integration testing?**

* **Integration Testing** is a type of software testing that focuses on verifying the interactions and interfaces between different components or systems.
* The primary goal of integration testing is to identify any issues that occur when individual modules or subsystems are combined to work together as a whole.

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

* The level of **risk** in any given situation—whether it's in software development, project management, or any other domain—is determined by a combination of **probability** and **impact**.
* These two factors help in assessing how likely a risk is to occur and how severe the consequences would be if it does.

1. **What is Alpha testing?**

* **Alpha Testing** is a type of **acceptance testing** that is performed by the **development team** or an **internal testing team** within the organization before the software is released to external users or customers. It is one of the final testing stages in the software development life cycle (SDLC), typically done after **unit testing**, **integration testing**, and **system testing.**

1. **What is beta testing?**

* **Beta Testing** is a type of software testing that occurs after **alpha testing** and is the final phase of testing before the software is released to the general public.
* During beta testing, a **select group of external users** (often called "beta testers") is given access to the software to test it in real-world environments.
* The primary goal is to identify any remaining issues, gather feedback, and ensure that the software performs as expected for a wider audience.

1. **What is component testing?**

* **Component Testing** (also known as **Module Testing** or **Unit Testing**) is a type of software testing that focuses on verifying individual components or modules of a software system in isolation, to ensure that each part functions correctly.
* This testing typically occurs early in the software development life cycle (SDLC), and its primary goal is to validate the correctness of each component before it is integrated with other parts of the system.

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

* Functional System Testing is a type of testing that focuses on evaluating the overall functionality of an entire system to ensure that it operates according to the specified functional requirements.
* It involves verifying whether the system behaves as expected and meets the defined business and user requirements.
* Unlike other testing types that may focus on individual components or the system’s performance, functional system testing looks at the system as a whole to confirm that all features and workflows perform their intended tasks correctly.

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

* Non-functional testing refers to the testing of non-functional aspects of a system, focusing on how the system performs rather than what it does.
* Non-functional testing evaluates attributes that define the quality of the system. These attributes are not related to specific functionalities, but they impact the user experience and system performance

1. **What is GUI Testing?**

* **GUI stands for** Graphical User Interface.
* **This GUI** Testing is the process of testing the graphical user interface of a software application to ensure that it meets its specifications and provides a smooth user experience.
* The primary goal of GUI testing is to verify that the interface is functional, visually appealing, and user-friendly.
* This involves checking how users interact with the application through buttons, menus, icons, forms, and other visual elements.

1. **What is Adhoc testing?**

* **Adhoc Testing** is an informal and unstructured type of software testing where testers try to find defects or issues without any specific test plan or documentation.
* It is generally performed without formal test cases, procedures, or guidelines. The goal is to explore the application in an attempt to identify potential issues that were not foreseen during the standard testing process.

1. **What is load testing?**

* **Load Testing** is a type of performance testing where the system is tested under a specific amount of load (number of users, transactions, or data) to determine how well it performs under normal or expected conditions.
* The primary goal of load testing is to verify that the application or system can handle the expected number of users or transactions without performance degradation or failure.

1. **What is stress Testing?**

* **Stress Testing** is a type of performance testing where a system or application is subjected to extreme conditions or beyond its expected operational limits to evaluate how it behaves under heavy load or stress.
* The goal of stress testing is to determine the system’s breaking point, how it recovers from failure, and whether it fails gracefully.

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

* **White Box Testing**, also known as **Clear Box Testing**, **Glass Box Testing**, or **Structural Testing**, is a type of software testing where the internal structures or workings of the application are known to the tester.
* In white box testing, the tester has full visibility into the code, algorithms, architecture, and design of the application. This allows the tester to test the internal logic, control flow, and the system’s internal components directly.
* **Types** :- web base testing, desktop base testing, mobile base testing, game base testing.

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

* **Black Box Testing** is a software testing method where the tester evaluates the functionality of an application without any knowledge of its internal workings or code.
* The focus is on testing the system based on its **inputs** and **outputs**. The tester does not need to understand the internal structure, algorithms, or code implementation; rather, they assess the system’s behavior and how it meets the requirements.
* **Techniques:-** Equivalence partitioning, Boundary value analysis, Decision tables, State transition testing, Use-case Testing.

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

* Defects in a product or process are typically categorized based on their impact, severity, or type. The common categories of defects are as follows
* 1. **Critical:** The defects **will cause** downstream damage.
* **2. Major:** The defects **could cause** a downstream damage.
* **3. Minor:** The defects are **highly unlikely to cause** the downstream damage.

1. **Mention what bigbang testing is?**

* **Big Bang Testing** is a type of software testing approach where all components or modules of the system are integrated at once, and then the system is tested as a whole.
* In this approach, developers or testers do not test each component or module individually before integration; instead, they integrate all components at once and then begin testing the system as a whole.

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

* **Exit criteria** refer to the conditions or set of requirements that must be met before a specific phase or process in a project, such as a software testing or development phase, can be considered complete.
* The purpose of exit criteria is to provide a clear and objective basis for determining when a particular phase of the project has been successfully finished, ensuring that the necessary tasks and standards have been met

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

* **Regression testing** should be performed whenever changes are made to a software application to ensure that new modifications or enhancements do not adversely affect the existing functionality.

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

* The **7 Key Principles** of software testing are fundamental are as under:-
* **1. Testing Shows the Presence of Defects**:- Testing can only confirm the existence of defects, not their absence. The goal of testing is to identify defects in the software, not to prove that the software is entirely free of defects
* **2. Exhaustive Testing is Impossible:-** It is practically impossible to test every possible input, scenario, or path in a complex system due to time and resource constraints.
* **3. Early Testing:-** Testing should begin as early as possible in the software development lifecycle, ideally during the requirements and design phases. Early testing helps to detect defects at an early stage when they are cheaper and easier to fix.
* **4. Defect clustering:-** Defects tend to cluster in specific areas of the application, rather than being spread evenly across the entire system. Testers should focus on these critical areas that are more likely to contain defects.
* **5. The 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.
* **6. Testing is context dependent:-**  Testing is basically context dependent. Testing is done differently in different contexts. Different kinds of sites are tested differently.
* **7. Absence of error fallacy: -** Even if testing does not uncover defects, it does not mean the software is free of errors. The software might still fail to meet the users' expectations or business requirements, even though no technical defects were found. This principle emphasizes that testing must not only verify the correctness of the software but also ensure that the software meets the user’s needs and performs as expected.

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

|  |  |  |  |
| --- | --- | --- | --- |
| **Aspect** | **QA** | **QC** | **TESTER** |
| Definition | 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 which ensure the identification of bugs/error/defects in the Software. |
| Focus | 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. |
| Activities | Process oriented activities | Product oriented activities | Product oriented activities |
| Nature of work | Preventive activities | It is a corrective process | It is a preventive process |
| Timing | It is a subset of software development life cycle(SDLC) | It is a considered as Quality Assurance(QA) | Testing is subset of Quality control(QC) |

1. **Difference between Smoke and Sanity?**

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Smoke Testing** | **Sanity Testing** |
| |  | | --- | | **Definition** |  |  |  | | --- | --- | |  |  |  |  |  |  | | --- | --- | --- | |  |  |  |  |  | | --- | |  |  |  |  |  | | --- | --- | --- | |  |  |  | | Visible vapor, gases, and particles produced by combustion. | |  |  | | --- | --- | |  |  |   Mental health state characterized by rational thought, clarity and emotional stability. |
| **Nature** | Physical substance or phenomenon. | Psychological condition or mental state. |
| **Usage** | Describes something tangible, like fumes or vapor from a fire. | Refers to mental well-being and the ability to think clearly. |
| **Context** | Often used in the context of fire, pollution, or smoking. | |  |  | | --- | --- | |  | Commonly used in psychology, law, or discussions of mental health. | |
| **Examples** | "The smoke from the fire filled the room." | "She was trying to keep her sanity during the stressful week." |

1. **Difference between verification and Validation**

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Verification** | **validation** |
| Definitions | The process of evaluating work-products (not the actual final product) of a development phase to determine whether they meet the specified requirements for that phase. | The process of evaluating software during or at the end of the development process to determine whether it satisfies specified business requirements. |
| **Objective** | To ensure that the product is being built according to the requirements and design specifications. In other words, to ensure that work products meet their specified requirements. | To ensure that the product actually meets the user’s needs, and that the specifications were correct in the first place. In other words, to demonstrate that the product fulfills its intended use when placed in its intended environment. |
| **Question** | Are we building the product right? | Are we building the right product? |
| **Evaluation Items** | Plans, Requirement Specs, Design Specs, Code, Test Cases | The actual product/software. |
| **Activities** | Reviews, Walkthroughs, Inspections. | Testing. |

1. **Explain types of Performance testing.**

* Performance testing is a type of software testing used to determine how a system performs under various conditions, such as load, stress, and scalability.
* The goal is to ensure that the system can handle expected traffic, meet performance requirements, and remain stable under stress.
* There are several types of performance testing, each with its focus. Here are the key types:
* **1. Load testing:** To ensure the system can handle the expected number of concurrent users or transactions without degradation in performance.
* **2. Stress testing:** To identify the system's breaking point (how much load it can tolerate before failing) and to observe how it recovers from failure.
* **3.Endurance testing :** To identify any potential memory leaks, resource depletion, or degradation in performance that may occur over time
* **4. Spike testing:** To determine how the system handles short-term, unexpected surges in demand and whether it can recover gracefully after the spike.
* **5.Volume testing :** To ensure the system can handle large datasets efficiently without performance degradation, such as slow query responses or system crashes
* **6. Scalability testing:** To identify whether the system can handle growth in terms of traffic, users, or data volume.

1. **What is Error, Defect, Bug and failure?**

* **ERROR:-** A discrepancy between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition. This can be a misunderstanding of the internal state of the software, an oversight in terms of memory management, confusion about the proper way to calculate a value, etc.
* **DEFECT:-** Commonly refers to several troubles with the software products, with its external behavior or with its internal features.
* **BUG:-** A fault in a program which causes the program to perform in an unintended or unanticipated manner. See: anomaly, defect, error, exception, and fault. Bug is terminology of Tester.
* **FAILURE:-** The inability of a system or component to perform its required functions within specified performance requirements. See: bug, crash, exception, and fault.

1. **Difference between Priority and Severity**

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Priority** | **Severity** |
| Definition | Describes the **impact** of the defect on the system. | Describes the **urgency** or order in which the defect should be fixed. |
| Focus | Concerned with how serious the defect is for the system. | Concerned with how soon the defect should be fixed. |
| Determined By | Typically determined by the **developer** or **technical team**. | Typically determined by **project managers** or **product owners**. |
| Impact | Affects the **system functionality** or behavior. | Affects the **business objectives** or deadlines. |
| Fixing Order | Determines the **seriousness of the issue** in terms of system performance. | Determines **which defect to fix first** based on business needs. |

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

* The **Bug Life Cycle** (also known as the **Defect Life Cycle**) refers to the various stages that a bug or defect goes through from its identification to its resolution and closure.
* It helps track the status of a bug and ensures proper handling and resolution of the issue within a software development project. The life cycle defines how a bug is managed by the testing and development teams.
* **1. New**: Bug is reported.
* **2**. **Assigned**: Bug is assigned to a developer.
* **3. Open (In Progress)**: Developer works on fixing the bug.
* **4**. **Fixed**: Bug is fixed, ready for testing.
* **5. Retested**: Tester verifies the fix.
* **6. Closed**: Bug is resolved and closed.
* **7. Reopened** (if needed): Bug is reopened if the fix doesn’t work.
* **8.** **Deferred**: Bug is postponed to a later release.
* **9**. **Rejected**: Bug is not a defect, rejected by the team.

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

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

1. **To create HLR & TestCase of**

* **1. (Instagram , Facebook) only first page**
* **2. Facebook Login Page : https://www.facebook.com/**

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| | **Aspect** |  |  | | --- | --- | --- | | **SDLC (Software Development Life Cycle)** | **STLC (Software Testing Life Cycle)** |
| **Definition** | The overall process of software development, from planning to deployment | The process of testing the software to ensure its quality and functionality. |
| **Focus** | Development, delivery, and maintenance of the software. | Testing and quality assurance of the software. |
| **Phases** | Requirements, Design, Development, Testing, Deployment, Maintenance. | Requirement Analysis, Test Planning, Test Design, Test Execution, Defect Reporting, Test Closure. |
| **Involvement** | |  |  | | --- | --- | |  | Developers, business analysts, project managers, and designers. | | Testers, test leads, and QA professionals. |
| **Timing** | |  |  | | --- | --- | |  | Happens throughout the entire software development process. | | Happens primarily during the testing phase after development. |
| **Goal** | To develop and deliver a functional software product. | |  |  |  | | --- | --- | --- | |  |  | To ensure that the software is free of defects and meets quality standards. | |

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

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | **Aspect** |  |  |  | | --- | --- | --- | --- |  |  | | --- | |  | | **Test Scenario** | **Test Case** | **Test Script** |
| **Definition** | |  | | --- | | High-level description of what to test. |  |  | | --- | |  |  |  | | --- | |  | | Detailed step-by-step instructions for testing. | Automated code to execute test cases. |
| **Purpose** | Identify what functionality needs testing. | Provide clear steps, inputs, and expected results. | Automate the execution of test cases. |
| **Level of Detail** | High-level, abstract, broad. | Specific, detailed, focused on input-output. | Written in programming or scripting languages. |
| **Example** | Verify login functionality with valid credentials. | Enter valid credentials and verify login. | A script that automates login verification. |
| **Usage** | Helps organize testing efforts. | Ensures testing is done thoroughly and consistently. | Performs tests automatically on the system. |

1. **Explain what Test Plan is? What is the information that should be covered?**

* A **Test Plan** is a detailed document that outlines the **strategy** and **approach** for testing a software application.
* It serves as a guide for the testing process, detailing the scope, objectives, resources, schedule, and procedures to ensure the software is tested thoroughly and meets the required quality standards.
* The Test Plan helps in organizing and controlling the testing efforts, ensuring that all aspects of the software are tested in an efficient and systematic manner.
* It provides clarity to the testing team about their responsibilities, the tools they will use, and the criteria for success
* **Information should be covered:- Test plan ID, introduction, scope, objective, Test strategy, test criteria, test environment, test tools etc…**

1. **What is priority?**

* **Priority** in the context of software testing refers to the **urgency** or **importance** with which a defect or issue should be addressed.
* It indicates how quickly a defect should be fixed or resolved based on the **business impact** or **customer needs**.
* Priority is usually determined by the **project stakeholders**, such as product owners, business analysts, or managers, and it helps the development team to understand which defects need attention first.

1. **What is severity?**

* **Severity** in the context of software testing refers to the **degree of impact** a defect or bug has on the functionality, performance, or overall behavior of the software.
* It describes how **serious** or **critical** a defect is in terms of the system's operation and how it affects the end-user experience or system stability.
* Severity is typically assessed by the **testing team** and is focused on the **technical impact** of the defect rather than the urgency to fix it.

1. **Bug categories are…**

* **1. Functional bugs :** Bugs related to core functionality not behaving as expected.
* **2. UI/UX Bugs (User Interface/User Experience Bugs):** Issues with user interface design and overall user experience.
* **3. Performance Bugs:** Bugs related to slow performance, high resource usage, or system freezes.
* **4. Security Bugs:** Vulnerabilities that expose the system to security risks.
* **5. Compatibility Bugs:** Issues caused by software failing to work across different platforms.
* **6. Regression Bugs:** Bugs introduced by new code changes that break previously working features
* **7. Configuration Bugs:** Errors due to incorrect or missing configuration settings.
* **8. Installation Bugs:** Bugs occurring during software installation or setup
* **9. Data Bugs :** Problems related to incorrect or corrupt data

1. **Advantage of Bugzila.**
2. **Difference between priority and severity.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  | | --- | | **Aspect** |  |  | | --- | |  |  |  | | --- | |  | | **Severity** | **Priority** |
| **Definition** | Describes the **impact** of a defect on the system. | Describes the **urgency** of fixing a defect. |
| **Focus** | |  | | --- | | Technical impact on the system or user. |  |  | | --- | |  | | Business impact and release schedule. |
| **Assigned by** | Testers or developers (based on technical analysis) | |  | | --- | | Project managers, product owners, or stakeholders. |  |  | | --- | |  | |
| |  |  |  | | --- | --- | --- | | **Example** |  |  |  |  |  |  | | --- | --- | --- | |  |  |  |  |  |  |  | | --- | --- | --- | |  |  |  | | A crash or major functionality failure (high severity). | Fixing a typo in a high-visibility area (low priority). |
| **Determining Factor** | How **critical** the defect is to system functionality. | How **urgent** it is to fix the defect, based on its impact. |
| **Levels** | Critical, Major, Normal, Minor. | High, Medium, Low. |

1. **What are the different Methodologies in Agile Development Model?**
2. **Explain the difference between Authorization and Authentication in Web testing. What are the common problems faced in Web testing?**

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Authentication** | **Authorization** |
| **Definition** | The process of **verifying** the identity of a user, ensuring that the user is who they claim to be | The process of determining **what actions or resources** a user is allowed to access after they are authenticated. |
| **Purpose** | To confirm the **identity** of the user attempting to access the system. | To define **permissions** and restrict access to certain resources based on the user’s role or credentials. |
| **Focus** | Focuses on the **identity** of the user. | Focuses on the **permissions** assigned to the user. |
| **Example** | A user provides a username and password to log in to a website | A user with an admin role can access the admin dashboard, while a regular user cannot. |
| **Process** | Involves validating credentials (e.g., username, password, biometrics). | Involves checking if the authenticated user has the right **permissions** or **access level** to perform a specific action. |

* **Common problems faced in Web testing:**

**Cross-Browser Compatibility Issues**

Cross-Device and Mobile Compatibility

Network Speed and Latency Issues

Session Management Issues

Security Vulnerabilities

Usability Issues

Performance and Load Testing

Integration Issues

1. **To create HLR & TestCase of WebBased (WhatsApp web , Instagram)**

* **1. WhatsApp Web :** [**https://web.whatsapp.com/**](https://web.whatsapp.com/)
* **2. Instagram**
* **To create HLR and TestCase on this Link. https://artoftesting.com/**

1. **Write a scenario of only Whatsapp chat messages**
2. **Write a Scenario of Pen**
3. **Write a Scenario of Pen Stand**
4. **Write a Scenario of Door**
5. **Write a Scenario of ATM**
6. **When to used Usablity Testing?**

* Following are the situation when we can use usability testing:
  1. During the Design Phase (Early Prototyping)
  2. After UI/UX Development
  3. Before Product Launch (Beta Testing)
  4. When Introducing New Features or Updates
  5. When Targeting a New Audience or Market
  6. When User Behavior Changes Over Time

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

* **GUI (Graphical User Interface) Testing** is a process used to verify that the graphical elements of an application are functioning as intended, are visually correct, and provide a positive user experience.
* **Here is the procedure for performing GUI testing:**
  1. **Understand requirements** and design.
  2. **Set up the testing environment** with appropriate platforms and devices.
  3. **Identify the UI elements** to test (buttons, links, forms, etc.).
  4. **Create test cases** based on functionality, usability, and visual design.
  5. **Perform functional testing** for button actions, forms, and interactions.
  6. **Perform usability testing** to evaluate intuitiveness and user experience.
  7. **Check compatibility** across browsers, devices, and operating systems.
  8. **Test for visual consistency** in layout, design, and brand guidelines.
  9. **Test performance** to assess how the UI handles load and network conditions.
  10. **Report defects** found during testing.
  11. **Re-test after fixes** to confirm defect resolution and no new issues.
  12. **Final validation** before the release.

1. **Write a scenario of Microwave Owen**
2. **Write a scenario of Coffee vending Machine**
3. **Write a scenario of chair**
4. **To Create Scenario (Positive & Negative)**

**1. Gmail receiving**

**2. Online shopping to buy product (flipkart)**

1. **Write a Scenario of Wrist Watch**
2. **Write a Scenario of Lift(Elevator)**
3. **Write a Scenario of whatsapp Group (generate group)**

1. **Write a Scenario of Whatsapp payment**