**MODULE -2 (MANUAL TESTING)**

**1.What is exploratory testing?**

Exploratory testing is a type of software testing that is performed in an unstructured and ad-hoc manner. Testers explore the software and try different scenarios, inputs, and interactions to identify bugs and issues without using pre-written test cases. In exploratory testing, the goal is to uncover as many defects as possible and provide valuable feedback to the development team.

**2.What is traceability matrix?**

Traceability Matrix (STM) is a document that links and [maps test cases](https://www.testbytes.net/blog/difference-between-test-case-vs-test-scenario/) to their respective requirements, ensuring that each requirement has been adequately tested. It serves as a verification tool to confirm that all software requirements, as defined in the requirements specification document, are covered by test scenarios and cases The matrix facilitates identifying missing tests, understanding the impact of changes, and ensuring [comprehensive test coverage](https://www.testbytes.net/resource/how-explain-100-test-coverage-is-not-possible/).

**3.What is boundary value testing?**

Boundary value analysis is a black-box testing technique. It is closely associated with [equivalence class partitioning](https://artoftesting.com/equivalence-class-partitioning). In this technique, we analyse the behaviour of the application with test data residing at the boundary values of the equivalence classes.

**4.What is equivalence partitioning testing?**

Equivalence class partitioning is a [software testing](https://en.wikipedia.org/wiki/Software_testing) technique that divides the input data of a software unit into partitions of equivalent data from which test cases can be derived. In principle, test cases are designed to cover each partition at least once. This technique tries to define test cases that uncover classes of errors, thereby reducing the total number of test cases that must be developed.

**5.What is integration testing?**

**Integration testing** is a type of software testing that focuses on verifying the interactions and interfaces between different components or modules of a system. It ensures that the integrated parts of the software work together as expected. During integration testing, various components are combined and tested as a whole to identify any issues related to data flow, communication, or functionality. This helps catch defects early in the development process and ensures smooth collaboration between different parts of the software.

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

A risk’s impact is determined as per how much damage it could do to the system. A security risk is certainly a huge red flag. Yet, we need to [analyse the impact](https://testsigma.com/blog/impact-analysis-in-testing/) by calculating the levels of the risks. For this, we need probability and impact value. Both these parameters range within High, Medium, and Low values. The security risk discussed in the above example would be awarded a high value for probability and impact, making it an immediate threat to the system

**7.What is alpha testing?**

**Alpha Testing** is a type of software testing performed to identify bugs before releasing the product to real users or to the public. Alpha Testing is one of the **user acceptance testing**. Alpha testing only because it is done early on, near the end of the development of the software. It is the last testing stage before the software is released into the real world.

**8.What is beta testing?**

Beta testing is the process of testing a software product or service in a real-world environment before its official release. It is an essential step in the software development lifecycle as it helps identify bugs and errors that may have been missed during the development process.

**9.What is component testing?**

Component testing is a type of software testing that involves testing each component of a software product individually. It validates the functionality and usability of each component before integrating it with others to form a complete software product. It makes it easier to determine which software component results in errors or bugs. It verifies the behaviour and functionality of individual components.

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

Functional testing is a software testing process that checks the functionality of a software application. It is conducted to verify that all the features of the application. It is usually used to detect and fix defects in the software before it is released to users.

**11.What is nonfunctional testing?**

Non-functional testing is a type of software testing to test non-functional parameters such as reliability, load test, performance and accountability of the software. The primary purpose of non-functional testing is to test the reading speed, stability and scalability of the software system as per non-functional parameters. The parameters of non-functional testing are never tested before the functional testing.

**12.What is GUI testing?**

**Graphical User Interface Testing (GUI) Testing** is the process for ensuring proper functionality of the graphical user interface (GUI) for a specific application. GUI testing generally evaluates a design of elements such as layout, colours and also fonts, font sizes, labels, text boxes, text formatting, captions, buttons, lists, icons, links, and content.

**13.What is Ad hoc testing?**

Adhoc testing is a type of software testing that is performed informally and randomly after the formal testing is completed to find any loophole in the system. For this reason, it is also known as Random or Monkey testing. Adhoc testing is not performed in a structured way so it is not based on any methodological approach. No documentation, no test cases, no test design this is the approach of Adhoc testing.

**14.What is load testing?**

**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. It determines the behaviour of the application when multiple users use it at the same time. It is the response of the system measured under varying load conditions.

**15.What is stress testing?**

Stress testing is defined as a type of software testing that verifies the stability and reliability of the system. It determines error handling of the system under extremely heavy load conditions. It how the system analyses work under extreme conditions. Stress testing is performed to ensure that the system would not crash under crunch situations. Stress testing is also known as **Endurance Testing.**

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

**White Box Testing** is a testing technique in which software’s internal structure, design, and coding are tested to verify input-output flow and improve design, usability, and security. In white box testing, code is visible to testers, so it is also called Clear box testing, Open box testing, Transparent box testing, Code-based testing, and Glass box testing.

Types of white box testing

* Unit testing
* Integration Testing
* Penetration Testing
* Mutation Testing

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

Black box testing is also referred to as **specification-based testing.** It involves performing testing based on the specification of the system under test. The knowledge of the internal architecture and the application code is not required in black-box testing.

Techniques:

### **Equivalence class partitioning**

[Equivalence class partitioning](https://artoftesting.com/equivalence-class-partitioning) involves partitioning the input data into logical groups or equivalence classes. All the data items lying in an equivalence class are assumed to be processed in the same way by the application when passed as input.  
  
E.g. for software that finds the square of a number, we can have different equivalence classes like – all positive numbers, negative numbers, decimal numbers.  
  
Its advantage is – the overall test execution time reduces as the number of test data greatly reduces.

### **Boundary value analysis**

[Boundary value analysis](https://artoftesting.com/boundary-value-analysis), is a black-box testing technique, closely associated with equivalence class partitioning. In this technique, we analyse the behaviour of the application with test data residing at the boundaries of the equivalence classes.

E.g. for equivalence classes with an input between 0 to 100, the test data using boundary value analysis would be 0 and 100.

Its advantage is – it is easier and faster to find defects as the density of defects at boundaries is more.

### **Decision tables**

Decision tables testing is used to test the application’s behaviour based on a different combination of input values. A decision table has a different set of input combinations and their corresponding expected outcomes on each row.

### **Cause-effect graph**

A cause-effect graph testing is carried out using a graphical representation of input i.e. cause and output i.e. effect. We can find the coverage of cause-effect graphs based on the percentage of combinations of inputs tested out of the total possible combinations.

### **State transition testing**

The state transition testing is based on a state machine model. In this technique, we test the application by graphically representing the transition between the different states of the application based on the different events and actions.

* **Use case testing**

Use case testing is a [type of testing](https://artoftesting.com/types-of-testing) that is carried out using use cases. In this technique, we test the application using use-cases, representing the interaction of the application with the different actors.

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

Error of commission

Error of omission

Error of clarity

Error of speed

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

Big bang testing is the type of integration testing. Big bang integration testingis a testing approach where all components or modules are integrated and tested as a single unit. This is done after all modules have been completed and before any system-level testing is performed. This is in contrast to incremental integration testing, in which components are tested one at a time or in small groups.

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

Software testing teams will use exit criteria to determine if a test plan or project can exit to the next stage or be considered complete. This isn't something that should be left up to the subjective and/or ad hoc decisions of a test admin, as it can directly impact the success of the next stage or project as a whole.

**21.When should regression testing be performed?**

Regression testing should be performed whenever your codebase has been modified or altered in any way as well as to verify any previously discovered issues marked as fixed. The more often the better: frequent partial regression testing will help your developers fix the reported defects on time, and your project to avoid any long-term pitfalls and technical debt caused by poor code quality. In some project might have the resources to perform the tests after the slightest changes have been introduced to the codebase, for most projects designing and maintaining such a multiplicity of regression tests may simply be infeasible.

**22.What is the 7 key principles? Explain in detail?**

### 1. **Testing Shows the Presence of Defects**

This testing principle states that software testing concerns the presence of defects, not their absence. It involves identifying bugs in the software product and fixing them before deploying and releasing it to the market.

It is important to note that testing does not make the product defect-free 100%. Instead, it reduces the number of defects. Even repetitive testing and bug repair may leave some bugs unfound.

### 2. **Exhaustive Testing is Impossible**

It tests and verifies every functionality of the application using both valid and invalid inputs. This principle states that it is not possible to test EVERYTHING. This is because the combinations of input and output are infinite. Hence, it is not possible to check your application for every combination. You just create some primary test cases according to your use cases, and if they pass, you need to assume that the application will work correctly.

3. **Early Testing**

This testing principle says that it is beneficial to start testing as early as possible in the [software development life cycle](https://artoftesting.com/software-development-life-cycle-sdlc) (SDLC). In fact, validating requirements starts even before coding. This uncovers potential bugs and errors in the software design and prevents the cost required to fix them if they were found in the later stages of development.

Early testing saves a lot of effort and time because resolving those issues in the later stages can become more complicated and critical and may require the team to rewrite some functionality.

### 4. **Defect Clustering**

Defect clustering refers to an application’s modules or features containing more bugs, generally resulting in operational failure. It is very less likely that bugs are distributed evenly throughout the application. Hence, testing should concentrate on such modules and features.

The most challenging part is how to determine the modules containing bugs. This entirely depends on the team’s knowledge and experience.

### 5. **Be Aware of Pesticide Paradox**

To avoid the concept of the Pesticide Paradox, it is essential to review and update test cases regularly, in addition to improving the existing test methods or implementing new ones.

### 6. **Testing is Context-Dependent**

Every software is developed with a different purpose. Hence, testing depends on the context of the software. As [different types of software](https://artoftesting.com/types-of-software) have distinct requirements, they require different types of testing. The same tests cannot detect all the major flaws in different types of applications, as users might have different ways of interacting with the product.

For instance, there are different types of cases for an E-Commerce application and a social media application. The same test cases on different products may not detect all the defects in one of the products.

7. **Absence of Errors Fallacy**

This principle states that it becomes unusable if your application is 99% bug-free but does not satisfy the end user’s requirements or business needs. There is a misconception among people that if you build 99% bug-free software, users will automatically use it. To make any software usable, it must satisfy users’ and business needs. So, run tests to check if the application meets the specified requirements.

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

Quality assurance: **Quality assurance** is process oriented. It is all about preventing defects by ensuring the processes used to manage and create deliverables works. Not only does it work, but is consistently followed by the team. Moreover, QA is about engineering processes that assure quality is achieved in an effective and efficient way.

Quality control: **Quality control**, alternatively, is product oriented. It is the function of software quality that determines the ending result is what was expected. Whereas QA is proactive, QC is reactive. QC detects bugs by inspecting and testing the product. This involves checking the product against a predetermined set of requirements and validating that the product meets those requirements.

Tester: **Testing** is a subset of QC. It is the process of executing a system in order to detect bugs in the product so that they get fixed. Testing is an integral part of QC as it helps demonstrate that the product runs the way it is expected and designed for.

**24.Difference between smoke and sanity**

| **Smoke Testing** | **Sanity Testing** |
| --- | --- |
| [Smoke testing](https://www.geeksforgeeks.org/smoke-testing-software-testing/) is done to assure that the acute functionalities of program is working fine. | [Sanity testing](https://www.geeksforgeeks.org/sanity-testing-software-testing/) is done to check the bugs have been fixed after the build. |
| Smoke testing is also called subset of acceptance testing. | Sanity testing is also called subset of regression testing. |
| Smoke testing is documented. | Sanity testing isn’t documented. |
| Smoke testing is performed by either developers or testers. | Sanity testing is normally performed by testers. |
| Smoke testing may be stable or unstable. | Sanity testing is stable. |
| Smoke testing is scripted. | Sanity testing is usually not scripted. |
| Smoke testing is done to measure the stability of the system/product by performing testing. | Sanity testing is done to measure the rationality of the system/product by performing testing. |
| Smoke testing is used to test all over function of the system/product. | Sanity testing is used in the case of only modified or defect functions of system/products. |
| Smoke testing can be performed either manually or by using automation tools. | Sanity testing is commonly executed manually, not by using any automation approach. |
| Smoke testing is performed when new product is built. | Sanity testing is conducted after the completion of regression testing. |
| It includes all the system’s essential basic functionality. | It includes only those modules where change in code is made. |
| Smoke Testing firstly performs on the initial build. Smoke testing is done first. | Sanity Testing is done on stable builds or for the introduced new features in the software. |
| Smoke testing can be carried out either way-manually or automatically. | Without using test cases or scripts sanity testing can be carried out. |
| There is end-to-end system verification done in smoke testing. | A specific component gets verified in sanity testing. |
| In the smoke testing process, the software build could be stable or unstable. | During sanity testing, the software build is comparatively stable. |
| For every new build release smoke testing is carried out. | Sanity testing is carried out when in-depth testing is not possible because of short time. |

**25.Difference between verification and validation model**

| **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. |
| Verification refers to the set of activities that ensure software correctly implements the specific function. | Validation refers to the set of activities that ensure that the software that has been built is traceable to customer requirements. |
| After a valid and complete specification, the verification starts. | Validation begins as soon as project starts. |
| Verification is for prevention of errors. | Validation is for detection of errors. |
| Verification is also termed as white box testing or static testing as work product goes through reviews. | Validation can be termed as black box testing or dynamic testing as work product is executed. |
| Verification finds about 50 to 60% of the defects. | Validation finds about 20 to 30% of the defects. |
| Verification is based on the opinion of reviewer and may change from person to person. | Validation is based on the fact and is often stable. |
| Verification is about process, standard and guideline. | Validation is about the product. |

**26.Explain the types of performance testing?**

* Load testing

Load testing is a [type of testing](https://artoftesting.com/types-of-testing) which involves evaluating the performance of the system under the expected workload. A typical load test includes determining the response time, throughput, error rate, etc during the course of the load test.

**Example**– For a newly developed application with an anticipated load of around 1000 concurrent users. We will create a load test script and configure it with 1000 virtual users and run it for say 1-hour duration. After the load test completion, we can analyse the test result to determine how the application will behave at the expected peak load.

* Stress testing

Stress testing is a type of performance testing where we evaluate the application’s performance at a load much higher than the expected load. Another aspect of the stress testing is to determine the break-point of the application.  
  
**Example** – For an application with an anticipated load of 1000 users we will run the test with 1200 users and check if the application is robust enough to not crash.

* Endurance testing

Endurance testing is also known as ‘Soak Testing’. It is done to determine if the system can sustain the continuous expected load for a long duration. Issues like memory leakage are found with endurance testing.  
  
**Example**– For an application like Income tax filing, the application is used continuously for a very long duration by different users. In this type of application, memory management is very critical. For an application like these, we can run the test for 24 hours to 2 days duration and monitor the memory utilization during the whole test execution

* Spike testing

In spike testing, we analyse the behaviour of the system on suddenly increasing the number of users. It also involves checking if the application is able to recover after the sudden burst of users.  
  
**Example** – For an e-commerce application running an advertisement campaign, the number of users can increase suddenly in a very short duration. Spike testing is done to analyse these types of scenarios.

* Volume testing

The volume testing is performed by feeding the application with a high volume of data. The application can be tested with a large amount of data inserted in the database or by providing a large file to the application for processing. Using volume testing, we can identify the bottleneck in the application.  
  
**Example** – For a newly developed e-commerce application, we can perform volume testing by inserting millions of rows in the database and then carry out the performance test execution.

**27.What is error, defect, bug and failure?**

* **Error** − Since an error is a mistake in the code, we are unable to execute or compile the code. Errors are brought up by the developers and automated test engineers.
* **Defect** − The discrepancy between actual results and anticipated outputs is known as the defect. Defect are founded by the testers and it is founded in early stage of development.
* **Bug** − It is a colloquial term used to describe a defect. Bug are reported by the test engineers.
* **Failure** − If the software has several flaws, it will fail or will be the cause of failure. During the development cycle, the manual test engineers discover the failure.

**28.Difference between priority and severity**

| **Priority** | **Severity** |
| --- | --- |
| The sequence in which the developer should resolve defects is specified by Defect Priority. | The defect severity of a fault is defined as the influence it has on the product's operation. |
| Priority is divided into three categories.   * Low * Medium * High | There are five levels of severity.   * Critical * Major * Moderate * Minor * Cosmetic |
| Priority has to do with scheduling. | The term "severity" refers to the degree to which something is functional or adheres to a set of standards. |
| The priority of a bug determines how quickly it should be repaired. | The severity of a problem on a product's functionality is indicated by its severity. |
| In consultation with the manager/client, the priority of faults is determined. | The defect's severity level is determined by the QA engineer. |
| When a problem has a high priority and low severity, it means it has to be corrected right away but isn't affecting the application. | When a fault has a high severity and a low priority, it means it has to be corrected, but not right now. |
| The priority status is determined by the needs of the consumer. | The product's technical aspect determines the severity level. |

**29.What is bug life cycle?**

A bug life cycle is the movement of a bug or defect in different stages of its lifetime, right from the beginning when it is first identified until the time is marked as verified and closed.

A bug is an error or mistake made in an application that makes a system behave differently than desired requirements. Finding a defect or bug in the application code by the developers is the responsibility of a testing team.   
  
The testing team makes sure that all the bugs are found before the application goes LIVE or is available to end-users.

**30.Explain the difference between functional and nonfunctional testing**

|  |  |
| --- | --- |
| **Functional testing** | **Non-functional testing** |
| It focuses on testing the functionality of the software or system. | It focuses on testing the non-functional aspects of the software or system. |
| Verifies whether the software meets the functional requirements. | Verifies whether the software meets the non-functional requirements such as performance, security, usability, reliability, and compatibility. |
| It involves testing the features and functionalities of the software, such as input/output, error handling, and user interface. | It involves testing the quality attributes of the software, such as response time, scalability, availability, and maintainability. |
| Tests are typically conducted using test cases or scenarios that validate the functional requirements. | Tests are conducted using various techniques such as load testing, stress testing, security testing, and usability testing. |
| It can be performed manually or using automated testing tools. | Often requires specialized testing tools and frameworks to measure and evaluate the non-functional requirements. |
| Done after unit testing and integration testing and before system testing. | It can be done at various stages of the development lifecycle, from design to deployment and maintenance. |

**31. To create HLR and Testcase of**

A) Instagram only first page

1. HLR

<https://github.com/PRATIBHABEDWAL/Tops_ST_assignment/blob/main/HRL%20of%20instagram%201.xlsx>

1. TESTCASE

<https://github.com/PRATIBHABEDWAL/Tops_ST_assignment/blob/main/Testcase%20of%20insta%201st%20page%20%201.xlsx>

B) Facebook login page: <https://www.facebook.com/>

1. HLR

<https://github.com/PRATIBHABEDWAL/Tops_ST_assignment/blob/main/HRL%20of%20facebook%20page%202.xlsx>

2.TESTCASE

<https://github.com/PRATIBHABEDWAL/Tops_ST_assignment/blob/main/Testcase%20facebook%201st%20page%202.xlsx>

**32.What is difference between SDLC (software development life cycle) AND STLC (software testing 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 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, development team makes the plans and designs based on the requirements. | In STLC, testing team (Test Lead or Test Architect) makes the plans and designs. |
| Goal of SDLC is to complete successful development of software. | Goal of STLC is to complete 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 SDLC phases. |
| Post deployment support, enhancement, and update are to be included if necessary. | Regression tests are run by QA team to check deployed maintenance code and maintains test cases and automated scripts. |
| Creation of reusable software systems is the end result of SDLC. | A tested software system is the end result of STLC. |

**33. What is the difference between test scenarios, test case and test script?**

Test scenario: Test scenario is the least detailed and high-level type of documentation. A test scenario is a description of an objective a user might face when using the program. They cover an end-to-end functionality which is to be tested. Test scenarios are derived from test artefacts such as Software Requirement Specification (SRS) and Business Requirement Specification (BRS) documents. Test scenarios flows are as per the functional requirements. They also serve as a quick tool to determine the most critical end-to-end transactions or the real use of the application.

[Test case](https://testworthy.us/Blogs/difference-between-test-scripts-cases-and-scenarios) : A [test case](https://testworthy.us/Blogs/difference-between-test-scripts-cases-and-scenarios) is a documented set of preconditions (prerequisites), procedures (inputs/actions) and postconditions (expected results) which a tester uses to determine whether a system under test satisfies requirements or works correctly.. Test cases are powerful artefacts that are also beneficial for future teammates, as well as a good source of knowing how a system and particular feature works.

Test script: A test script is the most detailed way to document software testing. It typically has 'steps' in the form of code that should be performed manually. These scripts also include expected results for each step, Test scripts are written in programming languages, such as Java, Python, Ruby, and are short programs used to test discrete parts of the software system. In other words, test scripts are automated sets of steps that have to be executed by the tester.

**34. What is test plan is? What is the information that should be covered?**

A Test Plan is a formal document derived from requirement documents like Software Requirement Specification, Use Case documents, etc. It describes in detail, the scope of testing and the different activities performed in testing.   
  
It is generally prepared by a test manager and approved by the different stakeholders of the application needed to carry out testing on a software product are all described in detail in a test plan. The test plan aids in estimating the amount of work required to verify the application’s quality.

A Test Plan needs to address the following-

* The overall scope of testing
* Risk Analysis
* Test estimate
* Resource Requirement
* Tools used
* Scheduling, review, and analysis of test design activities
* Creation of test cases and test data
* Identification of test monitoring and test control activities

**35. What are the different methodologies in agile development model?**

Agile frameworks are methods of organizing and dealing with software program development initiatives that follow the principles and values of the Agile Manifesto. Agile frameworks intend to supply value to clients faster and extra often, even also allowing groups to conform to converting requirements and remarks.

### ****Types of Agile Frameworks****

* **Models:** Crystal [Agile Software Development Methodology](https://www.geeksforgeeks.org/what-is-agile-framework-and-methodology-in-software-development/) places a strong emphasis on fostering effective communication and collaboration among team members, as well as taking into account the human elements that are crucial for a successful development process. This methodology is particularly beneficial for projects with a high degree of uncertainty, where requirements tend to change frequently.
* **Atern:** This methodology is tailored for projects with moderate to high uncertainty where requirements are prone to change frequently. Its clear-cut roles and responsibilities focus on delivering working software in short time frames. Governance practices set it apart and make it an effective approach for teams and projects.
* **Feature-driven development:**This approach is implemented by utilizing a series of techniques, like creating feature lists, conducting model evaluations, and implementing a design-by-feature method, to meet its goal. This methodology is particularly effective in ensuring that the end product is delivered on time and that it aligns with the requirements of the customer.
* **Scrum:** This methodology serves as a framework for tackling complex projects and ensuring their successful completion. It is led by a Scrum Master, who oversees the process, and a Product Owner, who establishes the priorities. The Development Team, accountable for delivering the software, is another key player.
* **Extreme programming (XP):** It uses specific practices like pair programming, continuous integration, and test-driven development to achieve these goals. Extreme programming is ideal for projects that have high levels of uncertainty and require frequent changes, as it allows for quick adaptation to new requirements and feedback.
* **Lean Development:** It is rooted in the principles of lean manufacturing and aims to streamline the process by identifying and removing unnecessary steps and activities. This is achieved through practices such as continuous improvement, visual management, and value stream mapping, which helps in identifying areas of improvement and implementing changes accordingly.
* **Unified Process:**Unified Process is a methodology that can be tailored to the specific needs of any given project. It combines elements of both waterfall and Agile methodologies, allowing for an iterative and incremental approach to development. This means that the UP is characterized by a series of iterations, each of which results in a working product increment, allowing for continuous improvement and the delivery of value to the customer.

**36.Explain the difference between authorization and authentication in web testing? What are the common problems faced in web testing?**

The key difference between authentication and authorization is that authentication verifies the identity of the user, while authorization grants or restricts access to resources within the application based on the user’s identity and other criteria.

In web testing, understanding the difference between authentication and authorization is crucial for designing and executing effective test cases. It helps testers to:

* Ensure that the authentication process is secure and effective in verifying the identity of the user.
* Verify that authorized users can access the resources they are supposed to, and that unauthorized users are denied access.
* Test different scenarios for authorization, such as testing different user roles or permissions, or testing how the application responds to incorrect or invalid authorization attempts.

Common problems faced in web testing

### 1. Integration

It requires integration testing as its result would define the overall applications’ performance at the user’s end

**2. Interoperability**

In software testing industry, they need to be tested for various functionalities to work on any environment, device and browser. It is called as Responsive web application

### Security

### Data integrity tests need to be performed if there are any chances of data loss. Testers take due time to perform security tests as they carefully deal with the unsecured data transfer.

### Intranet-based applications are bound to pre-defined usage of software and hardware required and they can be tested easily. But, when it comes to Internet-based applications, security measures are defined much more clearly by the [skilled testers](https://www.linkedin.com/pulse/7-traits-great-software-tester-himani-kankaria) and QA team.

### 4. Performance

Application’s Page/Loading Speed matters a lot. And seriously, we make sure our customer’s application loads much faster as delay in a single second can divert the user and we don’t want that, period.

Hardware testing, misunderstanding the required application’s features can surely lead to a breach of application performance. Integration and interoperability testing directly influence performance testing.

### 5. Usability

The testers definitely perform the usability testing of the web applications on the real browsers of the real devices.

### 6. Incoming & Outgoing of the apps

When your application has various entry and exit points, the application testers take deep care to test those points well.

### 7. Cross Browser

Web applications need to be tested on various browsers to ensure the website works perfectly fine in each and every browser. Here, [various browser testing tools can help](http://www.testing-whiz.com/blog/comparing-top-10-cross-browser-testing-tools).

In fact, an application needs to be considered to perform browser testing as similar browsers can be used in various devices with varying screen resolution and software and hardware specifications.

### 8. Lower Internet Speed/Bandwidth

While considering testing Internet-based applications, with low Internet speed, we can imagine that various web components can take more time to load and that can affect other components as well. So, this is quite important to test.

### ****9. Regulatory Compliance and Standards Based****

Web application is developed based on the regulatory Compliance and according to the standards specified globally.

### 10. Firewalls

### The web applications need to be tested across various firewalls. There are chances that certain ports might have been blocked and still the web application is expected to behave well. These scenarios need to be tested as well.

### 11. Web Services Requests

The parameter values of the requests made by the web services need to be considered to test. Earlier it was a headach12. Limited Project Delivery Time

### Project delivery time is a crucial factor while deciding a web application development vendor. So, when the time is limited to develop a web application, testing is not done appropriately.

**37.To create HLR and Testcase of web based (what’s up web, Instagram)**

A) WhatsApp Web ([https://web.what’sapp.com/](https://web.whatsapp.com/))

1.HLR

[https://github.com/PRATIBHABEDWAL/Tops\_ST\_assignment/blob/main/HLR%20of%20what’sup%20web%203.xlsx](https://github.com/PRATIBHABEDWAL/Tops_ST_assignment/blob/main/HLR%20of%20whatsup%20web%203.xlsx)

2.TESTCASE

[https://github.com/PRATIBHABEDWAL/Tops\_ST\_assignment/blob/main/what’s%20up%20web%20test%20case%203.xlsx](https://github.com/PRATIBHABEDWAL/Tops_ST_assignment/blob/main/whats%20up%20web%20test%20case%203.xlsx)

B) Instagram Web (<https://www.instagram.com/accounts/login/?hl=en>)

1.HLR

<https://github.com/PRATIBHABEDWAL/Tops_ST_assignment/blob/main/HLR%20OF%20web%20instagram%204.xlsx>

2.TESTCASE

<https://github.com/PRATIBHABEDWAL/Tops_ST_assignment/blob/main/WEB%20instagram%20testcase%204.xlsx>

**38.To create HLR and Testcase on this link (**[**https://artoftesting.com/contact-us**](https://artoftesting.com/contact-us))

A) HLR

<https://github.com/PRATIBHABEDWAL/Tops_ST_assignment/blob/main/HRL%20of%20art%20of%20testing%205.xlsx>

B) TESCASE

<https://github.com/PRATIBHABEDWAL/Tops_ST_assignment/blob/main/ART%20OF%20TESTING%20testcase%205.xlsx>

**39. Write the scenarios of only What’s up chat messages?**

1. Check the Chat window that contains the entire chat list.
2. Check the Chat window, which displays the contact numbers unsaved on mobile.
3. Check the Chat window displayed with all contacts with DP or without DP
4. Check the Chat window is displayed on the group chat list.
5. Check the Chat window, which displays the last updated chatting time.
6. Check the Chat window displays the name of all contacts on the chat window.
7. Check to click on one Chat contact, and then a new window should open with history.
8. Check the user can see all delivered and received messages.
9. Check the user can see the read or send time of messages.
10. Check the user can send and receive text messages in the individual chat box.
11. Check the user can send and receive documents in the individual chat box.
12. Check the user can send and receive photos in an individual chat box.
13. Check the user can send and receive videos in an individual chat box.
14. Check the user can send and receive audio in an individual chat box.
15. Verify the user can send and receive emotional icons in the individual chat boxes.
16. Check that the user can send and receive contacts in the individual chat boxes.
17. Check the user can send and receive Location in the individual chat box.
18. Check the user can send and receive GIFs in the individual chat boxes.
19. Check the user can send and receive Stickers in the individual chat boxes.
20. Verify the user can delete text, video, audio, locations, and documents in the individual chat boxes.
21. Check the user can send recorded voice mail in an individual chat box.
22. Check the user can delete the entire chat history in the individual chat box.
23. Check the user can see contact details in the individual chat box.
24. Verify the user can share images, links, and documents from media in the individual chat boxes.
25. Verify the user can search specific chat history using the search option in the individual chat box.
26. Check the user can video call in the individual chat box.
27. Check the user can voice call in the individual chat box.
28. Check the user can mute the individuals in the individual chat boxes.
29. Check the user can change the wallpaper.
30. Check the users have options like Report, Block, Clear Chat, Export Chat, and Add Shortcut.

**40.Write the scenario of pen**

1. Check the brand of the pen and the logo of the brand is clearly visible on the pen
2. Check the material of the pen
3. Check the dimensions of the pen
4. Check the type of the pen-ink pen, ball pen
5. Check the ink colour of the pen
6. Check the ink of the pen is waterproof or not
7. Verify the size of the tip of the pen
8. Verify if the text written by pen is erasable or not
9. Verify that the user is able to write fluently with pen on different types of the paper
10. Check the odour of the pen’s ink on writing over a surface.
11. Verify that the text written by the pen should have consistent ink flow without leaving any blob.
12. Check that the pen’s ink should not leak in case it is tilted upside down.
13. Verify the strength of the pen’s outer body. It should not be easily breakable.
14. Verify if the pen can support multiple refills or not.
15. In the case of an ink pen, verify that the user is able to refill the pen with all the supported ink types.
16. For ink pens, verify that the mechanism to refill the pen is easy to operate

**41.Write the scenarios of pen stand**

1. Verify the dimensions of the pen stand
2. Verify the material quality of the pen stand
3. Verify the weight of the pen stand
4. Verify the height of the pen stand
5. Verify the capacity of the pen stand which it holds the pen, pencils
6. Verify the durability of the pen stand
7. Verify the pen stand is with company logo or not
8. Verify the stand of the pen stand is proper or not
9. Verify if the material of pen stand is wooden then it is waterproof or not
10. Verify the online and offline price of the pen stand
11. Verify the shape of the pen stand
12. Verify the type of the pen stand for example, personalised pen stand
13. Verify the colour of the pen stand
14. Verify the brand of the pen stand
15. Verify the number of compartments of the pen stand
16. Verify the finishing type of pen stand
17. Verify the mounting type of the pen stand
18. Verify the special feature of the pen stand

**42.Write the scenarios of Door**

1. Verify the dimension of the door
2. Verify the door is waterproof or not
3. Verify the material quality of the door
4. Check the types of the locks in the door
5. Check if doors open inwards or outwards
6. Check the door id single or double folded
7. Verify the door is sliding or rotating
8. Check the position, quality and strength of hinges.
9. Check the type of locks in the door.
10. Check the number of locks in the door interior side or exterior side.
11. Verify if the door is having peek-hole or not.
12. Verify if the door is having stopper or not.
13. Verify if the door closes automatically or not – spring mechanism.
14. Verify if the door makes noise when opened or closed.

**43.Write the scenarios of ATM**

1. Verify that the touch of the ATM screen is smooth and operational.
2. Verify the type of ATM machine, if it has a touch screen, both keypad buttons only, or both.
3. Verify that on properly inserting a valid card different banking options appear on the screen.
4. Check that no option to continue and enter credentials is displayed to the user when the card is inserted incorrectly.
5. Verify that the user is presented with the option to choose a language for further operations.
6. Check that the user is asked to enter a pin number before displaying any card/bank account detail.
7. Verify that there is a limited number of attempts up to which the user is allowed to enter the pin code.
8. Verify that the user is presented with different account type options like- saving, current, etc.
9. Verify that the user is allowed to get account details like available balance.
10. Check that the correct amount of money gets withdrawn as entered by the user for cash withdrawal.
11. Verify that the user is only allowed to enter the amount in multiple denominations as per the specifications.
12. Verify that all the labels and controls including text boxes, buttons, images, and links are present on the screen.
13. Check the informative text written displayed on the screen is clearly visible and legible.
14. Verify that the size, colour, and UI of the different objects are as per the specifications.
15. Verify that the application’s UI is responsive i.e. it should adjust to different screen resolutions of ATM machines.
16. Verify that the user is not allowed to exceed the one-day transaction limit amount.
17. Verify that the user is allowed to do only one transaction per pin request.
18. Check that in case the ATM machine runs out of money, a proper message is displayed to the user.

**44.When to used Usability Testing?**

[Usability testing can be used at any time during the lifecycle of a service to ensure that user needs are continuously met](https://www.bing.com/ck/a?!&&p=5a2354c612db8a12JmltdHM9MTcxMTA2NTYwMCZpZ3VpZD0xYWM1OTExZi1jYjI5LTYxYjgtMGMyZS04MmM1Y2EzNDYwMTEmaW5zaWQ9NjAxMg&ptn=3&ver=2&hsh=3&fclid=1ac5911f-cb29-61b8-0c2e-82c5ca346011&u=a1aHR0cHM6Ly93d3cuZGhzLmdvdi9jeC9jeC1sZWFybmluZy91c2FiaWxpdHktdGVzdGluZy93aGF0LWlzLXVzYWJpbGl0eS10ZXN0aW5n&ntb=1). [It is recommended to start testing early and often, and continue testing through every project stage](https://www.bing.com/ck/a?!&&p=f9c9596f6f6f29a4JmltdHM9MTcxMTA2NTYwMCZpZ3VpZD0xYWM1OTExZi1jYjI5LTYxYjgtMGMyZS04MmM1Y2EzNDYwMTEmaW5zaWQ9NjAxNQ&ptn=3&ver=2&hsh=3&fclid=1ac5911f-cb29-61b8-0c2e-82c5ca346011&u=a1aHR0cHM6Ly93d3cudXNlcnRlc3RpbmcuY29tL2Jsb2cvaG93LXRvLWNvbmR1Y3QtdXNhYmlsaXR5LXRlc3Rpbmc&ntb=1). [Usability testing can occur at various stages to test each iteration of a product, from the initial development to its release](https://www.bing.com/ck/a?!&&p=da972c24b485580dJmltdHM9MTcxMTA2NTYwMCZpZ3VpZD0xYWM1OTExZi1jYjI5LTYxYjgtMGMyZS04MmM1Y2EzNDYwMTEmaW5zaWQ9NjAxNw&ptn=3&ver=2&hsh=3&fclid=1ac5911f-cb29-61b8-0c2e-82c5ca346011&u=a1aHR0cHM6Ly93d3cuY291cnNlcmEub3JnL2FydGljbGVzL3VzYWJpbGl0eS10ZXN0aW5n&ntb=1). [It uncovers pain points in the user journey and highlights barriers to good usability](https://www.bing.com/ck/a?!&&p=8c19b80a6bcb5b9fJmltdHM9MTcxMTA2NTYwMCZpZ3VpZD0xYWM1OTExZi1jYjI5LTYxYjgtMGMyZS04MmM1Y2EzNDYwMTEmaW5zaWQ9NjAxOQ&ptn=3&ver=2&hsh=3&fclid=1ac5911f-cb29-61b8-0c2e-82c5ca346011&u=a1aHR0cHM6Ly9jYXJlZXJmb3VuZHJ5LmNvbS9lbi9ibG9nL3V4LWRlc2lnbi91c2FiaWxpdHktdGVzdGluZy1ndWlkZS8&ntb=1). [It is recommended during the initial design phase of SDLC](https://www.bing.com/ck/a?!&&p=863f733434278f0fJmltdHM9MTcxMTA2NTYwMCZpZ3VpZD0xYWM1OTExZi1jYjI5LTYxYjgtMGMyZS04MmM1Y2EzNDYwMTEmaW5zaWQ9NjAyMQ&ptn=3&ver=2&hsh=3&fclid=1ac5911f-cb29-61b8-0c2e-82c5ca346011&u=a1aHR0cHM6Ly93d3cuZ3VydTk5LmNvbS91c2FiaWxpdHktdGVzdGluZy10dXRvcmlhbC5odG1s&ntb=1).

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

Performing GUI Testing might seem daunting at first, but by breaking it down into manageable steps, it becomes a lot simpler. Here is a straightforward, step-by-step guide to conducting an effective GUI Testing:

Step 1: Understand the Requirements

Step 2: Define the Test Cases

Step 3: Set Up the Test Environment

Step 4: Execute the Tests

Step 5: Report and Analyse

Step 6: Make Necessary Changes

Step 7: Review and Improvement

**46.Write scenario of Microwave oven**

1. Verify that the digital display is clearly visible and functions correctly.
2. Verify that the temperature regulator is smooth to operate.
3. Verify that the temperature regulator works correctly.
4. Check the maximum capacity of the oven and test its functioning with that volume of food
5. Check the oven’s functionality with different food at different temperatures.
6. Verify the oven’s functionality with different kinds of container material.
7. Verify that the power cord of the oven is long enough.
8. Verify that the usage instruction or user manuals have clear instructions.
9. Verify that the dimensions of the oven are as per the specification provided.
10. Verify that the oven’s material is optimal for its use as an oven and as per the specification.
11. Verify that the oven heats the food at the desired temperature properly.
12. Verify that the oven heats food at the desired temperature within a specified time duration.
13. Verify the ovens functioning with the maximum attainable temperature.
14. Verify the ovens functioning with minimum attainable temperature.
15. Verify that the oven’s door gets closed properly.
16. Verify that the oven’s door opens smoothly.

**47.Write the scenario of coffee vending machine**

1. Check if the machine can be switched on and off using the power buttons.
2. Check for the indicator lights when the machine is switched on-off.
3. Verify that the functioning of all the buttons work properly when pressed.
4. Verify that each button has an image/text with it, indicating the task it performce
5. Verify that coffee should not leak when not in operation.
6. Verify the amount of coffee served in single-serving is as per specification.
7. Verify that the digital display displays correct information.
8. Verify that the dimension of the coffee machine is as per the specification.
9. Verify that outer body, as well as inner part’s material, is as per the specification.
10. Verify that the machine’s body color as well brand is correctly visible and as per specification.
11. Verify the input mechanism for coffee ingredients-milk, water, coffee beans/powder, etc.
12. Verify that the quantity of hot water, milk, coffee powder per serving is correct.
13. Verify the power/voltage requirements of the machine.
14. Verify that machine should not make too much sound when in operation.
15. Check the amount of time the machine takes to serve a single serving of coff
16. Check the performance of the machine when used continuously until the ingredients run out of the requirements.

**48.Write a scenario of chair**

1. Verify that the dimension of chair is as per the specifications.
2. Verify that the weight of the chair is as per the specifications.
3. Check the height of the chair’s seat from floor.
4. Verify that the chair is stable enough to take an average human load.
5. Check the material used in making the chair-wood, plastic etc.
6. Check if the chair’s leg are level to the floor.
7. Check the usability of the chair as an office chair, normal household chair.
8. Check if there is back support in the chair.
9. Verify if the chair’s material is brittle or not.
10. Check if cushion is provided with chair or not.
11. Check the condition when washed with water or effect of water on chair.

**49.To create scenario (positive and Negative)**

1. Gmail
2. Verify that user can able to login with valid credentials
3. Verify that user can able to use forgot password functionality.
4. Verify that if user can put the invalid credentials shows an error message user can able to use.
5. Verify all colour, font size are visible.
6. Verify that application UI is responsive.
7. Verify that user can able to operate the all the buttons on the Gmail
8. Verify that there is limitation to attempt login.
9. Verify that user can received all read emails in inbox.
10. Verify that user can received all unread emails in inbox.
11. Verify that user can see all unread emails shows in/ highlights in bold.
12. Verify that user can see the attachment icon.
13. Verify that user can navigate to the page easily.
14. Verify that user can able to start the email
15. Verify that user can able to unstart the emails.
16. Verify that user can able see the count of read and unread emails.
17. Verify that recent received emails shows and correct email sender id, name, date and time.
18. Verify that to, subject, send are display on the new message window.
19. Verify that on bottom side of the new message screen display the all options like send, formatting option, attach files, insert link, insert emoji, insert file using drive, insert photos, more, discard draft .
20. Verify that minimize, maximize are available on the window of new message.
21. Verify that only text can accept in subject text box.
22. Verify that user can insert link in the email
23. Verify that user can not able to login with valid email.
24. Verify that user can not able to login with valid password.
25. Verify that user can not able to login with blank field.
26. Verify that if user can put the valid credentials shows an error message.
27. Verify that user can not able to receive the mail without sender name.
28. Verify that when user read the email counting is not decreases.
29. Verify that when user receives the mail it cannot shows in inbox .
30. Verify that user can not able to start the emails.
31. Verify that warning pop up message is displayed when we try to send a email without subject.
32. Verify that when user is entering the invalid email id it cannot receive.
33. Online shopping to buy product (Flipkart)
34. Verify that on the product page, the user can select the desired attribute of the product e.g. size, colour, etc.
35. Verify that the user can add to the cart one or more products.
36. Verify that users can add products to the wish list.
37. Verify that the user can see the previously added products on the cart page, after signing in to the application.
38. Verify that the user can successfully buy more than one product that were added to his/her cart.
39. Verify that the user cannot add more than the available inventory of the product.
40. Verify that the limit to the number of products a user can buy is working correctly.
41. Error message gets displayed, preventing the user from buying more than the limit.
42. Verify that the delivery can be declined during checkout for the places where shipping is not available.
43. Verify that the Cash on Delivery option of payment is working fine.
44. Verify that the different prepaid methods of payments are working fine.
45. Verify that product return functionality works correctly.

**50. Write a scenario of wrist watch**

1. Verify the type of watch – Analog or Digital
2. Verify the material of the watch
3. Verify that the time of the watch is displayed
4. Verify that the in digital watch all functions are properly working or not
5. Check the colour of the dial as per the specifications
6. Check that the shape of the dial
7. Check the dimensions of the watch
8. Verify the watch if it comes guaranty or warranty
9. Verify the brand of the watch
10. Check if the watch is having date and day displayed or not
11. Check the battery requirement of watch
12. Verify the watch is waterproof or not
13. Verify that the company logo of the watch is properly showing or not
14. Verify that various functionality of analogue watch is working or not
15. In the case of an analogue watch, check the correctness time displayed by the second, minute, and hour hand of the watch.
16. In the case of a digital watch, check the digital display for hours, minutes, and seconds is correctly displayed.
17. Verify the material watch strap
18. Verify that clock’s time can be corrected using the application in case of a digital clock.
19. Check if the second hand of the watch makes ticking sound or not.
20. Verify that clock’s time can be corrected using the key in case of an analogue clock
21. Check if the clock is having stopwatch, timers, and alarm functionality or not.
22. In the case of a digital watch, verify the format of the watch 12 hours or 24 hours
23. Verify if the dial has glass covering or plastic, check if the material is breakable or not.
24. Verify if the dial’s glass/plastic is resistant to minor scratches or not.

**51.Write the scenario of lift (Elevator)**

1. Verify the dimensions of the lift.
2. Verify the capacity of the lift in terms of the total weight.
3. Verify the type of door of the lift is as per the specification.
4. Verify the type of metal used in the lift interior and exterior.
5. Verify the buttons in the lift to close and open the door and numbers as per the number of floors.
6. Verify that the lift moves to the particular floor as the button of the floor is clicked.
7. Verify that the lift stops when the up/down buttons on a particular floor are pressed.
8. Verify if there is an emergency button to contact officials in case of any emergency
9. Verify the time duration for which the door remains open by default.
10. Verify if the lift interior is having proper air ventilation.
11. Verify lighting in the lift.
12. Verify that at no point the lift door should open while in motion.
13. Verify the performance of the floor – the time taken to go to a floor.
14. Verify that in case of power failure, the lift doesn’t free-fall and gets halted on the particular floor.
15. Verify lifts working in case the button to open the door is pressed before reaching the destination floor.
16. Verify that in case the multiple floor number button is clicked, the lift should stop on each floor.
17. Verify that in case of capacity limit is reached users are prompted with a warning alert- audio/visual.

**52.Write a scenario of a WhatsApp Group (generate group)**

1. Check if an admin can add others as Admin.
2. Check if the admin can remove it from the group.
3. Check admin can add users to the group.
4. Check admin can restrict users.
5. Check admin can remove others from admin.
6. Check if the admin can add people.
7. Check if the admin can add 250 people to a group.
8. Check the admin user can add people with the invite link.
9. Check the admin can delete people and add them back to the group.
10. Check if the admin user can able to delete people.
11. Check the admin user can delete all people in the group
12. Check if the admin user can ban users.

**53. Write a scenario of what’s up payment?**

1. Verify that the payment option is available or not in the what’s up system
2. Verify that on the what’s up home screen there are three dot menus at the right corner are present or not
3. Verify that the click on payment option is working or not
4. Verify that add payments method are present or not
5. Verify that it shows the terms and conditions policy for payment are showing or not
6. Verify that all buttons are working or not in the what’s app payment system
7. Verify that the list of bank names are showing or not
8. Verify that the verification process through SMS is working or not
9. Verify that the user verify their bank account or not
10. Verify that user is add more than one account in what’s app payment or not
11. Verify that user is add more than one account with one mobile no or not
12. Verify that user is easily use the what’s app payment system or not