Software Testing Assignment

Module–2(Manual Testing)

1. **What is Exploratory Testing?**

   Exploratory Testing is a type of software testing in which the tester is free to select any possible methodology to test the software. It is an unscripted approach to software testing. In exploratory testing, software developers use their learning, knowledge, skills, and abilities to test the software developed by themselves.

2. **What is traceability matrix?**

   Requirement Traceability Matrix (RTM) is a document that maps and traces user requirement with testcases. The main purpose of RTM is to validate that all requirements are checked via test cases such that no functionality is unchecked during software testing.

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

   Boundary value analysis is a methodology for designing test cases that concentrates software testing effort on cases near the limits of valid ranges. Boundary value analysis is a method which refines equivalence partitioning. Boundary value analysis generates test cases that highlight errors better than equivalence partitioning. The trick is to concentrate software testing efforts at the extreme ends of the equivalence classes.

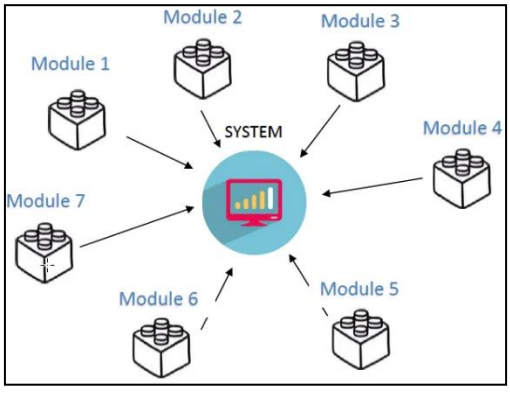
4. **What is Equivalence partitioning testing?**

   Equivalence partitioning is the process of defining the optimum number of tests by:

   - Reviewing documents such as the Functional Design Specification and Detailed Design Specification, and identifying each input condition within a function,
   - Selecting input data that is representative of all other data that would likely invoke the same process for that particular condition.

5. **What is Integration testing?**

   - Integration Testing is a level of the software testing process where individual units are combined and tested as a group.



For eg, Integration testing in an e-commerce website will be implementation by the use of APIs and spans several functions such as authentication, product selection, shopping cart, checkout, payment authorization, and purchase confirmation.

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

   Risk considerations can include:

   - Financial implication of software being released that is un-tested (support costs / possible legal action)
   - Software being delivered late to market
   - Potential loss of Life (safety critical systems)
   - Potential loss of face (may have financial implications as well)

7. **What is Alpha testing?**

   - Alpha Testing is performed and carried out at the developing organizations location with the involvement of developers.
   - Alpha testing aims to test the quality of the functionality and internal mechanics of the product. It enables the development team to immediately resolve any defects and bugs before the product reaches end users.
   - It comes under the category of both White Box Testing and Black Box Testing.

8. **What is beta testing?**

   - Beta Testing (field testing) is performed and carried out by users or you can say people at their own locations and site using customer data.
   - Beta testing is a pre-release type of acceptance test in which target users evaluate a digital asset or product to determine its overall effectiveness, as defined by functionality, usability, reliability and compatibility.

9. **What is component testing?**

   The testing of individual software components is called component testing. It is also called unit testing. The purpose is to validate that each unit of the software performs as designed.

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

    - Testing the attributes of a component or system that is related to functionality is called functional system testing.
    - This testing mainly involves black box testing and it is not concerned about the source code of the application.
    - Each & every functionality of the system is tested by providing appropriate input, verifying the output and comparing the actual results with the expected results.
    - Functional testing verifies that each function of the software application operates in conformance with the requirement specification.

11. What is Non-Functional Testing?

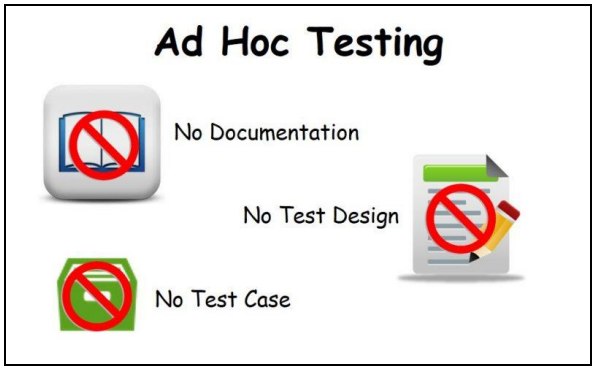
Testing the attributes of a component or system that do not relate to functionality, e.g. reliability, efficiency, usability, interoperability, maintainability and portabilityis called Non-Functional Testing.

12. What is GUI Testing?

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

13. What is Adhoc testing?

- Adhoc testing (Error Guessing) is an informal testing type with an aim to break the system. It does not follow any test design techniques to create test cases.



- Adhoc Testing does not follow any structured way of testing and it is randomly done on any part of application. Main aim of this testing is to find defects by random checking.

14. What is load testing?

- Load testing is a kind of performance testing which determines a system's performance under real-life load conditions. This testing helps determine how the application behaves when multiple users access it simultaneously.
- It is a type of non-functional testing. Load testing is commonly used for the Client/Server, Web based applications – both Intranet and Internet.

15. What is stress Testing?

- Stress testing is used to test the stability & reliability of the system.This test mainly determines the system on its robustness and error handling under extremely heavy load conditions.
- It even tests beyond the normal operating point and evaluates how the system works under those extreme conditions.
- Stress Testing is done to make sure 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?

- Testing based on an analysis of the internal structure of the component or system is called White Box Testing. White box testing is also called glass testing or open box testing.
- White box testing is the detailed investigation of internal logic and structure of the code.
- In order to perform white box testing on an application, the tester needs to possess knowledge of the internal working of the code.

Types of White box testing :

1. Unit Testing
2. Static Analysis
3. Dynamic Analysis
4. Statement Coverage
5. Branch Testing
6. Path Testing
7. Loop Testing

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

- Testing, either functional or non-functional,without reference to the internal structure of the component or system is called Black box testing.
- The testers have no knowledge of how the system or component isstructured inside the box.

Black box testing techniques:

1. Equivalence partitioning
2. Boundary value analysis
3. Decision tables
4. State transition testing
5. Use-case Testing

18. Mention what bigbang testing is?

- Big-bang integration testing is a type of integration testing that combines all the modules or components of a system into a single unit and tests them as a whole.
- 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.

- This approach is typically used when there is a tight deadline for delivering the software product, and all development teams are working in parallel on their respective components.

Benefits of Big Bang Integration Testing

1. **The simplest form of integration testing**: It is the simplest form of integration testing as all the modules are integrated at once and tested as a whole.
2. **Easy to implement**: It is easy to implement as all the modules are already present and just need to be integrated.
3. **Bugs can be identified at once**: All the errors and bugs can be identified at once as all the modules are tested together.
4. **Suitable for small projects**: This approach is suitable for small projects where all modules can be integrated at once.
5. **Potential problems are fixed early**: All components are tested at once, so potential problems can be identified and fixed early on.
6. **Save resources**: It can save resources by avoiding the need to test individual components separately.
7. **Uncover hidden dependencies**: It can uncover hidden dependencies between components that might not be apparent from testing them individually.
8. **Simplify the testing process**: It can simplify the testing process by avoiding the need to set up and configure testing environments for each component.

Limitations of Big Bang Integration Testing

1. **Can cause Delays**: All modules must be completed and ready for testing before integration can begin. This can cause delays if any module is not completed on time. So, it can cause disruptions to normal workflows.
2. **Difficult to identify the root cause of errors**: It can be difficult to identify the cause of errors when all modules are tested together. It can be difficult to identify the root cause of errors if they are discovered during the final test.
3. **Lower-quality software**: This approach can lead to lower-quality software due to the lack of testing of individual components.
4. **Time-consuming**: Big bang integration testing can be time-consuming because all the modules are integrated at once and tested together. This can lead to a lot of time being spent on debugging and fixing errors.
5. **Inefficient**: It can be inefficient because it does not allow for incremental testing. This means that errors can go undetected until all the modules are integrated and tested together.
6. **Not scalable**: It is not scalable because it requires all the modules to be integrated and tested together. This can be a problem for large projects with many modules.

7. **High risk**: It is a high-risk approach because all the modules are integrated and tested together. This can lead to a lot of errors and failures.

19. What is the purpose of exit criteria?

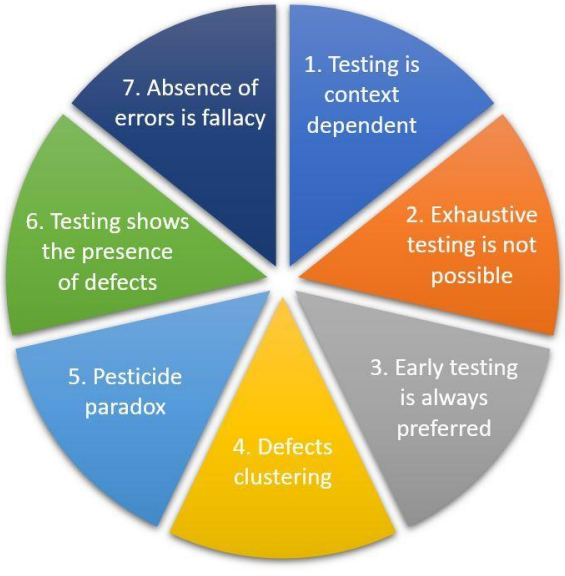
Exit criterion is used to determine whether a given test activity has been completed or not. Exit criteria can be defined for all of the test activities right from planning, specification and execution.

20. When should "Regression Testing" be performed?

Regression testing should be carried out:

- when the system is stable and the system or the environment changes
- when testing bug-fix releases as part of the maintenance phase

21. What is 7 key principles? Explain in detail?



7 key principles of software testing

1. Testing shows presence of defects
2. Exhaustive testing is not possible
3. Early testing
4. Defect clustering
5. Pesticide paradox
6. Testing is context dependent
7. Absence of errors fallacy

1. Testing shows presence of defects

Testing can show that defects are present, but cannot prove that there are no defects. Testing reduces the probability of undiscovered defects remaining in the software but, even if no defects are found, it is not a proof of correctness**.**

2. Exhaustive testing is not possible

Testing everything including all combinations of inputs and preconditions is not possible. So, instead of doing the exhaustive testing we can use risks and priorities to focus testing efforts. This is very unlikely that the project timescales would allow for this number of tests. So, accessing and managing risk is one of the most important activities and reason for testing in any project.

3. Early testing

Testing activities should start as early as possible in the software or system development life cycle, and should be focused on defined objectives. Testing activities should start as early as possible in the development life cycle. These activities should be focused on defined objectives – outlined in the Test Strategy.

4. Defect clustering

A small number of modules contain most of the defects discovered during pre-release testing, or are responsible for the most operational failures. Defects are not evenly spread in a system as they are 'clustered'. In other words, most defects found during testing are usually confined to a small number of modules. Similarly, most operational failures of a system are usually confined to a small number of modules.

5. Pesticide paradox

If the same tests are repeated over and over again, eventually the same set of test cases will no longer find any new defects. To overcome this "pesticide paradox", the test cases need to be regularly reviewed and revised, and new and different tests need to be written to exercise different parts of the software or system to potentially find more defects. Testing identifies bugs, and programmers respond to fix them. As bugs are eliminated by the programmers, the software improves. As software improves the effectiveness of previous tests erodes.

6. Testing is context dependent

Testing is context dependent which basically means that the way we test an e-commerce site will be different from the way we test a gaming site. All the developed software's are not identical. So we might use a different approach, methodologies, techniques, and types of testing depending upon the application type.

7. Absence of errors fallacy

It is possible that software which is 99% bug-free is still unusable. This can be the case if the system is tested thoroughly for the wrong requirement. Software testing is not mere finding defects, but also to check that software addresses the business needs. The absence of Error is a Fallacy i.e. Finding and fixing defects does not help if the system build is unusable and does not fulfill the user's needs & requirements.

22. Difference between QA v/s QC v/s Tester

QA	QC	Tester
Subset of SDLC	Subset of QA	Subset of QC
Process Oriented	Product Oriented	Product Oriented
Prevent defects	Find and fix defects	Find and fix defects
Ensure that processes and procedures are in place to achieve quality	Activities to ensure product quality	Validate the product against specifications

Proactive process	Reactive process	Reactive process
Whole project team involved	Testing team involved	Testing team involved

23. Difference between Smoke and Sanity?

Smoke Testing	Sanity Testing
Smoke Testing is performed to ascertainthat the critical functionalities of theprogram is working fine.	Sanity Testing is done to check thenew functionality / bugs have been fixed.
The objective of this testing is to verify "stability" of the system in order to rigorous testing	The objective of the testing is to verify the "rationality" of the system in order proceed with more rigorous testing
This testing is performed by the developers or testers	Sanity testing is usually performed by testers
Smoke testing is usually documented or scripted	Sanity testing is usually not documented and is unscripted
Smoke testing is a subset of Regression testing	Sanity testing is a subset of Acceptance testing
Smoke testing exercises the entire system from start to end	Sanity testing exercises only particular component of the entire system
Smoke testing is like General Health Check Up	Sanity Testing is like specialized health check up

24. Difference between verification and Validation

Verification	Validation
Verifies that outputs are according to inputs or not	Validates that the users accept the software or not
How the output is achieved?	What the output is achieved?
It is human based checking of documents and files	It is computer based execution of program
It is a static practice of checking documents, design code and program	It is a dynamic practice of validating and testing the actual product
It doesn't involve code execution	It involves code execution
Activities include reviews, inspection, meeting	Activities include types of testing like Unit, Smoke, Regression, System

25. Explain types of Performance testing.

1. **Load testing**
    - Load testing is a type of performance testing that evaluates an application's or system's performance under typical and expected user load.
    - The primary goal of load testing is to understand how the system processes user traffic and transactions, ensuring it remains stable and accessible under these conditions.
2. **Stress testing**
    - Stress testing is a critical performance testing methodology that focuses on determining how an application or system performs under extreme load conditions, often exceeding its maximum capacity.
    - Stress testing aims to understand how the system recovers from these intense stress situations, often called its resilience or robustness.
3. **Endurance testing**
    - The methodology of endurance testing involves simulating a load over a prolonged period, which can range from several hours to days or even weeks, to observe how the system performs under sustained use.
    - It requires creating realistic user scenarios and maintaining a consistent application load or system load.
4. **Spike testing**
    - Spike testing is a specialized form of performance testing that examines the resilience and adaptability of an application or system under sudden, extreme increases in load, referred to as 'spikes.'
    - The primary objective of spike testing is to ascertain whether the application or system can efficiently manage unexpected surges in load without experiencing performance degradation or failure.
5. **Volume testing**
    - Volume testing is a specialized form of performance testing designed to evaluate an application's or system's performance under a substantial volume of data or transactions.
    - The primary purpose of volume testing is to establish the maximum amount of data or transactions the system can handle while maintaining optimal performance levels.
6. **Scalability testing**
    - A scalability test is a type of load testing that measures the application's ability to scale up or down as a reaction to an increase in the number of users.

- In other words, it tests how the system is going to perform during a sudden spike or fall of user request loads.

26. What is Error, Defect, Bug and failure?

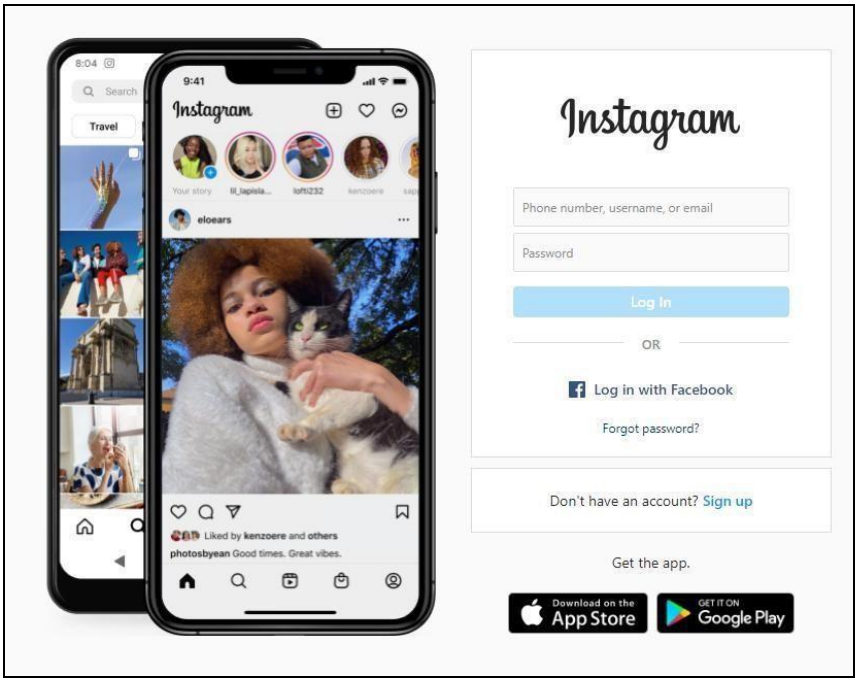
Any mistake in code is called **Error.** An error found by tester is called **Defect.** Defect accepted by developer is called **Bug.** If the build doesn't meet the requirements is called **Failure.**

27. Explain the difference between Functional testing and NonFunctional testing

Functional testing	NonFunctional testing
It is executed to analyze the functionality of components of an application as per the clients requirements.	It is executed to check the performance, reliability, scalability and other non functional aspects of an application.
It is executed at the early stages of development.	It is generally performed after functional testing.
Can be performed both manually and with automation tools.	Requires automation tools for effective testing.
Focuses on user requirements.	Focuses on user expectations.
Determines what the product is capable of	Determines how effectively the product works
Business requirements are the inputs of functional testing	Parameters like speed, scalability are the inputs of non functional testing
Examples of functional testing: Unit testing, Whitebox testing, Smoke testing, Sanity testing, Usability testing and Regression testing	Examples of nonfunctional testing: Performance testing, Load testing, Stress testing, Security testing, Installation testing and Crossbrowser testing

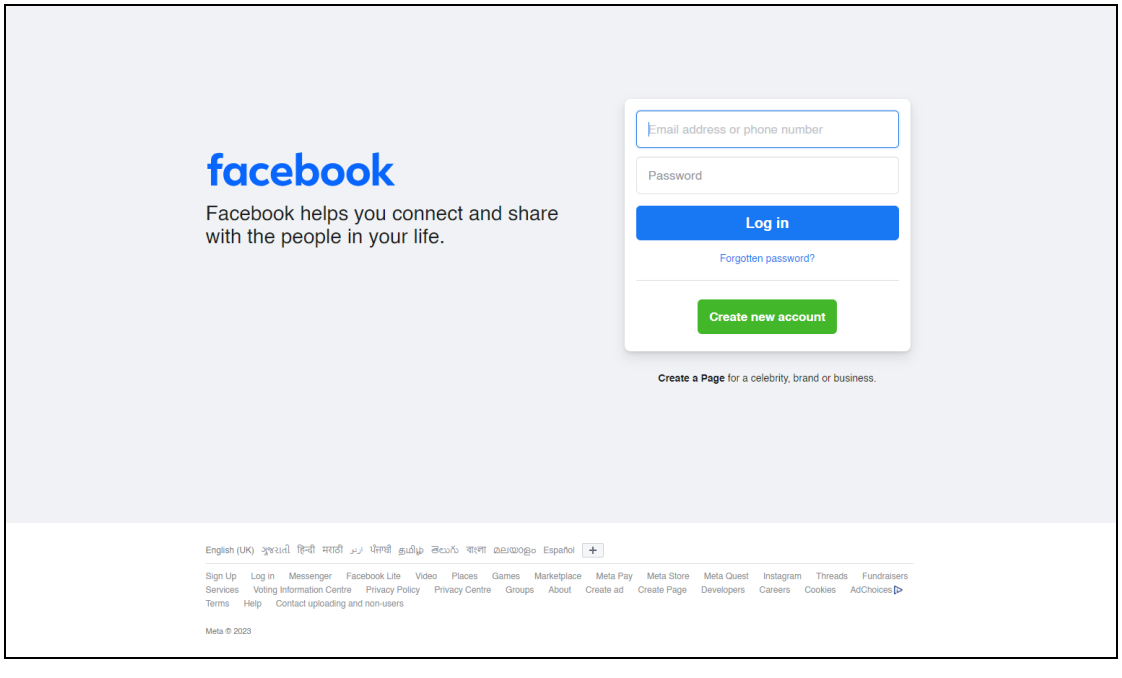
28. To create HLR &TestCase of

1)(Instagram , Facebook) only first page



HLR and Test case of Instagram_First page	Click Here

2) Facebook Login Page :https://www.facebook.com/



HLR and Test case of Facebook login page	Click Here

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

STLC	SDLC
It is a testing life cycle.	Its is a development life cycle.
Fewer people are involved.	More people are involved in all processes(number of developers)
STLC ensures that anything we produce meets customer needs and that the products are of high quality.	SDLC ensures that we are building the correct thing in the correct manner.
STLC is concerned with both the development and testing processes but it is primarly concerned with the testing process.	SDLC assures that we deliver high quality software which is as per client needs.
QA team analysis all the requirement from the requirement document and create a system test plan.	Business analysts gather all project related requirements from a stockholder and create a development plan.
Test architect or test manager creates a strategy to test a software application.	Development team create the high level and low level design of project based on clients requirement.

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

A **test scenario** is any functionality that a software testing company can examine. It is also called a Test Condition or Test Possibility. A **test case** is a document that lists the steps a tester needs to execute. A **test script** is a short program written in a programming language.

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

A document describing the scope, approach, resources and schedule of intended test activities is called Test Plan.The Test Plan (sometimes also referred to as a QA Test Plan) can be seen as the instruction manual or guide for testing effort. It describes the objectives of testing (what are you planning to verify and/or validate), the scope of testing (what will and will not be tested), together with the general and sometimes detailed schedule of the activities you want to perform (how and when are you testing).

Test plans should list the risks foreseen in the project and their respective levels so that testing can be prioritized by risk.

Perhaps the most important part of a test plan is the definition of resources needed. Resources can be seen as human (such as the people involved in the test) and technical (such as test environments, test tools and test data).

Components of a Test Plan

1. **Scope**: Details the objectives of the particular project. Also, it details user scenarios to be used in tests. The scope can specify scenarios or issues the project will not cover if necessary.
2. **Schedule**: Details start dates and deadlines for testers to deliver results.
3. **Resource Allocation**: Details which tester will work on which test.

4. **Environment**: Details the test environment ‘s nature, configuration, and availability.
5. **Tools**: Details what tools will be used for testing, bug reporting, and other relevant activities.
6. **Defect Management**: Details how bugs will be reported, to whom, and what each bug report needs to be accompanied by. For example, should bugs be reported with screenshots, text logs, or videos of their occurrence in the code?
7. **Risk Management**: Details what risks may occur during software testing and what risks the software itself may suffer if released without sufficient testing.
8. **Exit Parameters**: Details when testing activities must stop. This part describes the expected results from the QA operations, giving testers a benchmark to compare actual results.

32. What are the different Methodologies in Agile Development Model?

Different Methodologies in Agile Development Model :

1. Extreme Programming (XP)
2. Scrum
3. Kanban

33. Explain the difference between Authorization and Authentication in Web testing.What are the common problems faced in Web testing?

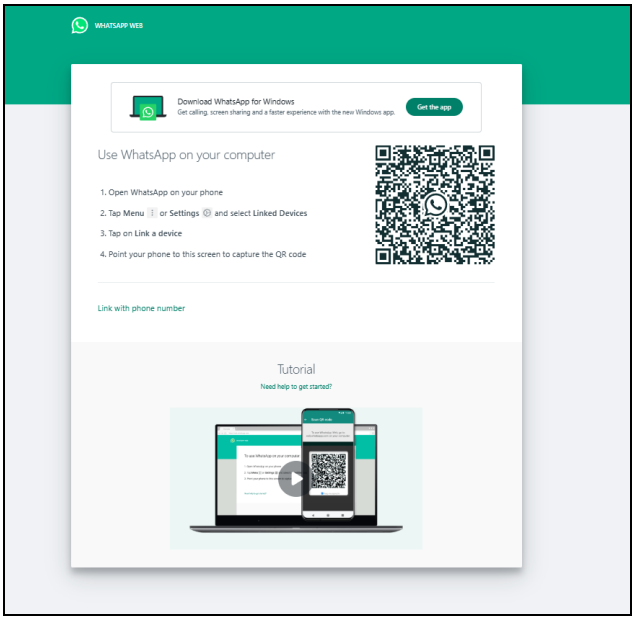
Authentication	Authorization
Determines whether users are who they claim to be	Determines what users can and cannot access
Challenges the user to validate credentials	Verifies whether access is allowed through policies and rules
It usually needs the users login details.	It needs users privilege or security levels.
Generally transmits information through an ID Token	Generally transmits information through an Access token.
Common methods include : username, password, answer to a security question, code sent via sms or mail	Permissions are granted and monitored by the organization.
Its changeable by the user.	It cannot be changed by the user.

Common problems faced in Web testing :

1. Problems that occur while navigating an application
2. Usability issues like broken links, form fields missing default focus, tab key not working, and all keyboardshortcuts not fully functional
3. Cross-browser compatibility
4. Responsiveness
5. Cross-device compatibility
6. Integration testing
7. Security
8. Performance testing
9. Application getting slow
10. Inconsistencies in the page layout across devices
11. Broken UI elements leading to misbehavior

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

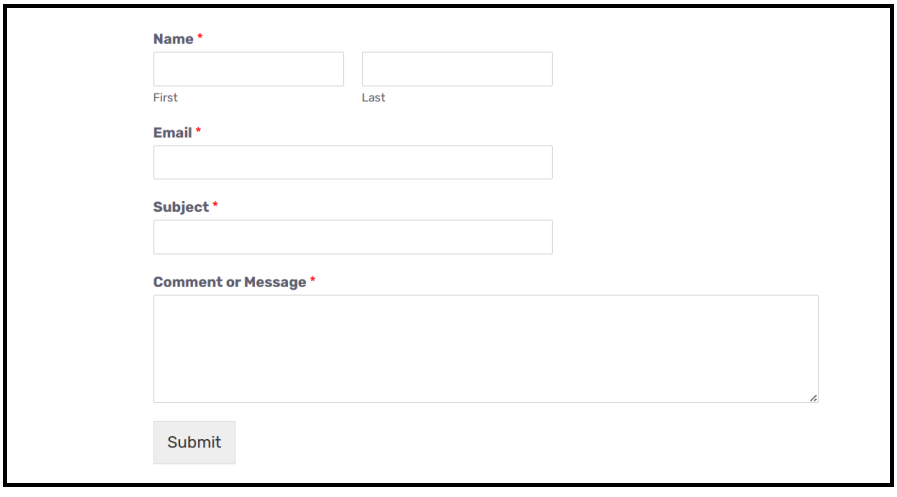
1. WhatsApp Web :https://web.whatsapp.com/



HLR and Test case of whatsapp web	Click Here

HLR and Test case of Instagram_Web Login page	Click Here

35. To create HLR and TestCase on this Link. https://artoftesting.com/



HLR and Test case of Art of Testing _Contact Us page	Click Here

36. Write a scenario of only Whatsapp chat messages

Scenario of Whatsapp chat messages	Click Here

37. Write a Scenario of Pen

Scenario of Pen	Click Here

38. Write a Scenario of Pen Stand

Scenario of Pen Stand	Click Here

39. Write a Scenario of Door

Scenario of Door	Click Here

40. Write a Scenario of ATM

Scenario of ATM	Click Here

41. When to used Usablity Testing?

The three primary goals of usability testing:

1. Discovering problems
2. Comparing against benchmarks
3. Comparing against other interfaces.

42. What is the procedure for GUI Testing?

The following checklist will ensure procedure of GUI Testing in Software Testing.

- Check all the GUI elements for size, position, width, length, and acceptance of characters or numbers. For instance, you must be able to provide inputs to the input fields.
- Check you can execute the intended functionality of the application using the GUI
- Check Error Messages are displayed correctly
- Check for Clear demarcation of different sections on screen
- Check Font used in an application is readable
- Check the alignment of the text is proper
- Check the Color of the font and warning messages is aesthetically pleasing
- Check that the images have good clarity
- Check that the images are properly aligned
- Check the positioning of GUI elements for different screen resolution.

43. Write a scenario of Microwave Oven

Scenario of Microwave Oven	Click Here

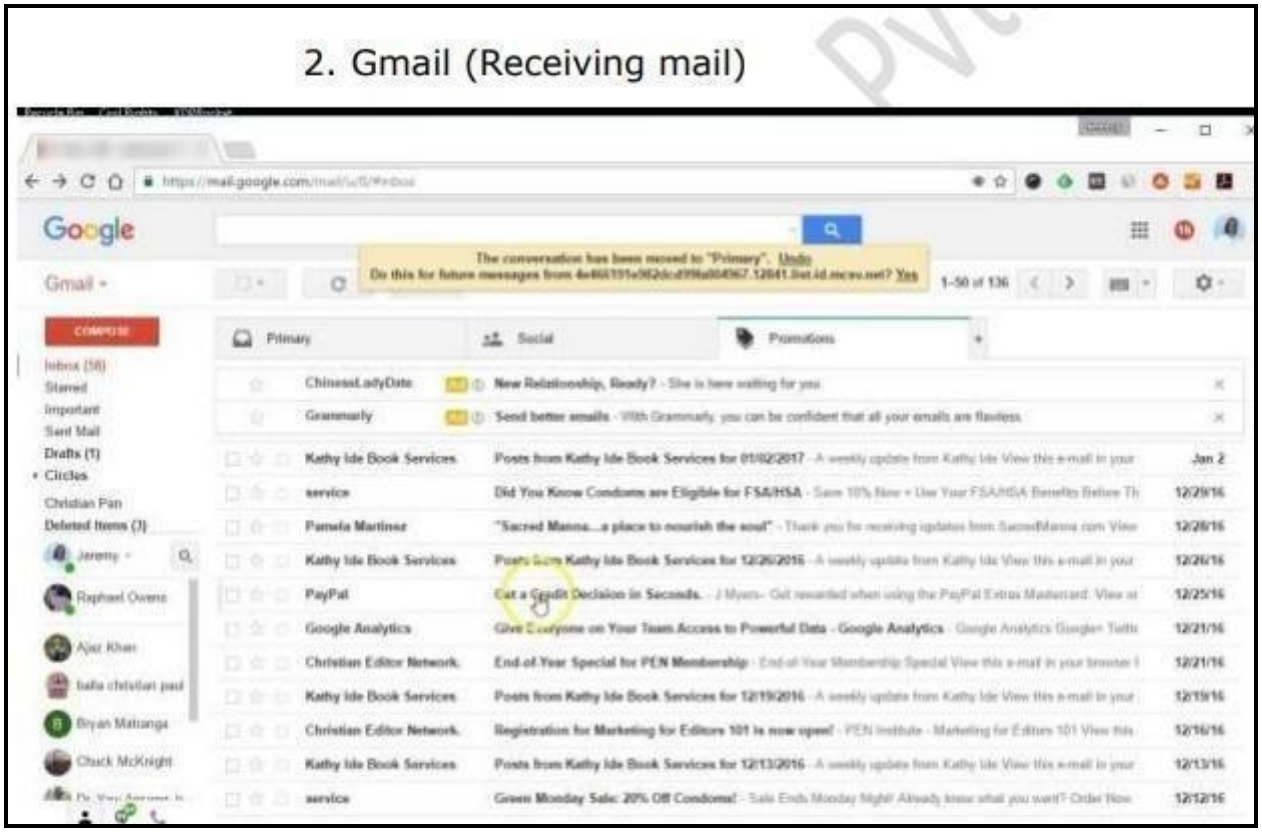
44. Write a scenario of Coffee vending Machine

Scenario of Coffee Vending Machine	Click Here

45. Write a scenario of chair

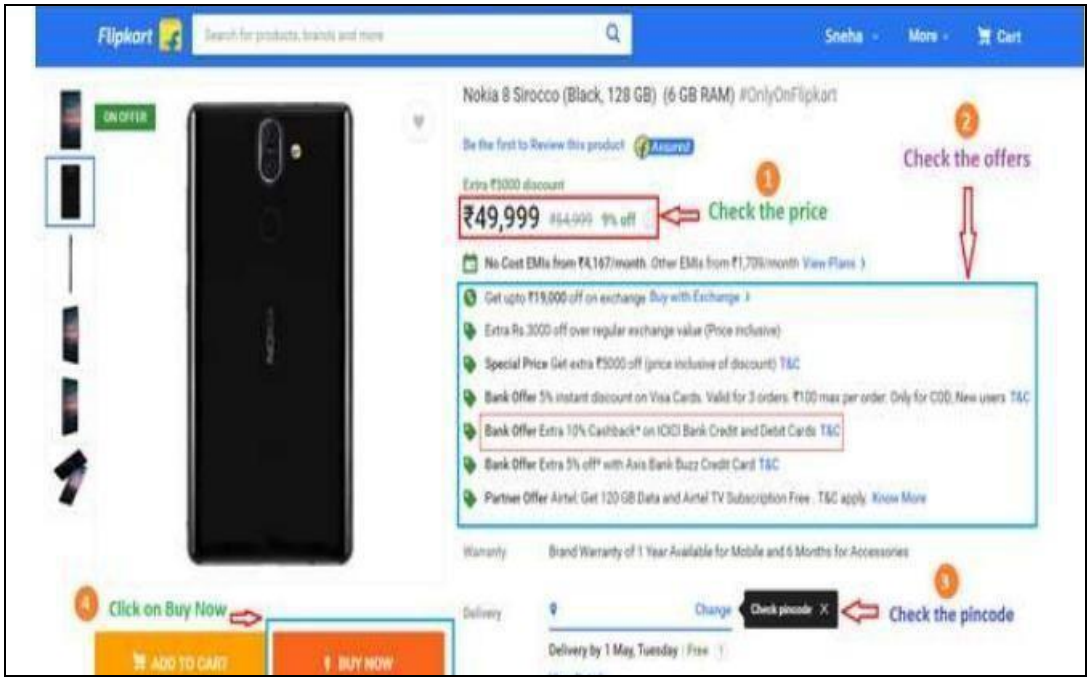
Scenario of Chair	Click Here

46. To Create Scenario (Positive & Negative)



Scenario of Gmail_receiving mail	Click Here

2. Online shopping to buy product (flipkart)



Scenario of Flipkart _buying a product	Click Here

47. Write a Scenario of Wrist Watch

Scenario of Wrist watch	Click Here

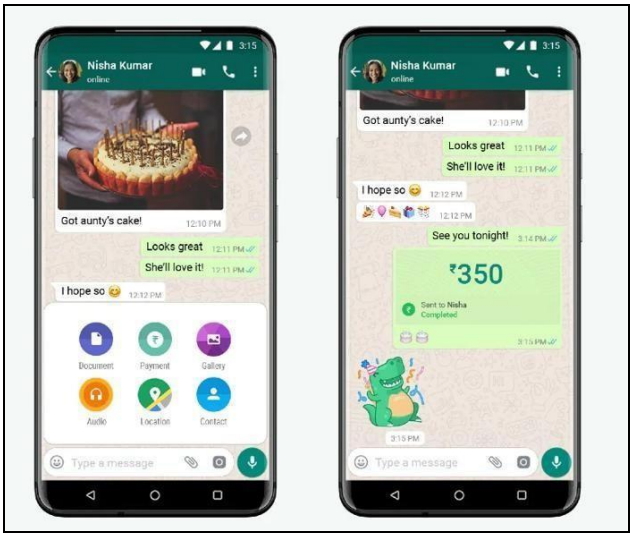
48. Write a Scenario of Lift(Elevator)

Scenario of Lift	Click Here

49. Write a Scenario of whatsapp Group (generate group)

Scenario of whatsapp group	Click Here

50. Write a Scenario of Whatsapp payment



Scenario of whatsapp payment	Click Here