Module -2

1.What is Exploratory Testing?

- Exploratory Testing is a concurrent process where
- Though the current trend in testing is to push for automation, exploratory testing is a new way of thinking. Automation has its limits
- Is not random testing but it is Adhoc testing with purpose of find bugs
- Is structured and rigorous
- Is cognitively (thinking) structured as compared to procedural structure of
- scripted testing. This structure comes from Charter, time boxing etc.
- Is highly teachable and manageable
- Is not a technique but it is an approach. What actions you perform next is
- governed by what you are doing currently

2. What is traceability matrix?

- Test conditions should be able to be linked back to their sources in the test basis, this is known as traceability.
- To protect against changes you should be able to trace back from every system component to the original requirement that caused its presence.
- A software process should help you keeping the virtual table up-to-date.Simple technique may be quite valuable.

Types:
1. Forward Traceability – Mapping of Requirements to Test cases
2.Backward Traceability – Mapping of Test Cases to Requirements
3.Bi-Directional Traceability - A Good Traceability matrix is the
4.References from test cases to basis documentation and vice versa.

Pros:
- Easy to identify the missing functionalities.
- Make obvious to the client that the software is being developed as per the requirements.
- To make sure that all requirements included in the test cases.
- If there is a change request for a requirement, then we can easily find out which test cases need to update.

Cons:
- Poor or unknown test coverage, more defects found in production
- It will lead to miss some bugs in earlier test cycles which may arise in

- later test cycles. Then a lot of discussions arguments with other teams and managers before release.
- Difficult project planning and tracking, misunderstandings between different teams over project dependencies, delays, etc.

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.
- At those points when input values change from valid to invalid errors are most likely to occur.
- Boundary Value Analysis (BVA) uses the same analysis of partitions as EP and is usually used in conjunction with EP in test case design.

4.What is Equivalence partitioning testing?

- EP can be used for all Levels of 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 - Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems.
- Integration Testing is a level of the software testing process where individual units are combined and tested as a group.Integration testing tests integration or interfaces between components, interactions to different parts of the system such as an operating system, file system and hardware or interfaces between systems.
- Integration testing is done by a specific integration tester or test team.
- Components may be code modules, operating systems, hardware and even complete systems.

- There are 2 levels of Integration Testing

1. Component Integration Testing
2. System Integration Testing
   - Component Integration Testing: Testing performed to expose defects in the interfaces and interaction between integrated components.

   - There is two types methods of Integration Testing:

1.Bing Bang Integration Testing
2. Incremental Integration Testing
i. Top Down Approach
ii. Bottom Up Approach

-When is Integration Testing performed?
- Integration Testing is performed after Unit Testing and before System Testing.
- Who performs Integration Testing?
  Either Developers themselves or independent Testers perform Integration Testing.

6.What determines the level of risk?
- Risks should be evaluated at the Business Level, Technological Level,
- Project Level and Testing Level.
- Risks are also used to decide where to start testing and where more testing is needed.
- 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 implications as well).

7.What is Alpha testing?

- It is always performed by the developers at the software development site.
- Sometimes it is also performed by Independent Testing Team.
- Alpha Testing is not open to the market and public
- It is conducted for the software application and project.
- It is always performed in Virtual Environment.
- It is always performed within the organization.
- It is the form of Acceptance Testing.
- Alpha Testing is definitely performed and carried out at the developing
- Organizations location with the involvement of developers.
- It comes under the category of both White Box Testing and Black Box Testing.

8.What is beta testing?

- It is always performed by the customers at their own site.
- It is not performed by Independent Testing Team.
- Beta Testing is always open to the market and public.
- It is usually conducted for software product.
- It is performed in Real Time Environment.
- It is always performed outside the organization.
- It is also the form of Acceptance 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.
- It is only a kind of Black Box Testing.
- Beta Testing is always performed at the time when software product
- and project are marketed.
- It is always performed at the user's premises in the absence of the
- development team.
- It is also considered as the User Acceptance Testing (UAT) which is
- done at customers or users area.
- Beta testing can be considered "pre-release" testing.
- Pilot Testing is testing to product on real world as well as collect data
- on the use of product in the classroom.

9.What is Component testing?

- Component testing is also known as Unit testing .
- Unit tests designed in the module design phase are executed on the code during this validation phase. Unit testing is the testing at code level and helps eliminate bugs at early stage, though all defects cannot be uncovered by unit testing.

10.What is Functional System Testing?

- Functional Testing: Testing based on an analysis of the specification of the functionality of a component or system.
- 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.

- This testing involves checking of User Interface, APIs, Database,security, client/ server applications and functionality of the Application under Test. The testing can be done either manually or using automation.

1.Web Based Testing :

- Are you able to login to a system after entering correct credentials?
- Does your payment gateway prompt an error message when you enter incorrect card number?
- Does your "add a customer" screen adds a customer to your records successfully?

2. Desktop Based Testing :

- Verifies Installation testing,Check for broken lines,
- Testing with different client accounts,Theme change and Print,
- Check for broken links. Warning messages. Resolution change effect on the application

3. Mobile Based Testing :

- To validate whether the application works as per as requirement whenever the application
- starts/stops
- To validate whether the application goes into minimized mode whenever there is an incoming
- phone call. In order to validate the same we need to use a second phone, to call the device

4. Game Based Testing :

-Takes more time to execute as testers look for game play issues,graphics issues , audio- visual issues, etc.

- Validates whether installation goes smoothly, the app works in minimized mode, the app allows
- social networking options, supports payment gateways, and many more.

10.What is Non-Functional Testing?

- 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 portability
- May be performed at all Test levels (not just Non Functional Systems Testing).
- Measuring the characteristics of the system/software that can be quantified on a varying scale- e.g. performance test scaling
- Non-functional testing includes, but is not limited to, performance testing, load testing, stress testing, usability testing, maintainability testing, reliability testing and portability testing
- It is the testing of "how" the system works. Non-functional testing may be performed at all test levels.

- The term non-functional testing describes the tests required to measure characteristics of systems and software that can be quantified on a varying scale, such as response times for performance testing.
- To address this issue, performance testing is carried out to check & f ine tune system response times. The goal of performance testing is to reduce response time to an acceptable level.
- Hence load testing is carried out to check systems performance at different loads i.e. number of users accessing the system.
- Non Functional Testing Example:
  1.Web Based Testing
  2. Desktop Based Testing
  3. Mobile Based Testing
  4. Game Based Testing

11.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.
- Check all the GUI elements for size, position, width, length and acceptance of characters or numbers. For instance, you must be able to provide inputs to the input fields.
- Check you can execute the intended functionality of the application using the GUI
- Check Error Messages are displayed correctly.
- Check for Clear demarcation of different sections on screen.
- Check Font used in application is readable
- Check the alignment of the text is proper
- Check the Color of the font and warning messages is aesthetically pleasing
- Check that the images have good clarity
- Check that the images are properly aligned
- Check the positioning of GUI elements for different screen resolution.

  Approach of GUI Testing:
  1.Menuale based Testing
  2.Record And Replay
  3. Model base Testing

12.What is Adhoc testing?

- Adhoc testing is an informal testing type with an aim to break the system.
- It does not follow any test design techniques to create test cases.
- In fact is does not create test cases altogether!

- This testing is primarily performed if the knowledge of testers in the system under test is very high.
- 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.
- Adhoc testing can be achieved with the testing technique called Error Guessing
- The Error guessing is a technique where the experienced and good testers are encouraged to think of situations in which the software may not be able to cope.
- Some people seem to be naturally good at testing and others are good testers because they have a lot of experience either as a tester or working with a particular system and so are able to find out its weaknesses.
- This is why an error guessing approach, used after more formal techniques have been applied to some extent, can be very effective.
- Types Of Adhoc Testing :

1.Buddy Testing
2.Pair Testing
3.Monkey Testing

13.What is load testing?

- Its a performance testing to check system behavior under load. Testing an application under heavy loads, such as testing of a web site under a range of loads to determine at what point the system's response time degrades or fails.
- Load testing is a kind of performance testing 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.
- This testing usually identifies-
- The maximum operating capacity of an application
- Determine whether current infrastructure is sufficient to run the application
- Sustainability of application with respect to peak user load
- Number of concurrent users that an application can support, and scalability to allow more users to access it.
- It is a type of non-functional testing. Load testing is commonly used for the Client/Server, Web based applications – both Intranet and Internet.

Why Load Testing :

- Load testing gives confidence in the system & its reliability and performance
- Load Testing helps identify the bottlenecks in the system under heavy user stress scenarios before they happen in a production environment.
- Load testing gives excellent protection against poor performance and accommodates complementary strategies for performance management and monitoring of a production environment.

Pros:
- Performance bottlenecks identification before production
- Improves the scalability of the system
- Minimize risk related to system down time.
- Reduced costs of failure
- Increase customer satisfaction

Cons:
- Need programming knowledge to use load testing tools.
- Tools can be expensive as pricing depends on the number of virtual users supported.

14.What is Stress Testing?

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

- Types of Stress Testing :

1.Application Stress Testing:
2.Transactional Stress Testing
3. Systemic Stress Testing
4. Exploratory Stress Testing

15.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.
- Structure-based testing technique is also known as ‘white-box’ or ‘glass-box’ testing technique because here the testers require knowledge of how the software is implemented, how it works.
- Different test cases may be derived to exercise the loop once, twice, and many times. This may be done regardless of the functionality of the software.
- Structure-based techniques are also used in system and acceptance testing, but the structures are different.

Cons :

- Testing based upon the structure of the code
- Typically undertaken at Component and Component Integration Test phases by development teams.
- White box testing is the detailed investigation of internal logic and structure of the code.
- White box testing is also called glass testing or open box testing. In order to perform white box testing on an application, the tester needs to possess knowledge of the internal working of the code.
- The tester needs to have a look inside the source code and find out which unit/chunk of the code is behaving inappropriately.

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

- Black-box testing: Testing, either functional or non-functional, without reference to the internal structure of the component or system.
- Specification-based testing technique is also known as 'black-box' or input/output driven testing techniques because they view the software as a black-box with inputs and outputs.
- The testers have no knowledge of how the system or component is structured inside the box. In black-box testing the tester is concentrating on what the software does, not how it does it.
- Specification-based techniques are appropriate at all levels of testing (component testing through to acceptance testing) where a specification exists.

Introduction Cons :

- The technique of testing without having any knowledge of the interior workings of the application is Black Box testing.
- What a system does, rather than HOW it does it.
- Typically used at System Test phase, although can be useful throughout the test lifecycle.
- The tester is oblivious to the system architecture and does not have access to the source code.
- Typically, when performing a black box test, a tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon.

Advantages :

- Well suited and efficient for large code segments.
- Code Access not required
- Clearly separates user's perspective from the developer's perspective through visibly defined roles.
- Large numbers of moderately skilled testers can test the application with no knowledge of implementation, programming language or operating systems.

Disadvantage:

- Limited Coverage since only a selected number of test scenarios are actually performed.
- Inefficient testing, due to the fact that the tester only has limited knowledge about an application.
- Blind Coverage, since the tester cannot target specific code segments or error prone areas.
- The test cases are difficult to design.

Techniques of Black Box Testing:

- There are four specification-based or black-box technique:
- Equivalence partitioning
- Boundary value analysis
- Decision tables
- State transition testing
- Use-case Testing
- Other Black Box Testing

17.Mention what are the categories of defects?

- Usability
- Reliability
- Performance
- Security

18.Mention what bigbang testing is?

- In Big Bang integration testing all components or modules is integrated simultaneously, after which everything is tested as a whole.
- Big Bang testing has the advantage that everything is finished before integration testing starts.
- The major disadvantage is that in general it is time consuming and difficult to trace the cause of failures because of this late integration.
- Here all component are integrated together at once, and then tested.

Advantege :

- Convenient for small systems

Disadvantage :

- Fault Localization is difficult.
- Given the sheer number of interfaces that need to be tested in this approach, some interfaces links to be tested could be missed easily.
- Since the integration testing can commence only after "all" the modules are designed, testing team will have less time for execution in the testing phase.
- Since all modules are tested at once, high risk critical modules are not isolated and tested on priority. Peripheral modules which deal with user interfaces are also not isolated and tested on priority.

19.What is the purpose of exit criteria?

- How do we know when to stop testing
- Run out of time?
- Run out of budget?
- The business tells you it went live last night
- Boss says stop
- All defects have been fixed?
- When out exit criteria have been met?
- Purpose of exit criteria is to define when we STOP testing either at the
- End of all testing – i.e. product Go Live
- End of phase of testing (e.g. hand over from System Test to UAT)

20.When should "Regression Testing" be performed?

- Testing of a previously tested program following modification to ensure that defects have not been introduced or uncovered in unchanged areas of the software, as a result of the changes made. It is performed when the software or its environment is changed.
- If the test is re-run and passes you cannot necessarily say the fault has been resolved because ..
- You also need to ensure that the modifications have not caused unintended side-effects elsewhere and that the modified system still meets its requirements – Regression Testing.
- Need of Regression Testing:
- Change in requirements and code is modified according to the requirement
- New feature is added to the software
- Defect fixing
- Performance issue fix
- Regression Testing Techniques :
- Retest All :
- This is one of the methods for regression testing in which all the tests in the existing test bucket or suite should be re-executed. This is very expensive as it requires huge time and resources.
- Regression Test Selection:
- Instead of re-executing the entire test suite, it is better to select part of test suite to be run
- Test cases selected can be categorized as 1) Reusable Test Cases 2) Obsolete Test Cases.
- Re-usable Test cases can be used in succeeding regression cycles.
- Obsolete Test Cases can't be used in succeeding cycles.
- Prioritization Of Test Cases :

- Prioritize the test cases depending on business impact, critical & frequently used functionalities. Selection of test cases based on priority will greatly reduce the regression test suite.
- Challenges Regression Testing :
- With successive regression runs, test suites become fairly large. Due to time and budget constraints, the entire regression test suite cannot be executed.
- Minimizing test suite while achieving maximum test coverage remains a challenge
- Determination of frequency of Regression Tests, i.e., after every modification or every build update or after a bunch of bug fixes, is a challenge.
- Regression Testing Tools:
- Quick Test Professional (QTP)
- Rational Functional Tester (RFT)
- Selenium

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

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.
- We test to find Faults
- As we find more defects, the probability of undiscovered defects remaining in a system reduces.
- However Testing cannot prove that there are no defects present.

2. Exhaustive Testing is Impossible! :

- Testing everything including all combinations of inputs and preconditions is not possible.
- So, instead of doing the exhaustive testing we can use risks and priorities to focus testing efforts.
- So, accessing and managing risk is one of the most important activities and reason for testing in any project.
- We have learned that we cannot test everything (i.e. all combinations of inputs and pre-conditions)
- That is we must Prioritise our testing effort using a Risk Based Approach.

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.
- Remember from our Definition of Testing, that Testing doesn't start once the code has been written!.

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
- 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
- An important consideration in test prioritisation!

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 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.
- Therefore we must learn, create and use new tests based on new techniques to catch new bugs.
- N.B It's called the "pesticide paradox" after the agricultural phenomenon, where bugs such as the boll weevil build up tolerance to pesticides, leaving you with the choice of ever-more powerful pesticides followed by ever-more powerful bugs or an altogether different approach.' – Beizer 1995.

6.Testing is Context Dependent:

- Testing is basically context dependent.
- Testing is done differently in different contexts.
- Different kinds of sites are tested differently.

For Example :

- Safety – critical software is tested differently from an e-commerce site
- Whilst, Testing can be 50% of development costs, in NASA's Apollo program it was 80% testing
- 3 to 10 failures per thousand lines of code (KLOC) typical for commercial software

- 1 to 3 failures per KLOC typical for industrial software.
- 0.01 failures per KLOC for NASA Shuttle code!
- Also different industries impose different testing standards

7.Absence of Errors Fallacy :
-If the system built is unusable and does not fulfill the user's needs and expectations then finding and fixing defects does not help.
- If we build a system and, in doing so, find and fix defects …
- It doesn't make it a good system
- Even after defects have been resolved it may still be unusable and/or does not fulfil the users' needs and expectations.

23 Difference between Smoke and Sanity?

S.N	Smoke Testing	Sanity Testing
1.	Smoke Testing is performed to ascertain that the critical functionalities of the program is working fine	Sanity Testing is done to check the new functionality / bugs have been fixed
2.	The objective of this testing is to verify "stability" of the system in order to with more rigorous testing	The objective of the testing is to verify the the "rationality" of the system in order proceed to proceed with more rigorous testing
3.	This testing is performed by the developers or testers	Sanity testing is usually performed by testers
4.	Smoke testing is usually documented or scripted is unscripted	Sanity testing is usually not documented and
5.	Smoke testing is a subset of Regression testing	Sanity testing is a subset of Acceptance testing
6.	Smoke testing exercises the entire system from end to end	Sanity testing exercises only the particular component of the entire system
7.	Smoke testing is like General Health Check Up	Sanity Testing is like specialized health check up

24.Difference between verification and Validation ?

Criteara	**Verification**	**Validation**
Definition	The process of evaluating work-products (not the actual f inal 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

25.Explain types of Performance testing ?

1. Load testing
2.Stress testing
3.Endurance testing
4.Spike testing
5.Volume testing
6.Scalability testing

1. **Load Testing :**
    - Its a performance testing to check system behavior under load. Testing an application under heavy loads, such as testing of a web site under a range of loads to determine at what point the system's response time degrades or fails.
    - Load testing is a kind of performance testing 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.
    - This testing usually identifies :
    - The maximum operating capacity of an application.
    - Determine whether current infrastructure is sufficient to run the application
    - Sustainability of application with respect to peak user load.
    - Number of concurrent users that an application can support, and scalability to allow more users to access it.
    - It is a type of non-functional testing. Load testing is commonly used for the Client/Server, Web based applications – both Intranet and Internet.

2.Stress Testing :

- System is stressed beyond its specifications to check how and when it fails. Performed under heavy load like putting large number beyond storage capacity, complex database queries, continuous input to system or database load .
- Stress testing is 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.
- Most prominent use of stress testing is to determine the limit, at which the system or software or hardware breaks.
- During festival time, an online shopping site may witness a spike in traffic, or when it announces a sale.
- When a blog is mentioned in a leading newspaper, it experiences a sudden surge in traffic.

3.Endurance testing

- 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.

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

Error :

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

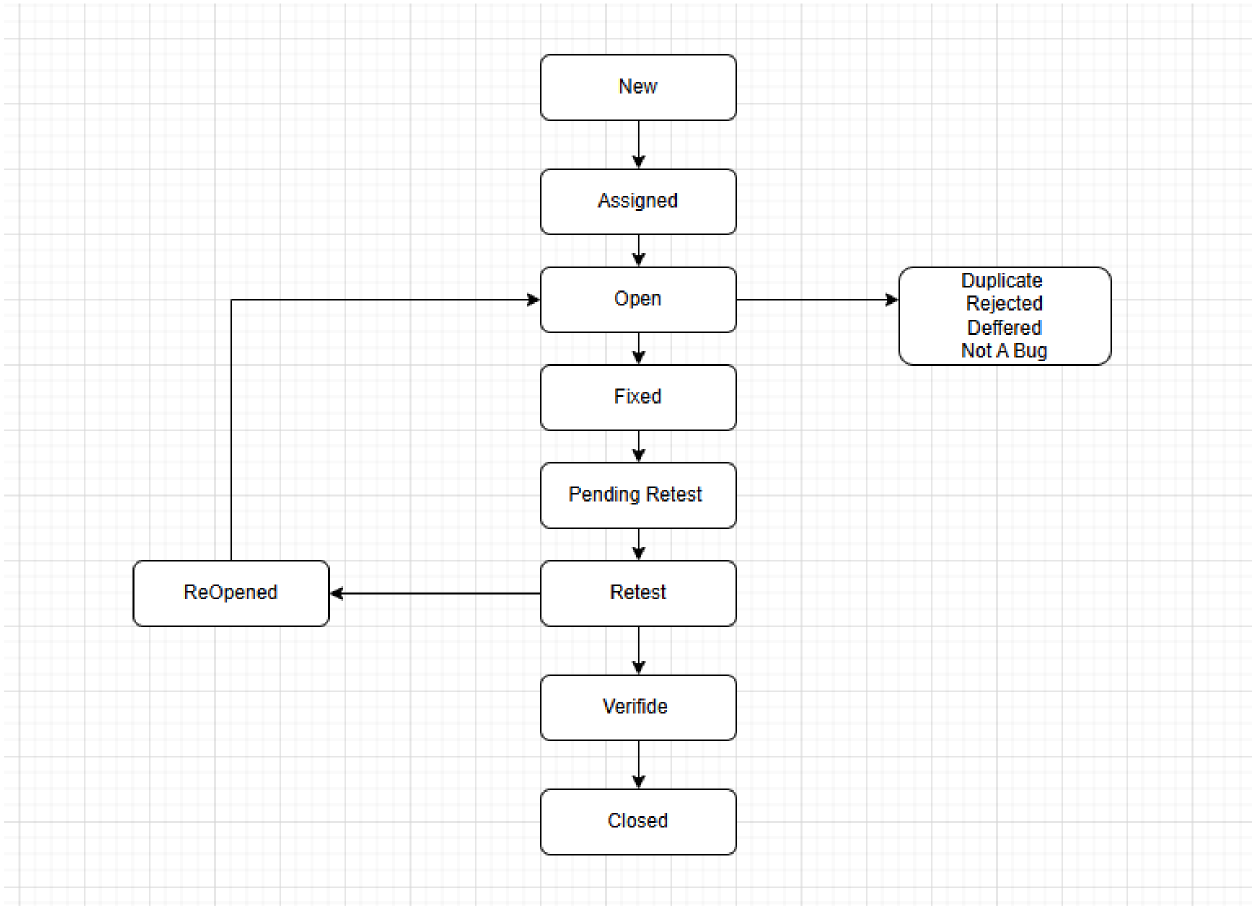
- The inability of a system or component to perform its required functions within specified performance requirements. See: bug, crash, exception, and fault.

27.Difference between Priority and Severity?

S.N	**Priority Testing**	**Severity Testing**
1	Priority is a term that defines how fast we need to fix a defect.	Severity is a term that denotes how severely a defect can affect the functionality of the software.
2	Priority is basically a parameter that decides the order in which we should fix the defects.	Severity is basically a parameter that denotes the total impact of a given defect on any software.
3	Severity relates to the standards of quality.	Priority relates to the scheduling of defects to resolve them in software.
4	The value of severity is objective.	The value of priority is subjective.

28.What is Bug Life Cycle?

- "A computer bug is an error, flaw, mistake, failure, or fault in a computer program that prevents it from working correctly or produces an incorrect result. Bugs arise from mistakes and errors, made by people, in either a program's source code or its design.
- The duration or time span between the first time defects is found and the time that it is closed successfully, rejected, postponed or deferred is called as 'Defect Life Cycle'.
- When a bug is discovered, it goes through several states and eventually reaches one of the terminal states, where it becomes inactive and closed.
- The process by which the defect moves through the life cycle is depicted next slide.



- As you can see from above diagram, a defect's state can be divided into Open or Closed.
- When a bug reaches one of the Closed or Terminal states, its lifecycle ends. Each state has one or more valid states to move to.

- This is to ensure that all necessary steps are taken to resolve or investigate that defect. For example, a bug should not move from Submitted state to resolved state without having it open.
- In a typical scenario, as soon as a bug is identified, it is logged into the bug tracking system with status as Submitted. After ascertaining the validity of the defect, it is given the “Open” Status.
- Defect Stages:
- New: When a new defect is logged and posted for the first time. It is assigned a status as NEW.
- Assigned: Once the bug is posted by the tester, the lead of the tester approves the bug and assigns the bug to the developer team
- Open: The developer starts analyzing and works on the defect fix
- Fixed: When a developer makes a necessary code change and verifies the change, he or she can make bug status as “Fixed.
- Pending retest: Once the defect is fixed the developer gives a particular code for retesting the code to the tester. Since the software testing remains pending from the testers end, the status assigned is “pending retest.”
- Retest: Tester does the retesting of the code at this stage to check whether the defect is fixed by the developer or not and changes the status to “Re-test.”
- Verified: The tester re-tests the bug after it got fixed by the developer. If there is no bug detected in the software, then the bug is fixed and the status assigned is “verified”.
- Reopen: If the bug persists even after the developer has fixed the bug, the tester changes the status to “reopened”. Once again the bug goes through the life cycle.
- Closed: If the bug is no longer exists then tester assigns the status “Closed.”
- Duplicate: If the defect is repeated twice or the defect corresponds to the same concept of the bug, the status is changed to “duplicate.
- Rejected: If the developer feels the defect is not a genuine defect then it changes the defect to “rejected.”
- Deferred: If the present bug is not of a prime priority and if it is expected to get fixed in the next release, then status “Deferred” is assigned to such bugs.
- Not a bug:If it does not affect the functionality of the application then the status assigned to a bug is “Not a bug”.

29.Explain the difference between Functional testing and NonFunctional testing?

S.N	Functional Testing	Non-Funactional Testing
1	Functional testing is performed using the functional specification provided by the client and verifies	Non-Functional testingchecksthe Performance, reliability, scalability and other non-functional aspects of the software system.

	the system against the functional requirements.	
2	Functional testing is executed first	Non functional testing should be performed after functional testing
3	Manual testing or automation tools can be used for functional testing	Using tools will be effective for this testing
4	Business requirements are the inputs to functional testing.	Performance parameters like speed , scalability are inputs to non-functional testing.
5	Functional testing describes what the product does	Nonfunctional testing describes how good the product works
6	Easy to do manual testing	Tough to do manual testing
7	Types of Functional testing · Unit Testing · Smoke Testing · Sanity Testing · Integration Testing · White box testing · Black Box testing · User Acceptance testing · Regression Testing	Types of Nonfunctional testing · Performance Testing · Load Testing · Volume Testing · Stress Testing · Security Testing · Installation Testing · Penetration Testing · Compatibility Testing · Migration Testing

30To create HLR & TestCase of

1)(Instagram , Facebook) only first page
- upload in PDF

2) Facebook Login Page : https://www.facebook.com/
- upload in PDF

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

	SDLC	**STLC**
Origin	Software Development Life Cycle	Software Testing Life Cycle
Definition	SDLC focuses on delivering a high-quality system that meets or exceeds user expectations, operates effectively within current and planned IT infrastructure, and is cost-efficient to maintain.	STLC defines the sequence of testing activities, specifying what tests to perform and when, to ensure thorough validation. While testing processes vary by organization, a structured test life cycle exists.
Focus	Covers both development and testing processes	Focuses solely on the testing process
Performed	SDLC phases complete before the start of STLC phases	STLC phases are executed after the SDLC phases
Objective	To successfully overcome challenges in software development and deliver a functional product	To identify defects or weaknesses in the software
Team Involved	Project Managers, Business Analysts, Designers, Developers	Quality Assurance and Testing Teams
Outcome	Delivery of a high-quality software product that meets business needs	Delivery of software with minimized bugs and verified quality

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

Test Scenarios	Test Cases	Test Script
In Any Functionality that can be Test	Is a Action Executed to Verify Particular features or Functionality	Is a Set Of Instruction To Test an app Automatically.
Is Derived from Test artifacts like Business requirement Specification And Software Requirement Specification	Is Mostly Derived from Test Scenarios	Is Mostly Derived from test case .
Help Test the end - to - end Functionality in an agile way	Help in Executive testing off an app	Help to Test specific Things Repeatedly .
Is More Focused What to Test .	Is Focused On What to Test And How to Test	In Focused on Expected Result .
Tack Less Time and Fewer Resource to create	Required more Resources and Time	Required less time for Testing but more resource fore script Creating and Updating.
Include End-To -End Functionality to be Test.	Include Test step ,Data ,Expected Result For Testing .	Include Different Command To Developed Script.

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

- Test Planning in STLC is a phase in which a Senior QA manager determines the test plan strategy along with efforts and cost estimates for the project.
- Moreover, the resources, test environment, test limitations and the testing schedule are also determined.
- The Test Plan gets prepared and finalized in the same phase.

Activities in Requirement Phase Testing:

- Create test cases, automation scripts (if applicable)
- Review and baseline test cases and scripts
- Create test data (If Test Environment is available)

Deliverables of Requirement Phase Testing:

- Test cases/scripts
- Test data

Test Environment Setup:

- Test Environment Setup decides the software and hardware conditions under which a work product is tested.
- It is one of the critical aspects of the testing process and can be done in parallel with the Test Case Development Phase.
- Test team may not be involved in this activity if the development team provides the test environment.
- The test team is required to do a readiness check (smoke testing) of the given environment.

Activities in Requirement Phase Testing :

- Understand the required architecture, environment set-up and prepare hardware and software requirement list for the Test Environment.
- Setup test Environment and test data
- Perform smoke test on the build

Deliverables of Requirement Phase Testing:

- Environment ready with test data set up
- Smoke Test Results

Test Execution:

- Test Execution Phase is carried out by the testers in which testing of the software build is done based on test plans and test cases prepared.
- The process consists of test script execution, test script maintenance and bug reporting.
- If bugs are reported then it is reverted back to development team for correction and retesting will be performed.

Activities in Requirement Phase Testing :

- Execute tests as per plan
- Document test results, and log defects for failed cases.
- Map defects to test cases in RTM
- Retest the Defect fixes
- Track the defects to closure

Deliverables of Requirement Phase Testing :

- Completed RTM with the execution status
- Test cases updated with results
- Defect reports

34.What is priority?

- Priority is Relative and Business-Focused. Priority defines the order in which we should resolve a defect. Should we fix it now, or can it wait? This priority status is set by the tester to the developer mentioning the time frame to f ix the defect. If high priority is mentioned then the developer has to fix it at the earliest. The priority status is set based on the customer requirements.
- For example: If the company name is misspelled in the home page of the website, then the priority is high and severity is low to fix it.
- Priority can be of following types:

- Low: The defect is an irritant which should be repaired, but repair can be deferred until after more serious defect has been fixed.
- Medium: The defect should be resolved in the normal course of development activities. It can wait until a new build or version is created.
- High: The defect must be resolved as soon as possible because the defect is affecting the application or the product severely. The system cannot be used until the repair has been done.
- Critical: Extremely urgent, resolve immediately.

35.What is severity?

- Severity is absolute and Customer-Focused. It is the extent to which the defect can affect the software. In other words it defines the impact that a given defect has on the system.
- Severity can be of following types :
- Critical: The defect that results in the termination of the complete system or one or more component of the system and causes extensive corruption of the data. The failed function is unusable and there is no acceptable alternative method to achieve the required results then the severity will be stated as critical.
- Major (High): The defect that results in the termination of the complete system or one or more component of the system and causes extensive corruption of the data. The failed function is unusable but there exists an acceptable alternative method to achieve the required results then the severity will be stated as major.
- Moderate (Medium): The defect that does not result in the termination, but causes the system to produce incorrect, incomplete or inconsistent results then the severity will be stated as moderate.
- Minor (Low): The defect that does not result in the termination and does not damage the usability of the system and the desired results can be easily obtained by working around the defects then the severity is stated as minor.
- Cosmetic: The defect that is related to the enhancement of the system where the changes are related to the look and field of the application then the severity is stated as cosmetic.

36.Bug categories are…

- Critical: The defect that results in the termination of the complete system or one or more component of the system and causes extensive corruption of the data. The failed function is unusable and there is no acceptable alternative method to achieve the required results then the severity will be stated as critical.
- Major (High): The defect that results in the termination of the complete system or one or more component of the system and causes extensive corruption of the data. The failed function is unusable but there exists an acceptable alternative method to achieve the required results then the severity will be stated as major.

- Moderate (Medium): The defect that does not result in the termination, but causes the system to produce incorrect, incomplete or inconsistent results then the severity will be stated as moderate.
- Minor (Low): The defect that does not result in the termination and does not damage the usability of the system and the desired results can be easily obtained by working around the defects then the severity is stated as minor.
- Cosmetic: The defect that is related to the enhancement of the system where the changes are related to the look and field of the application then the severity is stated as cosmetic.

37.Advantage of Bugzila.

- Bugzilla is an open-source issue/bug tracking system that allows developers effectively to keep track of outstanding problems with their product. It is written in Perl and uses MYSQL database.
- Bugzilla is a defect tracking tool, however it can be used as a test management tool as such it can be easily linked with other test case management tools like Quality Center, Testlink etc.
- This open bug-tracker enables users to stay connected with their clients or employees, to communicate about problems effectively throughout the data-management chain.
- Key features of Bugzilla includes
- Advanced search capabilities
- E-mail Notifications
- Modify/file Bugs by e-mail
- Time tracking
- Strong security
- Customization
- Localization

38.Difference between priority and severity

Features	Severity	Priority
Definition	Severity is a parameter to denote the impact of a particular defect on the software.	Priority is a parameter to decide the order in which defects should be fixed.
Purpose	Severity means how severe the defect is affecting the functionality.	Priority means how fast the defect has to be fixed.

Relation	Severity is related to the quality standard.	Priority is related to scheduling to resolve the problem.
Based On	It is based on the technical aspect of the product.	It is based on the customer's requirements.
Indication	It indicates the seriousness of the bug in the product functionality.	It indicates how soon the bug should be fixed.

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

- Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product.
- Agile Methods break the product into small incremental builds
- These builds are provided in iterations.
- Each iteration typically lasts from about one to three weeks.
- Every iteration involves cross functional teams working simultaneously on various areas like planning, requirements analysis, design, coding, unit testing, and acceptance testing.
- At the end of the iteration a working product is displayed to the customer and important stakeholders.

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

S.NO.	Authorization	Authantication
1	While in this process, users or persons are validated.	In the authentication process, users or persons are verified.
2	While this process is done after the authentication process.	It is done before the authorization process.
3	While it needs the user's privilege or security levels.	It needs usually the user's login details.

4	While it determines What permission does the user have?	Authentication determines whether the person is user or not.
5	Generally, transmit information through an Access Token.	Generally, transmit information through an ID Token.
6.	The authorization permissions cannot be changed by user as these are granted by the owner of the system and only he/she has the access to change it.	The authentication credentials can be changed in part as and when required by the user.
7.	The user authentication is visible at user end.	The user authorization is not visible at the user end.

41.To create HLR & TestCase of WebBased (WhatsApp web , Instagram) 1. WhatsApp Web : https://web.whatsapp.com/

1.Whatsapp Web : https://web.whatsapp.com/
- upload in PDF

2.Instagram Web :
- upload in PDF

3.To create HLR and TestCase on this Link. https://artoftesting.com/
- upload in PDF

42. Write a scenario of only Whatsapp chat messages ?

1.Verify that the user can set a chat wallpaper.
2.Verify that the user sets privacy settings like turning on/off last seen, online status, read receipts, etc.
3. Verify that the user can update notification settings like – notification sound, on/off, and show preview for both group and individual chats.
4.Verify that the user can take the complete chat backup of his chats.
5.Verify that the user can update the phone number that is used by the WhatsApp application.
6.Verify that the user can disable/delete his Whatsapp account.

7.Verify that the user can check data usage by images, audio, video, and documents in WhatsApp chats.
8. Verify delivery ticks, formatting (bold/italic/strikethrough), editing messages .
9. Verify images, GIFs, stickers, voice notes, document sharing.
10.Verify Delete/edit test flows and UI presentation.
11.Privacy settings like last seen and two-step verification activation.
12.Verify Show Particular Chat Show Payment Option.

43.Write a Scenario of Pen?

1.Verify that the length and the diameter of the pen are as per the specifications.
2.Verify the outer body material of the pen. Check if it is metallic, plastic, or any other material specified in the requirement specifications.
3.Check the color of the outer body of the pen. It should be as per the specifications.
4 .Verify that the brand name and/or logo of the company creating the pen should be clearly visible.
5.Verify that any information displayed on the pen should be legible and clearly visible.
6.Verify the type of pen, whether it is a ballpoint pen, ink pen, or gel pen.
7.Verify that the user is able to write clearly over different types of papers.
8.Check the weight of the pen. It should be as per the specifications. In case not mentioned in the specifications, the weight should not be too heavy to impact its smooth operation.
9.Verify if the pen is with a cap or without a cap.
10.Verify the color of the ink on the pen.
11.Check the odor of the pen's ink on writing over a surface.
12.Verify the pen's performance on different paper textures (smooth, rough, glossy, etc.) to ensure consistent ink dispersion and grip.
13.Verify the surfaces over which the pen is able to write smoothly apart from paper e.g. cardboard, rubber surface, etc.
14.Verify that the text written by the pen should have consistent ink flow without leaving any blob.
15.Check that the pen's ink should not leak in case it is tilted upside down.
16.Verify if the pen's ink should not leak at higher altitudes.
17.Verify if the text written by the pen is erasable or not.
18.Check the functioning of the pen by applying normal pressure during writing.
19.Verify the strength of the pen's outer body. It should not be easily breakable.
20.Verify that text written by pen should not get faded before a certain time as mentioned in the specification.
21.Check if the text written by the pen is waterproof or not.
22.Verify that the user is able to write normally by tilting the pen at a certain angle instead of keeping it straight while writing.

23.Check the grip of the pen, and whether it provides adequate friction for the user to comfortably grip the pen.
24.Verify if the pen can support multiple refills or not.
25.In the case of an ink pen, verify that the user is able to refill the pen with all the supported ink types.
26.For ink pens, verify that the mechanism to refill the pen is easy to operate.

44.Write a Scenario of Pen Stand?

1. Verify Pen Stand UI.
2. Verify the pen stand can hold the specified number of pens.
3. Test the stability on various surfaces (wood, glass, etc.)
4. Check the material durability under pressure.
5. Test resistance to water/spills.
6. Verify no sharp edges or unsafe parts.
7. Check dimensions accuracy against specs.
8. Drop test from a standard desk height.
9. Color fastness to cleaning agents.
10. Test compatibility with different pen sizes.
11. Pen stand holds pens without tipping over

45.Write a Scenario of Door?

1.Verify if the door is single door or bi-folded door.
2.Check if the door opens inwards or outwards.
3.Verify that the dimension of the doors are as per the specifications.
4.Verify that the material used in the door body and its parts is as per the specifications.
5.Verify that color of the door is as specified.
6.Verify if the door is sliding door or rotating door.
7.Check the position, quality and strength of hinges.
8.Check the type of locks in the door.
9.Check the number of locks in the door interior side or exterior side.
10.Verify if the door is having peek-hole or not.
11.Verify if the door has a stopper or not.
12.Verify if the door closes automatically or not – spring mechanism.
13.Verify if the door makes noise when opened or closed.
14.Check the door condition when used extensively with water.
15.Check the door condition in different climatic conditions- temperature, humidity etc.
16.Check the amount of force- pull or push required to open or close the door.

46.Write a Scenario of ATM?

1. Verify that all the labels and controls including text boxes, button images, and links are present on the screen.
2. Check the informative text written displayed on the screen is clearly visible and legible.
3. Verify that the size, color, and UI of the different objects are as per the specifications.
4. Verify that the application's UI is responsive i.e. it should adjust to different screen resolutions of ATM machines.
5. Verify the type of ATM machine, if it has a touch screen, both keypad buttons only, or both.
6. Verify that on properly inserting a valid card different banking options appear on the screen.
7. Check that no option to continue and enter credentials is displayed to the user when the card is inserted incorrectly.
8. Verify that the touch of the ATM screen is smooth and operational.
9. Verify that the user is presented with the option to choose a language for further operations.
10. Check that the user is asked to enter a pin number before displaying any card/bank account detail.
11. Verify that there is a limited number of attempts up to which the user is allowed to enter the pin code.
12. Verify that if the total number of incorrect pin attempts gets surpassed then the user is not allowed to continue further. And operations like temporary blocking of the card, etc get initiated.
13. Check that the pin is displayed in masked form when entered.
14. Verify that the user is presented with different account type options like- saving, current, etc.
15. Verify that the user is allowed to get account details like available balance.
16. Check that the correct amount of money gets withdrawn as entered by the user for cash withdrawal.
17. Verify that the user is only allowed to enter the amount in multiple denominations as per the specifications.
18. Verify that the user is prompted to enter the amount again in case the amount entered is less than the minimum amount configured.
19. Check that the user cannot withdraw more amount than the total available balance and a proper message should be displayed.
20. Verify that the user is provided the option to get the transaction details in printed form.
21. Verify that the user's session timeout is maintained.
22. Check that the user is not allowed to exceed one transaction limit amount.

23. Verify that the user is not allowed to exceed the one-day transaction limit amount.
24. Verify that the user is allowed to do only one transaction per pin request.
25. Check that in case the ATM machine runs out of money, a proper message is displayed to the user.
26. Verify that the applicable fee gets deducted along with the withdrawn amount in case the user exceeds the limit of the number of free transactions in a month.
27. Verify that the applicable fee gets deducted along with the withdrawn amount in case the user uses a card of a bank other than that of an ATM.
28. Check that the user is not allowed to proceed with the expired ATM card and that a proper error message gets displayed.
29. Verify that in case of sudden electricity loss before withdrawing cash, the transaction is marked as null and the amount is not withdrawn from the user's account.

.

47.When to used Usability Testing?

- Aesthetics and design are important. How well a product looks usually determines how well it works.
- There are many software applications / websites, which miserably fail, once launched, due to following reasons-
- Where do I click next?
- Which page needs to be navigated?
- Which Icon or Jargon represents what?
- Error messages are not consistent or effectively displayed.
- Session time not sufficient.
- Usability Testing identifies usability errors in the development cycle and can save a product from failure.

Goal Of Usability Testing :

- Effectiveness of the system
- Efficiency
- Accuracy
- User Friendliness

Pros:

- It helps uncover usability issues before the product is marketed
- It helps improve end user satisfaction.
- It makes your system highly effective and efficient.
- It helps gather true feedback from your target audience who actually use your system during usability test. You do not need to rely on "opinions" from random people.

Cons :

- Cost is a major consideration in usability testing. It takes lots of resources to set up a Usability Test Lab. Recruiting and management of usability testers can also be expensive.
- However, these costs pay themselves up in form of higher customer satisfaction, retention and repeat business. Usability testing is therefore highly recommended.

48.What is the procedure for 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.

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

49.Write a scenario of Microwave Owen?

1. Verify that the dimensions of the oven are as per the specification provided.
2. Verify that the oven's material is optimal for its use as an oven and as per the specification.
3. Verify that the oven heats the food at the desired temperature properly.
4. Verify that the oven heats food at the desired temperature within a specified time duration.
5. Verify the ovens functioning with the maximum attainable temperature.
6. Verify the ovens functioning with minimum attainable temperature.
7. Verify that the oven's plate rotation speed is optimal and not too high to spill the food kept over it.
8. Verify that the oven's door gets closed properly.

9. Verify that the oven's door opens smoothly.
10. Verify the battery requirement of the microwave oven and check that it functions smoothly at that power.
11. Verify that the text written over the oven's body is clearly readable.
12. Verify that the digital display is clearly visible and functions correctly.
13. Verify that the temperature regulator is smooth to operate.
14. Check the maximum capacity of the oven and test its functioning with that volume of food.
15. Check the oven's functionality with different kinds of food – solid, and liquid.
16. Check the oven's functionality with different food at different temperatures.
17. Verify the oven's functionality with different kinds of container material.
18. Verify that the power cord of the oven is long enough.
19. Verify that the usage instruction or user manuals have clear instructions.

50.Write a scenario of Coffee vending Machine

1.UI scenario – Verify that the dimension of the coffee machine is as per the specification.
2.Verify that outer body, as well as inner part's material, is as per the specification.
3.Verify that the machine's body color as well brand is correctly visible and as per specification.
4.Verify the input mechanism for coffee ingredients-milk, water, coffee beans/powder, etc.
5.Verify that the quantity of hot water, milk, coffee powder per serving is correct.
6.Verify the power/voltage requirements of the machine.
7.Verify the effect of suddenly switching off the machine or cutting the power. The machine should stop in that situation and in power resumption, the remaining coffee should not get come out of the nozzle.
8.Verify that coffee should not leak when not in operation.
9.Verify the amount of coffee served in single-serving is as per specification.
10.Verify that the digital display displays correct information.
11.Check if the machine can be switched on and off using the power buttons.
12.Check for the indicator lights when the machine is switched on-off.
13.Verify that the functioning of all the buttons work properly when pressed.
14. Verify that each button has an image/text with it, indicating the task it performs.
15.Verify that complete quantity of coffee should get poured in a single operation, no residual coffee should be present in the nozzle.
16.Verify the mechanism to clean the system work correctly- foamer.
17.Verify that the coffee served has the same and correct temperature each time it is served by the machine.
18.Verify that system should display an error when it runs out of ingredients.

19.Verify that pressing the coffee button multiple times leads to multiple serving of coffee.
20.Verify that there is the passage for residual/extra coffee in the machine.
21.Verify that machine should work correctly in different climatic, moistures and temperature conditions.
22.Verify that machine should not make too much sound when in operation.
23.Performance test – Check the amount of time the machine takes to serve a single serving of coffee.
24.Performance test – Check the performance of the machine when used continuously until the ingredients run out of the requirements.
25.Negative Test – Check the functioning of the coffee machine when two/multiple buttons are pressed simultaneously.
26.Negative Test – Check the functioning of coffee machine with a lesser or higher voltage than required.
27.Negative Test – Check the functioning of the coffee machine if the ingredient container's capacity is exceeded.

51.Write a scenario of chair

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

52.To Create Scenario (Positive & Negative)

1.Gmail(Receiving mail) - Positive

1. Verify that a newly received email is displayed as highlighted in the Inbox section.

2.Verify that a newly received email has correctly displayed sender email Id or name, mail subject and mail body(trimmed to a single line)

3.Verify that on clicking the newly received email, the user is navigated to email content.

4.Verify that the email contents are correctly displayed with the desired source formatting.

5.Verify that any attachments are attached to the email and are downloadable.

6.Verify that the attachments are scanned for viruses before download.

7.Verify that all the emails marked as read are not highlighted.

8.Verify that all the emails read as well as unread have a mail read time appended at the end on the email list displayed in the inbox section.

9.Verify that count of unread emails is displayed alongside 'Inbox' text in the left sidebar of Gmail.

10.Verify that unread email count increases by one on receiving a new email.

11.Verify that unread email count decreases by one on reading an email (marking an email as read).

12.Verify that email recipients in cc are visible to all users.

13.Verify that email recipients in bcc are not visible to the user.

14.Verify that all received emails get piled up in the 'Inbox' section and get deleted in cyclic fashion based on the size availability.

15.Verify that email can be received from non-Gmail email Ids like – yahoo, Hotmail etc.

16.Verify that on clicking the 'Compose' button, a frame to compose a mail gets displayed.

17.Verify that the user can enter email Ids in 'To', 'cc' and 'bcc' sections and also the user will get suggestions while typing the emails based on the existing emailIds in the user's email list.

18.Verify that the user can enter multiple comma-separated emailIds in 'To', 'cc' and 'bcc' sections.

19.Verify that the user can type the Subject line in the 'Subject' textbox.

20.Verify that the user can type the email in the email-body section.

21.Verify that users can format mail using editor-options provided like choosing font-family, font-size, bold-italic-underline, etc.

22.Verify that the user can attach a file as an attachment to the email.

23.Verify that the user can add images in the email and select the size for the same.

24.Verify that after entering emailIds in either of the ‘To’, ‘cc’ and ‘bcc’ .sections, entering Subject line and mail body and clicking ‘Send’ button, mail gets delivered to intended receivers.

25.Verify that sent mails can be found in ‘Sent Mail’ sections of the sender.

26.Verify that mail can be sent to non-gmail emailIds also.

27.Verify that all sent emails get piled up in the ‘Sent Mail’ section and get deleted in cyclic fashion based on the size availability.

28.Verify that the emails composed but not sent remain in the draft section.

29.Verify the maximum number of email recipients that can be entered in ‘To’, ‘cc’ and ‘bcc’ sections.

30.Verify the maximum length of text that can be entered in the ‘Subject’ textbox.

31.Verify the content limit of text/images that can be entered and successfully delivered as mail body.

32.Verify the maximum size and number of attachment that can be attached with an email.

33.Verify that only the allowed specifications of the attachment can be attached with an email.

34.Verify that if the email is sent without Subject, a pop-up is generated warning user about no subject line. Also, verify that on accepting the pop-up message, the user is able to send the email.

2.Gmail(Receiving mail) - Negative

1. Verify Incorrect Email Addresses Double-check the email address for errors like typos, extra spaces, or incorrect characters.
2. Verify Storage Limits If your Gmail account is full, it might not be able to receive new emails.
3. Verify Spam Filters Legitimate emails can sometimes be incorrectly marked as spam. Check your spam folder to see if the missing emails are there.
4. Verify that the sender's email address isn't on your block list.
5. Verify The sender might have encountered issues like a full inbox, a temporary outage with their email server, or a problem with their outgoing mail.
6. Verify Check your Gmail settings for any forwarding rules or filters that might be affecting the delivery of emails.
7. Verify In rare cases, Gmail itself might be experiencing an outage. Check the Google Workspace Status Dashboard to see if there are any reported issues.
8. Verify Look for legitimate emails that might have been misclassified as spam.
9. Verify Look for legitimate emails that might have been misclassified as spam.
10. Verify If you suspect the issue is on the sender's side, ask them to check their sent items, bounced messages, and their email server status.
11. Verify Check for any forwarding rules, filters, or blocked senders that might be interfering with email delivery.

2.Online shopping to buy product (flipkart)

Product Buy Flow :

1.Verify the initiation of the buy flow.

2.Test the accuracy of product details.

3.Validate responsiveness to changes in quantity.

4.Check behavior with out-of-stock items.

5.Test adding products from different categories.

6.Verify the accuracy and function of applied discounts.

7.Test the visibility and accuracy of delivery options.

8.Validate responsiveness of the payment selection step.

9.Test behavior with saved addresses.

10.Verify the accuracy of order summary details.

11.Test the visibility of promotional banners.

12.Validate responsiveness to changes in user location.

13.Test the redirection to the Order Confirmation page.

14.Check cancelling the payment process midway.

Payment :

1.Verify redirection to the secure payment gateway.

2.Test behavior with incorrect payment details.

3.Validate responsiveness to changes in payment methods.

4.Check the accuracy of applied discounts.

5.Test successful payment processing.

6.Verify the visibility of transaction details.

7.Test behavior with declined transactions.

8.Validate responsiveness to changes in user location.

9.Test the visibility of saved payment methods.

10.Verify accurate calculation of transaction charges.

11.Test the behavior of network interruptions during payment.

12.Validate responsiveness to changes in currency.

13.Test the redirection to the Order Confirmation page

53.Write a Scenario of Wrist Watch

1.Verify the type of watch – analog or digital.
2.In the case of an analog watch, check the correctness time displayed by the second, minute, and hour hand of the watch.
3.In the case of a digital watch, check the digital display for hours, minutes, and seconds is correctly displayed.
4.Verify the material of the watch and its strap.
5.Check if the shape of the dial is as per specification.
6.Verify the dimension of the watch is as per the specification.
7.Verify the weight of the watch.
8.Check if the watch is waterproof or not.
9.Verify that the numbers in the dial are clearly visible or not.
10.Check if the watch is having a date and day display or not.
11.Verify the color of the text displayed in the watch – time, day, date, and other information.
12.Verify that clock's time can be corrected using the key in case of an analog clock and buttons in case of a digital clock.
13.Check if the second hand of the watch makes ticking sound or not.
14.Verify the brand of the watch and check if its visible in the dial.
15.Check if the clock is having stopwatch, timers, and alarm functionality or not.
16.In the case of a digital watch, verify the format of the watch 12 hours or 24 hours.
17.Verify if the watch comes with any guarantee or warranty.
18.Verify if the dial has glass covering or plastic, check if the material is breakable or not.
19.Verify if the dial's glass/plastic is resistant to minor scratches or not.
20.Check the battery requirement of the watch.

54.Write a Scenario of Lift(Elevator)

1.Verify the dimensions of the lift.
2.Verify the type of door of the lift is as per the specification.
3.Verify the type of metal used in the lift interior and exterior.
4.Verify the capacity of the lift in terms of the total weight.
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 mishap.
9.Verify the performance of the floor – the time taken to go to a floor.

10.Verify that in case of power failure, the lift doesn’t free-fall and gets halted on the particular floor.
11.Verify lifts working in case the button to open the door is pressed before reaching the destination floor.
12.Verify that in case the door is about to close and an object is placed between the doors if the doors sense the object and again open or not.
13.Verify the time duration for which the door remains open by default.
14.Verify if the lift interior is having proper air ventilation.
15.Verify lighting in the lift.
16.Verify that at no point the lift door should open while in motion.
17.Verify that in case of power loss, there should be a backup mechanism to safely get into a floor or a backup power supply.
18.Verify that in case the multiple floor number button is clicked, the lift should stop on each floor.
19.Verify that in case of capacity limit is reached users are prompted with a warning alert- audio/visual.
20.Verify that inside lift users are prompted with the current floor and direction information the lift is moving towards- audio/visual prompt.

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

1.Verify Device Internet properly or not.
2.Verify Group name Character Limited or Not .
3.Verify that a user can successfully create a new group.\
4.Confirm that all added members receive an invitation and join successfully.
5.Verify Group Name Show Properly Or not.
6.Confirm that the group is immediately visible in the user's chat list.
7.Verify Group Admin Only Send Message or Other Members Send Message.
8.Verify Profile Picture Show Properly Or Not.
9.Verify Group Settings ,Group info Functionality Work Properly Or Not.
10.Verify Any Link Share in Group It Will Show Group all members or Not.
11.Verify Group info Settings in Audio Call , Video Call , Search Button is Work Properly Or Not.
12.Verify that the admin role can be assigned to a specific group member.
13.Test the functionality of admin-specific permissions, such as adding/removing members and changing group settings.
14.Check if admins receive notifications for important group activities, like member additions or removals.
15.Verify that only admins can create, add/remove, and make other users admin in the group.
16.Test the functionality of adding multiple members to a group simultaneously.

17.Check if new members have appropriate permissions, such as sending messages and sharing media.
18.Verify that existing members receive notifications for new member additions.
19.Test the ability to delete a group, ensuring it is removed from all members' accounts.
20.Verify that the admin can initiate group deletion process after removing all the members.
21.Check if a member can exit and delete the group for only themselves.
22.Verify that members receive notifications when removed from a group.
23.Test any recovery options available after group deletion, such as restoring archived content.

56.Write a Scenario of Whatsapp payment.

1. Verify Internet On Device .
2.User has WhatsApp installed and verified on their phone.
3. Verify Which version Support in Whatsapp Payment Option.
4. Verify Both sender and receiver have WhatsApp payment setup.
5.User's mobile number is UPI-linked with at least one bank account.
6.Sender has sufficient UPI balance.
7.Receiver's account is correctly linked and active.
8. Verify that a user can send money successfully to another user through WhatsApp.
9. Verify The receiver should receive a message with the payment notification.
10. Verify A confirmation message is sent to both the sender and receiver indicating that the payment was successful.
11.Verify Transaction recorded in WhatsApp Pay history.
12.Verify UI reflects updated balance.
13.Verify Both users receive digital receipt in chat.
14.Verify Ensure input fields (amount, note) are visible and labeled correctly .
15.Verify Verify masked display of UPI PIN/OTP entries.
16.Verify Confirm correct bank logo/name appears and matches account chosen.
17.Verify Show clear error.
18.Verify Funds should not be deducted.
19.Verify Transaction marked "Failed" in history.
20.Ensure data integrity, proper error handling, and secure transaction processing.
21.Test paying during poor network conditions .
22.If debited but not credited due to errors, reversal must trigger within X hours.
23.Refund notification sent to sender's chat.
24.Limit of PIN retries should hold to prevent brute-force.
25.Session timeout should automatically invalidate sensitive steps.