What is Exploratory Testing?

Exploratory Testing is widely used in Agile models and is all about discovery, investigation, and learning about the product.

Test cases are not created in advance but product is tested without any documentation. Its more like on the go testing.

What is traceability matrix?

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

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

It is basically a forward, backward or bi-directional traceable table consisting details about the product like which test cases are used for what requirements and description details about the requirement. It also denotes status of the testcase/ Requirement.

What is Boundary value testing?

Boundary Value Analysis is based on testing the boundary values of valid and invalid partitions. The behavior at the edge of the equivalence partition is more likely to be incorrect than the behavior within the partition, so boundaries are an area where testing is likely to yield defects.

It is a part of black box testing procedure where the divided code or modules are checked for the boundary values of valid and invalid partitions.

What is Equivalence partitioning testing?

In this technique, input data are divided into the equivalent partitions that can be used to derive test cases. Test cases are reduced into manageable chunks. It is applied when there is a range of input values.

The effectiveness of the testing is not compromised on test cases. It Works well with a large number of variables. It is a method that is used throughout testing.

What is Integration testing?

Integration testing is the process of testing the interface between two software units or modules. It focuses on determining the correctness of the interface.

The purpose of integration testing is to expose faults in the interaction between integrated units. It is done after unit testing is performed.

Integration test approaches – There are four types of integration testing approaches. Those approaches are the following:

1. Big Bang Integration
2. Top- Down Integration
3. Bottom-up Integration
4. Mixed Integration

What is Alpha testing?

Alpha Testing is performed by internal employees of the organization.

Alpha Testing is done within the organization.

During Alpha Testing only functionality and usability are tested in depth.

Long execution cycles may be needed for Alpha Testing.

What is beta testing?

Beta Testing is done by users.

Beta Testing is done in the user's environment. It is also called User Acceptance Testing.

Beta Testing usability, functionality, security, and reliability are tested in depth.

Short Execution cycle to make Beta Testing possible.

What is component testing?

Component testing is also called Unit Testing.

Unit Testing is a software testing technique by means of which individual units of software.

The software contains multiple modules when the each module is taken for testing it is tested as a unit by the developer for functionality errors.

It is correlated with the functional correctness of the independent modules.

What is functional system testing?

Functional is a type of software testing which is used to verify the functionality of the software application, whether the function is working according to the requirement specification.

To check Functional Aspects of software feature works as per the software requirements.

Each function is compared to the corresponding requirement to ascertain whether its output is consistent with the end user’s expectations. The testing is done by providing sample inputs, capturing resulting outputs, and verifying that actual outputs are the same as expected outputs.

Ex: Unit testing, regression testing etc.

What is Non-Functional Testing?

Non-Functional Testing is defined as a type of Software testing to check non-functional aspects (performance, usability, reliability, etc) of a software application. I

It is designed to test the readiness of a system as per nonfunctional parameters which are never addressed by functional testing.

Ex: Performance testing, Stress testing, useability testing etc.

What is GUI Testing?

Graphical User Interface Testing (GUI) Testing is the process for ensuring proper functionality of the graphical user interface (GUI) for a specific application.

GUI testing generally evaluates a design of elements such as layout, colors and also fonts, font sizes, labels, text boxes, text formatting, captions, buttons, lists, icons, links, and content. GUI testing processes may be either manual or automatic.

They are often performed by third-party companies, rather than developers or end users.

What is Adhoc testing?

Ad hoc Testing is an informal or unstructured software testing type that aims to break the testing process in order to find possible defects or errors at an early possible stage.

Ad hoc testing is done randomly and it is usually an unplanned activity which does not follow any documentation and test design techniques to create test cases.

There are different types of Adhoc testing and they are listed as below:

1. Buddy testing
   Two buddies mutually work on identifying defects in the same module. Mostly one buddy will be from development team and another person will be from testing team. Buddy testing helps the testers develop better test cases and development team can also make design changes early. It is done after unit testing.

2. Pair Testing
   Two testers are assigned modules, share ideas and work on the same machines to find defects. One person can execute the tests and another person can take notes on the findings. Roles of the persons can be a tester and scriber during testing.

3. Monkey Testing
   Randomly test the product or application without test cases with a goal to break the system.

What is load testing?

Load testing is the process of putting simulated demand on software, an application or website in a way that tests or demonstrates it's behavior under various conditions.

Load Testing is a type of Performance Testing. it determines the performance of a system, software product, or software application under real-life based load conditions when multiple users use it at the same time.

It is the response of the system measured under varying load conditions. The load testing is carried out for normal and extreme load conditions.

What is stress Testing?

Stress testing is also known as Endurance Testing or Torture Testing.

Stress Testing is a software testing technique that determines the robustness of software by testing beyond the limits of normal operation.

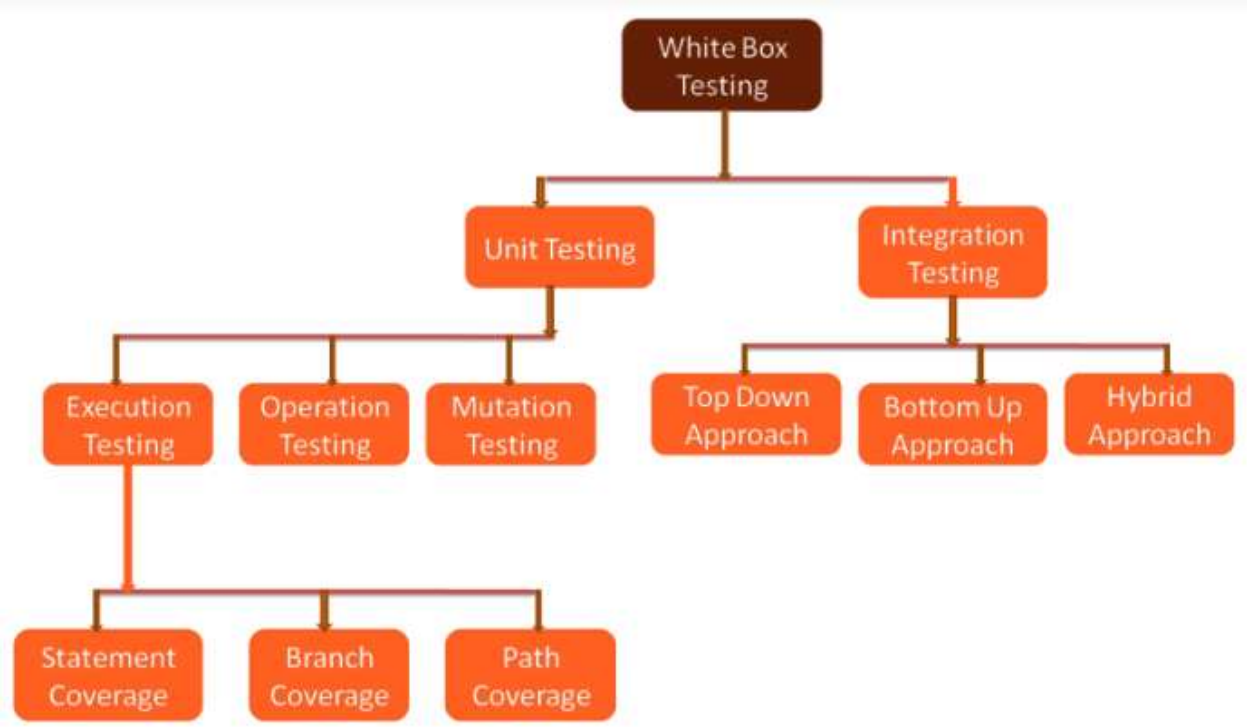
Stress testing emphasizes robustness, availability, and error handling under a heavy load rather than what is correct behavior under normal situations.

Stress testing is defined as a type of software testing that verifies the stability and reliability of the system. This test particularly determines the system on its robustness and error handling under extremely heavy load conditions. Stress testing is performed to ensure that the system would not crash under crunch situations.

What is white box testing and list the types of white box testing?

White box testing is an approach Where testers know the internal working of the test product.

Testers inspect and verify the inner workings of a software system—its code, infrastructure, and integrations with external systems.



Unit Testing :

Component testing is also called Unit Testing.

Unit Testing is a software testing technique by means of which individual units of software.

The software contains multiple modules when the each module is taken for testing it is tested as a unit by the developer for functionality errors.

It is correlated with the functional correctness of the independent modules.

- **Test coverage / code coverage**
  Test /code Coverage = (attempted test coverage / total coverage) * 100

  There are 3 types of test coverage
  1. Statement
     Check only True condition
     (Do ….While loop)

  2. Decision
     Branching true and false where false is only once tested
     (while loop)

  3. Condition
     Initial condition if true - executes
     Initial condition if false - stop

Integration Testing :

Integration testing is the process of testing the interface between two software units or modules. It focuses on determining the correctness of the interface.

The purpose of integration testing is to expose faults in the interaction between integrated units. It is done after unit testing is performed.

Integration test approaches – There are four types of integration testing approaches. Those approaches are the following:

1. Big Bang Integration
2. Top- Down Integration
3. Bottom-up Integration
4. Mixed/ Hybrid Integration

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

Black box testing is a type of software testing in which the functionality of the software is not known. The testing is done without the internal knowledge of the products code or how it works internally.

Different Black box techniques :

1. Equivalence Partitioning
2. Boundary Value Analysis
3. Decision Table Testing
4. State Transition Testing
5. Error Guessing
6. Graph-based Testing Methods
7. Comparison Testing
8. Use Case Technique

Mention what are the categories of defects?

Categories of defects are: Errors of commissions, Errors of omissions, Errors of clarity, and Error of speed and capacity.

1. **Error of Commission:**
   Commission means instruction or some kind of command given. Now the error in commission means the error in made in command or instruction. For example, suppose I wrote a loop which I was trying to run 10 times but I command it to run more than 10 times by mistake this is the error of commission.

2. **Errors of Omissions:**
   As name is already describing error of omission is some thing which happens accidentally. Omission word means something left out or executed. Practical most common example of this error is suppose we make a function in programming open its bracket but forget to close at the end.

3. **Error of Clarity:**
   The most common error in the natural languages. This error happens due to miss understanding between the developer and client. It travels most of the time from the requirements to the software.

4. **Error of Speed or Capacity:**
   The name of the error is itself enough i think to tell about it this error. Your software is working fine but not working in the required time this is the error of speed. When it comes to capacity it can be relevant to memory. For example, a small integer is declared where the long integer was required.

Mention what bigbang testing is?

Big Bang Integration Testing is an integration testing strategy.

In Big Bang testing all units are linked at once, resulting in a complete system.

it is difficult to isolate any errors found, because attention is not paid to verifying the interfaces across individual units.

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.

Exit criterion should be part of test plan and decided in the planning stage.

All tests planned must run, NO Critical or high severity defects that are left outstanding, level of requirement coverage met, NO outstanding defects, completely tested high risk areas and all tasks are completed within the projected cost in projected timelines.

When should "Regression Testing" be performed?

Regression Testing is applied where

1. A new requirement is added to an existing feature
2. A new feature or functionality is added
3. The codebase is fixed to solve defects
4. The source code is optimized to improve performance
5. Patch fixes are added
6. Changes in configuration

The system is checked using Regression testing everytime a new feature , update , patches or source code is changed to check for defects of compatibility between modules.

What is 7 key principles? Explain in detail?

1. Testing shows presence of defects (NO product is 100% defect free)

   As we test a product for defects , it is known that as time passes defect will be found even if there were none before. No product can be made that is 100% defect free.

2. Exhaustive testing is not possible

   Testing a product for infinite time is not possible because of time and cost.

3. Early testing

   AS soon as a working product is made testing should start. It is a good practice to start testing the product As early as possible which can make defects known and future possibility of product defects less frequent.

   It is much cheaper to fix a Defect in the early stages of testing.

4. Defect clustering

   Defect can be a single module or cluster of multiple modules. Tester has to find the defect and pinpoint it to certain module or cluster of modules.

5. Pesticide paradox

   As time passes the product is need of periodic testing. As we do the periodic testing on the product more defects will be found.

   If the same set of repetitive tests are conducted, the method will be useless for discovering new defects. To overcome this, the test cases need to be regularly reviewed & revised, adding new & different test cases to help find more defects.

6. Testing is context dependent

   No two products are same, so every product should be tested depending on their type methodologies , technologies, testcases would be different.

7. Absence of errors fallacy

   Being Error free and being useable are two different things.

   A product can be defect free but un-useable or Defective but useable. There is no error/ defect that does not mean it is useable or reliable.

   A product should fulfil the user's needs & requirements.

   The absence of Error is a Fallacy.

Difference between QA v/s QC v/s Tester

Quality Assurance (QA)	Quality Control (QC)	Testing
Process-oriented focuses on making the process of creating software better.	A product-oriented approach is a way to make sure the software meets all its requirements.	Testing the software system is about finding any mistakes or issues.
It works with the development process to help stop mistakes and ensure the software is of good quality. This means setting up and keeping standards, processes, procedures, and tools in place to ensure we're consistently producing high-quality software.	It's done after the development process and involves running test cases and seeing how the software reacts.	This usually happens after the software has been created, and it's all about ensuring that the software's quality is up to standard.
The goal is to keep improving our software development process for the best possible results.	The goal is to find any defects or errors in the software and fix them.	It involves running tests and looking at what comes out of them, finding any problems with the software, and ensuring that it does everything it's supposed to do.

Difference between Smoke and Sanity?

Smoke Testing	Sanity Testing
To check critical functionalities of the program is working fine.	To check the new functionality/bugs have been fixed.
The objective of this testing is to verify the "stability" of the system in order to proceed with more rigorous testing	The objective of the testing is to verify the "rationality" of the system in order to proceed with more rigorous testing
performed by the developers or testers.	performed by testers.
documented or scripted	not documented and is unscripted
subset of Acceptance testing	subset of Regression Testing
the entire system from end to end is checked	exercises only the particular component of the entire system

Difference between verification and Validation

Verification	Validation
process of checking if a product is developed as per the specifications.	process of ensuring that the product meets the needs and expectations of stakeholders.
tests the requirements, architecture, design, and code of the software product.	tests the usability, functionalities, and reliability of the end product.
does not require executing the code.	It emphasizes executing the code to test the usability and functionality of the end product.
A few activities involved in verification testing are requirements verification, design verification, and code verification.	The commonly-used validation activities in software testing are usability testing, performance testing, system testing, security testing, and functionality testing.
A few verification methods are inspection, code review, desk-checking, and walkthroughs.	A few widely-used validation methods are black box testing, white box testing, integration testing, and acceptance testing.
The quality assurance (QA) team would be engaged in the verification process.	The software testing team along with the QA team would be engaged in the validation process.
It targets internal aspects such as requirements, design, software architecture, database, and code.	It targets the end product that is ready to be deployed.

Explain types of Performance testing.

Performance testing is a non-functional software testing technique that determines how the stability, speed, scalability, and responsiveness of an application holds up under a given workload.

The goals of performance testing include evaluating application output, processing speed, data transfer velocity, network bandwidth usage, maximum concurrent users, memory utilization, workload efficiency, and command response times.

performance testing on the application, will concentrate on the various factors like Response time, Load, and Stability of the application.

Types of performance testing:

1. Load testing
   It's a performance testing to check system behaviour under load. Testing an application under heavy loads such as testing of a website under a range of loads to determine at what point the system's response time degrades or fails.

2. Stress testing
   Stress Testing is a software testing technique that determines the robustness of software by testing beyond the limits of normal operation. Done under high level of stress and user load.

3. Scalability testing
   Tests how many users the system can handle before performance dips below acceptable levels. By testing a software's capacity it helps developers anticipate issues in terms of scalability and future user-base growth.

4. Stability testing
   Checks that the software can handle and process a large amount of data at once without breaking, slowing down, or losing any information.

5. Soak Testing
   Simulates high traffic for an extended period of time. Checks the software's ability to tolerate extended periods of high traffic

What is Error, Defect, Bug and failure?

Error: A mistake in coding is called Error.

Defect: Error found by tester while testing is called Defect.

Bug: Defect accepted by development team after the defect report is given, then it is called Bug.

Failure: If a build does not meet the requirements then it Is Failure. A bulild should satisfy basic user /client Requirements.

"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."

Difference between Priority and Severity

Priority	Severity
a parameter to decide the order in which defects should be fixed.	a parameter to denote the impact of a particular defect on the software.
how fast defect has to be fixed.	how severe defect is affecting the functionality.
Scheduling to resolve the problem.	related to the quality standard.
Product manager decides the priorities of defects.	Testing engineer decides the severity level of the defect.
value is subjective.	Its value is objective.
value changes from time to time.	Its value doesn't change from time to time.
3 types: Low, Medium, High.	5 types: Critical, Major, Moderate, Minor, and Cosmetic.

Explain the difference between Authorization and Authentication in Web testing.

Authentication	Authorization
Authentication verifies who the user is.	Authorization determines what resources a user can access.
Authentication works through passwords, one-time pins, biometric information, and other information provided or entered by the user.	Authorization works through settings that are implemented and maintained by the organization.
Authentication is the first step of a good identity and access management process.	Authorization always takes place after authentication.
Authentication is visible to and partially changeable by the user.	Authorization isn't visible to or changeable by the user.

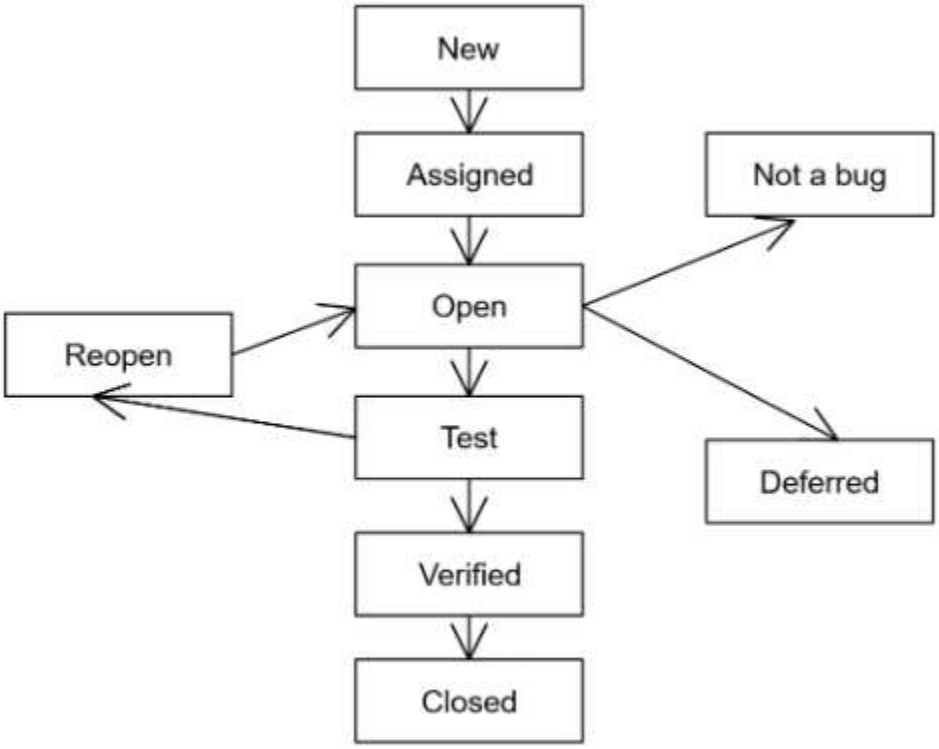
What determines the level of risk?

We often see situations where we have applied the best testing techniques and processes, and yet the testing wasn't completed in time or with quality.

It happens when we have not planned for risks in our testing process.

Determining the level of risk usually involves trying to assess not only the likelihood of an identified risk from actually occurring, but also the potential magnitude the consequences this risk could have on an organization and its stakeholder, should it occur.

What is Bug Life Cycle?



1. New
   A New defect added Potential defect that is raised and yet to be validated.
2. Assigned
   Assigned against a development team to address it but not yet resolved.
3. Active
   The Defect is being addressed by the developer and investigation is under progress.
   Defect if not a bug is flagged as not a bug.
   Defect if not possible to solve is differed as low priority.
4. Test
   The Defect is fixed and ready for testing.
5. Verified
   The Defect that is retested and the test has been verified by QA.
6. Closed
   The final state of the defect that can be closed.
7. Reopened
   When the defect is NOT fixed, QA reopens/reactivates the defect.
8. Deferred
   When a defect cannot be addressed in that particular cycle it is deferred to future release.

Explain the difference between Functional testing and Non-Functional testing.

Functional Testing	Non-Functional Testing
Verifies the operations and actions of an application.	Verifies the behaviour of an application.
based on requirements of customer.	based on expectations of customer.
helps to enhance the behavior of the application.	helps to improve the performance of the application.
easy to execute manually.	hard to execute non-functional testing manually.
tests what the product does.	describes how the product does.
based on the business requirement.	based on the performance requirement.
Ex: unit testing, smoke testing , integration testing, Regression testing.	Ex: Performance Testing, Load Testing, Stress Testing, Scalability Testing.

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

(Software Development Life Cycle)?

SDLC	STLC
SDLC, or software development life cycle, relates mainly to software development and includes all phases of software development, including testing	In essence, STLC is related to software testing, meaning that it is a software testing process that entails several phases.
As a whole, it covers the entire life cycle of the software and can be considered the predecessor.	Since it is part of SDLC and only involves testing, it is considered a child or successor.
SDLC aims to manage the entire process of software development from start to finish and to deliver a quality product that meets customer needs.	The focus is solely on test development and helps to make the testing process more sophisticated, consistent, and useful.
Phases of the SDLC are completed before those of the STLC.	Phases of STLC are carried out after the phases of SDLC.
Business Analysts and Product Analysts collect requirements and prepare a Development Plan during the Requirements collection phase of the SDLC.	The QA (Quality Assurance) team will analyze requirement documents such as functional and non-functional requirements, and then prepare a System Test Plan as part of the Requirement Analysis phase of the STLC.
SDLC process is to overcome any hurdle on the way to successful software development.	Testing is only intended to find any weaknesses or pitfalls in the system.
SDLC involves planning and designing software based on the requirements of the development team.	STLC involves the planning of tests by the testing team (Test Architect or Test Lead).
In this phase, programmers start writing code according to the designed document in any programming language to build the system from scratch.	Test cases and test scripts are developed by the testing team (quality assurance team) to verify the product's quality. They prepare the test environment and execute the tests.
As soon as the development team has written the code, they set up a test environment to validate it.	The testers ensure the test environment is prepared based on the prerequisites and conduct smoke tests to determine if the environment is stable enough for testing the product.

SDLC	STLC
The objective of this phase is to test the software. Among the activities included in this testing are Unit, Integration, System, Retest & Regression testing, etc., and the development team also participates in fixing reported bugs.	System integration testing is then conducted based on the test cases. All bugs and errors are reported, retested, and fixed. The product is also subject to regression tests and is signed off as soon as it meets the exit criteria.
As soon as the application has been approved by the various testing teams, it is deployed in a production environment for real end-users.	After the product has been deployed, smoke testing and sanity testing take place in the production environment, and the testing team prepares test reports and an analysis matrix for analyzing the product.
If necessary, post-deployment support, enhancement, and update are included.	The QA team runs regression tests to check deployed maintenance code. The team maintains test cases and automated scripts to make sure that tests are updated.
Throughout the SDLC, more people (developers) are needed.	The QA team runs regression tests to check deployed maintenance code. The team maintains test cases and automated scripts to make sure that tests are updated.
The end result of SDLC is the creation of reusable software systems.	STLC results in a tested software system.

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

Test Scenarios:

A test scenario is a description of an objective a user might face when using the product. It describes what can be positive and negative impacts on the product. A test scenario will require testing in a few different ways to ensure the scenario has been satisfactorily covered. It will be on possibility of success and failure.

Test cases:

Test cases describe a specific idea that is to be tested, without detailing the exact steps to be taken or data to be used. Every possibility of the product functionality is tested using the test cases. How detailed it should be checked is decided on test plan.

Test scripts:

A script typically has ‘steps’ that try to fully describe how to use the program — which buttons to press, and in which order — to carry out a particular action in the program.

When people talk about test scripts, they usually mean a line-by-line description of all the actions and data needed to perform a test.

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

A Test Plan is a detailed document that describes the test strategy, objectives, schedule, estimation, deliverables, and resources required to perform testing for a software product. Test Plan helps us determine the effort needed to validate the quality of the application under test.

The test plan serves as a blueprint to conduct software testing activities as a defined process, which is minutely monitored and controlled by the test manager.

Test plan Should cover:

1. Analyze the product
   Start with learning more about the product being tested, the client, and the end-users of similar products.

2. Design the Test Strategy
   Project objectives and how to achieve them. The amount of effort and cost required for testing. It has detailed documentation of Scope of Testing, Type of Testing, Risks and Issues, Test Logistics.

3. Define the Test Objectives
   This phase defines the goals and expected results of test execution.

4. Define Test Criteria
   Test Criteria refers to standards or rules governing all activities in a testing project.

   Tree types of criteria : Entry Criteria , Suspension Criteria, Exit Criteria.

5. Resource Planning
   This phase creates a detailed breakdown of all resources required for project completion. Resources include human effort, equipment, and all infrastructure needed for accurate and comprehensive testing.
   This part of test planning decides the project's required measure of resources (number of testers and equipment). This also helps test managers formulate a correctly calculated schedule and estimation for the project.

6. Plan Test Environment
   The test environment refers to the software and hardware setup on which QAs run their tests. Ideally, test environments should be real devices so testers can monitor software behavior in real user conditions.

7. Schedule & Estimation
   For test estimation, break down the project into smaller tasks and allocate the time and effort required for each.
   Then, create a schedule to complete these tasks in the designated time with a specific amount of effort.

Creating the schedule, however, does require input from multiple perspectives like Employee availability, number of working days, project deadlines, and daily resource availability.
Risks associated with the project which has been evaluated in an earlier stage.

8. Determine Test Deliverables
   Test Deliverables refer to a list of documents, tools, and other equipment that must be created, provided, and maintained to support testing activities in a project.

   Deliverables required before testing: Test Plan , Test Design

   Deliverables required during testing: Test Scripts, Simulators or Emulators (in early stages), Test Data, Error and execution logs

   Deliverables required after testing: Test Results, Defect Reports, Release Notes

What is priority?

Priority is a parameter that changes with time. Priority of fixing a Defect / bug is generally depends on its effect if its low it is delayed, if it is high it should have high priority to fix it.

It depends on Scheduling.

What is severity?

Severity is a parameter that does not changes with time. Severity of fixing a Defect / bug is generally depends on its effect on the overall system, if its low it is delayed, if it is high it should have high priority to fix it first.

Bug categories are…

Bugs by nature of testing there are 4 types:

1. Functional defects
   Functional bugs can be revealed during smoke, system, integration, regression, and user acceptance testing. A feasible share of test automation during functional testing is the key to making the process cost- and time-effective.

2. Performance defects
   Performance bugs are found during stress, load, stability, and scalability testing. This kind of testing is fully automated. The most popular performance testing tools are Apache JMeter and LoadRunner.

3. Usability defects
   Usability defects are revealed during usability testing, UX audit, or UX research. ScienceSoft applies expert-based (e.g., heuristic evaluation and cognitive walkthrough) and user-based

(interviews and surveys of the TA, executing scenarios by members of the TA) techniques to promptly detect usability issues.

4. Security defects
   Detecting security defects requires vulnerability assessment, penetration testing, security code review, and more. Software compliance assessment (e.g., for HIPAA, PCI DSS, GDPR) can be considered a part of security testing as well. Security testing can be both manual and automated, depending on the needs of each specific project.

Bugs by Severity and Priority 2 types:

1. Defects by severity
   To determine bug severity, test engineers consider how strongly it impacts the software functionality, performance, usability, etc. and how frequently it occurs. According to this classification, bugs can be critical, high-, medium-, and low-severity.

2. Defects by priority
   To prioritize bugs, test engineers look at the business impact of a defect (including the number of users affected, the threat to the company's image, the resulting business disruptions, losses, etc.). Thus, bugs can be classified as urgent, high-, medium-, and low-priority.

Advantage of Bugzila .

- Open source, free bug tracking tool.
- Automatic Duplicate Bug Detection.
- Search option with advanced features.
- File/Modify Bugs By Email.
- Move Bugs Between Installs.
- Multiple Authentication Methods (LDAP, Apache server).
- Time Tracking.
- Automated bug reporting; has an API to interact with system.
- Integrated email capabilities.
- Detailed permissions system.
- Optimized database structure to enhance performance.
- Robust security.
- Powerful query tool.
- Ideal for small projects.

What are the different Methodologies in Agile Development Model?

1. **Kanban**
   Kanban is a simple, visual means of managing projects that enables teams to see the progress so far and what's coming up next.

   Kanban projects are primarily managed through a Kanban board, which segments tasks into three columns: "To Do," "Doing," and "Done."

2. **Scrum**

   One of the most popular agile methodology examples is the agile scrum development methodology, which is depicted by various cycles of development. Similar to Kanban, Scrum breaks down the development phases into stages or cycles called 'sprints'. The development time for each sprint is maximized and dedicated, thereby managing only one sprint at a time.

   Scrum and agile methodologies focus on continuous deliverables, and thus this method lets designers adjust priorities to ensure that any incomplete or overdue sprints get more attention.

   Scrum Team has exclusive project roles such as a scrum master and a product owner with constant communications on the daily scrum where the activities are harmonized to devise the best way to implement the sprint.

3. **Extreme Programming (XP)**
   Extreme Programming (XP) is a methodology that emphasizes teamwork, communication, and feedback. It focuses on constant development and customer satisfaction. Similar to scrum, this method also uses sprints or short development cycles. This is developed by a team to create a productive and highly efficient environment.

   The extreme Programming technique is very supportive in a situation of constant and varying demands from the customers. It motivates the developers to accept changes in the customer's demands, even if they pop up in an advanced phase of the development process.

   In Extreme Programming, the project is tested from the initial stages by collecting feedback that progresses the output of the system. This also presents a spot check to implement easily any customer requirements.

4. **Crystal**
   Introduced by Mr Alistair Cockburn, one of the monumental persons in formulating the Agile manifesto for software development, Crystal is a group of smaller agile development methodologies comprising Crystal Yellow, Crystal Clear, Crystal Red, Crystal Orange, and more. Each has its peculiar and exclusive framework that is characterized by factors such as system criticality, team size, and project priorities. Depending on the nature of the project or system criticality such as Comfort (C), Essential Money (E), Discretionary Money (D), and Life (L), the kind of crystal agile methodology is chosen.

Similar to other methodologies of Agile, Crystal also addresses prompt delivery of software, regularity, less administration with high involvement of users, and customer satisfaction. The Crystal family advocates that each system or project is inimitable and necessitates the solicitation of diverse practices, processes, and policies to achieve the best results, earning the name of the most lightweight methods of agile methodology.

5. **Dynamic Systems Development Method (DSDM)**
   To address the need for a standard industry charter for the swift delivery of software, the Dynamic Systems Development Method (DSDM) was developed. DSDM gives a comprehensive structure that is defined and modified to create a plan, execute, manage, and scale the procedure of software development. Based on a business-driven approach and eight principles, the DSDM believes that modifications to the project are always expected, and quality with timely delivery must never be negotiated.

6. **Feature Driven Development (FDD)**
   Several industry-recognized best practices are incorporated into this iterative, customer-centric, and incremental agile method. Its primary goal is to consistently produce working software in a timely fashion.
   Lifecycle stages include developing an overarching model of the project; creating feature lists; planning by feature; designing by feature; and finally building by feature. Using this five-step process, large project teams will be able to move their products forward at a steady pace.

7. **Lean Software Development**
   Lean development is the application of Lean principles to software development.

   Lean principles got their start in manufacturing, as a way to optimize the production line to minimize waste and maximize value to the customer.

   It Follows a repeatable process, Requires particular quality standards, Relies on the collaboration of a group of specialized workers.

8. **Scaled Agile Framework (SAFe)**
   A set of workflow and organizational patterns for implementing agile practices at an enterprise scale is known as the Scaled Agile Framework (SAFe). Adopting SAFe allows you to take advantage of a framework that is relatively light while still maintaining the centralized decision-making required at the enterprise level for software development efficiency. To put it another way, SAFe takes the agile philosophy and applies it to software executives who are tasked with addressing more strategic issues.

When to used Usablity Testing?

Usability testing is a technique used to evaluate how easily and efficiently people can use a product, such as a website, software application, or physical product. It is typically conducted with a group of users who perform specific tasks and provide feedback on their experience.

Usability testing can be used at different stages of product development, including:

1. Conceptualization:
   Before you start building a product, usability testing can help you gather insights into what users want and need. You can conduct interviews or focus groups to understand user requirements and preferences.

2. Design
   During the design phase, usability testing can help you validate your design choices and identify any usability issues early on. You can create mockups or prototypes of the product and ask users to perform tasks and provide feedback.

3. Development
   Once the product is developed, usability testing can help you identify any bugs or issues that may affect usability. You can conduct testing on a small group of users before releasing the product to a larger audience.

4. Post-launch
   Even after the product has been launched, usability testing can help you gather feedback and improve the user experience. You can conduct surveys or user testing to identify areas for improvement.

Usability testing can be used at any stage of product development to improve the user experience and ensure that the product meets the needs of its users

What are the common problems faced in Web testing?

1. **Integration:**
   The challenging factors associated with this are inconsistency in the environment, infrastructure, interaction model used, performance and reliability issues etc. The Numerous Application Usage Paths, Intranet versus Internet based Applications used by end users will lead to the inconsistency issues.

2. **Interoperability:**
   Diverse information paths, smooth data retrieval and operating system compatibility are the most critical challenging factors associated with interoperability testing. The end users may use different types of browsers to access the application. Even on similar browsers, application may be rendered differently based on the Screen resolution/Hardware/Software Configuration. These are some main interoperability issues.

3. **Security:**
   Security testing checks how well the system saves itself from security attacks and how the confidentiality of the data is maintained with the site. The main challenges are with the cross site scripting, executing the malicious file, insecure communications, injection flaws, use of different authentication procedures for a single application with myriad of services etc.

4. **Performance:**
   Performance testing checks how fast a given input can be converted into an output. The main challenges faced in performance testing of web applications are on situations when large applications have to be tested with minimal hardware support, capability of the system to handle bulk data under various test bottleneck criterions.

5. **Usability:**
   Web-based applications present usability challenges that we don't often see in other types of designs. These challenges include scalability, interactivity, comprehension, issues that arise when frequent changes are made to the application etc.

What is the procedure for GUI Testing?

In GUI Testing aspects like Visual Design, Functionality, Security, Compliance, Usability, Performance

Is tested.

It extensively checks the user-interface of the application under test.

- Testing the size, position, height, width of the visual elements
- Verifying and testing the error messages are displayed or not
- Testing different sections of the display screen
- Verifying the usability of carousel arrows
- Checking the navigation elements at the top of the page
- Checking the message displayed, frequency and content
- Verifying the functionality of proper filters and ability to retrieve results.
- Checking alignment of radio buttons, drop downs
- Verifying the title of each section and their correctness
- Cross-checking the colors and its synchronization with the theme

You can even derive test cases for GUI in two ways:

1. Charts:
   The charts display the state of the system and check the state after some input.
2. Decision Tables:
   This helps in deciding the results for each input

There are three approaches available:

1. Manual Testing

   This approach involves human tester, where each screen is manually checked to validate each functionality by creating and executing test cases. It is a useful approach when part of UI or a feature is ready, the probability of defects is more at the initial stage, and human intervention is required.

2. Record and Replay Testing

   GUI record and replay tools are used to test applications for their user interface. Using such tools, testers run an application and record the user interaction with the app. A script runs to track and save the user actions, including cursor movements, which can be replayed several times to find the issues in the interface.

3. Model-based testing

   In this type of GUI testing, a model is created to understand and evaluate the system's behavior. This approach is useful in creating accurate test cases using system requirements. It is a structured, thorough, measurable form of testing.