**Manual Testing**

1)Why testing is required?

Testing is necessary because we all make mistakes. Some of those mistakes are unimportant, but some of them are expensive or dangerous. We need to check  everything and anything we produce because things can always go wrong – [**humans make mistakes all the time**](http://istqbexamcertification.com/when-do-defects-in-software-testing-arise/).

Since we assume that our work may have mistakes, hence we all need to check our own work. However some mistakes come from bad assumptions and blind spots, so we might make the same mistakes when we check our own work as we made when we did it. So we may not notice the flaws in what we have done.

2)Types of testing?

**Black box testing** – Internal system design is not considered in this type of testing. Tests are based on requirements and functionality.

**White box testing** – This testing is based on knowledge of the internal logic of an application’s code. Also known as Glass box Testing. Internal software and code working should be known for this type of testing. Tests are based on coverage of code statements, branches, paths, conditions.

**Unit testing** – Testing of individual software components or modules. Typically done by the programmer and not by testers, as it requires detailed knowledge of the internal program design and code. may require developing test driver modules or test harnesses.

**Incremental integration testing** – Bottom up approach for testing i.e continuous testing of an application as new functionality is added; Application functionality and modules should be independent enough to test separately. done by programmers or by testers.

**Integration testing** – Testing of integrated modules to verify combined functionality after integration. Modules are typically code modules, individual applications, client and server applications on a network, etc. This type of testing is especially relevant to client/server and distributed systems.

**Functional testing** – This type of testing ignores the internal parts and focus on the output is as per requirement or not. Black-box type testing geared to functional requirements of an application.

**System testing** – Entire system is tested as per the requirements. Black-box type testing that is based on overall requirements specifications, covers all combined parts of a system.

**End-to-end testing** – Similar to system testing, involves testing of a complete application environment in a situation that mimics real-world use, such as interacting with a database, using network communications, or interacting with other hardware, applications, or systems if appropriate.

**Sanity testing**– Testing to determine if a new software version is performing well enough to accept it for a major testing effort. If application is crashing for initial use then system is not stable enough for further testing and build or application is assigned to fix.

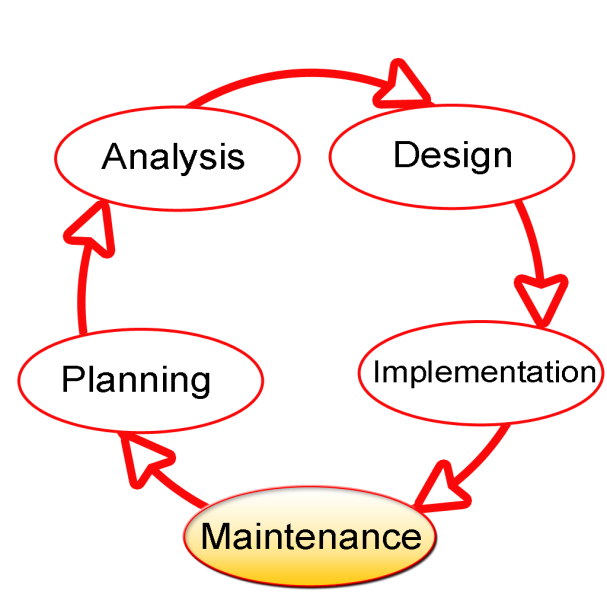
**Regression testing** – Testing the application as a whole for the modification in any module or functionality. Difficult to cover all the system in regression testing so typically automation tools are used for these testing types.

**Acceptance testing** -Normally this type of testing is done to verify if system meets the customer specified requirements. User or customer do this testing to determine whether to accept application.

**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.

3)what is sdlc and differen phaces in sdlc?

The **systems development life cycle** (**SDLC**), also referred to as the **application development life-cycle**, is a term used in [systems engineering](https://en.wikipedia.org/wiki/Systems_engineering), [information systems](https://en.wikipedia.org/wiki/Information_system) and [software engineering](https://en.wikipedia.org/wiki/Software_engineering) to describe a process for planning, creating, testing, and deploying an information system.[[1]](https://en.wikipedia.org/wiki/Systems_development_life_cycle#cite_note-1) The systems development life-cycle concept applies to a range of hardware and software configurations, as a system can be composed of hardware only, software only, or a combination of both



3.**Waterfall** model:

The **waterfall model** is a [sequential](https://en.wikipedia.org/wiki/Sequence) (non-iterative) [design](https://en.wikipedia.org/wiki/Design) process, used in [software development processes](https://en.wikipedia.org/wiki/Software_development_process), in which progress is seen as flowing steadily downwards (like a [waterfall](https://en.wikipedia.org/wiki/Waterfall)) through the phases of conception, initiation, [analysis](https://en.wikipedia.org/wiki/Analysis), [design](https://en.wikipedia.org/wiki/Software_design), construction, [testing](https://en.wikipedia.org/wiki/Software_testing), [production/implementation](https://en.wikipedia.org/wiki/Implementation) and [maintenance](https://en.wikipedia.org/wiki/Software_maintenance). Despite the development of new software development process models, the waterfall method is still the dominant process model with over a third of software developers still using it.[[1]](https://en.wikipedia.org/wiki/Waterfall_model#cite_note-1)

The waterfall development model originates in the [manufacturing](https://en.wikipedia.org/wiki/Manufacturing) and [construction](https://en.wikipedia.org/wiki/Construction) industries: highly structured physical environments in which after-the-fact changes are prohibitively costly, if not impossible. Because no formal software development methodologies existed at the time, this hardware-oriented model was simply adapted for software development.

4.what is **agile** method?

**Agile software development** describes a set of principles for [software development](https://en.wikipedia.org/wiki/Software_development) under which requirements and solutions evolve through the collaborative effort of self-organizing [cross-functional teams](https://en.wikipedia.org/wiki/Cross-functional_team).[[1]](https://en.wikipedia.org/wiki/Agile_software_development#cite_note-Collier_2011-1) It advocates adaptive planning, evolutionary development, early delivery, and continuous improvement, and it encourages rapid and flexible response to change.[[2]](https://en.wikipedia.org/wiki/Agile_software_development#cite_note-2) These principles support the definition and continuing evolution of many [software development methods](https://en.wikipedia.org/wiki/Software_development_methodologies)

5)What is scrum method?

Scrum is a management framework for incremental product development using one or more cross-functional, self-organizing teams of about seven people each. It provides a structure of roles, meetings, rules, and artifacts. Teams are responsible for creating and adapting their processes within this framework. Scrum uses fixed-length iterations, called Sprints. Sprints are no more than 30 days long, preferably shorter. Scrum teams try to build a potentially releasable (properly tested) product increment every Sprint

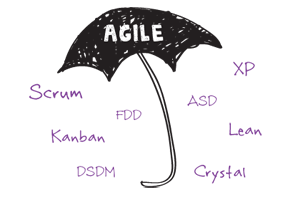
or

**Scrum** is an iterative and incremental [agile software development](https://en.wikipedia.org/wiki/Agile_software_development) framework for managing product development.[[1]](https://en.wikipedia.org/wiki/Scrum_(software_development)#cite_note-scrumalliance-1)[[2]](https://en.wikipedia.org/wiki/Scrum_(software_development)#cite_note-gunther-2) It defines "a flexible, [holistic](https://en.wikipedia.org/wiki/Holism) product development strategy where a development team works as a unit to reach a common goal",[[3]](https://en.wikipedia.org/wiki/Scrum_(software_development)#cite_note-TakeuchiNonaka-3) challenges assumptions of the "traditional, sequential approach"[[3]](https://en.wikipedia.org/wiki/Scrum_(software_development)#cite_note-TakeuchiNonaka-3) to product development, and enables teams to self-organize by encouraging physical [co-location](https://en.wikipedia.org/wiki/Colocation_(business)) or close online collaboration of all team members, as well as daily face-to-face communication among all team members and disciplines involved.

A key principle of Scrum is its recognition that during [product development](https://en.wikipedia.org/wiki/New_product_development), the customers can change their minds about what they want and need (often called requirements volatility[[4]](https://en.wikipedia.org/wiki/Scrum_(software_development)" \l "cite_note-4)), and that unpredicted challenges cannot be easily addressed in a traditional predictive or planned manner. As such, Scrum adopts an evidence-based [empirical approach](https://en.wikipedia.org/wiki/Empirical_process_(process_control_model))—accepting that the problem cannot be fully understood or defined, focusing instead on maximizing the team's ability to deliver quickly, to respond to emerging requirements and to adapt to evolving technologies and changes in market conditions.

6) what is the process of agile methodology

The various [agile Scrum](https://www.versionone.com/Agile101/Agile-Scrum-Velocity/) methodologies share much of the same philosophy, as well as many of the same characteristics and practices. But from an implementation standpoint, each has its own recipe of practices, terminology, and tactics. Here we have summarized a few of the main agile software development methodology contenders:



## Agile Scrum Methodology

[Scrum](https://www.versionone.com/agile-101/what-is-scrum/) is a lightweight agile project management framework with broad applicability for managing and controlling iterative and incremental projects of all types. Ken Schwaber, Mike Beedle, Jeff Sutherland and others have contributed significantly to the evolution of Scrum over the last decade. Scrum has garnered increasing popularity in the agile software development community due to its simplicity, proven productivity, and ability to act as a wrapper for various engineering practices promoted by other agile methodologies.

## 7) Product Backlog

The Product Backlog is dynamically changing and improving; it is never complete. We do not wait until it is complete before we start delivering the items. The first Sprint can be started as soon as the Product Backlog has enough stories defined.

The Product Backlog items will then be ordered based on their business value, in a way that the higher an item is, the sooner it will be delivered by the Development Team. As the items located at the top of the Product Backlog will be delivered earlier, they will also be more detailed and clearer when compared to the lower items.

## 8) Sprint Backlog

The Sprint Backlog is created during the Sprint Planning event, which is the first event in a Sprint. During the Sprint Planning event, the Scrum Team collaborates on creating the Sprint Backlog, which consists of the following:

* A number of items selected from the top of the Product Backlog, based on their estimated work and the estimated capacity of the Development Team;
* The Sprint Goal, which will help describe the real meaning of the items and direct the efforts of the Development Team.
* A detailed plan for delivery of the items and realization of the Sprint Goal during the Sprint. This detailed plan will continue to be updated during the Sprint.

8) what is product back log items

Ans: **Product Backlog Item**. In Scrum, a **product backlog item** is a unit of work small enough to be completed by a team in one Sprint iteration. **Backlog items** are decomposed into one or more tasks.

9) what is sprint planing meeting

Ans: The sprint planning meeting is attended by the product owner, ScrumMaster and the entire Scrum team. Outside stakeholders may attend by invitation of the team, although this is rare in most companies.During the sprint planning meeting, the product owner describes the highest priority features to the team. The team asks enough questions that they can turn a high-level user story of the product backlog into the more detailed tasks of the sprint backlog.

10) what is sprint review meeting

Ans: In Scrum, each **sprint** is required to deliver a potentially shippable product increment. This means that at the end of each **sprint**, the team has produced a coded, tested and usable piece of software. So at the end of each **sprint**, a **sprint review meeting**is held.

11) what is sprint retrospective

Ans: The **sprint retrospective** is a continuous improvement opportunity for a **Scrum** team to review its process (approaches to performing **Scrum**) and to identify opportunities to improve it. See also inspect and adapt.

12) what is sprint grooming

Ans: Product backlog refinement—sometimes called product backlog **grooming** in reference to keeping the backlog clean and orderly—is a meeting that is held near the end of one **sprint** to ensure the backlog is ready for the next **sprint**

13) what is burndown chart and velocity

Ans: The Scrum Burndown Chart is a visual measurement tool that shows the completed work per day against the projected rate of completion for the current project release. Its purpose is to enable that the project is on the track to deliver the expected solution within the desired schedule.

The rate of progress of a Scrum Team is called "velocity". It expresses the amount of e.g. story points completed per iteration. An import rule for calculating the velocity is that only stories that are completed at the end of the iteration are counted

14) what is user acceptance criteria test cases

Ans:User acceptance testing (UAT) is the last phase of the software testing process. During UAT, actual software users test the software to make sure it can handle required tasks in real-world scenarios, according to specifications. UAT is one of the final and critical software project procedures that must occur before newly developed software is rolled out to the market.

15) what is v model?

Ans:The **V** - **model** is SDLC **model** where execution of processes happens in a sequential manner in **V**-shape. It is also known as Verification and Validation**model**. **V** - **Model** is an extension of the waterfall**model** and is based on association of a testing phase for each corresponding development stage.

16) what is STLC?

Ans: The **process** of testing a software in a well planned and systematic way is known as **software testing life cycle**(**STLC**). Requirements gathering Requirements Analysis is done is this phase, software requirements are reviewed by test team. Design Test Planning, Test analysis and Test design is done in this phase.

17) what is defect?

Ans: A programmer while designing and building the **software** can make mistakes or error. These mistakes or errors mean that there are flaws in the **software**. These are called **defects**. When actual result deviates from the expected result while **testing** a**software** application or product then it results into a **defect**

**18)** **how to arise a defect and what we specify while logging defect?**

**Ans:** Defect logging, a process of finding defects in the application under test or product by testing or recording feedback from customers and making new versions of the product that fix the defects or the clients feedback.Defect tracking is an important process in software engineering as Complex and business critical systems have hundreds of defects. One of the challenging factors is Managing, evaluating and prioritizing these defects. The number of defects gets multiplied over a period of time and to effectively manage them, defect tracking system is used to make the job easier.

**19)** **defect lifecycle**

**Ans:** Defect life cycle, also known as Bug Life cycle is the journey of a defect cycle, which a defect goes through during its lifetime. It varies from organization to organization and also from project to project as it is governed by the software testing process and also depends upon the tools used.

**20)** **What is unit testing?**

**Ans: Unit testing** is a software **testing** method by which individual units of source code, such as functions, methods, and class are tested to determine whether they are fit for use. Intuitively, one can view a **unit** as the smallest testable part of an application.

**21)** **when do we use regression testing?**

**Ans:** Regression testing a black box testing technique that consists of re-executing those tests that are impacted by the code changes. These tests should be executed as often as possible throughout the software development life cycle.

**22)** What is integration testing?

**Ans: Integration testing** (sometimes called **integration** and **testing**, abbreviated I&T) is the phase in **software testing** in which individual **software** modules are combined and **tested** as a group. It occurs after unit **testing** and before validation **testing**.

23) when do we use integration testing?

Ans: There are two major ways of carrying out an integration test, called the bottom-up method and the top-down method. Bottom-up integration testing begins with [unit testing](http://searchsoftwarequality.techtarget.com/definition/unit-testing), followed by tests of of progressively higher-level combinations of units called modules or builds. In top-down integration testing, the highest-level modules are tested first and progressively lower-level modules are tested after that. In a comprehensive software development environment, bottom-up testing is usually done first, followed by top-down testing. The process concludes with multiple tests of the complete application, preferably in scenarios designed to mimic those it will encounter in customers' computers, systems and [network](http://searchnetworking.techtarget.com/definition/network)s.

24) when do we use smoke testing and sanity testing?

Ans: Smoke Testing is a kind of Software Testing performed after software build to ascertain that the critical functionalities of the program is working fine. It is executed "before" any detailed functional or regression tests are executed on the software build. The purpose is to reject a badly broken application, so that the QA team does not waste time installing and testing the software application.In Smoke Testing, the test cases chosen cover the most important functionality or component of the system. The objective is not to perform exhaustive testing, but to verify that the critical functionalities of the system is working fine.

Sanity testing is a kind of Software Testing performed after receiving a software build, with minor changes in code, or functionality, to ascertain that the bugs have been fixed and no further issues are introduced due to these changes. The goal is to determine that the proposed functionality works roughly as expected. If sanity test fails, the build is rejected to save the time and costs involved in a more rigorous testing

25) what is unit testing?

Ans: **Unit testing** is a software **testing** method by which individual units of source code, such as functions, methods, and class are tested to determine whether they are fit for use. Intuitively, one can view a **unit** as the smallest testable part of an application.

26) what is UAT?

Ans: In software development, user acceptance testing (UAT) - also called **beta** testing,**application** testing, and end user testing - is a phase of software development in which the software is tested in the "real world" by the intended audience.

27) what is alpha and beta testing

Ans: Alpha testing performed by Testers  who are usually internal employees of the organization. Alpha Testing performed at developer's site

Beta testing is performed by Clients or End Users who are not employees of the organization. Beta testing is performed at client location or end user of the product

28) when do we use white box testing and block box testing?

Ans: White box testing is a testing technique, that examines the program structure and derives test data from the program logic/code. The other names of glass box testing are clear box testing, open box testing, logic driven testing or path driven testing or structural testing.

Black-box testing is a method of software testing that examines the functionality of an application based on the specifications. It is also known as Specifications based testing. Independent Testing Team usually performs this type of testing during the software testing life cycle.

31) when do we use automation testing?

Ans: Test engineers strive to catch them before the product is released but they always creep in and they often reappear, even with the best manual **testing processes**. Test Automation software is the best way to increase the effectiveness, efficiency and coverage of your software testing

32) what tester will do in each phase of SDLC?

Ans: Requirements gathering and Analysis:All types of estimation and examination of user needs are done in this phase.

System Design:In the second phase a basic system planning is done. After collecting the all statistics and data, a system design is done.

Implementation:In the next phase implementation of project is done. Respect to the system design, correct development is made to expand that design. According to the project programming language will be chosen.

System Testing:After the implementation phase, system testing phase take place to recognize the result of application. Testing is done to recognize the original result and the predictable result.

Operation Maintenance:It is the ultimate phase of SDLC, where the application which is implemented is spread to users who are answerable for conserving and using it for appropriate actions. The implemented application should be available for any adjustment.

33) difference between load and performance testing?

Ans: Performance testing, a non-functional testing technique performed to determine the system parameters in terms of responsiveness and stability under various workload. Performance testing measures the quality attributes of the system, such as scalability, reliability and resource usage..

Load testing is performance testing technique using which the response of the system is measured under various load conditions. The load testing is performed for normal and peak load conditions.

34) different types of non-functional testing types?

Ans: Non-Functional testing is a software testing technique that verifies the attributes of the system such as memory leaks, performance or robustness of the system. Non-Functional testing is performed at all test levels.

Baseline testing

Compatibility testing

Compliance testing

Endurance testing

Load testing

Localization testing

Internationalization testing

Performance testing

Recovery testing

Resilience testing

Security testing

Scalability testing

Stress testing

35) what is test case?

Ans: A **test case** is a document, which has a set of **test** data, preconditions, expected results and postconditions, developed for a particular **test** scenario in order to verify compliance against a specific requirement.

36) what is test planning/test strategy document

Ans: Test planning, the most important activity to ensure that there is initially a list of tasks and milestones in a baseline plan to track the progress of the project. It also defines the size of the test effort.It is the main document often called as master test plan or a project test plan and usually developed during the early phase of the project.

Test Strategy is also known as test approach defines how testing would be carried out. Test approach has two techniques:

**Proactive -**An approach in which the test design process is initiated as early as possible in order to find and fix the defects before the build is created.

**Reactive -**An approach in which the testing is not started until after design and coding are completed

37) what is Exit and Entry criteria

Ans: Entry criterion is used to determine when a given test activity should start. It also includes the beginning of a level of testing, when test design or when test execution is ready to start.

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.

38) what is TDD and BDD (cucumber framework)

Ans: **Cucumber** is a testing tool that supports **Behavior Driven Development** (**BDD**)**framework**. It defines application behavior using simple English text, defined by a language called Gherkin. **Cucumber** allows automation functional validation that is easily read and understood.

TDD is a Software Development methodology in which first test cases are written in the form of stories and then allowed to fail. Then the developers write code to pass the testcases and thus implementing the story or requirement.

39) how do we write test cases in BDD format

Ans:

40) what is priority and severity in defect?

Ans: **Defect Priority** (Bug Priority) indicates the importance or urgency of fixing a defect. Though priority may be initially set by the Software Tester, it is usually finalized by the Project/Product Manager.

**Defect Severity** or Impact is a classification of **software defect** (bug) to indicate the degree of negative impact on the quality of **software**. ISTQB Definition. **severity**: The degree of impact that a **defect** has on the development or operation of a component or system.

**41)** **how to estimate test cases?**

**Ans:** In order to be a successful in estimating, the software test project and proper execution are significant as the software development life cycle. Software testing estimation techniques play a very important role in making the good reputation with the client while bidding the project for testing.One of the most important factors while estimating testing efforts is the experience on varied projects for the software test life cycle. Apparently one cannot just put some number of days for any task or taking the old time formula of one third of the development effort. This is one of the most widely used estimation technique by the companies offering software testing services. It is merely due to the fact that this method is not based on any scientific principle or technique

42) what are test design techniques

Ans:  **Test Design** is creating a set of inputs for given software that will provide a set of expected outputs. ... Broadly speaking there are two main categories of **Test Design Techniques**.

43) if we dont have time to test call test cases what we will do

Ans: If we have enough time to test the application then it is not a problem at all. But if there isn’t enough time for through testing of application, in this situation it won’t possible to test each & every combination of scenario. The Risk analysis is playing vital role in Software Testing, we recommend that you should use risk analysis to determine where testing should be focused.

44) how we learn the functionality of system?

Ans: It provides a platform (hardware abstraction layer) to run high-level **system software**and application **software**. ... Device drivers, including also computer BIOS and device firmware, provide basic **functionality** to operate and control the hardware connected to or built into the computer.

45) who will assign the work?

Ans:

46) what is requirement traceability matrix

Ans: Requirements tracing, a process of documenting the links between the requirements and the work products developed to implement and verify those requirements. The RTM captures all requirements and their traceability in a single document delivered at the conclusion of the life cycle.

47) what are typical environments we have in projects

Ans: **typical** set of software you need for everyday \*manual\* testing tasks? **We** could include these in our deployment so that user could save more time for testing, not for **environment** prep & tuning. Sure

48) what is development environment

Ans: In computer program and **software** product **development**, the **development environment** is the set of processes and programming tools used to create the program or **software** product. The term may sometimes also imply the physical**environment**.

49) what is QA environment

Ans: Development, Test, **QA**, and Production **Environments**. A **QA environment** is where you test your upgrade procedure against data, hardware, and software that closely simulate the Production **environment** and where you allow intended users to test the resulting Wave set application.

50) What is staging environment

Ans: A stage or **staging environment** is an **environment** for **testing** that exactly resembles the production **environment**. In other words, it's a complete but independent copy of the production **environment**, including the database. **Staging** provides a true basis for QA **testing** because it precisely reproduces what is in production

51) what is production environment

Ans: **Production environment** is a term used mostly by developers to describe the setting where software and other products are actually put into operation for their intended uses by end users.

52) how to deal the production defects?

Ans: major responsibilities were performing function testing, executing regression suites, running smoke tests etc.

53) Types of testing?

**Ans:**

**Black box testing** – Internal system design is not considered in this type of testing. Tests are based on requirements and functionality.

**White box testing** – This testing is based on knowledge of the internal logic of an application’s code. Also known as Glass box Testing. Internal software and code working should be known for this type of testing. Tests are based on coverage of code statements, branches, paths, conditions.

**Unit testing** – Testing of individual software components or modules. Typically done by the programmer and not by testers, as it requires detailed knowledge of the internal program design and code. may require developing test driver modules or test harnesses.

**Incremental integration testing** – Bottom up approach for testing i.e continuous testing of an application as new functionality is added; Application functionality and modules should be independent enough to test separately. done by programmers or by testers.

**Integration testing** – Testing of integrated modules to verify combined functionality after integration. Modules are typically code modules, individual applications, client and server applications on a network, etc. This type of testing is especially relevant to client/server and distributed systems.

**Functional testing** – This type of testing ignores the internal parts and focus on the output is as per requirement or not. Black-box type testing geared to functional requirements of an application.

**System testing** – Entire system is tested as per the requirements. Black-box type testing that is based on overall requirements specifications, covers all combined parts of a system.

**End-to-end testing** – Similar to system testing, involves testing of a complete application environment in a situation that mimics real-world use, such as interacting with a database, using network communications, or interacting with other hardware, applications, or systems if appropriate.

**Sanity testing**– Testing to determine if a new software version is performing well enough to accept it for a major testing effort. If application is crashing for initial use then system is not stable enough for further testing and build or application is assigned to fix.

**Regression testing** – Testing the application as a whole for the modification in any module or functionality. Difficult to cover all the system in regression testing so typically automation tools are used for these testing types.

**Acceptance testing** -Normally this type of testing is done to verify if system meets the customer specified requirements. User or customer do this testing to determine whether to accept application.

**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.