Modul 2 Assignment

- ## Errors, Defects and Failures

Error	**a human action that produces an incorrect result can manifest as**
defect	**A flaw in a component or system that cause the component or system to fail the component or system to fail to**
defect	**Deviation of the component or From its expected delivery, service, or result**

- "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"

- 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.

➢ Quality: -

Quality - ‘The degree to which a component, system or process meets specified requirements and/or user/customer needs and expectations’

Key aspects of quality for the customer include:

- ➢ Good design – looks and style
- ➢ Good functionality – it does the job well
- ➢ Reliable – acceptable level of breakdowns or failure
  Consistency
- ➢ Durable – lasts as long as it should
- ➢ Good after sales service
- ➢ Value for money

➢ **Risk**: -

Risk – ‘A factor that could result in future negative

➢ Who does Testing?

- Software Tester
- Project Lead/Manager
- End User

basis of their experience and knowledge such as software Tester, Software Quality Assurance Engineer, and QA Analyst ect

❖ What do Tester?

- Automation
- Performance
- Usability
- Security

Or alternatively may work more generally doing

- Test Analysis
- Test Design
- Test Execution
- Test Environment management

❖ Role of Software Tester

- ➢ Speed up development process by identifying bugs at an early stage
- ➢ Reduce the organization's risk of legal liability
- ➢ Maximize the value of the software
- ➢ Assure successful launch of the product, save money, time and reputation of the company by discovering bugs and design flaws at an early stage before failures occurs in production, or in the field
- ➢ Promote continual improvement

❖ Testing v/s Debugging

The responsibility for each activity is very different, i.e.

Tester’s test

Developers debug

● Testing

- ➢ It involves the identification of bug/error/defect in the software without correcting it.
- ➢ Normally professionals with a Quality Assurance background are involved in the identification of bugs. Testing is performed in the testing phase.
- ➢ Testing can show failures that are caused by defects

● Debugging

- ➢ It involves identifying, isolating and fixing the problems/bug. Developers who code the software conduct

debugging upon encountering an
error in the code.

- Debugging is the part of White box
or Unit Testing.
Debugging can be performed in the
development phase while
conducting Unit Testing or in
phases while fixing the reported bugs
- Debugging identifies the cause of a
defect, repairs the code and checks
that the defect has been fixed.

❖ Test Process document

- Test Analysis (Cont...)
From testing perspective, we look at the test basis in order to see
what could be tested. These are the test conditions. A test
condition is simply something that we could test.
- While identifying the test conditions we want to identify as many
conditions as we can and then we select about which one to take
forward and combine into test cases. We could call them test
- Test conditions can be identified for test data as well as for test inputs
and test outcomes, for example, different types of record, different
sizes of records or fields in a record.

➢ Q.A vs Q.C AND TESTING

Quality Assurance **Quality Control** **Testing**

1 (Q.A) Activities which ensure the implementation of processes, procedures and standards in context to verification of developed software and intended requirements.

1 (Q.C) Activities which ensure the verification of developed software with respect to documented (or not in some cases) requirements.

1 (testing) Activities which ensure the identification of bugs/error/defects in the Software.

2 (Q.A) Focuses on processes and procedures rather than conducting actual testing on the system.

2 (Q.C) Focuses on actual testing by executing Software with intend to identify bug/defect through implementation of procedures and process.

2 (testing) focuse on actual testing

3(Q.A) Process oriented activities.

3(Q.C) Product oriented activities.

3 (testing) Product oriented activities.

4 (Q.A) Preventive activities.

4 (Q.C) It is a corrective process.

4 (testing) It is a preventive process.

5 (Q.A) It is a subset of Software Test Life Cycle (STLC).

5 (QC) can be considered as the subset of Quality Assurance.

5(Testing) is the subset of Quality Control

➢Functional vs Non-Functional

- **Function testing** is performed using the functional specification provided Functional by the client and verifies the system against the functional requirements.
- **Nonfunctional testing** check the performance reliability, scalability and other non-functional aspects of the software system.
- **Functional testing** is executed first
- **Nonfunctional testing** should be performed after functional testing
- **Function testing** Manual testing or automation tools can be used for functional testing
- **Nonfunctional testing** Using tools will be effective for this testing
- Business requirements are the inputs to **functional testing**
- Performance parameters like speed, scalability are inputs to **non-functional testing**.
- **Functional testing** describes what the product does
- **Nonfunctional testing** describes how good the product works
- Easy to do manual testing / Tough to do manual testing

➢ Functional Testing

➢ **Types of function testing**

➢ **types of non-functional testing**

➢ Smoke testing
➢ Sanity testing
➢ Integration testing
➢ White box testing
➢ Black box testing
➢ User acceptance testing
➢ Regression testing

performance testing
load testing
volume testing
stress testing
security testing
penetration testing
compatibility testing
Migration testing
Installation testing

➢ Functional Testing

➢ Black Box Testing
➢ While Box Testing
➢ Experience Based Testing
➢ Smoke Testing

- Sanity Testing
- End to End Testing

Black box -

- Based on requirement
- From the requirement test are created
- Specification model can be used for systematic test case design

Techniques

- Equivalence partitioning
- Boundary value analysis
- Decision table
- State transition testing
- Use case testing

Equivalence Partitioning (E.P.)

- Aim is to treat groups of inputs as equivalent and to select one

  representative input to test them all
- Selecting input data that is representative of all other data that would

  likely invoke the same process for that particular testing.

Boundary Value Analysis(B.V.A.)

- Boundary value analysis is a methodology for designing test that concentrates software testing effort on cases near the the limits of valid ranges

- Boundary value analysis generates test cases that highlight errors better than equivalence partitioning
- Boundary value analysis is a method which refines equivalence

  Partitioning.

➢ Decision Table

Table based technique where

- Inputs to the system are recorded
- Outputs to the system are defined
- Inputs are usually defined in terms of actions which are Boolean (true and false)

➢ State Transaction Testing

- State Transition: A transition between two states of a component or

  system.

➢ Experience based testing

- In experience-based techniques, people's knowledge, skills and

  background is of prime importance to the test conditions and

- **Techniques –**

  1 grey box

  2 error guessing

  3 exploratory testing

❖ 1 Grey Box Testing

- Grey Box testing is a technique to test the application with limited knowledge of the internal workings of an application.
- In software testing, the term the more you know the better carries a lot of weight when testing an application.
- Having this knowledge, the tester is able to better prepare test Data.

❖ 2 Ad hoc Testing (Error Guessing)

- 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.

- **Types of Adhoc Testing**

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

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. This testing usually happens after unit testing completion.

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.

Monkey Testing

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

3 Exploratory Testing

- Exploratory testing is a concurrent process where
- Test design, execution and logging happen simultaneously
- Testing is often not recorded
- Makes use of experience, heuristics and test patterns

End to End testing

- Modern software systems are complex and are interconnected with multiple sub-systems
- A sub-system may be different from the current system or may be owned by another organization.
- If any one of the sub-system fails, the whole software system could collapse.
- This is major risk and can be avoided by End-to-End testing.
- End-to-End testing verifies the complete system flow. It increase test coverage of various sub-systems.
- It helps detect issues with sub-systems and increases confidence in the overall

STLC – Software Testing Life Cycle

STLC Phases

1. Requirement Analysis
2. Test Planning
3. Test case development
4. Test Environment setup
5. Test Execution
6. Test Cycle closure

What is Entry and Exit Criteria in STLC?

- Entry Criteria: Entry Criteria gives the prerequisite items that must be completed before testing can begin
- Exit Criteria: Exit Criteria defines the items that must be completed before testing can be concluded
- In an Ideal world, you will not enter the next stage until the exit criteria for the previous stage is met. But practically this is not always possible.

Requirement Analysis

- Requirement Phase Testing also known as Requirement Analysis in which test team studies the requirements from a testing point of view to identify testable requirements and the QA team may interact with various stakeholders to understand requirements in detail.

Activities in Requirement Phase Testing

- Identify types of tests to be performed.
- Gather details about testing priorities and focus.
- Prepare Requirement Traceability Matrix (RTM).
- Identify test environment details where testing is supposed to be carried out.
- Automation feasibility analysis (if required).

Deliverables of Requirement Phase Testing

- RTM
- Automation feasibility report. (if applicable)

Test Planning

- 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.

Activities in Requirement Phase Testing

- Preparation of test plan/strategy document for various types of testing
- Test tool selection
- Test effort estimation
- Resource planning and determining roles and responsibilities.
- Training requirement

Deliverables of Requirement Phase Testing

- Test plan /strategy document.
- Effort estimation document.

Test Case Development

The Test Case Development Phase involves the creation, verification and rework of test cases & test scripts after the test plan is ready.

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.

➢ 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.

➢ 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

- ➢ Test cases updated with results

Test Cycle Closure

Test Cycle Closure phase is completion of test execution which involves several activities like test completion reporting, collection of the completion matrices and test results.

Activities in Requirement Phase Testing

- Evaluate cycle completion criteria based on Time, Test coverage, Cost, Software, Critical Business Objectives, Quality
- Prepare test metrics based on the above parameters.
- Document the learning out of the project
- Prepare Test closure report

Deliverables of Requirement Phase Testing

- Test Closure report
- Test metrics

Self-Testing & Independent testing

- Tests by the person who wrote the item.
- Tests by another person within the same team, like another programmer.
- Tests by the person from some different group such as an independent test team -
- Tests by a person from a different organization or company, such as outsourced testing or certification by an external body.

❖ Usability testing

Need For Usability Testing

Aesthetics and design are important. How well a product looks usually determines how well it works.

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 system early in

development

cycle and can save a product from failure.

➢ Usability Testing Example

- ➢ The user should not be able to type in drop-down select lists.
- ➢ All buttons on a page should be accessible by keyboard shortcuts and the user
- ➢ should be able to perform all operations using a keyboard...
- ➢ Page text should be left-justified.

➢ Goal of Usability Testing

Effectiveness of the system

Efficiency

Accuracy

User Friendliness

❖ Pros & Cons Usability Testing

❖ 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

❖ Compatibility testing

Types of Compatibility Testing

- Hardware
- Operating Systems
- Software

- Network
- Browser
- Devices
- Mobile
- Versions of the software
- Backward compatibility Testing
- Forward compatibility testing

❖ 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.

❖ Approach of GUI Testing

➢ MANUAL BASED TESTING

Under this approach, graphical screens are checked manually by testers

in conformance with the requirements stated in business requirements

document.

➢ RECORD AND REPLAY

GUI testing can be done using automation tools. This is done in 2 parts. During

Record, test steps are captured into the automation tool. During playback, the

recorded test steps are executed on the Application under Test. Example of

such tools - QTP.

MODEL BASED TESTING

A model is a graphical description of system's behavior. It helps us understand and predict the system behavior. Models help in a generation of

efficient test cases using the system requirements.

- **Security testing**
  - You need to test how secure your web application is from both external and internal threats.
  - The goal of security testing is to identify the threats in the system and measure its potential vulnerabilities.
  - It also helps in detecting all possible security risks in the system and help developers in fixing these problems through coding.
- **Types of Security Testing**
  - Vulnerability Scanning
  - Security Scanning
  - Risk Assessment
  - Security Auditing
  - Ethical hacking
  - Posture Assessment
- **ROLES YOU MUST KNOW!**
  - Hackers - Access computer system or network without authorization
  - Crackers – Break into the systems to steal or destroy data
  - Ethical Hacker – Performs most of the breaking activities but with permission from Owner
  - Script Kiddies or packet monkeys – Inexperienced Hackers with programming language skill

❖ Security with SDLC

SDLC Phases	Security Processes
Requirements	Security analysis for requirements and check abuse/misuse cases
Design	Security risk analysis for designing. Development of test plan including security tests
Coding and Unit Testing	Static and Dynamic Testing and Security white box testing
Integration Testing	Black Box Testing
System Testing	Black Box Testing and Vulnerability scanning
Implementation	Penetration Testing, Vulnerability Scanning
Support	Impact analysis of Patches

➢ Security Testing Techniques

In security testing, different methodologies are followed, and they are as follows:

- Tiger Box: This hacking is usually done on a laptop which has a collection of OSs and hacking tools. This testing helps penetration testers and security testers to conduct vulnerabilities assessment and attacks.
- Black Box: Tester is authorized to do testing on everything about the network topology and the technology.
- Grey Box: Partial information is given to the tester about the is hybrid of white and black box models.

- ## Security Testing Examples

Web Based Testing & Desktop Based Testing

- Secure pages should use the HTTPS protocol.
- Sensitive fields like passwords and credit card information should not have to autocomplete enabled.
- Verify CAPTCHA functionality.

Mobile Based Testing:

- Every payment gateway app used security code or OTP
- Must be used mobile tracking mode on

Game Based Testing

- Secure your rewards which you get from your game playing
- Without current round completing, you cannot going in next round.

- ### Performance testing

Software performance testing is a means of quality assurance (QA). It involves testing software applications to ensure they will perform well under their expected workload.

The focus of Performance testing is checking a software program

- **Speed** – Determines whether the application responds quickly
- **Scalability** – Determines maximum user load the software application can handle.
- **Stability** – Determines if the application is stable under varying loads

ADD

Types of Performance Testing

- Load testing
- Stress testing
- Endurance

Testing

- Spike testing
- Volume testing
- Scalability testing

Performance Problems

Long Load time –

Load time is normally the initial time it takes an application to start.

This should generally be kept to a minimum.

While some applications are impossible to make load in under a minute, Load time should be kept under a few seconds if possible.

Poor response time –

Response time is the time it takes from when a user inputs data into the application

until the application outputs a response to that input.

Generally, this should be very quick.

Again, if a user has to wait too long, they lose interest.

Poor scalability –

A software product suffers from poor scalability when it cannot handle the expected number of users or when it does not accommodate a wide enough range of users.

Load testing should be done to be certain the application can handle the anticipated number of users.

➢ Performance Problems

- Bottlenecks are obstructions in system which degrade overall system performance.
- Bottlenecking is often caused by one faulty section of code.
- Bottle necking is generally fixed by either fixing poor running processes or adding additional Hardware.

➢ Some common performance bottlenecks are

- CPU utilization
- Memory utilization
- Network utilization
- Operating System limitations
- Disk usage

Performance Parameters

Processor Usage

➢ Bandwidth	Throughput
➢ Private bytes	Amount of connection pooling
➢ Committed memory	Maximum active sessions
➢ Memory pages/second	Hit ratios

- Page faults/second
- CPU interrupts per second
- Disk queue length
- Network output queue length
- Network bytes total per second
- Response time

Hits per second

Rollback segment

Database locks

Top waits

Thread counts

Garbage collection

Performance Testing Examples

- **Web Based Testing & Desktop Based Testing:**

  Check the page load on slow connections.

  Check the database query execution time.

  Check the performance of database stored procedures and triggers.

- **Mobile Based Testing:**

  Check the database query execution time.

  Check the response time for any action under a light, normal, moderate, and

  Check CPU and memory usage under peak load conditions

- **Game Based Testing:**

  Determine whether the current infrastructure is sufficient for the smooth

  Determine the number of users an app can support and its scalability rate

  Accommodates strategies for performance management.

Stress testing

- Stress Testing is done to make sure that the system would not crash under crunch situations.
- Stress testing is also known as endurance testing.

- It even tests beyond the normal operating point and

  evaluates how the system works under those extreme conditions.
- Most prominent use of stress testing is to determine the limit, at

  which the system or software or hardware breaks.

❖ Need For Stress Testing

- To check whether the system works under abnormal conditions.
- Displaying appropriate error message when the system is under stress.
- System failure under extreme conditions could result in enormous revenue loss
- It is better to be prepared for extreme conditions by executing Stress Testing.

❖ Types of Stress Testing

- Application Stress Testing:
- Transactional Stress
- Testing:
- Systemic Stress Testing:
- Exploratory Stress Testing:

Stress Testing

➢ - Stress Tester

- Neo Load
- App Perfect

❖ Application Response

- Hit time: Average time to retrieve an image or a page
- Time to the first byte: Time taken to return the first byte of data or
- Page Time: Time taken to retrieve all the information in a page

Failures

- Failed Connections: Number of failed connections refused by the client
- Failed Rounds: Number of rounds it gets failed
- Failed Hits: Number of failed attempts done by the system

❖ Load testing: -

Load testing - It's a performance testing to check system behavior under load. Testing an application under heavy loads, such as testing of a web site under arrange of loads to determine at what point the system's response time degrades or fails.

- 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.

➢ Need For Load Testing

- Popular toy store Toysrus.com, could not handle the increased traffic generated by their advertising campaign resulting in loss of both marketing dollars, and potential toy sales.
- Facebook (FB)

➢ 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.

Strategies of Load Testing

- Manual Load Testing
- In house (Organization) developed load testing tools
- Open-source load testing tools
- Enterprise (Record and Play) load testing tools

Pros and Cons of Load Testing

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.

Acceptance Testing

User Acceptance Testing

- Acceptance testing: Formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system.
- The goal of acceptance testing is to establish confidence in the system.
- Acceptance testing is most often focused on a validation type testing.
- Usually, Black Box Testing method is used in Acceptance Testing.

Alpha Testing

- It is always performed by the developers at the software development site.
- Sometimes it is also performed by Independent Testing Team.

- It is always performed in Virtual Environment.
- It is the form of Acceptance Testing.
- It comes under the category of both White Box Testing and Black Box Testing.

➢ Beta Testing (Field Testing)

- It is always performed by the customers at their own site.
- It is not performed by Independent Testing Team.
- It is performed in Real Time Environment.
- It is also the form of Acceptance Testing.
- It is only a kind of Black Box Testing.

➢ Maintenance testing

- Maintenance testing: Testing the changes to an operational system or
- the impact of a changed environment to an operational system
- Modification
- software upgrades
- Operating system changes
- system tuning
- emergency fixes
- Software Retirement (may necessitate data archiving tests)

➢ Objectives of Maintenance

- Develop tests to detect problems prior to placing
- Correct problems identified in the live environment
- Test the completeness of needed training material
- Involve users in the testing of software changes

- Corrective maintenance: identifying and repairing defects Adaptive maintenance: adapting the existing solution to the new platforms.
- Perfective Maintenance: implementing the new requirements

➢ Test case authoring or static testing

➢ Static Testing

- Static Testing: Static testing techniques involve examination

of the project's documentation, software and other

information about the software products without executing

them

- Reviews, Static Analysis, and dynamic testing have the same objective – identifying defects

➢ Static Testing (Cont...)

- Reviews are a way of testing software work products (including code)
- and can be performed well before dynamic test execution.
- A review could be done entirely as a manual activity, but there is also tool support available.
- The main manual activity is to examine a work product and make comments about it.

➢ Use of Static Testing

- Since static testing can start early in the life cycle so early feedback on quality issues can be established.
- Types of the defects that are easier to find during the static testing are:
- deviation from standards,
- missing requirements,
- design defects,
- non-maintainable code,
- inconsistent interface specifications.

➢ Informal Reviews

- ➢ Informal reviews are applied many times during the early stages of the life cycle of the document.
- ➢ A two-person team can conduct an informal review. In later stages these reviews often involve more people and a meeting.
- ➢ The most important thing to keep in mind about the informal reviews is that they are not documented.

➢ Formal Reviews

➢ A formal review process consists of six main steps:

- ➢ Planning
- ➢ Kick-off
- ➢ Preparation
- ➢ Review meeting
- ➢ Rework
- ➢ Follow-up

➢ Planning

- ➢ The first phase of the formal review is the Planning phase.
- ➢ The entry check is done to ensure that the reviewer's time is not wasted on a document that is not ready for review.
- ➢ For the formal reviews the moderator performs the entry check and also defines the formal exit criteria.
- ➢ The documents should not reveal a large number of major defects.
- ➢ The documents to be reviewed should be with line numbers.
- ➢ The documents should be cleaned up by running any automated checks that apply.
- ➢ The author should feel confident about the quality of the document so that he can

➢ Formal Review – Kick Off

- ➢ This kick-off meeting is an optional step in a review procedure.

- At customer sites, we have measured results up to 70% more major defects found per page as a result of performing a kick-off.
- The success factor for a thorough preparation is the number of pages checked per hour. This is called the checking rate.
- Usually, the checking rate is in the range of 5 to 10 pages per hour.

➢ Formal Review – Preparation

- The success factor for a thorough preparation is the number of pages checked per hour. This is called the checking rate.
- In this step the reviewers review the document individually using the related documents, procedures, rules and checklists provided.
- Usually, the checking rate is in the range of 5 to 10 pages per hour.
- The author should feel confident about the quality of the document so that he can

➢ Formal Review – Planning

- The first phase of the formal review is the Planning phase.
- In this phase the review process begins with a request for review by the author to the moderator (or inspection leader.
- The entry check is done to ensure that the reviewer's time is not wasted on a document that is not ready for review.

➢ Types of Review – Inspection

Inspection:

- It is the most formal review type
- It is led by the trained moderators
- During inspection the documents are prepared and checked thoroughly by the
- reviewers before the meeting

Build Release Process

❖ What is Build in testing?

- After developing the software module, developers convert the source codes into a standalone form or an executable code.
- Build is in the testing phase; it may have already undergone testing or not. Software testing team checks this build.
- Builds occur prior to the release, and they are generated more frequently.

❖ What is Release in testing?

- The release is the final application after completing development and testing.
- It is possible for one release to have several builds.
- The main difference between Build and Release in Software Testing is that Build is a version of a software the development team hands over to the testing team for testing purposes while Release is a software the testing team hands over to the customer.

❖ Standalone Application

Software Release:

- The release is the final application after completing development and testing.
- After testing the build, the testing team certifies that software and delivers it to
- It is possible for one release to have several builds.
- Moreover, the release is based on builds, and it can have several builds.

Defect Management and Tracking

- Defect is the variance from a desired product attribute (it can be a wrong, missing, or extra data).

It can be of two types –

- Defect from the product or a variance from customer/user expectations.
- It is a flaw in the software system and has no impact until it affects the user/customer and operational system.

- With the knowledge of testing so far gained, you can now be able to categorize the defects you have found.
- Some defects address security or database issues while others may refer to functionality or UI issues.

➢ Types of Defects

- Data Quality/Database Defects: Deals with improper handling of data in the database.

Examples:

- Values not deleted/inserted into the database properly
- Improper/wrong/null values inserted in place of the actual values
- Critical Functionality Defects: The occurrence of these bugs hampers the crucial functionality of the application. Examples: - Exceptions
- Functionality Defects: These defects affect the functionality of the application. **Examples:**

All JavaScript errors

- Buttons like Save, Delete, Cancel not performing their intended functions

➢ Types of Defects

Security Defects: Application security defects generally involve improper handling of data sent from the user to the application. These defects are the most severe and given highest priority for a fix.

User Interface Defects: As the name suggests, the bugs deal with problems related to UI are usually considered less severe

➢ Bug (Defect) 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'.

➢ The process by which the defect moves through the life cycle is depicted next slide.

➢ Bug (Defect) Life Cycle

- 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.

➢ 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."

➢ Defect Stages (Cont...)

- 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."

➢ Defect 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
- **For example**: If an application or web page crashes when a remote link is clicked, in this case clicking the remote link by an user is rare but the impact of application crashing is severe. So the severity is high but priority is low.
- 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.

- Defect Severity (Cont...)

Severity can be of following types:

- 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

<u>Defect Priority</u>

- 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 fix 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

➢ Scenario of Defect Priority – Severity

- High Priority & High Severity: An error which occurs on the basic functionality of the application and will not allow the user to use the system. (Eg. A site maintaining the student details, on saving record if it, doesn’t allow to save the record then this is high priority and high severity bug.)
- High Priority & Low Severity: The spelling mistakes that happens on the cover page or heading or title of an application.
- High Severity & Low Priority: An error which occurs on the functionality of the application (for which there is no workaround) and will not allow the

user to use the system but on click of link which is rarely used by the end user.

- Low Priority and Low Severity: Any cosmetic or spelling issues which is within a paragraph or in the report (Not on cover page, heading, title).

Agile testing principal

1 Customer satisfaction through early and continuous software delivery –

Customers are happier when they receive working software at regular intervals, rather than waiting extended periods of time between releases.

2 Accommodate changing requirements throughout the development process – The

ability to avoid delays when a requirement or feature request changes.

3 Frequent delivery of working software – Scrum accommodates this principle since the team operates in software sprints or iterations that ensure regular delivery of working software.

4 Collaboration between the business stakeholders and developers throughout the project – Better decisions are made when the business and technical team are aligned.

5 Support, trust, and motivate the people involved – Motivated teams are more likely to deliver their best work than unhappy teams.

6 Enable face-to-face interactions – Communication is more successful when

7 Working software is the primary measure of progress – Delivering functional software to the customer is the ultimate factor that measures progress.

8 Agile processes to support a consistent development pace – Teams establish arepeatable and maintainable speed at which they can deliver working software, and they repeat it with each release.

9 Attention to technical detail and design enhances agility – The right skills and good design ensures the team can maintain the pace, constantly improve the product, and sustain change.

10 Simplicity – Develop just enough to get the job done for right now.

11 Self-organizing teams encourage great architectures, requirements, and designs –Skilled and motivated team members who have decision-making power, take ownership, communicate regularly with other team members, and share ideas that deliver quality products.

12 Regular reflections on how to become more effective – Self-improvement, process improvement, advancing skills, and techniques help team members work more efficiently.

Scrum

Scrum: SCRUM is an agile development method which concentrates particularly on how to manage tasks within a team-based development environment. Basically, Scrum is derived from activity that occurs during rugby match. Scrum believes in empowering the development team and advocates working in small teams (say- 7 to 9members).It consists of **<u>three roles and their responsibilities are explained as follows</u>:**

1 Scrum Master: Master is responsible for setting up the team, sprint meeting and removes obstacles to progress

2 Product owner: The Product Owner creates product backlog, prioritizes the backlog and is responsible for the delivery of the functionality at each iteration

3 Scrum Team: Team manages its own work and organizes the work to complete the sprint or cycle

Sprint: Sprint is a time-boxed period in which the scrum team needs to finish the set amount of work. Each sprint has a specified timeline, i.e., 2 weeks to 1 month. The scrum team agrees with this timeline during the sprint planning meeting.

- Product Backlog is a collection of activities that need to be done within the project. When we want to develop software, then we need to perform the 'n' number of activities.

For example, we need to develop the e-commerce website then we have to do the 'n'

number of activities such we need to create the login page, payment system, cart system, etc. and these 'n' number of activities which are needed to develop the software is known as the product backlog.

- Sprint Backlog

We know that in a scrum, we break the scrum into 'n' number of sprints and the objective of a sprint is to bring the small functionality of the software and ship it to the client for demo. In Product backlog, we have to do all the activities which are required to develop the software while in the sprint backlog, a small set of product backlog activities are performed within that sprint. The 'n' number of sprint backlogs is equal to the 1 product backlog.

- Burndown Chart

Burndown chart is the outcome of the sprint, which shows the progress in a sprint. Aftereach sprint, we need to examine the progress of each sprint. The burndown chart tells how you are working on the sprint. In the burndown chart, the graph starts from some time, i.e., where the activity gets started, and at the end of the sprint, the graph reaches to zero where the activity ends. It is generally an inclined line from top to bottom.

Scrum Ceremonies

Let's look at the following Scrum Ceremonies:

1 sprint planning

Scrum consists of a number of sprints which have a different set of modules used to deliver the software. Before starting the sprint planning, we have a meeting known as sprint planning, and in sprint planning, we discuss what we are going to do in a sprint. In sprint planning, product owner discusses about each feature of a product and estimates the effort involved by the development team.

2.Daily Scrum
In Scrum, meetings are conducted daily for 15 minutes by Master, where

ScrumMaster is the person who manages the meeting. Meeting consists of scrum master, developers, testers, designers, product owner, the client where product and.

3.Sprint Review

After the completion of each sprint, the meeting is conducted with a client in which a product is shown to the client for demo and team discuss the features they added in the project.

Scrum Board

- Product Backlog: Product Backlog is a set of activities that need to be done to develop the software.
- Sprint Backlog: Sprint Backlog is a backlog that has taken some of the activities from the product backlog which needs to be completed within this sprint.
- Scrum Board: Scrum Board is a board that shows the status of all the activities that need to be done within this sprint. Scrum Board consists of four statuses:
- Open The 'Open' status means that the tasks which are available in 'Open' are not yet started.
- In progress The 'In progress' status means the developers completed their tasks.
- Testing The 'testing' means that the task is in a testing phase.
- Closed The 'closed' means the task has been completed.