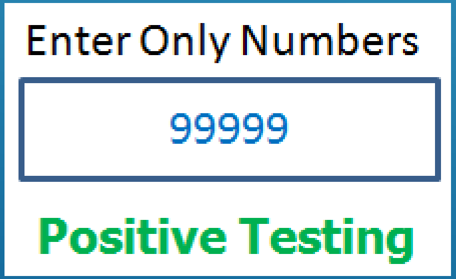
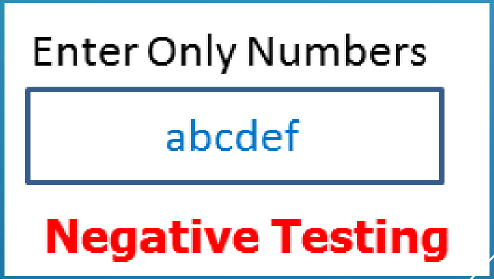
Positive Testing :

Positive Testing is a type of testing which is performed on a software application by providing the valid data sets as an input. It checks whether the software application behaves as expected with positive inputs or not. Positive testing is performed in order to check whether the software application does exactly what it is expected to do.



Negative Testing:

Negative Testing is a testing method performed on the software application by providing invalid or improper data sets as input. It checks whether the software application behaves as expected with the negative or unwanted user inputs. The purpose of negative testing is to ensure that the software application does not crash and remains stable with invalid data inputs.



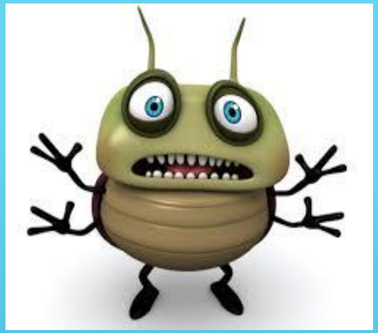
TODAY'S AGENDA

- Testing Techniques
- What is a bug?
- Bug Lifecycle
- Bug Template
- Severity Vs Priority
- Testing Levels
  - White Box Testing
  - Blackbox Testing

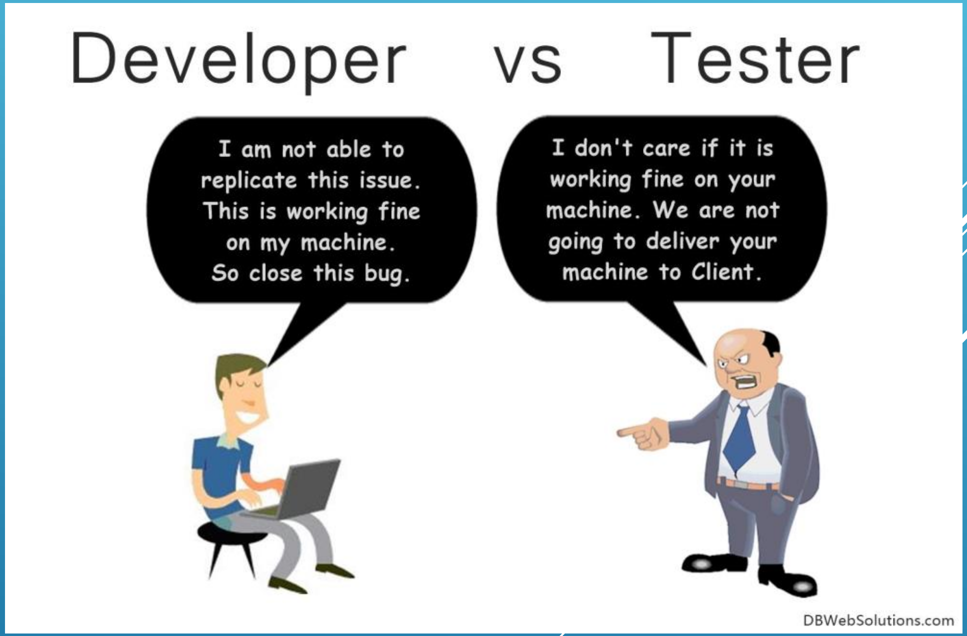
WHAT IS A BUG?

A software bug is an error, failure, or fault/defect in a computer program or system that causes it to produce an incorrect or unexpected results, or to have behave in unintended ways.

WHY ARE THERE BUGS IN THE SOFTWARE?

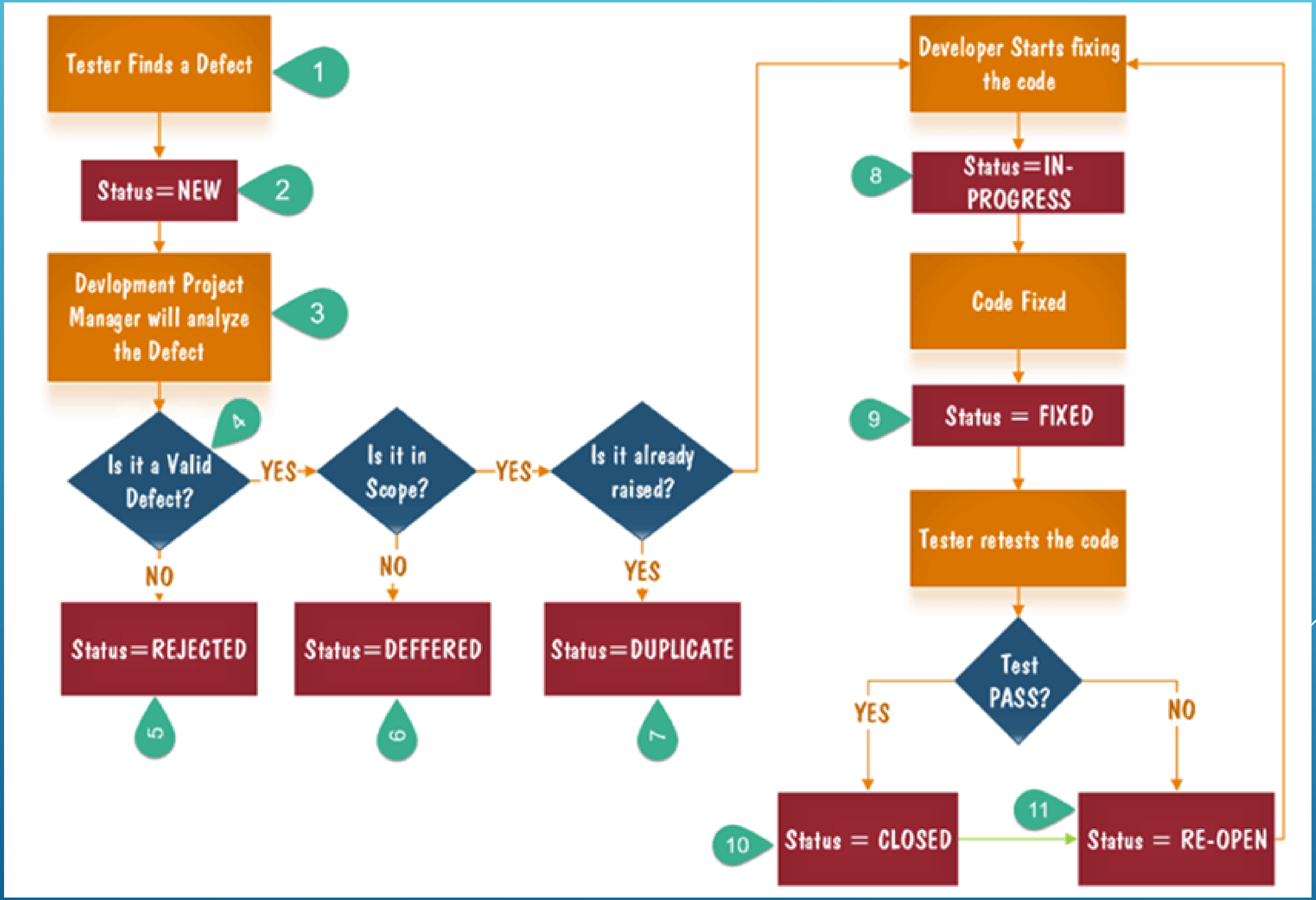


- Miscommunication or no communication
- Software complexity
- Programming errors
- Changing requirements
- Lack of skilled testers/developers

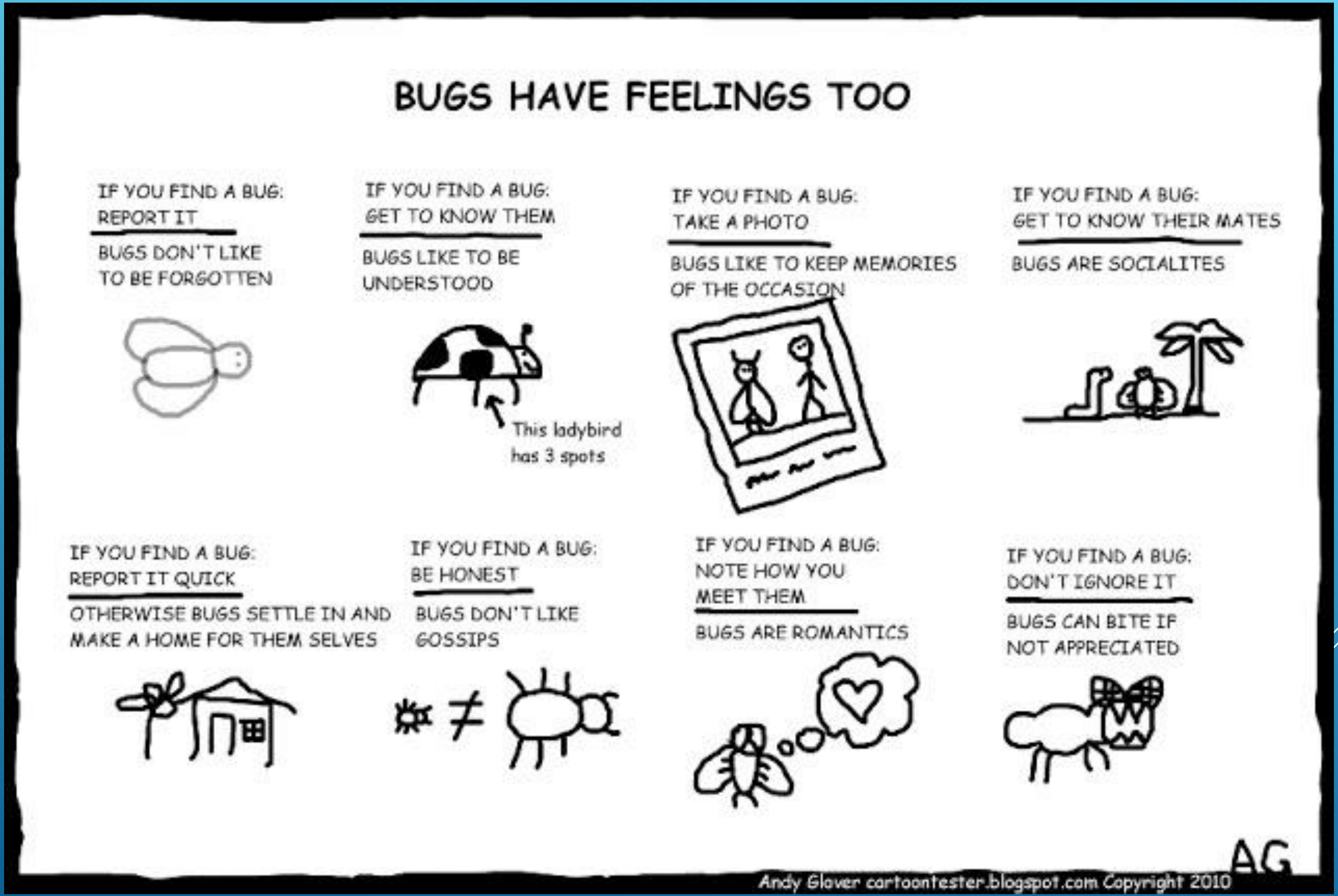

Developer vs Tester
I am not able to replicate this issue. This is working fine on my machine. So close this bug.
I don't care if it is working fine on your machine. We are not going to deliver your machine to Client.
DBWebSolutions.com

BUG LIFE CYCLE

A Defect/Bug life cycle, is a cycle of a defect from which goes through covering the different states in its entire life. This starts as soon as any new defect is found by a tester and comes to an end when a tester closes that defect assuring that it won't get reproduced again.



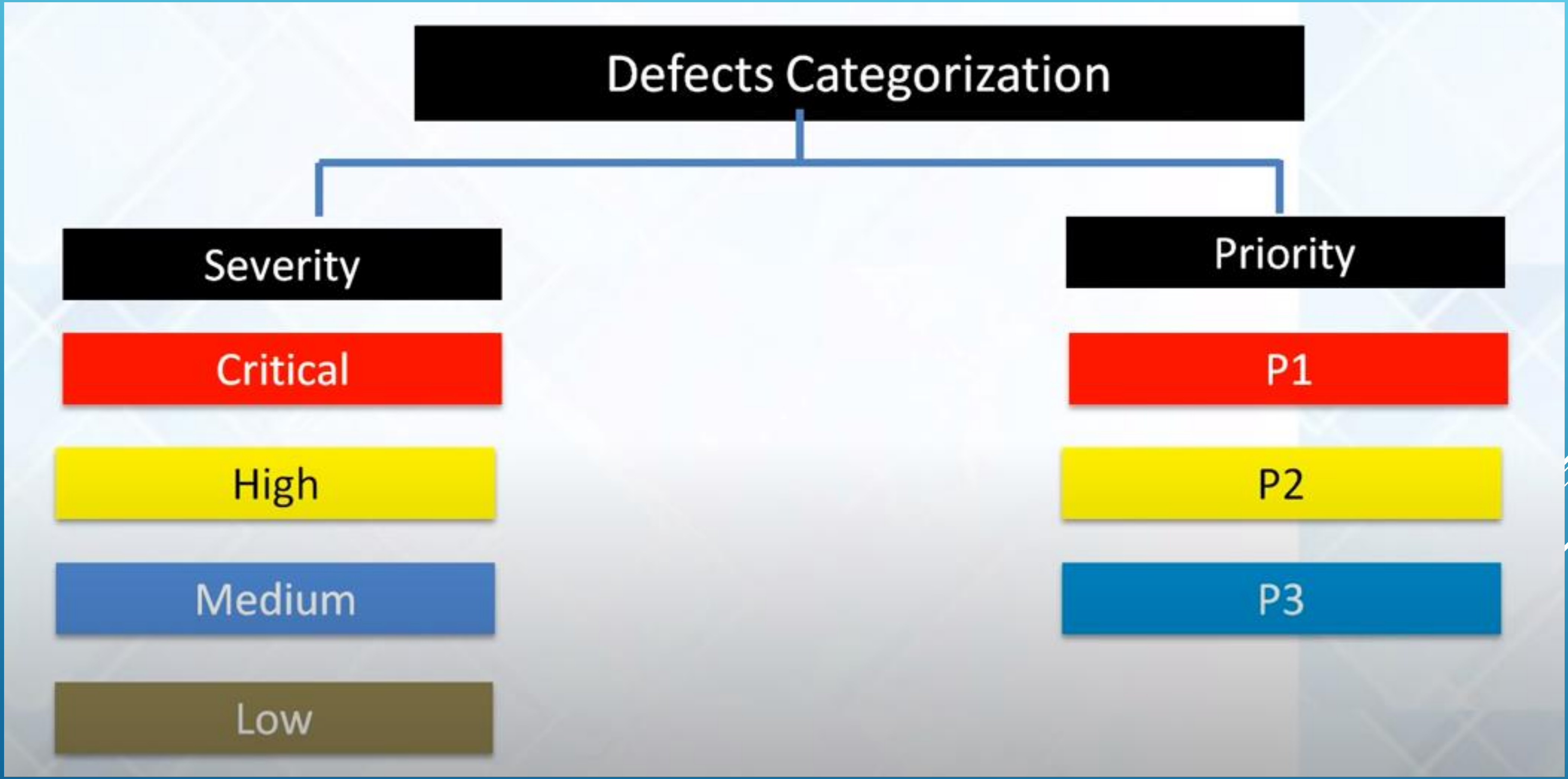
WHY AND HOW?


BUGS HAVE FEELINGS TOO
IF YOU FIND A BUG: REPORT IT
BUGS DON'T LIKE TO BE FORGOTTEN
IF YOU FIND A BUG: GET TO KNOW THEM
BUGS LIKE TO BE UNDERSTOOD
This ladybird has 3 spots
IF YOU FIND A BUG: TAKE A PHOTO
BUGS LIKE TO KEEP MEMORIES OF THE OCCASION
IF YOU FIND A BUG: GET TO KNOW THEIR MATES
BUGS ARE SOCIALITES
IF YOU FIND A BUG: REPORT IT QUICK
OTHERWISE BUGS SETTLE IN AND MAKE A HOME FOR THEM SELVES
IF YOU FIND A BUG: BE HONEST
BUGS DON'T LIKE GOSSIPS
IF YOU FIND A BUG: NOTE HOW YOU MEET THEM
BUGS ARE ROMANTICS
IF YOU FIND A BUG: DON'T IGNORE IT
BUGS CAN BITE IF NOT APPRECIATED
AG
Andy Glover cartoontester.blogspot.com Copyright 2010

BUG TEMPLATE

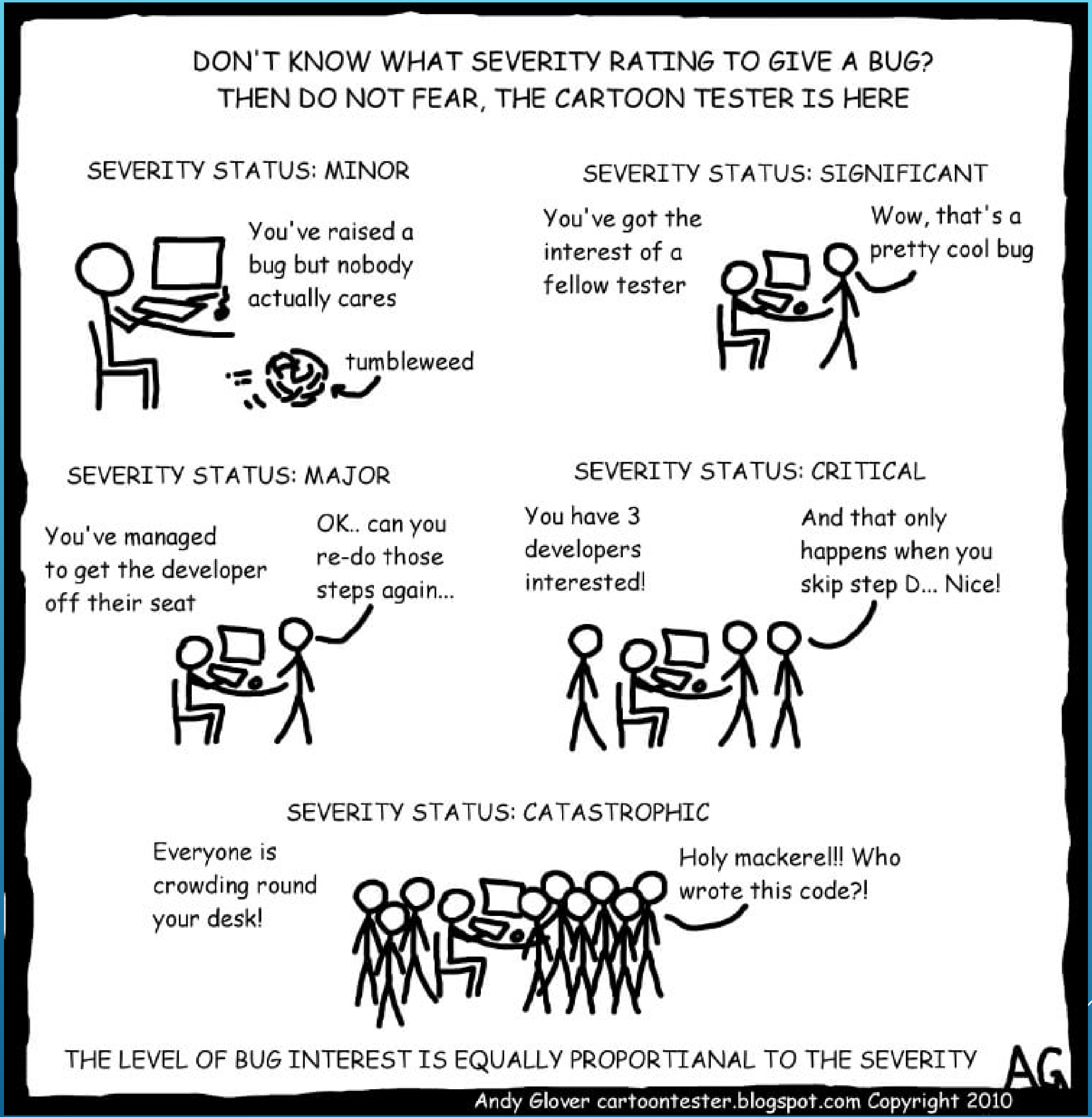
- **Bug ID** - Unique identification number for the defect.
- **Bug Description** - Detailed description of the Defect including information about the module in which Defect was found.
- **Version** - Version of the application in which defect was found.
- **Status** - Status of the defect.
- **Severity** - which describes the impact of the defect on the application
- **Priority** - which is related to defect fixing urgency.
- **Steps to reproduce** - Detailed steps along with screenshots with which the developer can reproduce the defects.
- **Date Raised** - Date when the defect is raised
- **Reference** - where in you Provide reference to the documents like . requirements, design, architecture or maybe even screenshots of the error to help understand the defect.
- **Detected By** - Name/ID of the tester who raised the defect
- **Fixed by** - Name/ID of the developer who fixed it
- **Date Closed** - Date when the defect is closed

SEVERITY VS PRIORITY



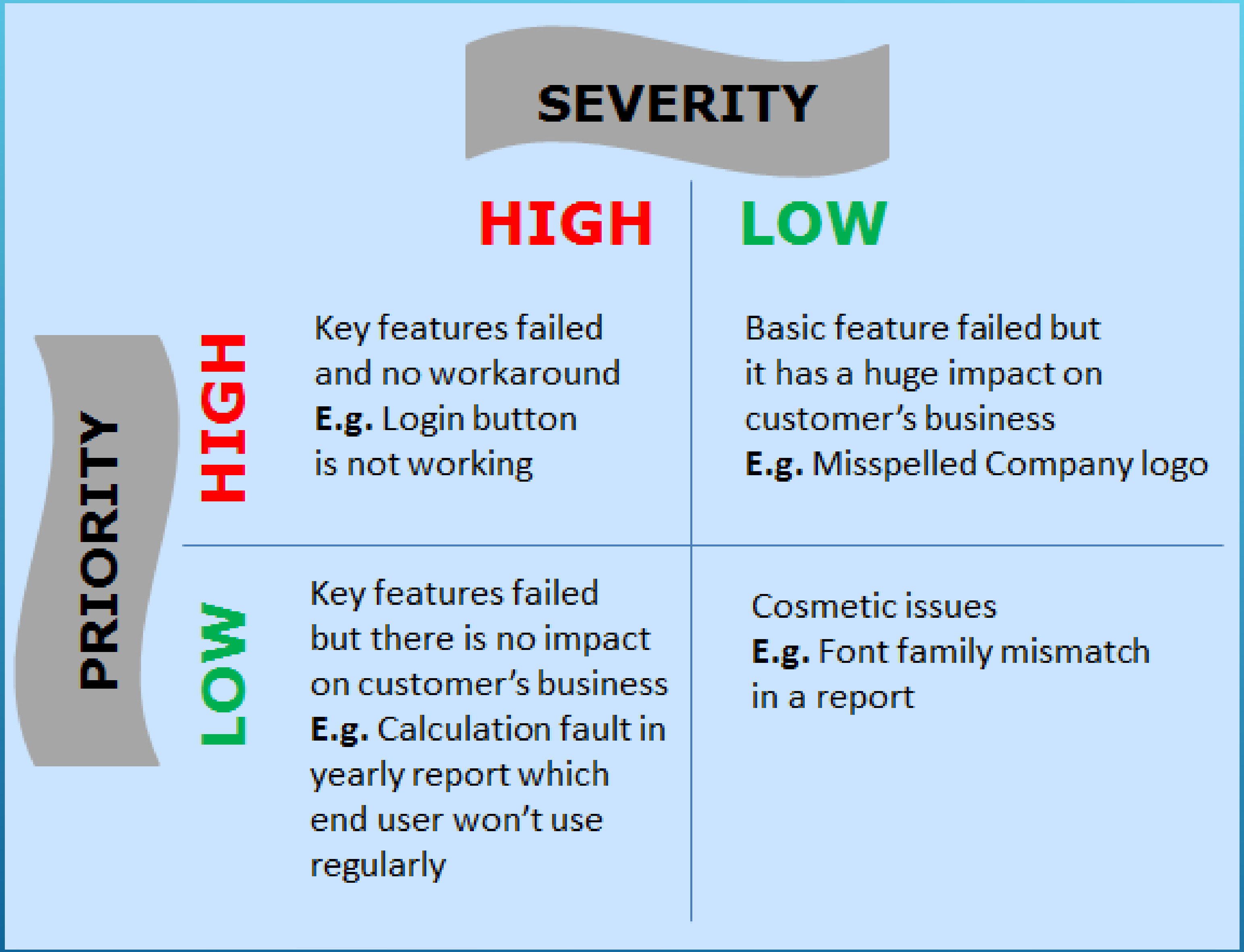
WHAT IS SEVERITY?

- Bug/Defect severity can be defined as the impact of the bug on customer's business.
- Severity decide by the Tester.
- Types of Severity
  - Critical/Blocker – We can't move forward after braking big functionality
  - High/Major – After braking big functionality we can still move forward.
  - Medium/Minor – undesirable behaviour but system is still functional.
  - Low/Trivial – It won't cause any major break-down of the system.

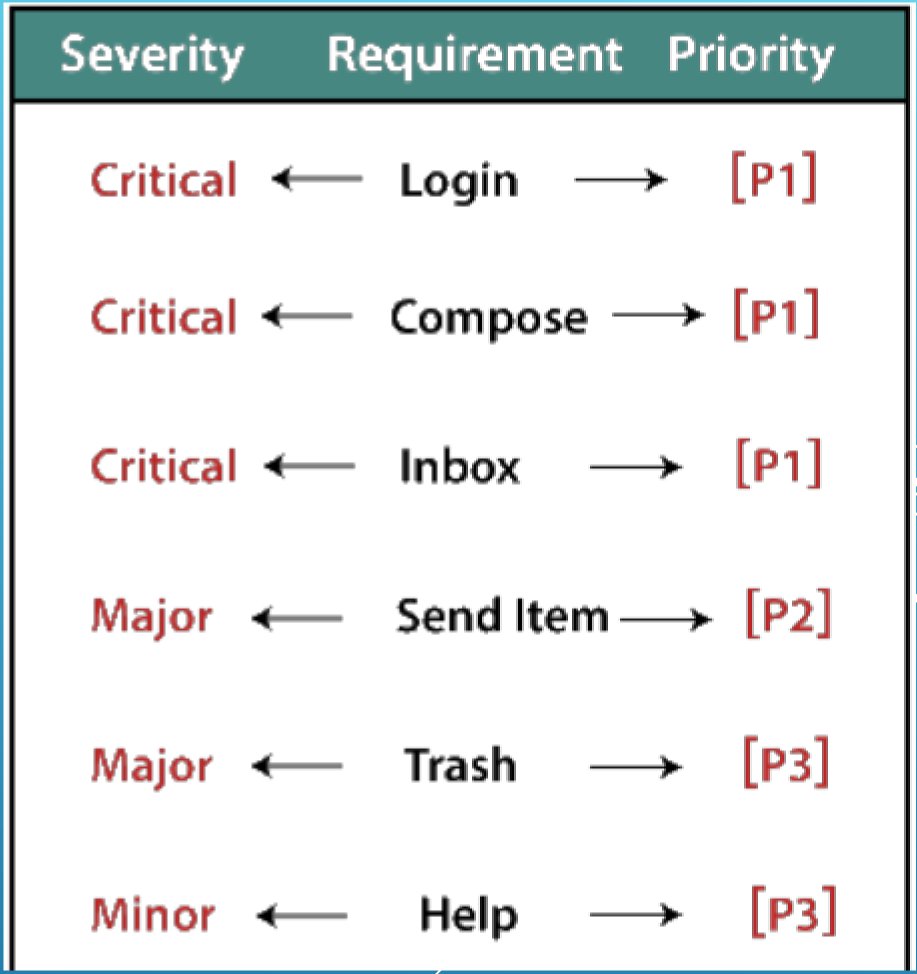

DON'T KNOW WHAT SEVERITY RATING TO GIVE A BUG? THEN DO NOT FEAR, THE CARTOON TESTER IS HERE
SEVERITY STATUS: MINOR
You've raised a bug but nobody actually cares
tumbleweed
SEVERITY STATUS: SIGNIFICANT
You've got the interest of a fellow tester
Wow, that's a pretty cool bug
SEVERITY STATUS: MAJOR
You've managed to get the developer off their seat
OK.. can you re-do those steps again...
SEVERITY STATUS: CRITICAL
You have 3 developers interested!
And that only happens when you skip step D... Nice!
SEVERITY STATUS: CATASTROPHIC
Everyone is crowding round your desk!
Holy mackerel!! Who wrote this code?!
THE LEVEL OF BUG INTEREST IS EQUALLY PROPORTIANAL TO THE SEVERITY
AG
Andy Glover cartoontester.blogspot.com Copyright 2010

WHAT IS PRIORITY?

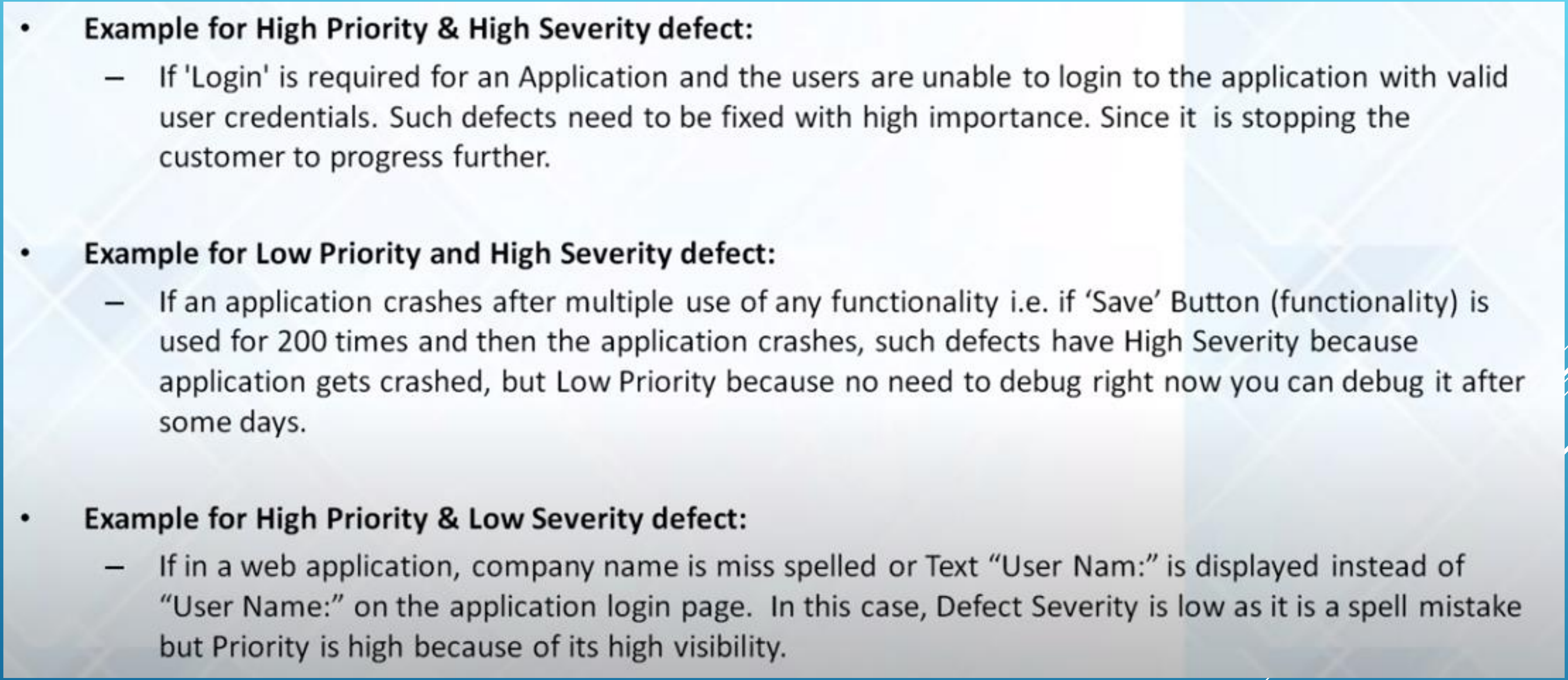
- Bug/Defect priority can be defined as how soon the bug should be fixed.
- Priority define by Developer.
- **Types of Priority:**
  - P1 (High) –Defect must be fixed as soon as possible as it is severely affecting the system and cannot be unused.
  - P2 (Medium) – It can be fixed as per normal queue.
  - P3 (Low) – As no impact on the system so can be fixed after more serious defects have been fixed.



PRIORITY \ SEVERITY	HIGH	LOW
HIGH	Key features failed and no workaround **E.g.** Login button is not working	Basic feature failed but it has a huge impact on customer's business **E.g.** Misspelled Company logo
LOW	Key features failed but there is no impact on customer's business **E.g.** Calculation fault in yearly report which end user won't use regularly	Cosmetic issues **E.g.** Font family mismatch in a report



Severity	Requirement	Priority
Critical ⟵	Login ⟶	[P1]
Critical ⟵	Compose ⟶	[P1]
Critical ⟵	Inbox ⟶	[P1]
Major ⟵	Send Item ⟶	[P2]
Major ⟵	Trash ⟶	[P3]
Minor ⟵	Help ⟶	[P3]

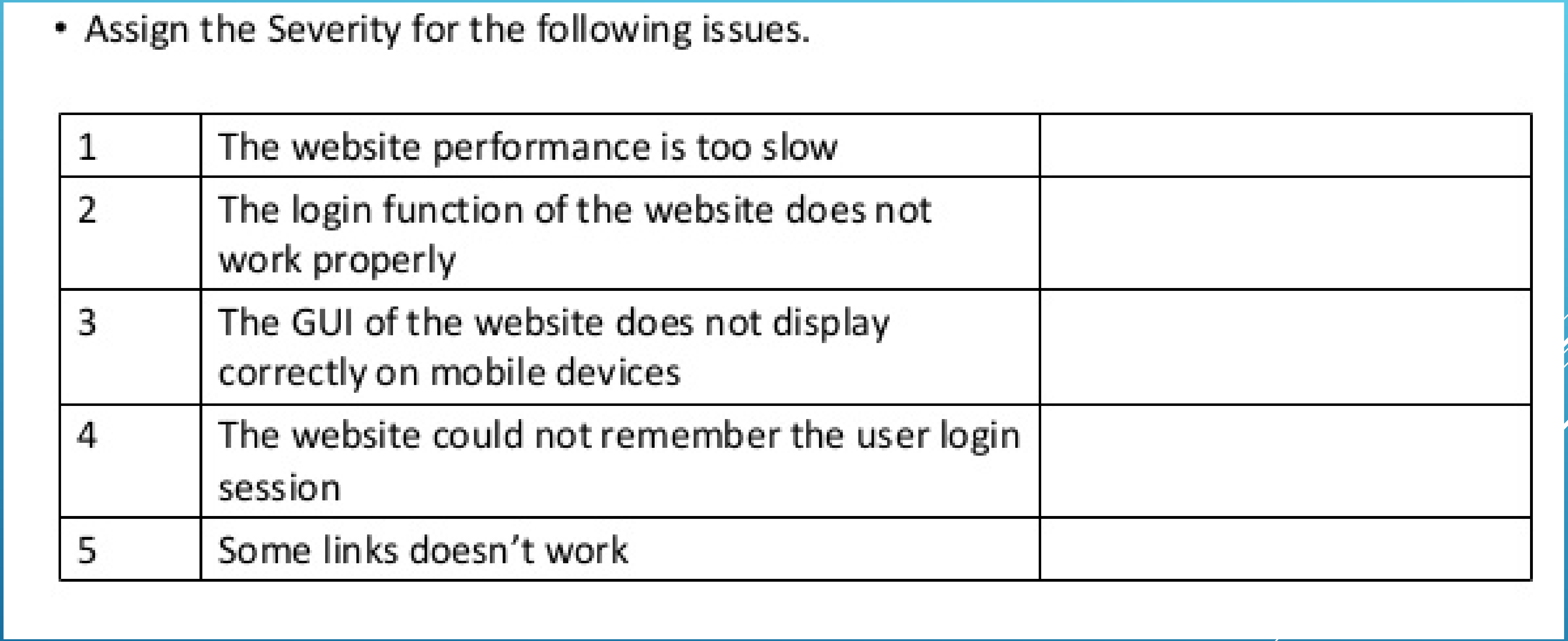


- **Example for High Priority & High Severity defect:**
  - If 'Login' is required for an Application and the users are unable to login to the application with valid user credentials. Such defects need to be fixed with high importance. Since it is stopping the customer to progress further.

- **Example for Low Priority and High Severity defect:**
  - If an application crashes after multiple use of any functionality i.e. if ‘Save’ Button (functionality) is used for 200 times and then the application crashes, such defects have High Severity because application gets crashed, but Low Priority because no need to debug right now you can debug it after some days.

- **Example for High Priority & Low Severity defect:**
  - If in a web application, company name is miss spelled or Text “User Nam:” is displayed instead of “User Name:” on the application login page. In this case, Defect Severity is low as it is a spell mistake but Priority is high because of its high visibility.

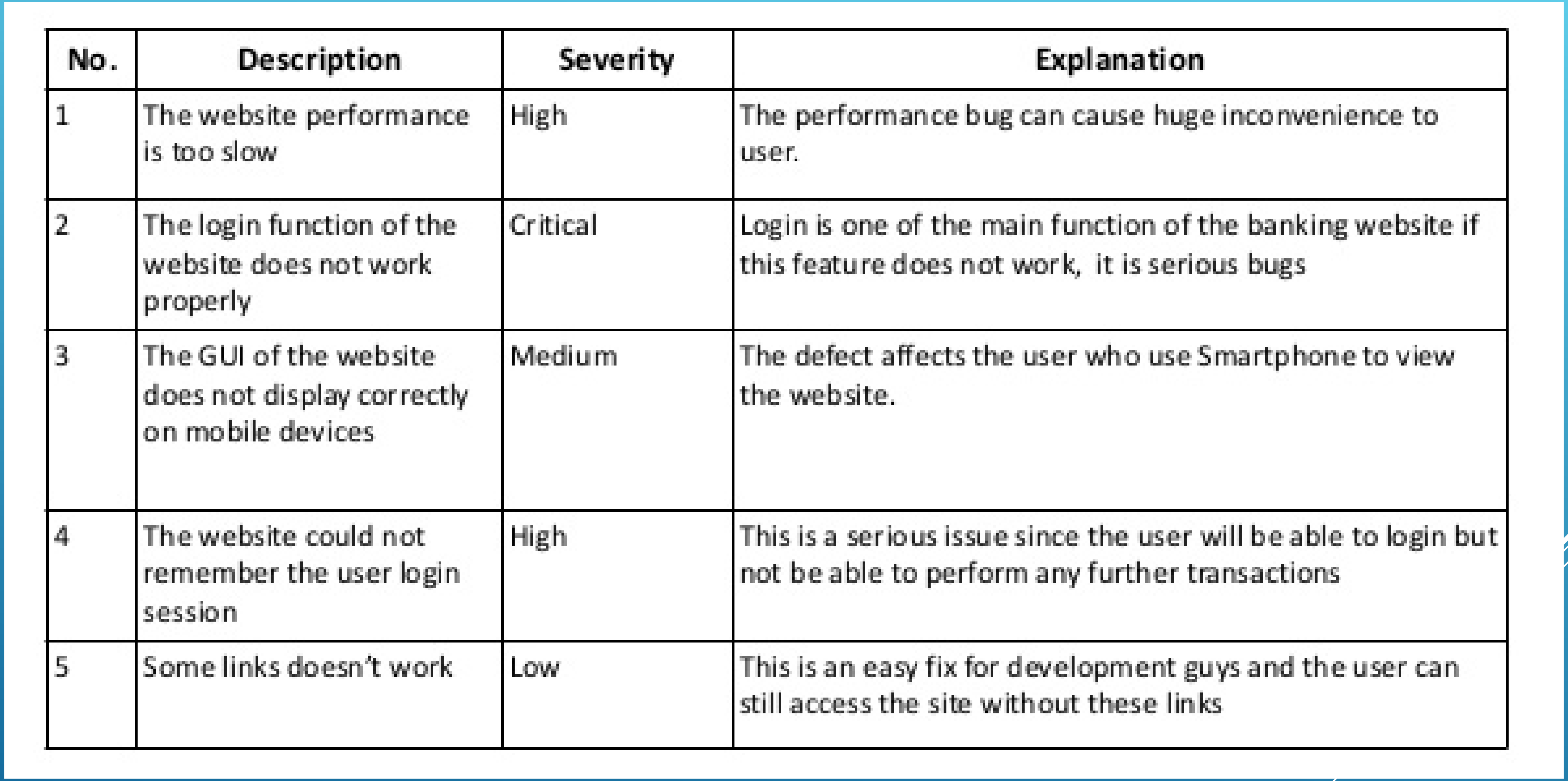
EXERCISE



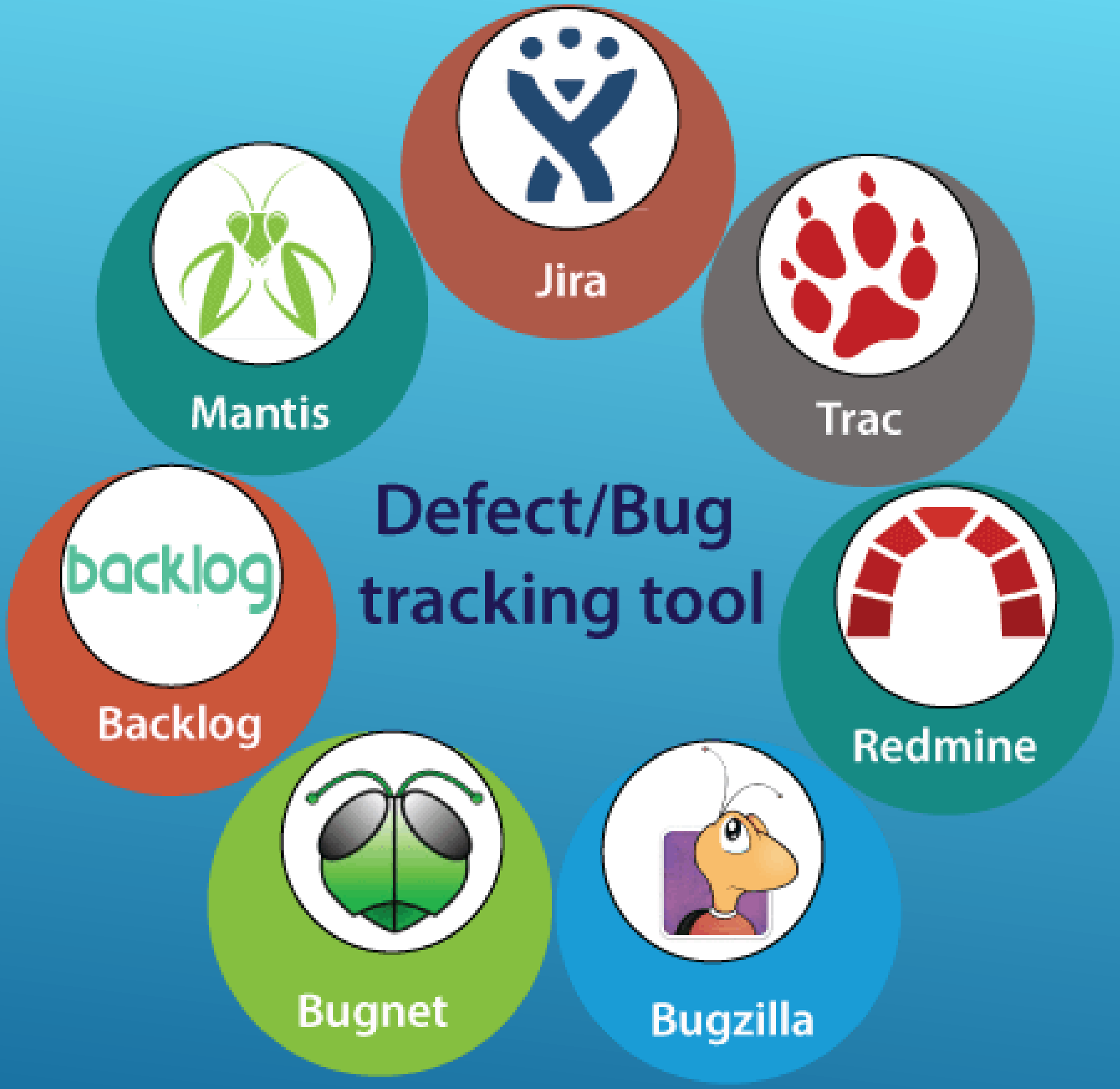
- Assign the Severity for the following issues.

1	The website performance is too slow	
2	The login function of the website does not work properly	
3	The GUI of the website does not display correctly on mobile devices	
4	The website could not remember the user login session	
5	Some links doesn’t work	

SOLUTION



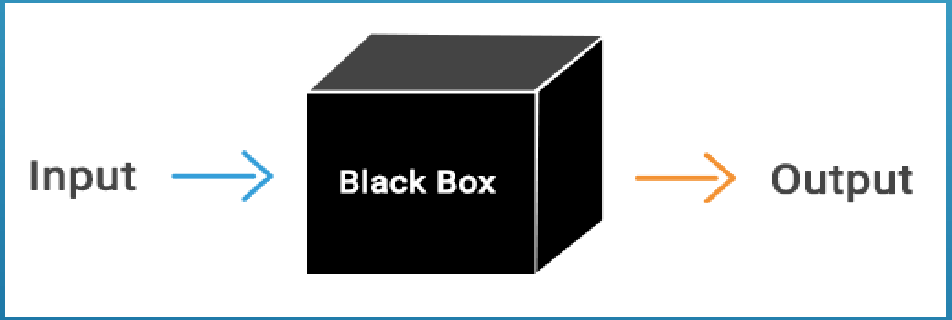
No.	Description	Severity	Explanation
1	The website performance is too slow	High	The performance bug can cause huge inconvenience to user.
2	The login function of the website does not work properly	Critical	Login is one of the main function of the banking website if this feature does not work, it is serious bugs
3	The GUI of the website does not display correctly on mobile devices	Medium	The defect affects the user who use Smartphone to view the website.
4	The website could not remember the user login session	High	This is a serious issue since the user will be able to login but not be able to perform any further transactions
5	Some links doesn't work	Low	This is an easy fix for development guys and the user can still access the site without these links

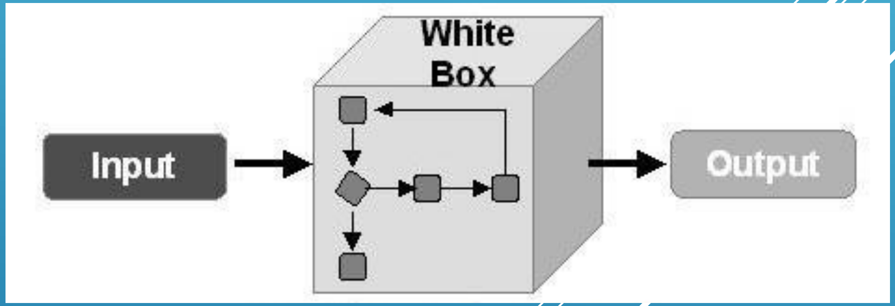

Defect/Bug tracking tool
Jira
Trac
Redmine
Bugzilla
Bugnet
Backlog
backlog
Mantis

Black Box Testing

&

White Box Testing



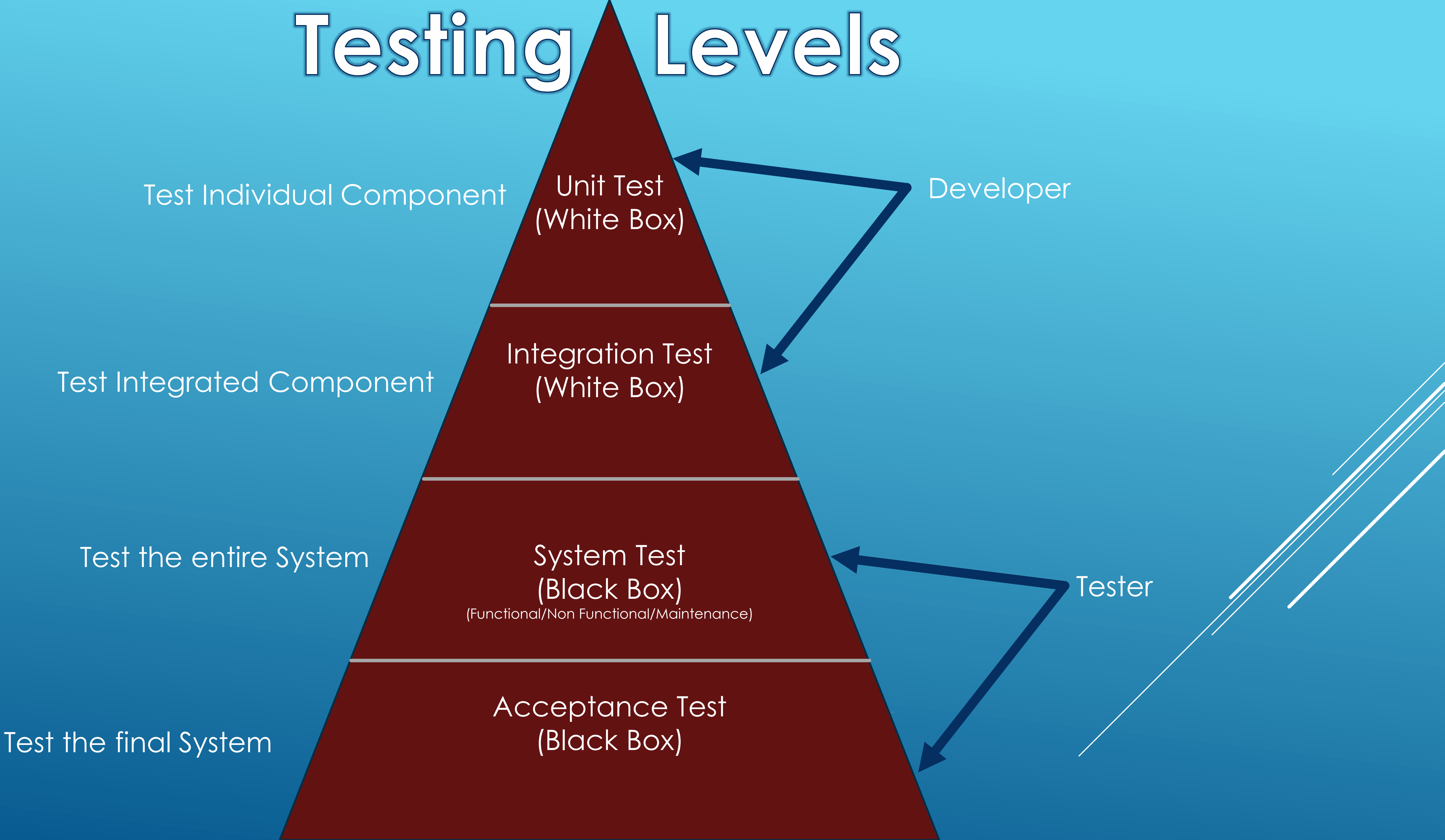


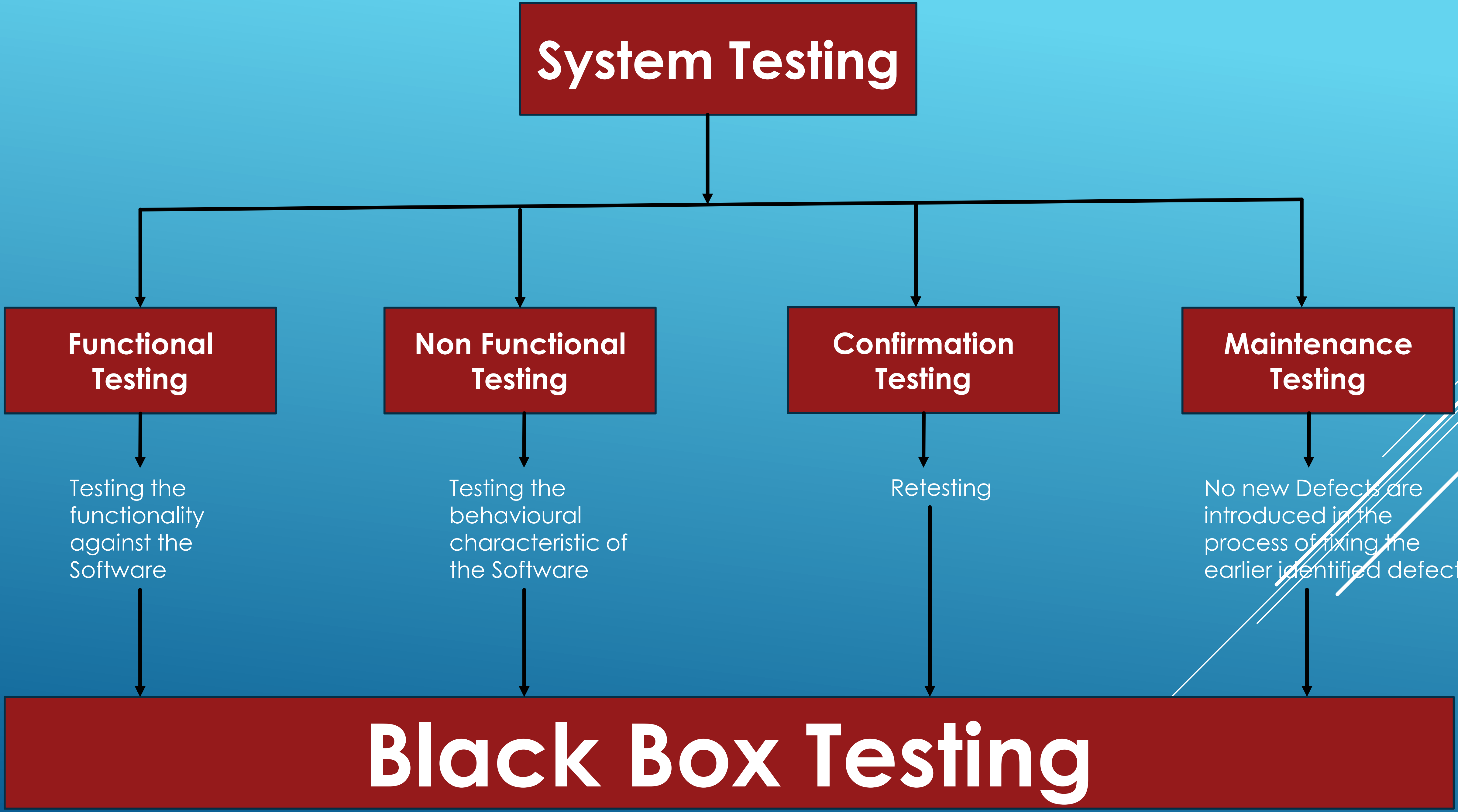
Parameter	Black Box testing	White Box testing
Definition	It is a testing approach which is used to test the software without the knowledge of the internal structure of program or application.	It is a testing approach in which internal structure is known to the tester.
Also Known as	It also knowns as data-driven, box testing, data-, and functional testing.	It is also called structural testing, clear box testing, code-based testing, or glass box testing.
Base of Testing	Testing is based on external expectations; internal behaviour of the application is unknown.	Internal working is known, and the tester can test accordingly.
Usage	This type of testing is ideal for higher levels of testing like System Testing, Acceptance testing.	Testing is best suited for a lower level of testing like Unit Testing, Integration testing.
Programming knowledge	Programming knowledge is not needed to perform Black Box testing.	Programming knowledge is required to perform White Box testing.
Implementation knowledge	Implementation knowledge is not requiring doing Black Box testing.	Complete understanding needs to implement White Box testing.
Objective	The main objective of this testing is to check what functionality of the system under test.	The main objective of White Box testing is done to check the quality of the code.

Parameter	Black Box testing	White Box testing
Basis for test cases	Testing can start after preparing requirement specification document.	Testing can start after preparing for Detail design document.
Tested by	Performed by the end user, developer, and tester.	Usually done by tester and developers.
Testing method	It is based on trial and error method.	Data domain and internal boundaries can be tested.
Time	It is less exhaustive and time-consuming.	Exhaustive and time-consuming method.
Code Access	Code access is not required for Black Box Testing.	White box testing requires code access. Thereby, the code could be stolen if testing is outsourced.

Parameter	**Black Box testing**	**White Box testing**
Skill level	Low skilled testers can test the application with no knowledge of the implementation of programming language or operating system.	Need an expert tester with vast experience to perform white box testing.
Techniques	Equivalence partitioning is Black box testing technique is used for Blackbox testing. Equivalence partitioning divides input values into valid and invalid partitions and selecting corresponding values from each partition of the test data. Boundary value analysis checks boundaries for input values.	Statement Coverage, Branch coverage, and Path coverage are White Box testing technique. Statement Coverage validates whether every line of the code is executed at least once. Branch coverage validates whether each branch is executed at least once Path coverage method tests all the paths of the program.

Testing Levels




System Testing
Functional Testing
Non Functional Testing
Confirmation Testing
Maintenance Testing
Testing the functionality against the Software
Testing the behavioural characteristic of the Software
Retesting
No new Defects are introduced in the process of fixing the earlier identified defects
Black Box Testing

FUNCTIONAL TESTING

- Unit Testing
- Integration Testing
- User Acceptance testing
- Smoke Testing
- Sanity Testing

Smoke Testing	Sanity Testing
Smoke Testing is performed to find out that the critical functionalities of the program is working fine	Sanity Testing is done 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 thorough testing	The objective of the testing is to verify the logic of the system in order to proceed with more thorough testing
This testing is performed by the developers or testers	Sanity testing in software testing is usually performed by testers
Smoke testing is usually documented or scripted	Sanity testing is usually not documented and is unscripted
Smoke testing is a subset of Acceptance testing	Sanity testing is a subset of Regression Testing
Smoke testing exercises the entire system from end to end	Sanity testing exercises only the particular component of the entire system
Smoke testing is like General Health Check Up	Sanity Testing is like specialized health check up

NON-FUNCTIONAL TESTING

- Performance Testing
  - Load Testing
  - Endurance Testing
  - Stress Testing
- Compatibility Testing
  - Cross Browser Testing
  - Cross Device testing
  - Cross Version Testing
  - Cross Operating System Testing
- Usability Testing
- GUI Testing
- Security Testing

CONFIRMATION TESTING

- Retesting

MAINTENANCE TESTING

- Regression Testing

Regression Testing	Re-testing
Regression Testing is carried out to confirm whether a recent program or code change has not adversely affected existing features	Re-testing is carried out to confirm the test cases that failed in the final execution are passing after the defects are fixed
The purpose of Regression Testing is that new code changes should not have any side effects to existing functionalities	Re-testing is done on the basis of the Defect fixes
You can do automation for regression testing, Manual Testing could be expensive and time-consuming	You cannot automate the test cases for Retesting
Regression testing is done for passed test cases	Retesting is done only for failed test cases
Regression testing checks for unexpected side-effects	Re-testing makes sure that the original fault has been corrected
Regression testing is only done when there is any modification or changes become mandatory in an existing project	Re-testing executes a defect with the same data and the same environment with different inputs with a new build

Functional Testing	Non-Functional Testing
Functional testing is performed using the functional specification provided by the client and verifies the system against the functional requirements.	Non-Functional testing checks the Performance, reliability, scalability and other non-functional aspects of the software system.
Functional testing is executed first	Non-functional testing should be performed after functional testing
Manual Testing or automation tools can be used for functional 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	Non functional testing describes how good the product works
Easy to do Manual Testing	Tough to do Manual Testing

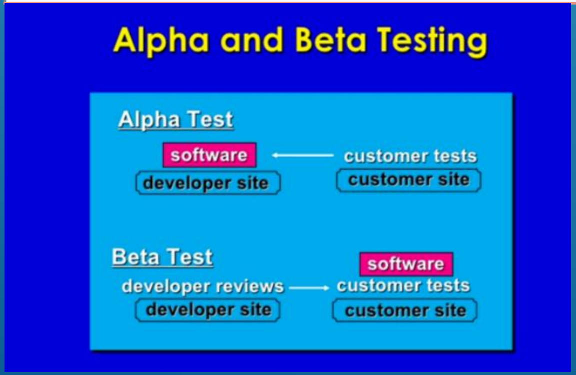
User Acceptance Testing (UAT)

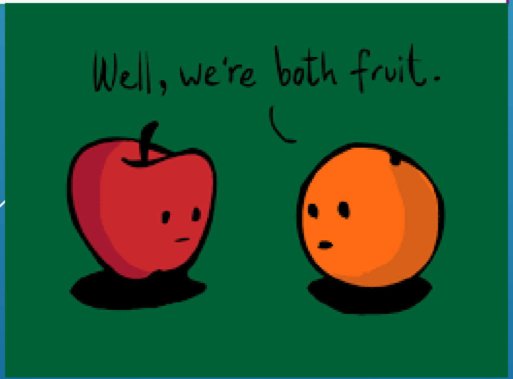
User Acceptance testing is the last phase of the software testing process. During UAT, actual software users test the software to make sure it can handle required tasks in real world scenarios, according to specifications.

DEV
QA
UAT
PROD

Deliver
Fix Defects
Test
Client
Organisation
USER ACCEPTANCE TESTING

Alpha Testing	Beta Testing
Alpha testing performed by Testers who are usually internal employees of the organization	Beta testing is performed by Clients or End Users who are not employees of the organization
Alpha Testing performed at developer's site	Beta testing is performed at a client location or end user of the product
Alpha testing involves both the white box and black box techniques	Beta Testing typically uses Black Box Testing
Alpha testing requires a lab environment or testing environment	Beta testing doesn't require any lab environment or testing environment. The software is made available to the public and is said to be real time environment
Critical issues or fixes can be addressed by developers immediately in Alpha testing	Most of the issues or feedback is collected from Beta testing will be implemented in future versions of the product
Alpha testing is to ensure the quality of the product before moving to Beta testing	Beta testing also concentrates on the quality of the product, but gathers users input on the product and ensures that the product is ready for real time users.





HOMEWORK

- Create test cases for nopCommerce registration page (positive and negative)

- Raise a bug using Jira for one failed registration test case.

Deadline: Wednesday Midnight Latest