**Challenges in Software Testing:**

1)  **Complexity of Software**: As software systems become more complex, testing becomes more challenging. Interactions between different components, integration issues, and the need for testing across various platforms and devices add to the complexity.

2) **Changing Technology**: Rapid advancements in technology and the adoption of new tools and frameworks can make it challenging for testing teams to keep up. Learning and implementing new testing tools can be time-consuming.

3) **Regression Testing**: As software evolves, changes to one part of the code can unintentionally introduce defects in other areas. Regression testing is essential to ensure that new features or bug fixes do not negatively impact existing functionalities. However, running comprehensive regression tests can be time-consuming.

4)  **Incomplete Requirements:** Testing without clear and complete requirements can lead to ambiguity in test cases. Incomplete or changing requirements make it difficult to create comprehensive test cases, leading to potential issues being overlooked.

5) **Time Constraints**: Often, software development projects are subject to tight deadlines. Testing is sometimes compressed, leaving limited time for thorough testing. This can result in inadequate test coverage and potentially missed defects.

6)  **Communication Issues**: Effective communication between development and testing teams is essential. Miscommunication or a lack of collaboration can result in misunderstandings about requirements, leading to ineffective testing.

**Difference between Defect, Bug, Error and Failure**

1) **Defect** – When developers are doing Unit testing to check whether functionality is perfect or not in their local set up, now they compare actual result with the expected result, there is any difference between the actual result and expected result, this leads to a defect. Basically this occurred during the development phase.

2) **Bug** – When Tester is testing the application, if they find variation between actual result and expected result, this will be considered as Bug. Also defects accepted by developers can be considered as bugs. Bug is also an informal name given to the defect.

3) **Error** – Mistakes done in the program while writing the code, because of which the developer is not able to write and compile the code is called an Error.

4) **Failure** – Once software is developed, it is tested and verified by our testers. Finally end users are facing issues in the production, it is termed as failure.

**Severity and Priority**

1) **Severity** – Impact of Bug on customer business workflow is called Severity.

If the impact is high, it can be considered as high severity.

If the impact is low, it can be considered as low severity.

2) **Priority** – Importance given to fix the bug is called Priority.

**Types of Severity:**

1) Blocker

2) Critical

3) Major

4) Minor

1) **Blocker** – Lets say, when the Tester is testing the application it is not allowing the user to proceed further, it may say internal server error or might take the user to a blank page, the functionality itself is blocked. This is considered a Blocker bug.

2) **Critical** – If the main functionality of the application is not working or not according to customer requirement specification, it can be considered a Critical bug. In other words, if Tester is sure that this bug affects customer business flow, then it can be termed as Critical bug. Note that it is not blocking the user, but customer business work flow is affected largely because of this bug.

3) **Major** – If the Tester is not sure that this bug which is found is affecting the customer business workflow or not, it can be considered as a Major bug. In other words, if minor functionalities of an application is affected, it can be considered a Major bug.

4) **Minor** – If the Tester is sure that this bug is definitely not affecting the customer business workflow, it can be considered a minor bug. These are usually with respect to the look and feel of the application, alignment issues, spell issues etc.

**Types of Priority**

1) P0 – High Priority, the developer has to fix the bug immediately.

2) P1 – Medium Priority, developers can fix the bug in upcoming builds.

3) P2 – Low Priority, developers can fix the bug in the coming releases.

Blocker and Critical bugs can be considered as High severity.

Examples:

**High Severity High Priority** – While logging into the login page, a blank page is displayed. It is a blocker bug and Priority is P0 because this has a large impact on customer business workflow.

**High Severity Low Priority** – When a user clicks on the Terms and conditions page, a blank page is displayed. It is High Severity low priority because this does not affect customer business workflow.

**Low Severity High Priority** – When logging in to the login page, let’s say login spelling is wrong (logone), now this low severity, but it will impact the user as it is displayed in the first page itself.

**Low Severity Low Priority** – Lets say there is a spelling mistake in the inside contact information page with respect to email Id, phone etc or any color formation issue or alignment issues. This can be considered as a Low severity and low priority.