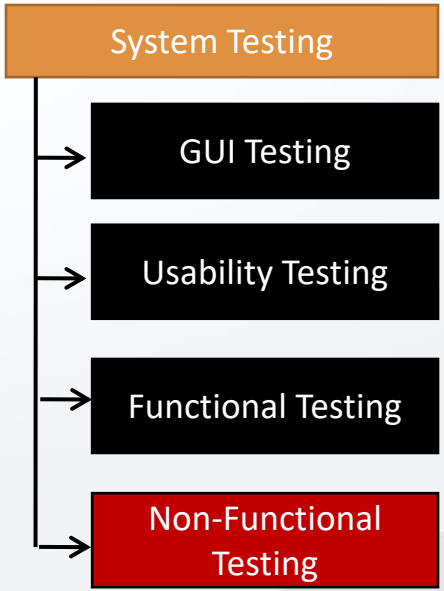
SESSION 5

Non-functional Testing

1. Performance Testing
    - Load Testing
    - Stress Testing
    - Endurance Testing
    - Spike Testing
    - Volume Testing
2. Security Testing
3. Recovery Testing
4. Compatibility Testing
5. Configuration Testing
6. Installation Testing
7. Sanitation/Garbage Testing



Load Testing

Load testing is a type of performance testing which involves evaluating the performance of the system **under the expected workload**.

A typical load test includes determining the response time, throughput, error rate, etc during the course of the load test.

Example – For a newly developed application with an anticipated load of around 1000 concurrent users. We will create a load test script and configure it with 1000 virtual users and run it for say 1-hour duration. After the load test completion, we can analyse the test result to determine how the application will behave at the expected peak load.

Stress Testing

Stress testing is a type of performance testing where we evaluate the application's performance at a load **much higher than the expected load**.

Another aspect of the stress testing is to determine the break-point of the application, the point at which the application fails to respond in the correct manner.

Example – For an application with an anticipated load of 1000 users we will run the test with 1200 users and check if the application is robust enough to not crash.

Endurance Testing (or) Soak Testing

Endurance testing is also known as **Soak Testing**.

It is done to determine if the system can **sustain the continuous expected load for a long duration**.

Issues like memory leakage are found with endurance testing.

Example – For an application like Income tax filing, the application is used continuously for a very long duration by different users. In this type of application, memory management is very critical. For an application like these, we can run the test for 24 hours to 2 days duration and monitor the memory utilization during the whole test execution.

Spike Testing

In **spike testing**, we analyse the behaviour of the system on **suddenly increasing the number of users**.

It also involves checking if the application is able to recover after the sudden burst of users.

Example – For an e-commerce application running an advertisement campaign, the number of users can increase suddenly in a very short duration. Spike testing is done to analyse these types of scenarios.

Volume Testing

The **volume testing** is performed by **feeding the application with a high volume of data**.

The application can be tested with a large amount of data inserted in the database or by providing a large file to the application for processing.

Using volume testing, we can identify the bottleneck in the application with a high volume of data.

Example – For a newly developed e-commerce application, we can perform volume testing by inserting millions of rows in the database and then carry out the performance test execution.

Performance Testing Types - Summary

Test Type	Goal	Method	
Load Testing	Understand performance under expected load	Gradual load increase	Expected Load
Stress Testing	Identify breaking points	Extreme or unrealistic load	Beyond Expected Load
Endurance Testing	Assess long-term stability	Sustained load over time	Expected Load for long time
Spike Testing	Evaluate response to sudden traffic bursts	Short-duration load spikes	Sudden change in Load
Volume Testing	Handle large data volumes	Large datasets or data transfer rates	High volumes of data

Security Testing

- Security Testing checks if the software is secure and protects sensitive information.
- It aims to find and fix vulnerabilities that could be exploited by hackers.
- Focus on
  - User Authentication
  - User Authorization/ access control
  - Data Encryption & Decryption
  - Network Security
  - System Software Security
  - Client-side Application Security
  - Server-side Application Security
- **Example:** Testing an online banking application to make sure that user account information is protected from unauthorized access.

Recovery Testing

- Recovery Testing assesses how well a system can recover after a failure or crash. It checks if the software can resume normal operation without losing data.
- Software should be recovery tested for failures like :
  - Power supply failure
  - The external server is unreachable
  - Wireless network signal loss
  - Physical conditions
  - Database server Down
  - API’s response failed etc.
- **Example:** Simulating a sudden power outage and checking if a word processing software can recover the document being edited when power is restored.

Compatibility Testing

- Compatibility Testing ensures that the software works well on different devices, browsers, and operating systems. It verifies that the application is compatible with a variety of environments.
- Focus on
  - Operating System compatibility
  - Browser compatibility (Cross browser testing)
  - Hard ware compatibility (Configuration testing)
  - Backward compatibility
  - Forward compatibility
- **Example:** Testing a mobile app on different smartphones and ensuring it functions correctly on various screen sizes and resolutions.

Configuration Testing

- Configuration Testing checks if the software works correctly with different configurations or settings. It ensures that the application adapts well to various setups.
- **Example:** Testing a video game on different computers with varied hardware configurations to ensure smooth gameplay on various setups.

Installation Testing

- Installation Testing assesses the process of installing and uninstalling the software. It checks if the installation is smooth and if the software can be removed without causing issues.
- **Example:** Installing a new software version and checking if it sets up properly without errors or conflicts with existing installations

Sanitation/Garbage Testing

- Sanitation/Garbage Testing involves checking for unnecessary or leftover data in the system. It ensures that the software cleans up after itself and doesn't leave unused or "garbage" data behind.
- **Example:** Testing a messaging app to make sure that deleted messages are completely removed from the system and don't linger in the background.

Functional Testing vs. Non-Functional Testing

Functional Testing:

- **Focus:** Ensures that the software functions as expected, performing its intended tasks.
- **What It Checks:** Specific features, actions, and behaviors outlined in the requirements.

Non-Functional Testing:

- **Focus:** Evaluates how well the software performs under various conditions and assesses aspects beyond specific functionalities.
- **What It Checks:** Performance, security, usability, and other aspects related to the user experience.

Major Difference:

- Functional Testing is about WHAT the software does, ensuring individual features work correctly.
- Non-Functional Testing is about HOW WELL the software performs, assessing aspects like speed, security, and overall user experience.