System Testing

System Testing is an essential part of the **software development lifecycle**. It is conducted to evaluate the software system as a whole and ensure that it meets specified requirements.

---

1. Definition

- System testing is a level of testing where the complete and integrated software is tested to verify that it meets all specified requirements.
- It involves testing the end-to-end functioning of the application, covering all modules and features.

2. Objectives of System Testing

- **Verify Requirements**: Ensure all user and system requirements are met.
- **Identify Defects**: Detect any bugs or issues that could impact software performance.
- **Assess Usability**: Evaluate how the system will function in a real-world environment.
- **Validate Integration**: Confirm that different modules work together without issues.

3. Types of System Testing

- **Functional Testing**: Checks if each function of the software works as per the requirements.
- **Non-Functional Testing**: Tests aspects like performance, scalability, and reliability.
- **End-to-End Testing**: Simulates real-world scenarios to see how the system behaves.
- **User Interface (UI) Testing**: Verifies the look and feel of the application, ensuring it aligns with user expectations.

4. Examples of System Testing

- Testing an online banking system by performing tasks like login, checking account balance, and transferring funds.
- Testing an e-commerce website from browsing products to completing a purchase.

5. Importance of System Testing

- Ensures a quality product by catching issues early.
- Improves customer satisfaction by delivering reliable software.
- Reduces maintenance costs and post-release issues.

6. System Testing Process

1. **Requirement Analysis**: Understand all system requirements.

2. **Test Planning**: Define the scope and approach for system testing.
3. **Test Case Design**: Write test cases for different scenarios.
4. **Test Execution**: Execute test cases and log results.
5. **Defect Reporting**: Report any issues to the development team.
6. **Test Closure**: Evaluate the testing process and results to ensure completeness.

7. Challenges in System Testing

- High dependency on complete and integrated modules.
- Requires comprehensive test cases for covering different scenarios.
- Managing test data to simulate real-world usage.