Q 1. Software Testing Life Cycle (STLC)

The **Software Testing Life Cycle (STLC)** refers to a systematic process that defines the phases involved in testing software to ensure its quality and compliance with requirements. Each phase has specific objectives, deliverables, and entry/exit criteria.

Phases of STLC

1. **Requirement Analysis**
    - **Goal**: Understand the testing requirements.
    - **Activities**:
        - Analyze software requirements from the SRS document.
        - Identify testable requirements.
        - Prepare a **Requirement Traceability Matrix (RTM)**.
    - **Deliverables**:
        - Requirement Traceability Matrix (RTM).
        - Automation feasibility report.
2. **Test Planning**
    - **Goal**: Define the scope, strategy, and resources for testing.
    - **Activities**:
        - Create a test plan document.
        - Estimate testing effort, schedule, and resources.
        - Define entry and exit criteria.
    - **Deliverables**:
        - Test plan and test strategy.
        - Resource allocation plan.
3. **Test Case Design**
    - **Goal**: Create detailed test cases and test data.
    - **Activities**:
        - Write test cases covering all scenarios (positive and negative).
        - Prepare test data.
        - Review test cases for accuracy.
    - **Deliverables**:
        - Test cases and test scripts.
        - Test data sets.

4. **Environment Setup**
    - **Goal**: Prepare the hardware/software environment for testing.
    - **Activities**:
        - Set up test servers, databases, and tools.
        - Configure the testing environment as per requirements.
        - Validate the test environment.
    - **Deliverables**:
        - Environment setup confirmation.
        - Test environment access credentials.
5. **Test Execution**
    - **Goal**: Execute test cases and log defects.
    - **Activities**:
        - Execute test cases manually or via automation.
        - Record the actual results.
        - Log defects in a defect-tracking tool.
    - **Deliverables**:
        - Test execution results.
        - Defect reports.
6. **Test Closure**
    - **Goal**: Complete testing activities and evaluate quality.
    - **Activities**:
        - Conduct test closure meeting.
        - Document test summary report.
        - Archive test artifacts for future reference.
    - **Deliverables**:
        - Test summary report.
        - Lessons learned and best practices.

Q 2. Types of Testing

There are various types of software testing, broadly categorized into **functional** and **non-functional testing**:

Functional Testing

Verifies the system against functional requirements.

1. **Unit Testing**
    - Tests individual components or modules.
    - Usually automated and done by developers.
2. **Integration Testing**
    - Tests interactions between modules or systems.
    - Types:
        - **Top-down**: Testing higher-level modules first.
        - **Bottom-up**: Testing lower-level modules first.
3. **System Testing**
    - Validates the entire system as a whole.
    - Performed in an environment close to production.
4. **Acceptance Testing**
    - Ensures the system meets business requirements.
    - Types:
        - **User Acceptance Testing (UAT)**.
        - **Operational Acceptance Testing (OAT)**.
5. **Regression Testing**
    - Verifies that new changes don’t break existing functionality.
6. **Smoke Testing**
    - Quick tests to verify basic functionality before deeper testing.
7. **Sanity Testing**
    - Focused testing to verify minor fixes or changes.

Non-Functional Testing

Focuses on system performance, usability, reliability, etc.

1. **Performance Testing**
    - Measures system performance under different conditions.
    - Types:
        - **Load Testing**: Normal load conditions.

        - **Stress Testing**: Beyond normal load limits.
        - **Spike Testing**: Abrupt load increases.

2. **Security Testing**
    - Ensures the system is protected against vulnerabilities (e.g., hacking, data breaches).
3. **Usability Testing**
    - Validates ease of use and user experience.
4. **Compatibility Testing**
    - Checks system behavior across different environments (browsers, OS, devices).
5. **Reliability Testing**
    - Ensures consistent performance under expected conditions.
6. **Localization and Internationalization Testing**
    - **Localization Testing**: Tests content and UI for a specific locale.
    - **Internationalization Testing**: Ensures system supports multiple languages and regions.
7. **Accessibility Testing**
    - Ensures the system is usable by people with disabilities.

Summary Table

Type	**Description**
Unit Testing	Tests individual components.
Integration Testing	Tests interactions between modules.
System Testing	Validates the system as a whole.
Performance Testing	Checks speed, scalability, and stability.
Security Testing	Validates data and system security.
Regression Testing	Ensures changes haven’t broken functionality.