TUTORIAL 6

6.Explore and summarize different static and dynamic testing techniques.

Static vs Dynamic Testing Techniques

What is Software Testing?

Software testing is the process of evaluating and verifying that a software product or application does what it is supposed to do. There are **two main types** of software testing:

- **Static Testing**
- **Dynamic Testing**

1. STATIC TESTING

Definition:

Static testing involves checking the code, requirement documents, and design documents **without executing the program**. It helps to detect errors early in the software development lifecycle.

Static Testing Techniques

A. Review Techniques

1. Informal Review

- No formal process.
- Developers or testers read documents or code.
- Example: A developer asking a teammate to glance over code or design.

2. Walkthrough

- Author explains the document/code to a group.
- Group gives suggestions but doesn't find faults.
- Step-by-Step:

    - Prepare document.
    - Invite team.
    - Explain content.
    - Note feedback.

3. Technical Review

- Conducted by a team of technical experts.
- Goal: Examine the quality and logic of the code/design.
- Focus: Use of correct algorithms, architecture.

4. Inspection

- Most formal review method.
- Uses a predefined process and roles (Moderator, Author, Reader, Recorder).
- Step-by-Step:
    - Plan inspection session.
    - Distribute materials.
    - Conduct inspection.
    - Log defects.
    - Review and follow-up.

B. Static Analysis

5. Code Analysis using Tools

- Tools scan source code to detect issues (e.g., syntax errors, unused variables).
- Does **not execute** the program.
- Examples of tools: SonarQube, Pylint, Checkstyle

6. Data Flow Analysis

- Checks variable definitions and usages.
- Detects issues like:

- Use before definition
- Unused variables

7. Control Flow Analysis

- Understands possible paths through the program.
- Identifies unreachable code, infinite loops.

Summary of Static Testing

Technique	Description	Purpose
Informal Review	Casual review	Quick feedback
Walkthrough	Author-led session	Clarify content
Technical Review	Expert feedback	Improve technical quality
Inspection	Formal process	Detect defects
Static Analysis	Tool-based scan	Find bugs without running code

2. DYNAMIC TESTING

Definition:

Dynamic testing involves **executing the code** to validate the software behavior against expected results. It is done at different levels of software testing.

Dynamic Testing Techniques

A. White Box Testing (Structural Testing)

1. Unit Testing

- Tests individual units/modules.
- Done by developers.
- Tools: JUnit, NUnit

2. Integration Testing

- Tests interaction between modules.
- Types:
    - Top-down
    - Bottom-up
    - Big Bang

3. Code Coverage Techniques

- Measures how much code is tested.
- Types:
    - Statement Coverage
    - Branch Coverage
    - Path Coverage

B. Black Box Testing (Functional Testing)

4. Equivalence Partitioning

- Divide input into valid and invalid partitions.
- Test one input from each partition.

5. Boundary Value Analysis

- Test at edge values (min, max, just inside/outside).
- Example: For input range 1–100, test: 0, 1, 100, 101

6. Decision Table Testing

- Test combinations of conditions and actions.
- Useful in business logic.

7. State Transition Testing

- Tests behavior for various states and transitions.
- Example: ATM (Idle → Card Inserted → PIN Entry)

8. Error Guessing

- Based on tester's intuition and experience.
- Guess areas where software may fail.

C. Non-functional Testing

9. Performance Testing

- Tests speed, scalability.
- Tools: JMeter, LoadRunner

10. Security Testing

- Tests for vulnerabilities.
- Examples: SQL Injection, Broken Authentication

Summary of Dynamic Testing

Technique	Type	Description
Unit Testing	White Box	Tests smallest components
Integration Testing	White Box	Checks interaction of modules
Equivalence Partitioning	Black Box	Divide inputs into groups
BVA	Black Box	Test at boundary values
State Transition	Black Box	Test state changes
Decision Table	Black Box	Rules-based testing
Error Guessing	Black Box	Based on experience
Performance Testing	Non-functional	Test system performance

Step-by-Step for Students

🔹 Step 1: Understand the Types

- Learn difference between **Static** (no execution) and **Dynamic** (with execution).

🔹 Step 2: Learn Techniques under Each Type

- Static: Reviews, inspections, static analysis.
- Dynamic: Unit tests, BVA, state transition, etc.

🔹 Step 3: Take a Sample Scenario

Example: Online Banking Login

- Static: Review requirement docs, check code for syntax
- Dynamic: Write test cases → valid login, invalid login → execute → verify results

🔹 Step 4: Use Tools (Optional)

- Static: SonarQube, Pylint
- Dynamic: JUnit (Java), Selenium (UI testing), Postman (API)

🔹 Step 5: Document and Report

Create a test document with:

- Technique used
- Test case steps
- Result
- Defect (if any)

Practical Example for Students

- **Static Testing:**
  - Review login page HTML/CSS.
  - Check code indentation, naming conventions.
  - Use Pylint to check the Python login module.

Dynamic Testing:

TC_ID	Test Scenario	Test Data	Expected Result
TC_001	Valid Login	user123 / pass@123	Success
TC_002	Invalid Login	user123 / wrongpass	Error Message

Conclusion

Static Testing	Dynamic Testing
Done without executing code	Code is executed
Early defect detection	Validates actual behavior
Less expensive	More resource-intensive
Examples: Code reviews, inspections	Examples: Unit tests, UI tests