Software Testing

CSC224

Lecture 7 Notes

Learning Objectives:

- Understand the role and importance of software testing in the development lifecycle.
- Explore the different types and levels of software testing.
- Learn best practices for writing and executing test cases.
- Understand the role of automation in software testing.

Prelude

- **1. What is Software Testing?**
- **Definition:**
  Software testing is the process of evaluating a software system to identify and fix defects, ensure it meets specified requirements, and validate its functionality and performance under defined conditions.
- **Key Goals:**
  - Verify that the software works as intended.
  - Identify and fix bugs before deployment.
  - Validate that the software meets user needs and expectations.
- **Importance of Testing:**
  - Enhances software quality.
  - Reduces maintenance costs by identifying issues early.
  - Builds user trust by ensuring reliability and usability.

2. The Software Testing Process

- **Phases of Testing:**

1. Test Planning:

- Define objectives, scope, and schedule of testing activities.
- Identify resources and tools required.

2. Test Design:

- Develop test cases and scenarios based on requirements.
- Determine expected results.

3. Test Execution:

- Run the test cases and document results.
- Report identified defects.

4. Defect Tracking and Reporting:

- Log defects in a tracking system.
- Monitor and verify defect resolution.

5. Test Closure:

- Summarize test activities.
- Ensure all planned tests are executed and objectives met.

3. Levels of Software Testing

- Testing is performed at different levels of software development to ensure comprehensive validation:
- **A. Unit Testing**
  - Tests individual components or functions of the software.
  - Conducted by developers during implementation.
  - Tools: JUnit (Java), pytest (Python).
- **B. Integration Testing**
  - Verifies the interaction between integrated components.
  - Identifies issues with data flow or module communication.
  - **Types:**
    - **Top-Down Testing:** Test higher-level modules first.
    - **Bottom-Up Testing:** Start testing from lower-level modules.

3. Levels of Software Testing

- **C. System Testing**
  - Tests the complete, integrated system to ensure it meets requirements.
  - Includes functional and non-functional testing.
- **D. Acceptance Testing**
  - Conducted by end-users or clients to ensure the software satisfies their needs.
  - **Types:**
    - **Alpha Testing:** Performed in a controlled environment by internal users.
    - **Beta Testing:** Conducted by external users in a real-world environment.
      - E.g. Beta-version

4. Types of Software Testing

- **A. Functional Testing**
  - Verifies that the software performs the intended functions.
  - Based on functional requirements.
  - Example: Ensuring a login system accepts valid credentials.
- **B. Non-Functional Testing**
  - Tests system attributes like performance, usability, and security.
  - Examples:
    - **Performance Testing:** Checks response time and scalability.
    - **Usability Testing:** Assesses user-friendliness.
    - **Security Testing:** Identifies vulnerabilities.

4. Types of Software Testing

- **C. Regression Testing**
  - Ensures new changes don’t break existing functionality.
  - Typically automated due to repetitive nature.
- **D. Exploratory Testing**
  - Tester explores the application without predefined test cases.
  - Useful for discovering edge cases or unexpected behaviors.
- **E. Automated Testing**
  - Uses tools to execute prewritten test scripts.
  - Examples:
    - Selenium for web applications.
    - Appium for mobile apps.
  - **Benefits:** Faster execution, consistent results, and reusability.

5. Writing Effective Test Cases

- **What is a Test Case?**
  A set of inputs, execution conditions, and expected results to validate specific software functionality.
- **Components of a Good Test Case:**
  - **Test Case ID:** Unique identifier for tracking.
  - **Description:** Brief summary of the test.
  - **Preconditions:** Requirements or setup needed before execution.
  - **Steps:** Clear, step-by-step instructions.
  - **Expected Results:** The correct outcome if the test passes.
  - **Actual Results:** Outcome observed during execution.

Example Test Case:

Test Case ID	**TC001**
Description	Validate login functionality.
Preconditions	User has a valid username and password.
Steps	1. Open login page. 2. Enter valid credentials. 3. Click "Login."
Expected Result	User is redirected to the dashboard.
Actual Result	To be filled after execution.

6. Testing Automation

- **What is Test Automation?**
  The use of software tools to execute predefined test scripts and compare actual results with expected outcomes.
- **Advantages:**
  - Faster execution of repetitive tests.
  - Reduces human error.
  - Facilitates continuous integration and deployment.
- **Popular Automation Tools:**
  - Selenium (Web applications).
  - JUnit/TestNG (Unit testing).
  - Postman (API testing).
  - Appium (Mobile testing).

7. Testing Metrics and KPIs

- To measure the effectiveness of testing efforts:
- **Test Coverage:** Percentage of code or requirements covered by tests.
- **Defect Density:** Number of defects per module or function.
- **Defect Removal Efficiency (DRE):** Ratio of defects removed during testing to total defects found.
- **Pass/Fail Rate:** Percentage of test cases that pass or fail.

8. Example: Testing an E-Commerce Application

Test Case ID	**TC002**
Description	Validate "Add to Cart" functionality.
Preconditions	User is logged in. Product exists in inventory.
Steps	1. Search for a product. 2. Click "Add to Cart."
Expected Result	Product is added to the cart with correct details.

9. Key Takeaways

- Testing is an integral part of the software development lifecycle, ensuring quality and reliability.
- Different levels and types of testing address various aspects of the system.
- Writing effective test cases and using automation tools can significantly enhance the testing process.
- Continuous testing and monitoring are essential for maintaining software quality in production.

Discussion Questions

- Compare functional and non-functional testing. Why are both important?
- Discuss the advantages and challenges of automated testing.
- How would you prioritize test cases in a project with tight deadlines?

Practical Activity

- **Objective:** Develop and execute test cases for a sample application.
- **Task:**
  - Choose a system (e.g., library management or online shopping).
  - Identify functional and non-functional requirements.
  - Write at least three test cases, including preconditions, steps, and expected results.
  - Execute the test cases and document the actual results.