COMSATS UNIVERSITY ISLAMABAD, ABBOTTABAD CAMPUS

SOFTWARE TESTING

ASSIGNMENT 1

SAUD KHAN

FA21-BSE-033

SUBMITTED TO

SIR MUKHTIAR ZAMIN

Table of Contents

Test Plan:

A test plan is a detailed document outlining the objectives, scope, approach, resources, and schedule for a software testing project. It serves as a roadmap for the testing process, ensuring that all aspects of the software are thoroughly examined to meet quality standards before deployment.

Evolution Of Test Plan:

Following is the evolution of test plan:

1. Early Template (1980s – Early 1990s)
2. Structured Template (Mid 1990s – 2000s)
3. Methodology-Driven Template (2000s – 2010s)
4. Comprehensive & Metrics-Focused Template (2010s – Till Now)

1. Early Template (1990s -Early 1990s);

Structure: This was very basic template of test plan (like a checklist we use).

Focus: Simple verification that features were present and seemed to work on a basic level.

Quality Consideration: These templates primarily addressed the most basic need to have a plan but lacked the detail to ensure comprehensive or thorough testing.

Template:

Section	Explanation
Introduction	One or two sentences describing the project
Feature to test	A simple list of functionalities
Test Result	Columns for pass/fail next to each feature

Example:

Basic word processing software in the late 1980s.

Introduction: Our project aims to develop a basic word processing software to enable users to create, save, and print documents efficiently.

Feature to test:

- Typing
- Saving
- printing

Test Results:

- typing: Pass
- saving: pass
- printing: Fail

2. Structured Template (Mid 1990s – 2000s):

Structure: Gained organization and formality, often driven by the rise of structured development methodologies.

Focus: Ensuring the plan aligned with requirements and had a clear purpose.

Quality Considerations: A more defined structure improved communication and set clearer expectations. Testers began thinking more deeply about the "why" of testing.

Template:

Section	Explanation
Introduction	Brief project overview
Objectives	Concise statements of what testing aims to achieve.
Scope	Clear definition of what's included/not included.
References	Linkage to formal requirements documents.
Test cases	Inputs, Steps, Expected Results for each test.
Environment	Basic description of hardware/software needed.

Example:

Develop a reliable inventory management software to streamline our business operations.

Introduction: Our project aims to develop a reliable inventory management software to streamline our business operations.

Objectives:

- Verify the functionality of essential features such as inventory tracking and reporting.
- Identify and rectify any defects in the software to ensure smooth operation.
- Ensure compliance with the requirements outlined in the project specifications.

Scope: This test plan encompasses the testing of core functionalities of the inventory management software, including data entry, storage, retrieval, and reporting. It does not extend to testing peripheral features such as user interface enhancements or integration with external systems.

References: This test plan is based on the formal requirements documented in the Project Requirements Specification (PRS) version 2.0.

Test Cases:

1. Data Entry:
    - Input: Enter product details into the system.
    - Steps: Access the data entry module, input product information, and save.
    - Expected Results: Product details are accurately recorded in the database without errors.
2. Data Storage:
    - Input: Save inventory data to the database.
    - Steps: Perform data entry, save the information to the database, and verify integrity.
    - Expected Results: Inventory data is stored securely in the database and can be retrieved without loss.
3. Reporting:
    - Input: Generate inventory reports.
    - Steps: Access the reporting module, select report parameters, and generate the report.
    - Expected Results: Inventory reports are generated accurately and formatted appropriately for analysis.

Environment:

- Hardware: Minimum requirements include a desktop PC with 2GB RAM and a Pentium processor.
- Software: Compatible with Windows 95/98 operating system, Microsoft Access database software installed.

3. Methodology-Driven Template (2000s – 2010s):

Structure: Reflects Agile and iterative development trends, with flexibility built in.

Scope: Adapting testing to the pace of development, outlining the types of tests used at different stages.

Quality Considerations: Explicitly considering testing methodology ensures a more targeted and efficient approach aligned with the project's overall development strategy.

Template:

Section	Explanation
Introduction	Brief project overview.
Objectives	Concise statements of what testing aims to achieve.
Scope	Clear definition of what's included/not included.
Test approach	Unit Testing, Integration Testing, System Testing, etc.
Test cases	May include prioritization (High, Medium, Low).
Environment	Basic description of hardware/software needed.
Roles responsibilities	Who performs what types of tests.
Risks	Potential issues & contingency plans.

Example:

development of an e-commerce platform to facilitate online shopping for our customers.

Introduction:

Our project involves the development of an e-commerce platform to facilitate online shopping for our customers.

Objectives:

- Ensure the functionality and reliability of the e-commerce platform.
- Identify and rectify any defects in a timely manner to maintain project timelines.
- Align testing activities with Agile development principles to adapt to changing requirements.

Scope:

This test plan covers the testing of the e-commerce platform's core functionalities, including user registration, product browsing, shopping cart management, and checkout processes. It excludes testing of advanced features such as recommendation engines or social media integration.

Test Approach:

- Unit Testing: Conducted by individual developers to validate the functionality of code units.
- Integration Testing: Verifies the interaction between integrated modules to ensure smooth operation.
- System Testing: Validates end-to-end functionality and performance of the entire system.
- Acceptance Testing: Conducted by stakeholders to confirm that the system meets specified requirements.

Test Case:

- High Priority:
    1. User Registration: Ensure users can register successfully.
    2. Product Search: Verify that products can be found using search functionality.
    3. Checkout Process: Test the end-to-end functionality of the checkout process.
- Medium Priority:
    1. Shopping Cart Management: Verify the accuracy of items added to and removed from the shopping cart.
    2. Payment Processing: Test payment processing functionality with different payment methods.
    3. Order Confirmation: Verify that users receive order confirmation emails after completing a purchase.
- Low Priority:
    1. Account Settings: Test the functionality to update user account settings.

2. Newsletter Subscription: Verify that users can subscribe to newsletters.
3. Customer Support: Test the functionality to contact customer support.

Environment:

- Hardware: Minimum requirements include a server with 4GB RAM and a dual-core processor.
- Software: Compatible with Linux-based server environment, Apache web server, MySQL database.

Roles and Responsibilities:

- Developers: Responsible for unit testing and fixing defects identified during testing.
- QA Engineers: Responsible for integration testing, system testing, and creating and executing test cases.
- Product Owners: Responsible for acceptance testing and ensuring the final product meets business requirements.

Risks:

- Potential issues may arise due to third-party integration dependencies.
- Contingency plan: Regular communication and collaboration with third-party vendors to address integration challenges promptly.

4. Comprehensive & Metrics-Focused Template (2010s – Till Now):

Structure: Emphasize automation, continuous testing, and data-driven insights. Testers and developers collaborate closely.

Scope: Maximizing feedback loops, measuring effectiveness, and treating testing as an integral part of software quality assurance.

Quality Considerations:

Comprehensive templates drive a proactive approach to quality, measure progress, and enable informed decision-making about test strategy adjustments over time.

Template:

Section	Explanation

Introduction	Brief project overview.
Objective	Concise statements of what testing aims to achieve.
Scope	Clear definition of what's included/not included.
Test approach	May include automation, continuous testing, security testing, performance testing.
Test cases	Highly detailed, may link to code repositories or test scripts.
Environment	Cloud simulators, different device types, etc.
Roles & responsibilities	May include developers in some testing activities.

Example:

mobile banking application to provide convenient banking services to our customers.

Introduction:

Our project involves the development of a mobile banking application to provide convenient banking services to our customers.

Objective:

- Ensure the functionality, security, and performance of the mobile banking application.
- Maximize feedback loops between testers and developers to identify and address issues promptly.
- Use data-driven insights to continuously improve the quality of the application throughout the development lifecycle.

Scope:

This test plan encompasses the testing of all aspects of the mobile banking application, including user authentication, account management, fund transfers, and transaction history. It includes security testing, performance testing, and compatibility testing across multiple devices and platforms.

Test Approach:

- Automation Testing: Implement automated test scripts for regression testing, smoke testing, and sanity testing to expedite the testing process and improve test coverage.
- Continuous Testing: Integrate testing activities seamlessly into the development pipeline to provide immediate feedback on code changes and ensure continuous quality assurance.

- Security Testing: Conduct penetration testing and vulnerability assessments to identify and mitigate potential security risks.
- Performance Testing: Evaluate the responsiveness and scalability of the application under different load conditions to ensure optimal performance.

Test Cases:

Detailed test cases are documented in a test management tool and linked to the code repository for traceability. Test cases cover various scenarios, including positive and negative test cases, edge cases, and boundary conditions, to ensure comprehensive test coverage.

Environment:

- Cloud Simulators: Utilize cloud-based testing environments to simulate different network conditions and device types for compatibility testing.
- Various Device Types: Test the application on a range of mobile devices, including smartphones and tablets, running different operating systems such as iOS and Android.

Roles & Responsibilities:

- Testers: Responsible for creating test cases, executing tests, and reporting defects.
- Developers: Collaborate closely with testers to understand testing requirements, review test cases, and fix defects identified during testing.
- Automation Engineers: Develop and maintain automated test scripts to support continuous testing efforts.
- Security Experts: Conduct security testing activities to identify and address potential vulnerabilities in the application.

Summary:

In the early days, testing was basic, like using checklists to ensure basic functionality. Then, it became more structured with formal documents outlining objectives and scope. As projects evolved, testing adapted, specifying types of testing for efficiency. Today, testing is comprehensive, using automation and examining all aspects of software, including performance and safety. Detailed test plans track progress and aid decision-making.