ACCELQ

TEST CASE DESIGN

A Guide for QA Engineers

With Examples

Test Case Design

Concise plan by Quality Analysts detailing testing strategy, steps, and expected outcomes to identify software defects and ensure product quality

Different Types of Testing

Formal
Detailed preconditions and expected outcomes.

Informal
Where testing occurs without set conditions, discovering outcomes as tests progress.

Integration Testing
Ensures different systems communicate and function together.

Performance Testing
Evaluates system performance under various conditions.

UI Testing
Focuses on visual elements and accessibility across multiple devices.

Security Testing
Identifies system vulnerabilities and protects against threats.

Database Testing
Ensures data compliance with data privacy laws and prevents unauthorized access.

Usability Testing
Assesses ease of use and ability to complete intended actions.

Functionality Testing
Verifies if specific functionalities like user login.

Different Types of Test Case Design Techniques

Specific-Based Techniques

- **Boundary Value Analysis (BVA)**
  Identifies errors at input value boundaries, assuming system stability within these limits.
- **Equivalence Partitioning (EP)**
  Divides input data into classes for testing each class equally.
- **Decision Table Testing**
  Uses a cause-and-effect table for mapping various inputs and outputs.
- **State Transition Diagrams**
  Tests application behavior under different input sequences, useful for workflow-specific systems.
- **Use Case Testing**
  Executes business scenarios and end-user functions to cover the entire system.

Structure-Based Techniques

- **Statement Coverage Testing**
  Executes all source code statements to measure executed vs. total statements.
- **Decision Testing Coverage**
  Executes at least one branch from each decision point to check for unexpected behaviors.
- **Condition Testing**
  Thoroughly tests all conditions in the source code for errors.
- **Multiple Condition Testing**
  Tests different condition combinations simultaneously for complete coverage.
- **All Path Testing**
  Identifies all executable paths and potential faults in the code.

Experience Based Techniques

Error Guessing
Predicts errors based on the tester's experience, skills, and intuition.

Exploratory Testing
Combines test design and execution, often used when time is limited.

How to Plan and Design Test Cases?

Prepare the Test Environment
Equip with required software versions and hardware specifications and document them for future reference.

Test Case Requirements
Design specifications, use cases, and software usability as the foundation.

Design the Test Case
Develop test cases for each software requirement, ensuring compliance with specifications.

What are the Best Practices for Test Case Design?

01 **Clear and Concise Test Cases**
Simple steps, straightforward language, and one expected result per test case with each unique ID

02 **Comprehensive Testing Coverage**
Employ techniques like BVA and EP for maximum functionality to detect and fix bugs

03 **Regular Updates with New Requirements**
Continuously update test cases to reflect new requirements and for future understanding

04 **Adherence to Testing Scope**
Stick to the defined scope - Avoid assumptions, focusing on actual requirements

05 **Leverage AI-Powered Automation Tools**
For efficient writing, tracking, and managing of test cases

Test Case Design is Crucial for Building Good-Quality Software

Essential for efficient and effective software testing.

Provides clear visibility on testing coverage.

Prevents release of poor-quality software with defects and bugs.

Ensures software is thoroughly tested, safe, and market-ready.