Test Plan

Created by Teodor Coste

A). Objective

The primary purpose of the testing activities outlined in this test plan is to ensure that the software system or application meets its designed specifications and requirements. This involves a rigorous process of verification and validation, aimed at identifying and resolving any discrepancies between the actual functionalities of the system and its expected outcomes.

Key Goals:

1. Verifying Software Functionality:
    - Conduct a thorough examination of the software's functional aspects.
    - Include tests covering all user scenarios and edge cases for comprehensive functionality coverage.
2. Ensuring Performance Standards:
    - Validate system performance under various conditions, including load and stress.
    - Focus on responsiveness, speed, and resource utilization.
3. Validating Security Measures:
    - Rigorously test the software's security aspects, including data protection and resilience against threats.
    - Include tests for authentication, authorization, data encryption, and vulnerability scanning.
4. Enhancing User Experience:
    - Assess the software from an end-user perspective for intuitiveness and ease of use.
    - Include tests for usability, accessibility, and compatibility.

Alignment with Project Requirements and Stakeholder Expectations:

- Align testing activities closely with the project's objectives and stakeholder requirements.
- Engage with stakeholders throughout the testing process.

Overall Objective:

- Deliver a software product that is technically sound, secure, and offers a seamless user experience.

B). Scope Inclusions

Define the boundaries of the testing process, detailing what will be tested and possibly what will be excluded. This section sets clear expectations and helps focus the testing efforts.

1. Test Environments
    - Development Environment: For initial testing phases.
    - Staging Environment: A replica of the production environment.
    - Specialized Environments: Environments with specific configurations or conditions.
2. Defect Reporting Procedure
    - Defect Tracking Tools: Examples include JIRA or Bugzilla.
    - Defect Lifecycle: How defects are reported, triaged, and resolved.
    - Severity and Priority Levels: Criteria for categorizing defects.
3. Test Strategy
    - Types of Testing: Functional, usability, performance, security.
    - Testing Methodologies: Agile, Waterfall, or hybrid approaches.
    - Test Coverage: Aspects of the application to be tested.
4. Test Schedule
    - Key Phases: Planning, execution, closure.
    - Milestones: Significant checkpoints in the testing process.
    - Deadlines: For each major testing activity.
5. Test Deliverables
    - Test Plans and Test Cases: Documentation of testing procedures.
    - Bug Reports: Documenting defects found.
    - Test Summary Reports: Overview of testing activities and outcomes.
6. Entry and Exit Criteria
    - Entry Criteria: Prerequisites for commencing testing.
    - Exit Criteria: Conditions for concluding the testing phase.
7. Test Execution
    - Entry Criteria: Criteria for starting test execution.
    - Exit Criteria: When to end test execution.
8. Test Closure
    - Entry Criteria: Starting the test closure phase.
    - Exit Criteria: Completing the test closure phase.
9. Tools
    - Test Management: Examples like TestRail.
    - Automation Tools: Examples like Selenium, Cypress or Playwright.
    - Performance Testing Tools: Examples like JMeter.
10. Risks and Mitigations
    - Identify potential risks and strategies for mitigation.
11. Approvals
    - This plan requires approval from:
      Project Manager: [Name]
      QA Lead: [Name]
      Stakeholders: [Names]