Software Test Plan

[Software Name / Version / Release]

Prepared by: Your Name

Date: August 25, 2025

# 1. Introduction

# 1.1 Purpose

This exhaustive Software Test Plan (STP) serves as a comprehensive blueprint for all testing activities related to [Software Name/Version/Release]. It details the testing strategy, objectives, scope, resources, timelines, methodologies, risks, and deliverables.

# 1.2 Scope

Inclusions: All modules, features, interfaces, performance aspects.  
Exclusions: Unsupported hardware, unapproved plugins, deferred features.  
Boundaries: Controlled environments (web, mobile, desktop).

# 1.3 References

SRS, HLD/LLD, Project Charter, User Stories, Regulatory Docs (GDPR, HIPAA), Standards (IEEE 829, ISO 29119, ISTQB Glossary, WCAG 2.1), Tool Docs (JIRA, Selenium).

# 1.4 Definitions

STP: Software Test Plan; SRS: Software Requirements Specification; RTM: Requirements Traceability Matrix; UAT: User Acceptance Testing; API: Application Programming Interface; CI/CD: Continuous Integration/Deployment.

# 1.5 Overview

Sections: Introduction, Objectives, Scope, Strategy, Environment, Deliverables, Roles, Schedule, Criteria, Risks, Defects, Metrics, Resources, Assumptions, Configuration, Communication, Appendices, Approvals.

# 2. Test Objectives

SMART objectives: 100% requirement coverage, <2s response, no high vulnerabilities, >85% usability, defect leakage <5%, risk reduction 80%, automate 70% regression.

# 3. Test Scope

# 3.1 Features to Be Tested

Module 1: User Management (login, RBAC, load test).  
Module 2: Business Logic (workflow, integrations, accuracy 99.99%).  
Module 3: Reporting (dashboards, export).  
Cross-cutting: Localization, responsiveness.

# 3.2 Features Not Tested

Deferred AI module, vendor APIs, deprecated endpoints.

# 3.3 Limitations

Time constraints, reliance on mock services.

# 3.4 In-Scope Scenarios

Positive/negative paths, boundary conditions.

# 4. Test Strategy

# 4.1 Approach

Risk-based, V-Model with agile feedback.

# 4.2 Types

Functional: smoke, sanity, regression.  
Non-Functional: performance, security, compatibility, usability, accessibility.  
Specialized: localization, installation.

# 4.3 Levels

Unit, Integration, System, UAT.

# 4.4 Techniques

Black-box, White-box, Exploratory, Pairwise, Model-based.

# 4.5 Automation

Selenium, RestAssured, Cypress. CI: Jenkins, GitHub Actions.

# 4.6 Manual Focus

Exploratory, usability, new features.

# 4.7 Design Process

Derive from SRS via RTM, peer reviews.

# 5. Test Environment

# 5.1 Hardware

Servers: Dual Xeon, 32GB RAM.  
Clients: Desktops, iPhone 14, Samsung S23.  
Network: latency simulation.

# 5.2 Software

OS: Win 11, macOS, CentOS.  
Browsers: Chrome, Firefox, Edge, Safari.  
DB: Oracle, MySQL.  
Tools: TestRail, JIRA, Jenkins, JMeter, ZAP.

# 5.3 Architecture

Dev → Test → Staging → Prod.

# 5.4 Setup/Teardown

Ansible, Docker, seeded data, validation checklists.

# 5.5 Maintenance

Daily backups, versioning, incident response.

# 6. Test Deliverables

Test Plan, Test Cases, Scripts, Logs, Reports, Summary, Lessons Learned.

# 7. Roles and Responsibilities

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Role | Responsibilities | Skills | Assigned To | Contact |
| Test Manager | Planning, approvals | ISTQB Adv. | [Name] | [Email] |
| Test Lead | Reviews, tracking | Leadership | [Name] | [Email] |
| Sr. Test Eng | Complex tests, automation | Scripting | [Name] | [Email] |
| Test Eng | Execution, defects | QA basics | [Name] | [Email] |
| Automation Eng | CI/CD integration | Java/Python | [Name] | [Email] |

# 8. Test Schedule

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Phase | Start | End | Duration | Dependencies | Resources |
| Planning | [Date] | [Date] | 5d | SRS | Test Manager |
| Design | [Date] | [Date] | 10d | Planning | QA Team |
| Execution | [Date] | [Date] | 25d | Environment | QA Team |
| Closure | [Date] | [Date] | 5d | Reports | Test Manager |

# 9. Entry and Exit Criteria

Entry: Approved SRS, stable build.  
Exit: All tests executed, no open criticals, coverage met.  
Suspension: >30% blocked.  
Go/No-Go: Defect trends, stakeholder input.

# 10. Risk Management

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Category | Description | Impact | Owner |
| R1 | Schedule | Dev delays | High | PM |
| R2 | Resources | Tester attrition | High | HR |
| R3 | Quality | High defect influx | Medium | QA Lead |

# 11. Defect Management

Lifecycle: New → Assigned → Fixed → Retest → Closed.  
Severity: Blocker, Critical, Major, Minor, Trivial.  
Priority: P1, P2, P3.  
Tools: JIRA, Bugzilla.  
RCA: Fishbone, Pareto.

# 12. Test Metrics

Coverage %, Defect Density, Pass Rate, Defect Leakage, and Automation ROI.

# 13. Resources

Human resources, training, budget, and facilities.

# 14. Assumptions, Dependencies, Constraints

Assumptions: Stable requirements.  
Dependencies: Dev/IT support.  
Constraints: Fixed release dates.  
Change Control: Formal assessment.

# 15. Configuration Management

Git for cases/scripts, tagged releases, audits.

# 16. Communication Plan

Daily stand-ups, weekly status reports, escalation matrix.

# 17. Appendices

# 17.1 RTM

Req ID | Description | Test Cases | Status

# 17.2 Sample Test Case

ID: TC-001 | Title: Login Test | Expected: Redirect to dashboard.

# 17.3 Sample Test Script

Example: Selenium Python login script.

# 17.4 Risk Register

Extended version of Section 10.

# 17.5 Glossary

Extended project-specific terms.

# 17.6 Change History

Version | Date | Change | Author | Reviewer

# 18. Approval and Sign-Off

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Title | Role | Signature | Date |
| [Name] | PM | Project Manager |  |  |
| [Name] | QA Lead | Test Manager |  |  |
| [Name] | Dev Manager | Development |  |  |
| [Name] | Stakeholder | Client |  |  |